

PIM Working Group  
Internet Draft  
Intended status: Standards Track  
Expires: December 1, 2018

Dave Allan  
Ericsson  
Jeff Tantsura  
Nuage  
Ian Duncan  
Ciena  
June 1, 2018

A Framework for Computed Multicast Applied to SR-MPLS  
draft-allan-pim-sr-mpls-multicast-framework-00

Abstract

This document describes a multicast solution for SR-MPLS. It is consistent with the Segment Routing architecture in that an IGP is augmented to distribute information in addition to the link state. In this solution it is multicast group membership information sufficient to synchronize state in a given network domain. Computation is employed to determine the topology of any loosely specified multicast distribution tree.

Status of this Memo

This Internet-Draft is submitted to IETF 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 in December 1st, 2018.

Copyright and License Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction.....	3
1.1. Authors.....	3
1.2. Requirements Language.....	3
2. Changes from the last version.....	3
3. Conventions used in this document.....	4
3.1. Terminology.....	4
4. Solution Overview.....	5
4.1. Mapping source specific trees onto the segment routing architecture.....	6
4.2. Role of the Routing System.....	6
4.3. MDT Construction Requirements.....	6
4.4. Simplification and Pruning - theory of operation.....	7
5. Elements of Procedure.....	7
5.1. Triggers for Computation.....	7
5.2. FIB Determination.....	8
5.2.1. Information in the IGP.....	8
5.2.2. Computation of individual segments.....	8
5.3. FIB Generation.....	12
5.4. FIB installation.....	12
6. Related work.....	13
6.1. IGP Extensions.....	13
6.2. BGP Extensions.....	13
7. Observations.....	14
8. Acknowledgements.....	14
9. Security Considerations.....	14
10. IANA Considerations.....	14
11. References.....	14
11.1. Normative References.....	14
11.2. Informative References.....	15
12. Authors' Addresses.....	15

## 1. Introduction

This memo describes a solution for multicast for SR-MPLS in which source specific multicast distribution trees (MDTs) are computed from information distributed via an IGP. Computation uses information in the IGP to determine if a given node in the network has a role as a root, a leaf or replication point in a given MDT. Unicast tunnels are employed to interconnect the nodes determined to have a role.

Therefore multicast topological instructions only need be installed in nodes that have one of these three roles to fully instantiate an MDT.

Although this approach might appear to be computationally intensive, a significant amount of computation can be avoided if and when the computing agent determines that the node it is computing for has no role in a given MDT. If there will be no need to install a multicast topological instruction in that node for the given MDT, the computing agent can abandon computation for the MDT and move on to other tasks, such as converging other MDTs. This permits a computed approach to multicast convergence to be computationally tractable.

This approach is proposed as a solution for networks for which an implementation of an alternative data plane, such as BIER, offers technical or economic challenges.

### 1.1. Authors

David Allan, Jeff Tantsura, Ian Duncan

### 1.2. Requirements Language

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

## 2. Changes from the last version

Clarification in motivation.

Editorial corrections and improvements.

Clarification of the description of upstream pruning in section 5.2.2

Alignment of terminology with current segment routing practice.

### 3. Conventions used in this document

#### 3.1. Terminology

Candidate replication point (CRP) - is a node that potentially needs to install a multicast topological instruction to replicate multicast traffic as determined at an intermediate step in multicast segment computation. It will either resolve to having no role or a role as a replication point once multicast has converged.

Candidate role - refers to any potential combination of roles on a given multicast segment as determined at some intermediate step in MDT computation. For example, a node with a candidate role may be a leaf and may also be a candidate replication point.

Computing agent- refers to the agent that will compute the FIB for the MDTs in a given network on behalf of one node (distributed model) or multiple nodes (SR controller(s) in a centralized model).

Downstream - refers to the direction along the shortest path to one or more leaves for a given multicast distribution tree

Multicast convergence - is when all computation and multicast topological instruction installation to ensure the FIB reflects the multicast information in the IGP is complete.

MDT - multicast distribution tree. Is a tree composed of one or more multicast segments.

Multicast segment - is a portion of the multicast tree where only the root and the leaves have been specified, and computation based upon the current state of the IGP database is employed to determine and install the required topological instructions to implement the segment. For SR-MPLS a multicast segment is implemented as a p2mp LSP. A multicast segment is identified by a multicast SID.

Multicast SID - Is the topological instruction that is used to implement a multicast segment. As per a unicast SR-MPLS segment, the rightmost 20 bits of a multicast SID is encoded as a label. It is drawn from the SRGB for the domain.

Pinned path - Is a unique shortest path extending from a leaf upstream towards the root for a given multicast segment. Therefore, it is a component of the multicast segment that it has been determined must be there. It will not necessarily extend from the leaf all the way to the root during intermediate computation steps. A pinned path can result from pruning operations.

Role - refers specifically to a node that is either a root, a leaf, a replication node, or a pinned waypoint for a given MDT.

Unicast convergence - is when all computation and topological instruction installation to ensure the FIB reflects the unicast information in the IGP is complete.

Upstream - refers to the direction along the shortest path to the root of a given MDT.

#### 4. Solution Overview

This memo describes a multicast architecture in which multicast topological instructions are only installed in those nodes that have roles as a root, a leaf, or a replication point for a given multicast segment. The a-priori established mesh of unicast tunnels (using node-SIDs) are used as interconnect between the nodes that have a role in a given multicast SID. Hence on an outgoing interface where the next node in that path of the MDT is not immediately adjacent, the operation will typically be a CONTINUE of the multicast SID and a PUSH of the node-SID.

A loosely specified MDT is composed of a single multicast segment and the routing of the MDT is delegated entirely to computation driven by information in the IGP database.

Explicitly routed MDTs are expressed as a tree of concatenated multicast segments where both the leaves of each segment and the waypoints coupling a given segment to the upstream and/or downstream segment(s) is specified in information flooded in the IGP by the overall root of the MDT. The segments themselves will be computed as per a loosely specified MDT.

A PE acting as an overall root for a given tree is expected to be configured by the operator as to where to source multicast traffic from, be it an attachment circuit, interworking function for client technology or other. Similarly, a leaf for a given tree is expected to be configured by the operator as to the disposition of received multicast traffic.

A computed segment is guaranteed to be loop free in a stable fault free system. A concatenation of segments to construct an MDT will similarly be loop free as any collision of segments can be disambiguated in the data plane via the SIDs.

This architecture significantly reduces the number of multicast topological instructions that needs to be installed in the data plane

to support multicast. This also means that the impact of many failures in the network on multicast traffic distribution will be recovered by unicast local repair or unicast convergence with subsequent multicast convergence acting in the role of network re-optimization (as opposed to restoration).

#### 4.1. Mapping source specific trees onto the segment routing architecture

A computed source specific tree for a given multicast group corresponds to one or more multicast segments in the SR architecture. Each multicast segment is assigned a SID, typically by management configuration of the node that will be the overall root for the source specific tree. The root node then uses the IGP to advertise this information to all nodes in the IGP area/domain.

A multicast group is implemented as the set of source specific trees from all nodes that have registered transmit interest to all nodes that have registered receive interest in a multicast group.

#### 4.2. Role of the Routing System

The role of the IGP is to communicate topology information, multicast capability and associated algorithm, multicast registrations, unicast to node-SID bindings, multicast to SID bindings and waypoints in multi-segment MDTs. No changes to topology or unicast to node-SID binding advertisements are proposed by this memo.

The multicast registrations/bindings will be in the form of source, group, transmit/receive interest and the SID to use for the source specific multicast tree. Registrations are originated by any node that has send or receive interest in a given multicast group. Nodes will use the combination of topology and multicast registrations to determine the nodes that have a role in each source specific tree and the SID information to then derive the required FIB state.

#### 4.3. MDT Construction Requirements

A multicast segment in an MDT is constructed such that between any pair of nodes that have a role in the segment and are connected by a unicast tunnel, there is not another node on the shortest path between the two with a role in that segment. This ensures that copies of a packet forwarded by a multicast segment will traverse a link only once in a stable system and avoids the potential scenario whereby a packet needs to be replicated twice on a given interface.

Note that this can be satisfied by a minimum cost shortest path tree, but this is not an absolute requirement. The pruning rules specified

in this memo will meet this requirement without necessarily producing an absolute minimum cost multicast segment (or incurring the associated computational cost).

#### 4.4. Simplification and Pruning - theory of operation

The role of nodes in a given multicast segment is determined by first producing an inclusive shortest path tree with all possible paths between the root and leaves, and then applying a set of simplification and pruning rules repeatedly until either an acyclic tree is produced, or no further prunes are possible.

For the majority of multicast segments these rules will authoritatively produce a minimum cost tree. For those segments that are not able to be authoritatively resolved, there is a set of pruning operations applied that are not guaranteed to produce a tree that meets the requirements of 3.3, therefore these trees require auditing and potential correction according to a further set of agreed rules. This avoids the necessity and computational overhead of an exhaustive search of the solution space.

A computing agent during computation of a segment may conclude that none of the nodes that it is computing on behalf of will have a role at any point in the computation process and abandon computation of that segment.

#### 5. Elements of Procedure

##### 5.1. Triggers for Computation

MDT computation is triggered by changes to the IGP database. These are in the form of either changes in registered multicast group interest, addition or removal of a multi-segment MDT descriptor, or topology changes.

A change in registered interest for a group will require re-computation of all MDTs that implement the multicast group.

A topology change will require the computation of some number of multicast segments, the actual number will depend on the implementation of tree computation but at a minimum will be all trees for which there is not an optimal shortest path solution as a result of the topology change.

## 5.2. FIB Determination

### 5.2.1. Information in the IGP

Group membership information for a multicast segment is obtained from the IGP. This is true for single segment MDTs as well as multi-segment MDTs. Included in the multi-segment MDT specification is the waypoint nodes in MDT and the upstream and downstream SIDs. The specified node is expected to cross connect the SIDs to join the segments together acting in the role of leaf for the upstream segment and root for the downstream segment.

When a waypoint in an MDT descriptor does not exist in the IGP, the assumption is that the node identified by the waypoint SID has failed. The response of the other nodes in the system in FIB determination is to add the leaves of the downstream segment to the upstream segment.

An example of this would be consider a node "x", and another node "y". At some point in time, "x" advertises a tree that identifies "y" as a waypoint that cross connects upstream SID "a" to downstream SID "b". At some later point node "y" fails. The other nodes in the network will compute segment "a" as if it included all leaves and waypoints in segment "b". All apriori state installed for segment "b" would be removed as the failure of "y" has required "b" to be subsumed by "a".

### 5.2.2. Computation of individual segments

FIB generation for a multicast segment is the result of computation, ultimately as applied to all source specific trees in the network. All computing agents in a given network computing a tree for a given multicast segment must implement a common algorithm for tree generation, as all MUST agree on the solution.

One algorithm is as follows:

All possible shortest paths to the set of leaves for the MDT is determined. Then simplification and pruning rules are repeatedly applied until no further prunes are possible or the MDT is determined to be resolved.

The distinction between simplification rules and pruning rules is the former will not change the candidate role of a node with respect to the MDT under consideration and therefore can be performed in any order, while the latter will affect candidate node roles and must be



performed in an agreed order between all participating computing agents.

The philosophy of the application of these rules could be expressed as "simplify as much as possible, and prune that which cannot be". The rules are:

- 1) Simplification: Eliminate any links and nodes not on a potential shortest path from the root to the leaves for the MDT under consideration.
- 2) Simplification: Replace any nodes that do not have a potential role in the MDT with links.

This will be nodes that are not a leaf, a root or a candidate replication point. For example:

Root-----A-----B

B is a leaf. A is not but is in a potential shortest path from root to B. However, A will have no role in the MDT that serves B as it provides simple transit therefore is replaced with a direct connection between the root and B.

Root-----B

Note that such simplification also needs to avoid the creation of duplicate parallel links. For example:

```

      /-----A-----\
Root                               B
      \-----C-----/

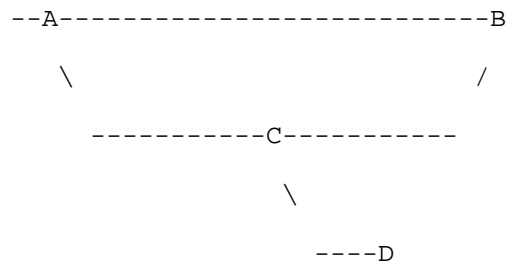
```

Where A and C have no role and the cost root-A-B = cost root-C-B, they can be replaced with a single link from Root to B.

- 3) Simplification: Eliminate of fewer hop paths

When for a given set of leaves, a node has multiple downstream links that converge on a common downstream point, and that set of leaves is only a subset of the leaves reachable on one or more of the links, any link that only serves that subset of leaves can be eliminated.

For example:



Link AB is cost 2, link AC and CB are cost 1 (cost of link CD does not affect the example).

B and D are leaves of a root upstream of A. From A, link AB can reach leaf B. Path AC can reach leaf B and D. In this case path A-B can be eliminated from consideration. The set of leaves reachable via link A-B is a subset of that reachable by A-C, and the paths from A that serves that subset converges at B.

#### 4) Prune: upstream links.

The normal procedure is to determine the best-closest upstream leaf or pinned path and then compare all upstream adjacencies with that metric. Note that the best-closest upstream leaf or pinned path may not be directly connected to the node under consideration. Where there is more than one equally close upstream leaf or pinned path, the highest ranked is selected with the ranking being that a leaf is ranked superior to a pinned path, and the lowest unicast SID is selected when the leaf/pinned path ranking is equal.

Then examine each of the remaining upstream adjacencies:

- a. If the upstream adjacency extends closer to the root than the closest leaf or pinned path, then that adjacency can be pruned.
- b. If the upstream adjacency extends the same distance towards the root as the best-closest adjacency, then it can be eliminated as it has already been ranked lower than the best-closest adjacency. Note that this would include non-leaf and non-pinned path candidate replication points.
- c. If the upstream adjacency is a candidate replication point closer than the best-closest leaf or pinned path, then it is left alone.

When for a given node all possible upstream adjacencies that can be pruned have been identified, each is removed, and any simplifications that can be performed as a result of the prune are performed. This is the equivalent of a localized check for 2 and 3 above and is then performed iteratively in response to changes to the graph as a result of pruning.

The procedure is to implement all simplifications of type 1, 2 and 3 above, then loop on type 4 prunes until such time as the MDT is fully resolved from the point of view of the node under consideration, or no further prunes are possible. Step 4 is required to be performed in a specific order if there is more than one computing agent generating topological instructions for a given multicast segment. This memo suggests that the nodes are processed according to a ranking of nodes from closest to the root to the farthest, and from lowest unicast SID to the highest within a given distance from the root.

At the end of pruning and simplification, either:

- 1) The node whom the computing agent is computing for has no role in the multicast segment under consideration
- 2) A unique shortest path to the root has been determined for all leaves in the multicast segment that are downstream of the node under consideration (also termed as a pinned path from the root to every leaf).
- 3) A unique shortest path to the root has not been determined for all leaves downstream of the node under consideration in the multicast segment.

If 1 or 2 then the multicast segment is considered to be resolved, and for 2, the computation can progress directly to the topological instruction generation step for that segment.

If 3 (not all downstream leaves have a unique shortest path), additional pruning steps are applied. These steps are NOT guaranteed to produce a lowest cost tree, and therefore require an additional audit and possible modification to ensure when forwarding a maximum of one copy of a packet will traverse an interface.

For segments not authoritatively resolved by the above rules, a prune that will not authoritatively result in a minimum cost tree is applied. For the purpose of interoperability, the following rule is applied: A computing agent will select the closest node to the root with a candidate role that does not have a unique shortest path to the root. Where more than one such node exists, the one with the

lowest node-SID is selected. For that node, the best upstream link is selected and all other upstream links pruned. The best upstream link is defined as the link with the closest node with a candidate role that potentially serves the highest number of leaves. Where there is a tie, once again the node with the lowest unicast SID is selected.

Once the links have been pruned, rules 2 through 4 are repeatedly applied until either the tree is fully resolved, or again no further prunes are possible, in which case the next closest remaining unresolved node has the same prune applied.

For all segments not resolved by the initial prune rules, they are audited to ensure all nodes that have a role in the tree do not have a node with a role between them and their upstream node on the tree. If they do, the old upstream adjacency is removed, and the superior one added.

### 5.3. FIB Generation

The topology components that remain at the end of the simplification and pruning operations will reflect all nodes that have a role in a given multicast segment plus the necessary tunnels (as all intervening multi-path scenarios will have been simplified away). From this the topological instructions to put in the FIB can be generated:

All nodes that have a role in a given multicast segment and have nodes upstream in the segment will need to accept the multicast SID for the MDT from at minimum, all upstream interfaces.

All nodes that have a role in a given segment and have nodes immediately downstream in the segment will need to replicate packets simply labelled with the multicast SID onto those interfaces.

All nodes that have a role in a given segment and have nodes reachable via a tunnel downstream set the FIB to push the tunnel unicast SID for the downstream node onto any replicated copies of a received packet, and identify the set of interfaces on the shortest path for the tunnel SID.

### 5.4. FIB installation

FIB installation needs to acknowledge two aspects of the hybrid tunnel and role model of multicast tree construction. The first is that because of the sparse state model simple tree adds, moves, and changes may require the installation of topological instructions where they did not previously exist, and such changes may impact

existing services. The second is that it is possible to retain the knowledge to prioritize computation of those trees impacted the failure of a node with a role.

To address this in the distributed model, there are three stages of topological instruction installation for multicast convergence:

1) Immediate:

- a. Installation of topological instructions for multicast segments impacted by the failure of a node in the network, and installation of topological instructions for segments in nodes that have not previously had a role in the given segment.
- b. Installation of topological instructions for waypoints in multi-segment MDTs.

2) After T1: Update topological instructions for nodes that both had and have a role in a given multicast segment.

3) After T2: Removal of topological instructions for nodes that transition from having a role to not having a role for a given multicast segment.

T1 and T2 are network wide configurable values.

When an SR-Controller is used, it is only necessary to properly sequence the installation of state. This also suggests that when there is more than one SR-Controller, the division of responsibility should be on the basis of MDT ownership.

## 6. Related work

### 6.1. IGP Extensions

The required IGP changes are documented in [MCAST-ISIS] and [MCAST-OPSF].

### 6.2. BGP Extensions

This memo will require the specification of a new PMSI Tunnel Attribute (SPRING P2MP tunnel, tentatively 0x0c) to order to integrate into the multicast framework documented in RFC 6514

## 7. Observations

This technique is not confined to SR-MPLS:

- with the provision of a global label space (to be employed as per a multicast SID), an MPLS-LDP network would also provide the requisite mesh of unicast tunnels and be capable of implementing this approach to multicast.
- It is also possible to envision an SRv6 implementation but would require the ability to rewrite the SRH at each hop.

This memo focuses on an implementation based upon nodes that are IGP speakers and converge independently so is written in a form that assumes a node, computing agent and IGP speaker are one in the same. It should be observed that the relative frugality of data plane state would suggest that separation of computation from nodes in the data plane combined with management or "software defined networking" based population of the multicast FIB entries may also be useful modes of network operation.

## 8. Acknowledgements

Thanks to Uma Chunduri for his detailed review and suggestions.

## 9. Security Considerations

For a future version of this document.

## 10. IANA Considerations

This document requires the allocation of a PMSI tunnel type to identify a SPRING P2MP tunnel type from the P-Multicast Service Interface Tunnel (PMSI Tunnel) Tunnel Types registry.

## 11. References

### 11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 11.2. Informative References

- [MCAST-ISIS] Allan et.al., "IS-IS extensions for Computed Multicast applied to MPLS based Segment Routing", IETF work in progress, draft-allan-isis-spring-multicast-00, July 2016
- [MCAST-OSPF] Allan et.al., "OSPF extensions for Computed Multicast applied to MPLS based Segment Routing", IETF work in progress, draft-allan-ospf-spring-multicast-00, July 2016
- [SR-ARCH] Filsfils et.al., "Segment Routing Architecture", IETF work in progress, draft-ietf-spring-segment-routing-15, January 2018
- [RFC6514] Aggarwal et.al., "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", IETF RFC 6514, February 2012
- [RFC7385] Andersson & Swallow "IANA Registry for P-Multicast Service Interface (PMSI) Tunnel Type Code Points", IETF RFC 7385, October 2014

## 12. Authors' Addresses

Dave Allan (editor)  
Ericsson  
2455 Augustine Drive  
Santa Clara 95054  
USA  
Email: david.i.allan@ericsson.com

Jeff Tantsura  
Email: jefftant.ietf@gmail.com

Ian Duncan  
Ciena  
iduncan@ciena.com  
5050 Innovation Drive  
Kanata, ON K2K 0J2

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: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 queue-limit {
  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 = 'comp-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 mdr-queuing-policy {
    when "../.../sub-type = 'mdrr-queuing-policy'" {
      description
        "MDRR queue handling properties for the traffic " +
        "classified into current queue";
    }
    leaf mdr-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](mailto:mjethanandani@gmail.com)

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

Email: [nstrahle@juniper.net](mailto:nstrahle@juniper.net)

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

Email: [exa@juniper.net](mailto:exa@juniper.net)

Ing-Wher Chen  
Jabil

Email: [ing-wher\\_chen@jabil.com](mailto:ing-wher_chen@jabil.com)



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: September 7, 2022

A. Choudhary  
Cisco Systems  
I. Chen  
The MITRE Corporation  
March 06, 2022

YANG Model for QoS Operational Parameters  
draft-asechoud-rtgwg-qos-oper-model-10

Abstract

This document describes a YANG model for Quality of Service (QoS) 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 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 (<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 . . . . .	2
2. Terminology . . . . .	2
3. QoS Operational Model Design . . . . .	3
4. Modules Tree Structure . . . . .	4
5. Modules . . . . .	6
5.1. ietf-qos-oper . . . . .	6
6. Security Considerations . . . . .	14
7. Acknowledgement . . . . .	14
8. References . . . . .	15
8.1. Normative References . . . . .	15
8.2. Informative References . . . . .	16
Authors' Addresses . . . . .	16

## 1. Introduction

This document defines a base YANG [RFC6020] [RFC7950] data module for Quality of Service (QoS) operational parameters. Remote Procedure Calls (RPC) or notification definition is currently not part of this document and will be added later if necessary. QoS configuration modules are defined by [I-D.ietf-rtgwg-qos-model].

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].

## 1.1. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [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 Operational Model Design

QoS operational model include QoS policy applied to an interface in each direction of traffic. For each QoS policy applied to an interface the model further includes counters for associated Classifiers, Meters and Queues in a particular direction. To modularize and for reusability, grouping have been defined for various counters of classifier, Meters and Queues. The target is assumed to be interface but the groupings can be used for any other target type where QoS policy is applied.

[I-D.ietf-rtgwg-qos-model] defines various building blocks for applying a QoS Policy on a target. It includes QoS Policy configuration, which is a container of various classifiers and corresponding actions which are configured for traffic conditioning. This drafts defines the various counters for these building blocks. ietf-qos-oper module defined in this draft augments ietf-interfaces [RFC8343] module.

Classifier statistics contains counters for packets and bytes matched to the traffic in a direction and also average rate at which traffic is hitting a classifier. Classification criterion may be based on IP, MPLS or Ethernet. Counters defined in this draft are agnostic to underlying data plane technology.

Statistics of 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. Metering statistics includes counters corresponding to various rates configured. A metering container is referred by a metering identifier. This identifier could be a classifier name if the metering configuration is inline with classifier or it could be metering template name if the metering is configured as separate entity and associated with the classifier.

Queuing statistics includes counters corresponding to various queues associated with the policy. A queuing container is referred by queuing identifier. This identifier could be a classifier name if the queuing configuration is inline with classifier and hence there is one-to-one mapping between a classifier and a queue or it could be a separate queue identifier if one or more than one classifiers are associated with a queue.

#### 4. Modules Tree Structure

This document defines counters for classifiers, meters and queues.

Classifier statistics consists of list of classifier entries identified by a classifier entry name. Classifier counters include matched packets, bytes and average rate of traffic matching a particular classifier.

Metering statistics consists of meters identified by an identifier. Metering counters include conform, exceed, violate and drop packets and bytes.

Queuing counters include instantaneous, peak, average queue length, as well as output conform, exceed, tail drop packets and bytes.

Named statistics is defined as statistics which are tagged by a name. This could be aggregated or non-aggregated. Aggregated named statistics is defined as counters which are aggregated across classifier entries in a policy applied to an interface in a particular direction. Non-aggregated named statistics are counters of classifier, metering or queuing which have the same tag name but maintained separately.

```

module: ietf-qos-oper
  augment /if:interfaces/if:interface:
    +--ro qos-interface-statistics
      +--ro stats-per-direction* []
        +--ro direction?          identityref
        +--ro policy-name?        string
        +--ro classifier-statistics* []
          +--ro classifier-entry-name? string
          +--ro classified-pkts?      uint64
          +--ro classified-bytes?     uint64
          +--ro classified-rate?      uint64
        +--ro named-statistics* []
          +--ro stats-name?         string
          +--ro aggregated
            +--ro pkts?      uint64
            +--ro bytes?    uint64
            +--ro rate?     uint64
          +--ro non-aggregated
            +--ro classifier-statistics* []
              +--ro classifier-entry-name? string
              +--ro classified-pkts?      uint64
              +--ro classified-bytes?     uint64
              +--ro classified-rate?      uint64

```

```

+--ro metering-statistics* []
|   +--ro meter-id?          string
|   +--ro conform-pkts?      uint64
|   +--ro conform-bytes?     uint64
|   +--ro conform-rate?      uint64
|   +--ro exceed-pkts?       uint64
|   +--ro exceed-bytes?      uint64
|   +--ro exceed-rate?       uint64
|   +--ro violate-pkts?      uint64
|   +--ro violate-bytes?     uint64
|   +--ro violate-rate?      uint64
|   +--ro meter-drop-pkts?   uint64
|   +--ro meter-drop-bytes?  uint64
+--ro queueing-statistics* []
|   +--ro queue-id?          string
|   +--ro output-conform-pkts? uint64
|   +--ro output-conform-bytes? uint64
|   +--ro output-exceed-pkts?  uint64
|   +--ro output-exceed-bytes? uint64
|   +--ro queue-current-size-bytes? uint64
|   +--ro queue-average-size-bytes? uint64
|   +--ro queue-peak-size-bytes?  uint64
|   +--ro tailed-drop-pkts?      uint64
|   +--ro tailed-drop-bytes?     uint64
|   +--ro red-drop-pkts?         uint64
|   +--ro red-drop-bytes?        uint64
|   +--ro red-ecn-marked-pkts?   uint64
|   +--ro red-ecn-marked-bytes?  uint64
|   +--ro wred-stats* []
|   |   +--ro profile-id?      uint64
|   |   +--ro red-drop-pkts?   uint64
|   |   +--ro red-drop-bytes?  uint64
|   |   +--ro red-ecn-marked-pkts? uint64
|   |   +--ro red-ecn-marked-bytes? uint64
+--ro metering-statistics* []
|   +--ro meter-id?          string
|   +--ro conform-pkts?      uint64
|   +--ro conform-bytes?     uint64
|   +--ro conform-rate?      uint64
|   +--ro exceed-pkts?       uint64
|   +--ro exceed-bytes?      uint64
|   +--ro exceed-rate?       uint64
|   +--ro violate-pkts?      uint64
|   +--ro violate-bytes?     uint64
|   +--ro violate-rate?      uint64
|   +--ro meter-drop-pkts?   uint64
|   +--ro meter-drop-bytes?  uint64
+--ro queueing-statistics* []

```

```

+--ro queue-id?                string
+--ro output-conform-pkts?      uint64
+--ro output-conform-bytes?     uint64
+--ro output-exceed-pkts?      uint64
+--ro output-exceed-bytes?     uint64
+--ro queue-current-size-bytes? uint64
+--ro queue-average-size-bytes? uint64
+--ro queue-peak-size-bytes?   uint64
+--ro tailed-drop-pkts?        uint64
+--ro tailed-drop-bytes?       uint64
+--ro red-drop-pkts?           uint64
+--ro red-drop-bytes?          uint64
+--ro red-ecn-marked-pkts?     uint64
+--ro red-ecn-marked-bytes?    uint64
+--ro wred-stats* []
    +--ro profile-id?          uint64
    +--ro red-drop-pkts?       uint64
    +--ro red-drop-bytes?      uint64
    +--ro red-ecn-marked-pkts? uint64
    +--ro red-ecn-marked-bytes? uint64

```

## 5. Modules

### 5.1. ietf-qos-oper

```
<CODE BEGINS>file "ietf-qos-oper.yang"
```

```

module ietf-qos-oper {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-oper";
  prefix oper;
  import ietf-interfaces {
    prefix if;
    reference
      "RFC8343: A YANG Data Model for Interface Management";
  }
  organization "IETF RTG (Routing Area) Working Group";
  contact
    "WG Web:  <http://tools.ietf.org/wg/rtgwg/>
    WG List:  <mailto:rtgwg@ietf.org>
    Editor:    Aseem Choudhary
               <mailto:asechoud@cisco.com>";
  description
    "This module contains a collection of YANG definitions for
    qos operational specification.
    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 2022-03-06 {
  description
    "Latest revision for qos operational statistics";
  reference
    "RFC XXXX: YANG Model for QOS Operational Parameters";
}
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";
}
grouping classifier-entry-stats {
  description
    "
      This group defines the classifier filter counters of
      each classifier entry
    ";
  leaf classified-pkts {
    type uint64;
    description
      " Number of total packets which filtered
        to a classifier-entry";
  }
  leaf classified-bytes {
    type uint64;
    description
      " Number of total bytes which filtered
        to a classifier-entry";
  }
  leaf classified-rate {
    type uint64;
```

```
    units "bits-per-second";
    description
      " Rate of average data flow through a
        classifier-entry";
  }
}
grouping named-stats {
  description
    "QoS matching statistics associated with a stats-name";
  leaf pkts {
    type uint64;
    description
      " Number of total matched packets associated
        to a statistics name";
  }
  leaf bytes {
    type uint64;
    description
      " Number of total matched bytes associated
        to a statistics name";
  }
  leaf rate {
    type uint64;
    units "bits-per-second";
    description
      " Rate of average matched data which is associated
        to a statistics name";
  }
}
grouping queue-stats {
  description
    "Queuing Counters";
  leaf output-conform-pkts {
    type uint64;
    description
      "Number of packets transmitted from queue ";
  }
  leaf output-conform-bytes {
    type uint64;
    description
      "Number of bytes transmitted from queue ";
  }
  leaf output-exceed-pkts {
    type uint64;
    description
      "Number of packets transmitted from queue ";
  }
  leaf output-exceed-bytes {
```



```
    type uint64;
    description
      "Number of bytes transmitted from queue ";
  }
  leaf queue-current-size-bytes {
    type uint64;
    description
      "Number of bytes currently buffered ";
  }
  leaf queue-average-size-bytes {
    type uint64;
    description
      "Average queue size in number of bytes";
  }
  leaf queue-peak-size-bytes {
    type uint64;
    description
      "Peak buffer queue size in bytes ";
  }
  leaf tailed-drop-pkts {
    type uint64;
    description
      "Total number of packets tail-dropped ";
  }
  leaf tailed-drop-bytes {
    type uint64;
    description
      "Total number of bytes tail-dropped ";
  }
  leaf red-drop-pkts {
    type uint64;
    description
      "Total number of packets dropped through RED mechanism";
  }
  leaf red-drop-bytes {
    type uint64;
    description
      "Total number of bytes dropped through RED mechanism";
  }
  leaf red-ecn-marked-pkts {
    type uint64;
    description
      "Total number of packets ECN marked through RED mechanism";
  }
  leaf red-ecn-marked-bytes {
    type uint64;
    description
      "Total number of bytes ECN marked through RED mechanism";
  }
```

```
    }  
  }  
  grouping meter-stats {  
    description  
      "Metering counters";  
    leaf conform-pkts {  
      type uint64;  
      description  
        "Number of conform packets";  
    }  
    leaf conform-bytes {  
      type uint64;  
      description  
        "Bytes of conform packets";  
    }  
    leaf conform-rate {  
      type uint64;  
      units "bits-per-second";  
      description  
        "Traffic Rate measured as conforming";  
    }  
    leaf exceed-pkts {  
      type uint64;  
      description  
        "Number of packets counted as exceeding";  
    }  
    leaf exceed-bytes {  
      type uint64;  
      description  
        "Bytes of packets counted as exceeding";  
    }  
    leaf exceed-rate {  
      type uint64;  
      units "bits-per-second";  
      description  
        "Traffic Rate measured as exceeding";  
    }  
    leaf violate-pkts {  
      type uint64;  
      description  
        "Number of packets counted as violating";  
    }  
    leaf violate-bytes {  
      type uint64;  
      description  
        "Bytes of packets counted as violating";  
    }  
    leaf violate-rate {
```

```
        type uint64;
        units "bits-per-second";
        description
            "Traffic Rate measured as violating";
    }
    leaf meter-drop-pkts {
        type uint64;
        description
            "Number of packets dropped by meter";
    }
    leaf meter-drop-bytes {
        type uint64;
        description
            "Bytes of packets dropped by meter";
    }
}
grouping classifier-entry-statistics {
    description
        "Statistics for a classifier entry";
    leaf classifier-entry-name {
        type string;
        description
            "Classifier Entry Name";
    }
    uses classifier-entry-stats;
}

grouping queuing-stats {
    description
        "Statistics for a queue";
    leaf queue-id {
        type string;
        description
            "Queue Identifier";
    }
    uses queue-stats;
    list wred-stats {
        config false;
        description
            "Qos RED statistics for each color of traffic";
        leaf profile-id {
            type uint64;
            description
                "profile-id for each color of traffic";
        }
        leaf red-drop-pkts {
            type uint64;
            description
```

```
        "Total number of packets dropped through RED mechanism";
    }
    leaf red-drop-bytes {
        type uint64;
        description
            "Total number of bytes dropped through RED mechanism";
    }
    leaf red-ecn-marked-pkts {
        type uint64;
        description
            "Total number of packets ECN marked through RED mechanism";
    }
    leaf red-ecn-marked-bytes {
        type uint64;
        description
            "Total number of bytes ECN marked through RED mechanism";
    }
}

grouping metering-stats {
    description
        "Statistics for a meter";
    leaf meter-id {
        type string;
        description
            "Meter Identifier";
    }
    uses meter-stats;
}

augment "/if:interfaces/if:interface" {
    description
        "Augments Qos Target Entry to Interface module";

    container qos-interface-statistics {
        config false;
        description
            "Qos Interface statistics";

        list stats-per-direction {
            description
                "Qos Interface statistics for ingress or egress direction";

            leaf direction {
                type identityref {
                    base direction;
                }
            }
        }
    }
}
```

```
        description
            "Direction of the traffic flow either inbound
             or outbound";
    }
    leaf policy-name {
        type string;
        description
            "Policy entry name for single level policy as well as
             for Hierarchical policies. For Hierarchical policies,
             this represent relative path as well as the last level
             policy name.";
    }

    list classifier-statistics {
        description
            "Classifier Statistics for each Classifier Entry in a
             Policy applied in a particular direction";
        reference
            "RFC3289: Section 6";
        uses classifier-entry-statistics;
    }

    list named-statistics {
        config false;
        description
            "Statistics for a statistics-name";
        leaf stats-name {
            type string;
            description
                "Statistics name";
        }
        container aggregated {
            description
                "Matched aggregated statistics for a statistics-name";
            uses named-stats;
        }
        container non-aggregated {
            description
                "Statistics for non-aggregated statistics-name";
            list classifier-statistics {
                description
                    "Classifier Statistics for each Classifier Entry in a
                     Policy applied in a particular direction";
                uses classifier-entry-statistics;
            }
            list metering-statistics {
                config false;
                description
                    "Statistics for each Meter associated with
```

```
        the Policy";
    reference
        "RFC2697: A Single Rate Three Color Marker
        RFC2698: A Two Rate Three Color Marker";
    uses metering-stats;
}
list queueing-statistics {
    config false;
    description
        "Statistics for each Queue associated with
        the Policy";
    uses queueing-stats;
}
}
list metering-statistics {
    config false;
    description
        "Statistics for each Meter associated with the Policy";
    reference
        "RFC2697: A Single Rate Three Color Marker
        RFC2698: A Two Rate Three Color Marker";
    uses metering-stats;
}
list queueing-statistics {
    config false;
    description
        "Statistics for each Queue associated with the Policy";
    uses queueing-stats;
}
}
}
}
```

<CODE ENDS>

## 6. Security Considerations

## 7. Acknowledgement

MITRE has approved this document for Public Release, Distribution Unlimited, with Public Release Case Number 20-0518. The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.

## 8. References

### 8.1. Normative References

- [I-D.ietf-rtgwg-qos-model]  
Choudhary, A., Jethanandani, M., Aries, E., and I. Chen,  
"YANG Models for Quality of Service (QoS)", draft-ietf-  
rtgwg-qos-model-07 (work in progress), March 2022.
- [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>>.
- [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>>.
- [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>>.

## 8.2. Informative References

[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>>.

## Authors' Addresses

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

Email: [asechoud@cisco.com](mailto:asechoud@cisco.com)

Ing-Wher Chen  
The MITRE Corporation

Email: [ingwherchen@mitre.org](mailto:ingwherchen@mitre.org)



INTERNET-DRAFT  
Intended status: Informational

S. Hu  
F. Qin  
Z. Li  
China Mobile  
T. Chua  
Singapore Telecommunications Ltd  
V. Lopez  
Telefonica  
D. Eastlake  
Z. Wang  
J. Song  
Huawei  
March 11, 2019

Expires: September 10, 2019

Architecture for Control Plane and User Plane Separated BNG  
draft-cuspd-t-rtgwg-cu-separation-bng-architecture-04.txt

## Abstract

This document defines an architecture for Broadband Network Gateway (BNG) devices with control plane (CP) and user plane (UP) separation. A BNG-CP is a user control management component while a BNG-UP takes responsibility as the network edge and user policy implementation component. Both BNG-CP and BNG-UP are core components for fixed broadband services and are deployed separately at different network layers.

## 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.....	3
1.1 Motivation.....	3
2. Terminology.....	4
3. CU Separated BNG Architecture.....	5
3.1 Internal Interfaces Between the CP and UP.....	7
4. Usage of the CU Separation BNG.....	8
5. Security Considerations.....	10
6. IANA Considerations.....	10
Normative References.....	11
Informative References.....	11
Authors' Addresses.....	12

## 1. Introduction

A Broadband Network Gateway (BNG) device is defined as an Ethernet-centric IP edge router, and the aggregation point for user traffic. It performs Ethernet aggregation and packet forwarding via IP/MPLS, and supports user management, access protocols termination, QoS, policy management, etc.

This document describes an architecture for BNG devices with control plane (CP) and user plane (UP) separation. A BNG-CP is a user control management component while a BNG-UP takes responsibility as the network edge and user policy implementation components. Both BNG-CP and BNG-UP are core components for fixed broadband services and are deployed separately at different network layers in the network.

### 1.1 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 CU separation conception is defined in [TR-384]. The main idea of Control-Plane and User-Plane 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). Note that the CU separation concept has also been introduced in the 3GPP 5G architecture [3GPP.23.501].

## 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.

The following acronyms are used as specified below:

AAA: Authentication Authorization Accounting.

BNG: Broadband Network Gateway. A broadband remote access server (BRAS (Broadband Access Server), B-RAS or BBRAS) that 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).

CP: Control Plane. The CP is a user control management component which manages the UP's resources such as the user entry and user's QoS policy

DHCP: Dynamic Host Configuration Protocol.

EMS: Element Management System.

IPoE: IP over Ethernet.

MANO: Management and Orchestration.

NFV: Network Function Virtualization.

NFVI: NFV Infrastructure.

PPPoE: Point-to-Point Protocol over Ethernet.

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.

### 3. CU Separated BNG Architecture

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. In a cloud-based BNG, we find that tearing these two functions apart can make a difference. The user management function can be centralized and deployed as a concentrated module or device, called the BNG-CP (Control Plane). The other functions, such as the router function and forwarding engine, can be deployed in the form of the BNG User Plane. Thus, the Cloud-based BNG architecture is made up of control plane and user plane.

The following figure describes the architecture of CU separated BNG:

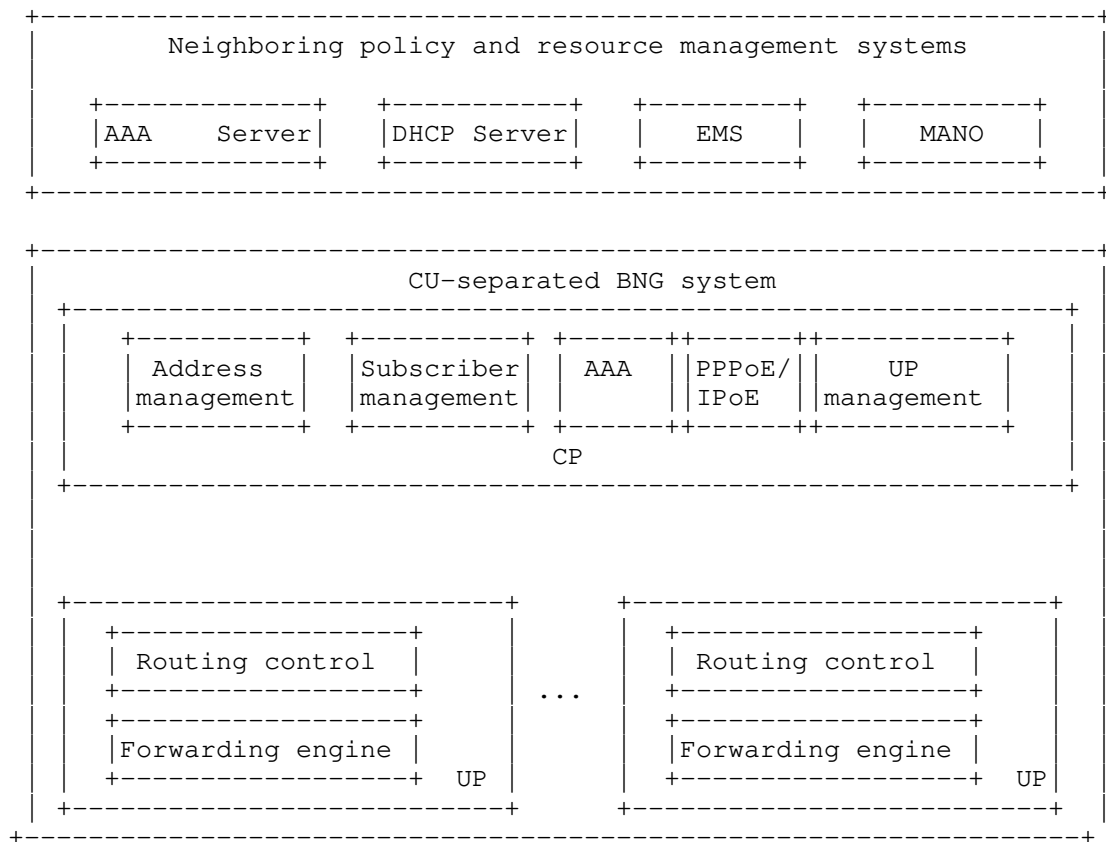


Figure 1. Architecture of CU Separated BNG

As in Figure 1, the BNG Control Plane could be virtualized and centralized, which provides significant 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 should support:

- (1) Address management: unified address pool 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) PPPoE/IPoE: process user dialup packets via PPPoE/IPoE.
- (5) UP management: management of UP interface status, and the setup, deletion, and maintenance of channels between CP and UP.

The User Plane should support:

- (1) Control plane functions including routing, multicast, and MPLS.
- (2) Forwarding plane functions including traffic forwarding, QoS and traffic statistics collection.

### 3.1 Internal Interfaces Between the CP and UP

To support the communication between the Control Plane and User Plane, several interfaces are involved. Figure 2 illustrates the internal interfaces of CU Separated BNG.

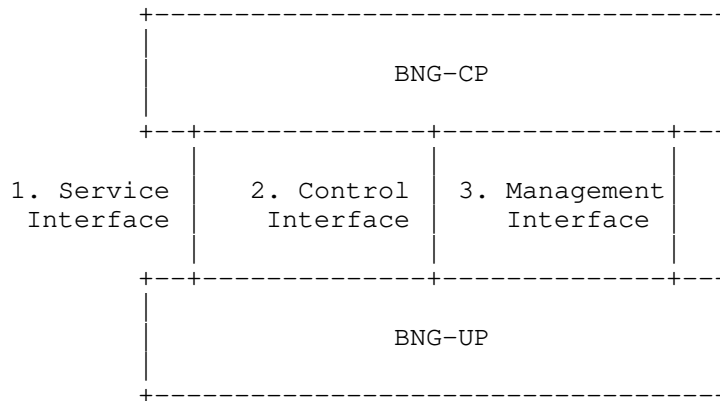


Figure 2. Internal Interfaces Between the CP and UP of the BNG

**Service Interface:** The CP and UP use this interface to establish tunnels with each other and transmit PPPoE and IPoE packets over those tunnels. VXLAN is commonly used for such tunnels as discussed in [hu-nvo3-vxlan-gpe-extension-for-vbng].

**Control Interface:** The CP uses this interface to deliver service entries, and the UP uses this interface to report service events to the CP. The requirements of this interface are introduced in [cuspdrt-gtwg-cusp-requirements], and the carrying protocol is presented in [cuspdrt-gtwg-cu-separation-bng-protocol] which specifies the Simple Control and User Plane Separation protocol (S-CUSP). The information model of this interface is presented in [cuspdrt-gtwg-cu-separation-infor-model].

**Management Interface:** The CP uses this interface to deliver configurations to the UP. This interface uses NETCONF [cuspdrt-gtwg-cu-separation-yang-model].

#### 4. Usage of the CU Separation BNG

In the CU separated BNG scenario, there are several processes when a home user accesses the Internet:

- (1) User dialup packets via PPPoE or IPoE from the BNG-UP are sent to the BNG-CP through the BNG-UP's Service Interface.
- (2) BNG-CP processes the dialup packet. Confirming the user's authorization with the outside neighboring systems in the management network, the BNG-CP makes the decision to permit or deny the user access.
- (3) After that, the BNG-CP tells the UP to do perform authorized forwarding actions with appropriate QoS policies.
- (4) If the user is certificated and permitted, the UP forwards the traffic into the Internet with appropriate QoS policies such as limited bandwidth, etc. Otherwise, the user is denied to access the Internet.

In actual deployments, a CU separated BNG device is composed of a CP and one or more UPs. The CP is usually centrally deployed and takes responsibility as a user control management component managing UP's resources such as the user entry and forwarding policy. The UPs are distributed and act as a network edge and user policy implementation component.

In order to fulfill a service, neighboring policy and resource management systems are deployed outside the BNG. In the neighboring systems, different service systems such as RADIUS/DIAMETER server, DHCP server and EMS are included. If a BNG-CP is virtualized as a NFV, the NFVI management system MANO is also included here. A BNG-CP has connections with the outside neighboring systems to transmit management traffic.

The deployment scenario is shown in the following figure:



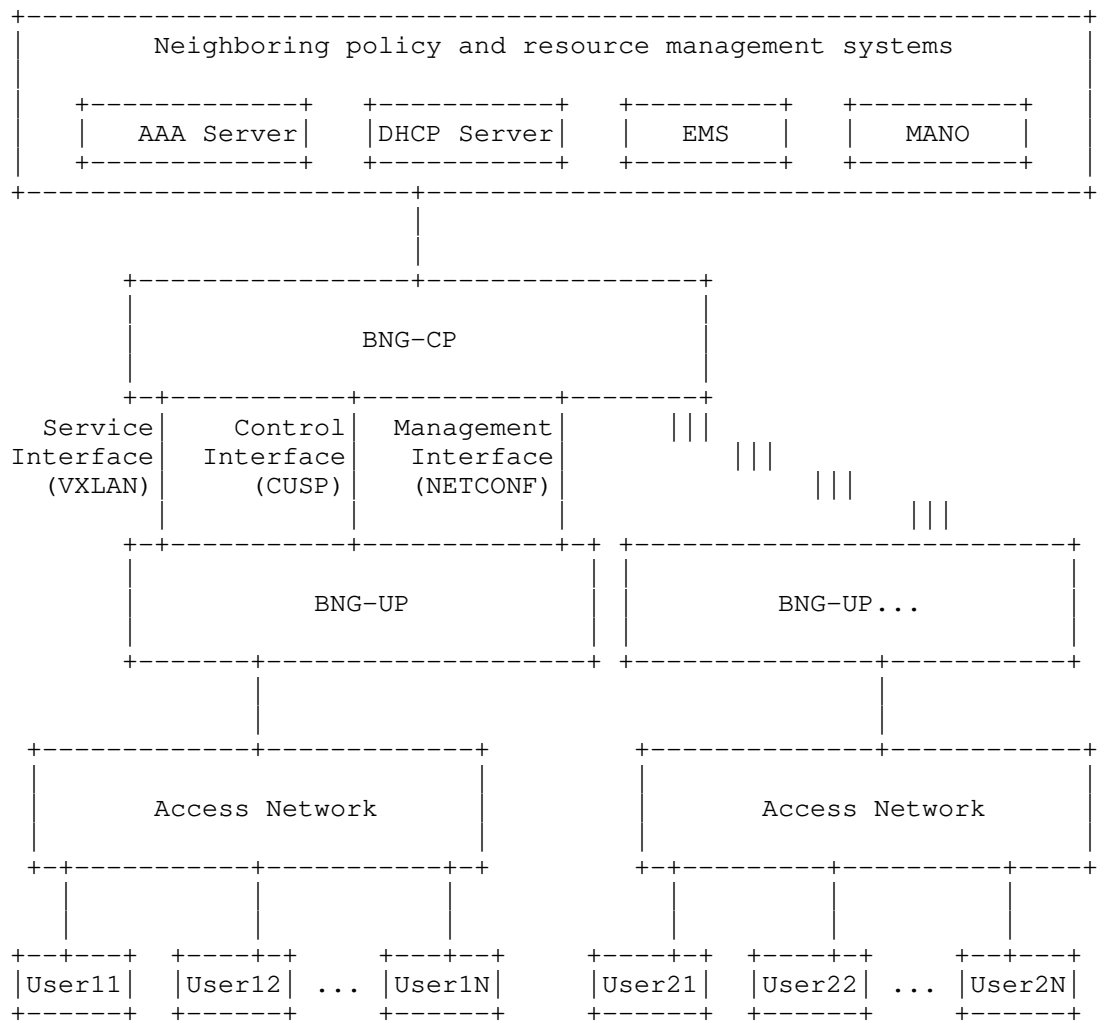


Figure 3. Deployment Example

## 5. Security Considerations

The Service, Control, and Management Interfaces between the CP and UP might be across the general Internet or other hostile environment. Thus, appropriate protections **MUST** be implemented to provide integrity, authenticity, and secrecy of traffic over those interfaces. For example, the implementation of IPSEC, DTLS, or TLS as appropriate. However, 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 might be in a more benign environment.

## 6. IANA Considerations

This document requires no IANA actions.

## 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>>.
- [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

- [\_3GPP.23.501] "System Architecture for the 5G System", 3GPP GPP TS 23.501 15.0.0, 2018.
- [cuspdtd-rtgwg-cu-separation-bng-deployment] Gu, R., "Deployment Model of Control Plane and User Plane Separated BNG", draft-cuspdtd-rtgwg-cu-separation-bng-deployment, work in progress, 2018.
- [cuspdtd-rtgwg-cu-separation-bng-protocol] Wang, Z., "Control-Plane and User-Plane separation BNG control channel Protocol", draft-cuspdtd-rtgwg-cu-separation-bng-protocol, work in progress, 2018.
- [cuspdtd-rtgwg-cu-separation-infor-model] Wang, Z., "Information Model of Control-Plane and User-Plane separation BNG", draft-cuspdtd-rtgwg-cu-separation-infor-model, work in progress, 2018.
- [cuspdtd-rtgwg-cusp-requirements] Hu, S., "Requirements for Control Plane and User Plane Separated BNG Protocol", draft-cuspdtd-rtgwg-cusp-requirements, work in progress, 2018.
- [cuspdtd-rtgwg-cu-separation-yang-model] Hu, F., "YANG Data Model for Configuration Interface of Control-Plane and User-Plane separation BNG", draft-cuspdtd-rtgwg-cu-separation-yang-model, work in progress, 2018.
- [hu-nov3-vxlan-gpe-extension-for-vbng] Huang, L., "VXLAN GPE Extension for Packets Exchange Between Control and User Plane of vBNG", draft-hu-nvo3-vxlan-gpe-extension-for-vbrg, work in progress, 2017.
- [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

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

Victor Lopez  
Telefonica  
Spain

Email: victor.lopezalvarez@telefonica.com

Donald Eastlake, 3rd  
Huawei Technologies  
1424 Pro Shop Court  
Davenport, FL 33896  
USA

Phone: +1-508-333-2270  
Email: d3e3e3@gmail.com

Zitao Wang  
Huawei Technologies  
101 Software Avenue, Yuhua District  
Nanjing, Jiangsu 210012  
China

Email: wangzitao@huawei.com

Jun Song  
Huawei Technologies  
101 Software Avenue, Yuhua District  
Nanjing, Jiangsu 210012  
China

Email: song.jun@huawei.com

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.



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:

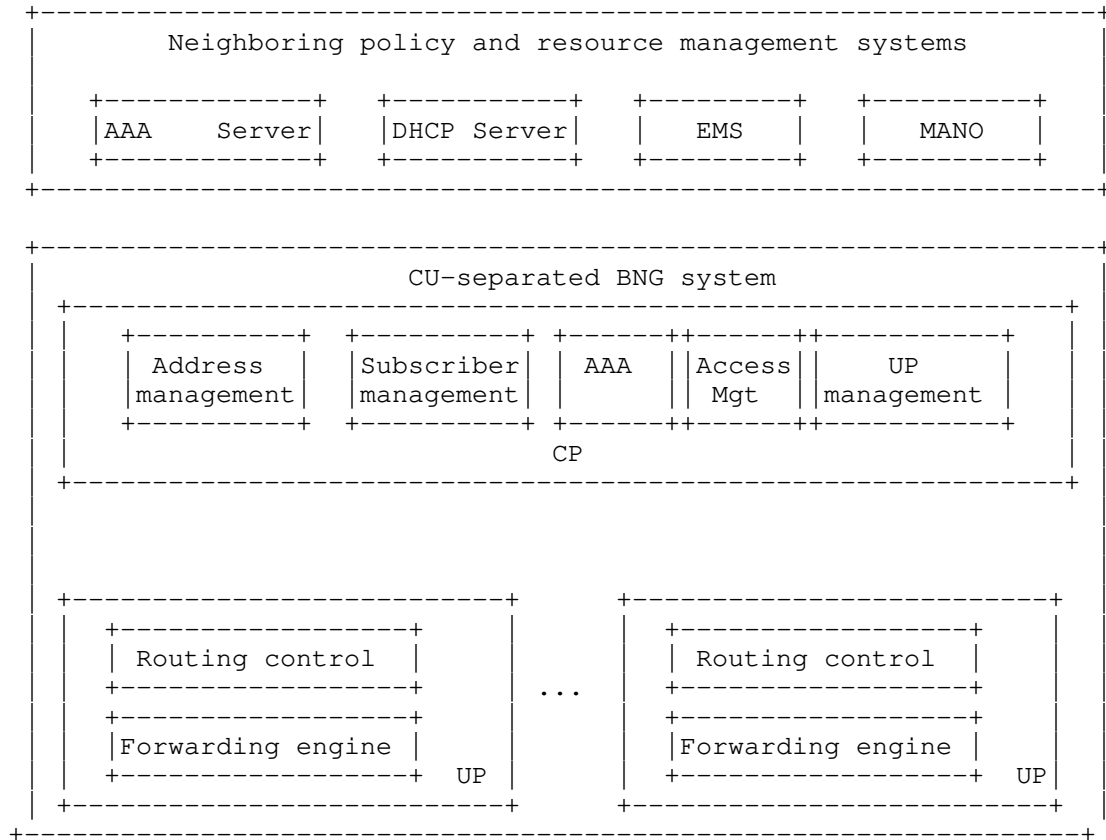


Figure 1: Architecture of CU Separated BNG

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.

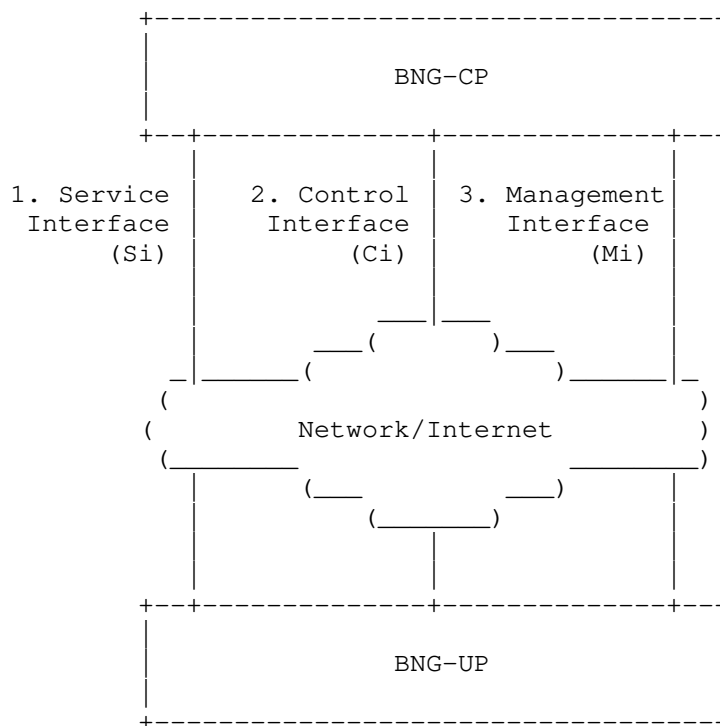


Figure 2: Interfaces Between the CP and UP of the BNG

### 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

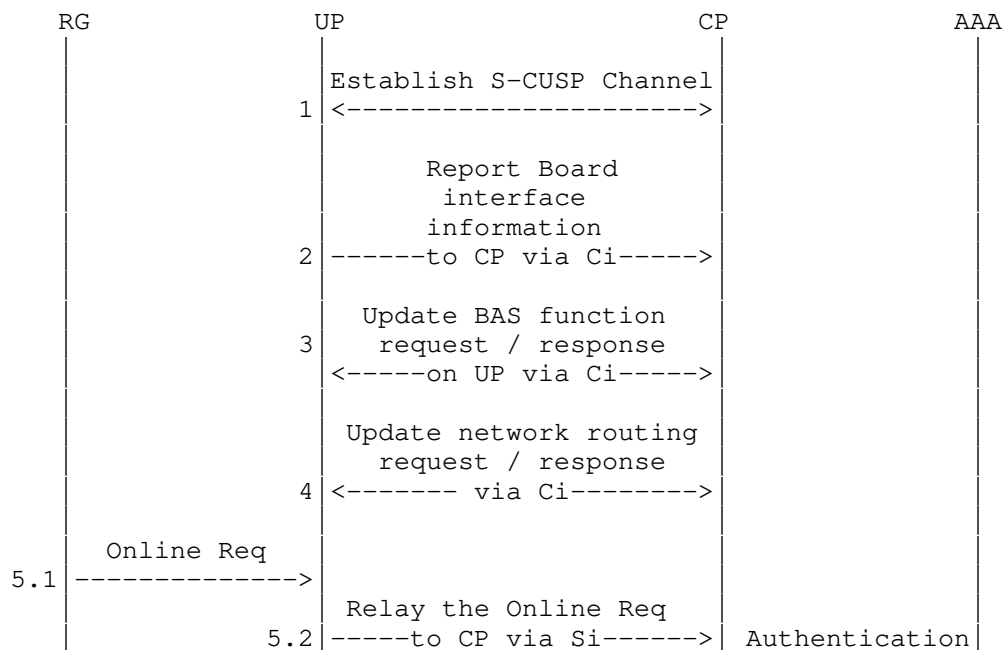
The CP uses the Control Interface to deliver subscriber session states, network routing entries, etc. to the UP (see Section 6.2.7)). The UP uses this interface to report subscriber service statistics, subscriber detection results, etc. to the CP (see Sections 6.3 and 6.4). A carrying protocol for this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



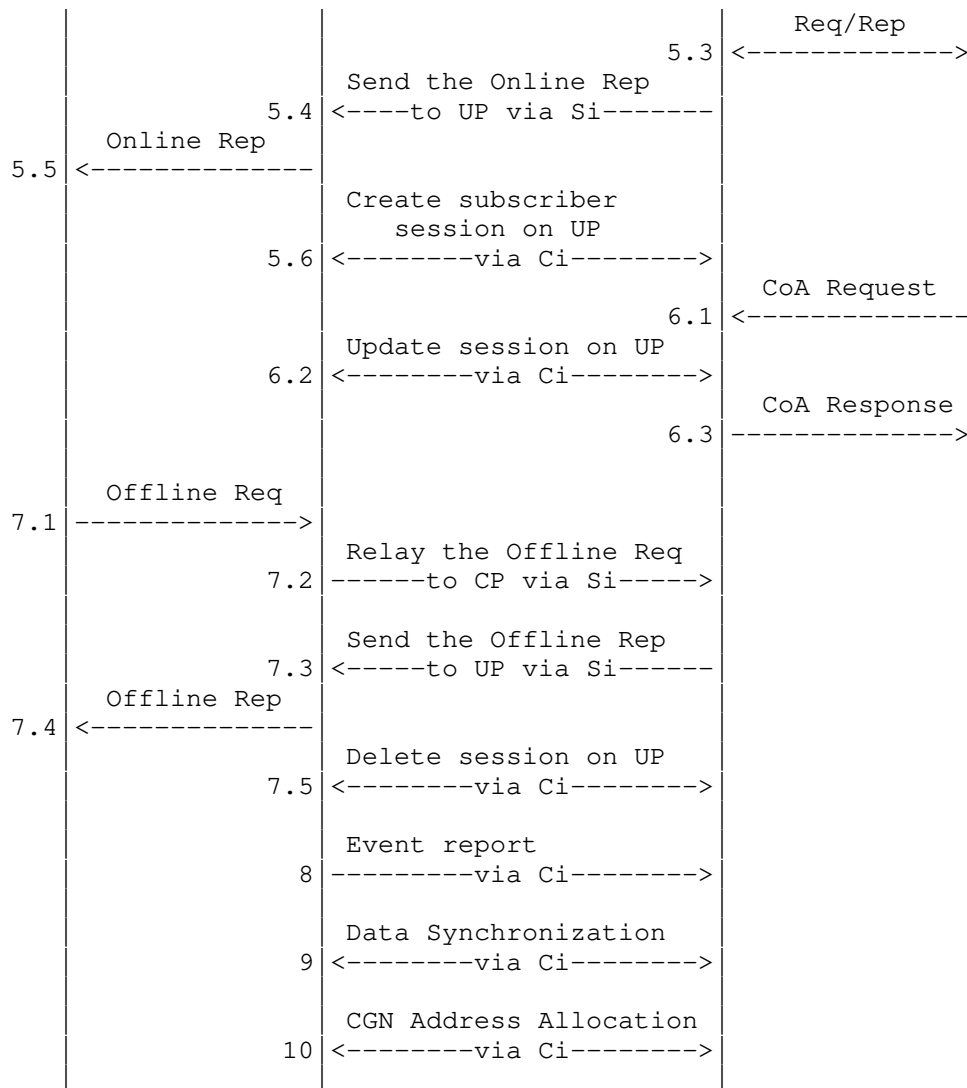


Figure 3: BNG CUPS Procedures Overview

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.

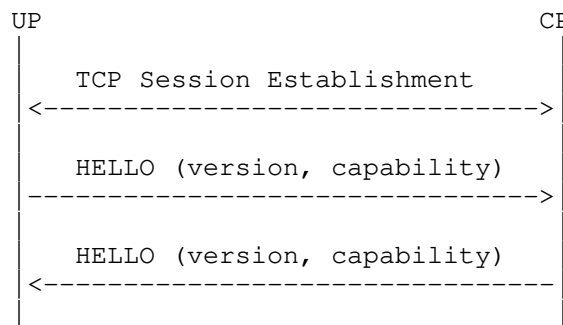


Figure 4: S-CUSP Session Establishment

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 support 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.

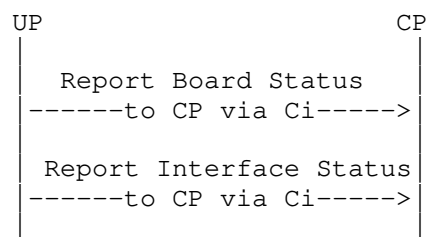


Figure 5: UP Board and Interface Report

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).

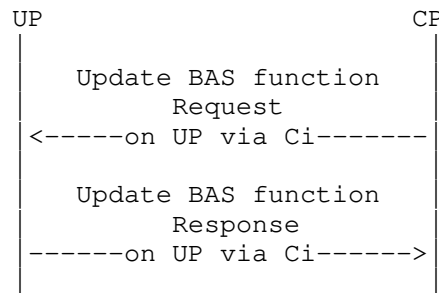


Figure 6: Update BAS Function

#### 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.

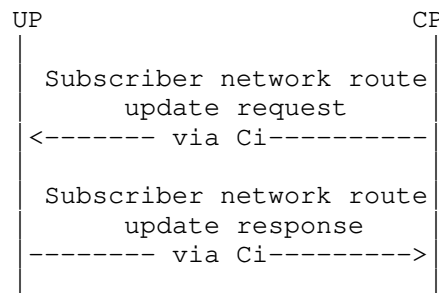


Figure 7: Update Network Routing

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.

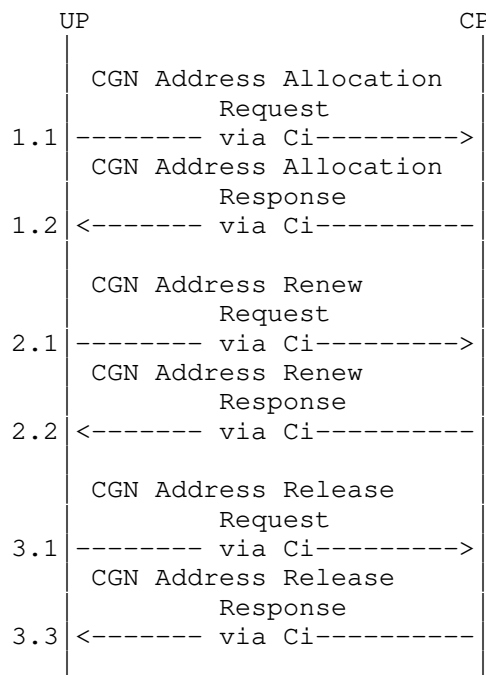


Figure 8: CGN Public IP Address Allocation

#### 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.

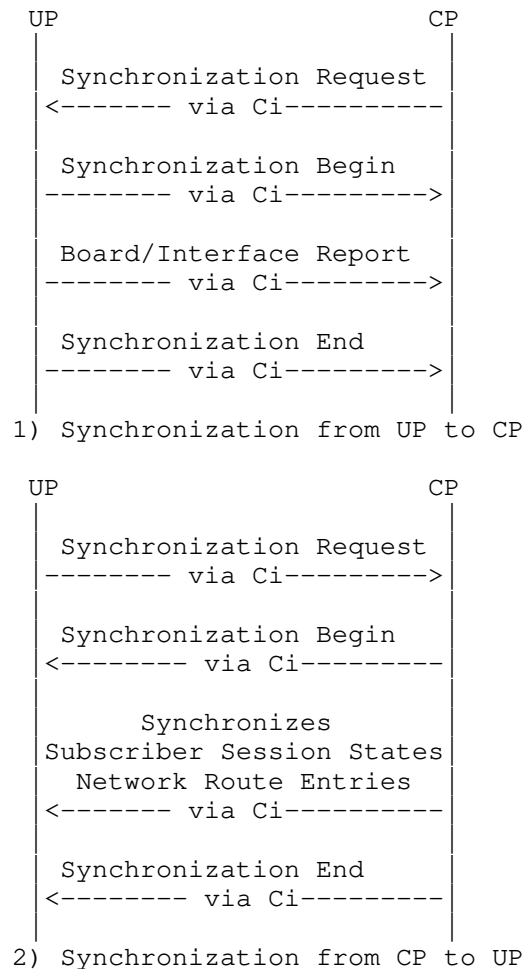


Figure 9: Data Synchronization

#### 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.)

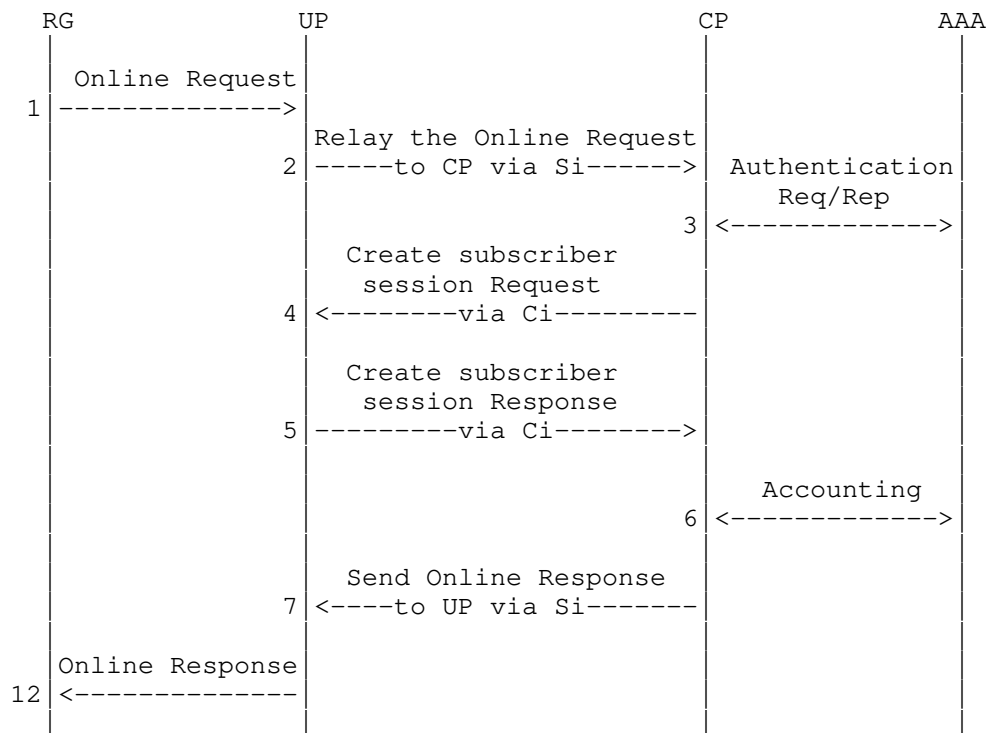


Figure 10: Subscriber Session Create

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.

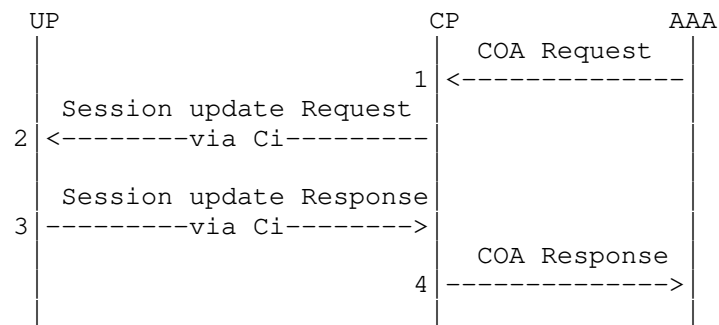


Figure 11: Subscriber Session Update

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.

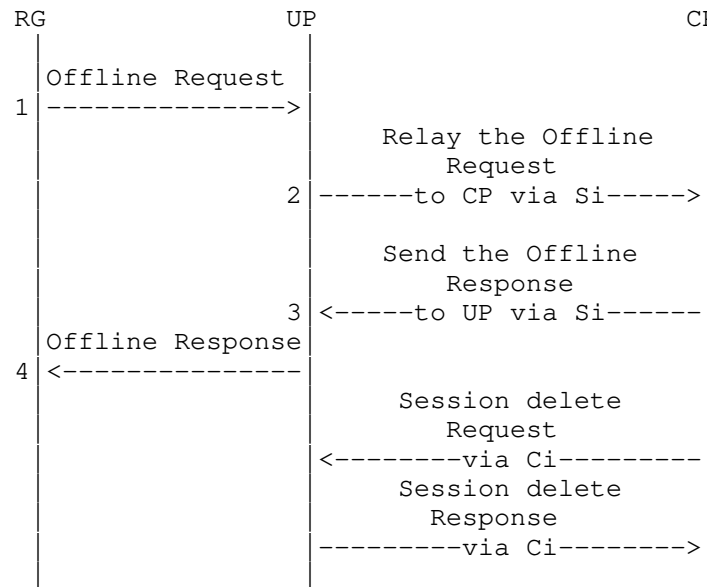


Figure 12: Subscriber Session Delete

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

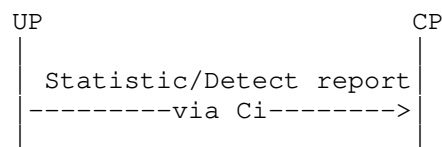


Figure 13: Events Report

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.

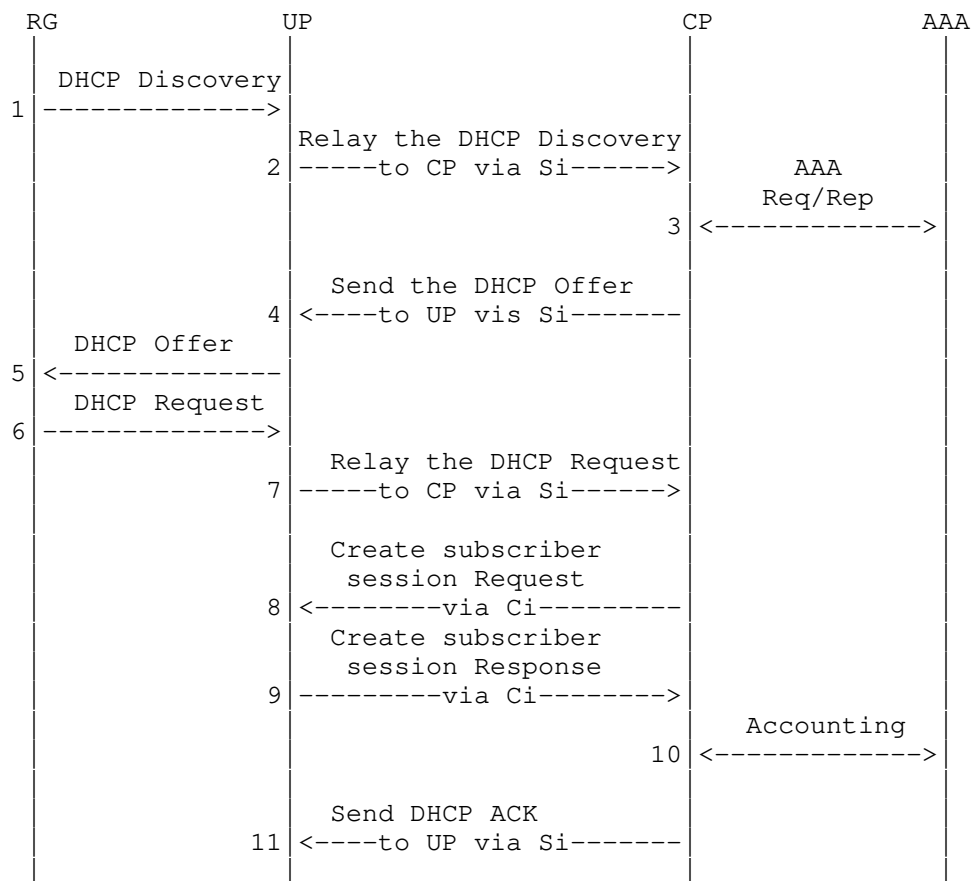




Figure 14: DHCPv4 Access

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>]
  
```

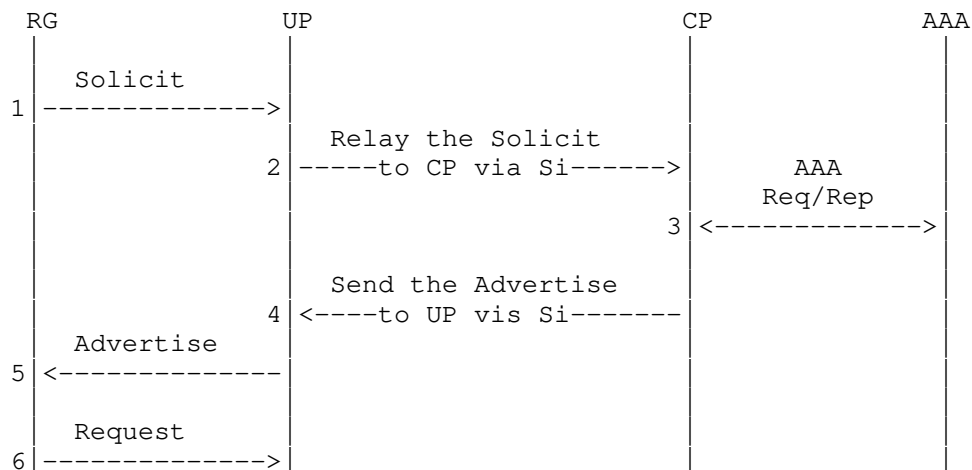
The UP will reply with an Update\_Response message, the format of the Update\_Response message is as follows:

```

<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.



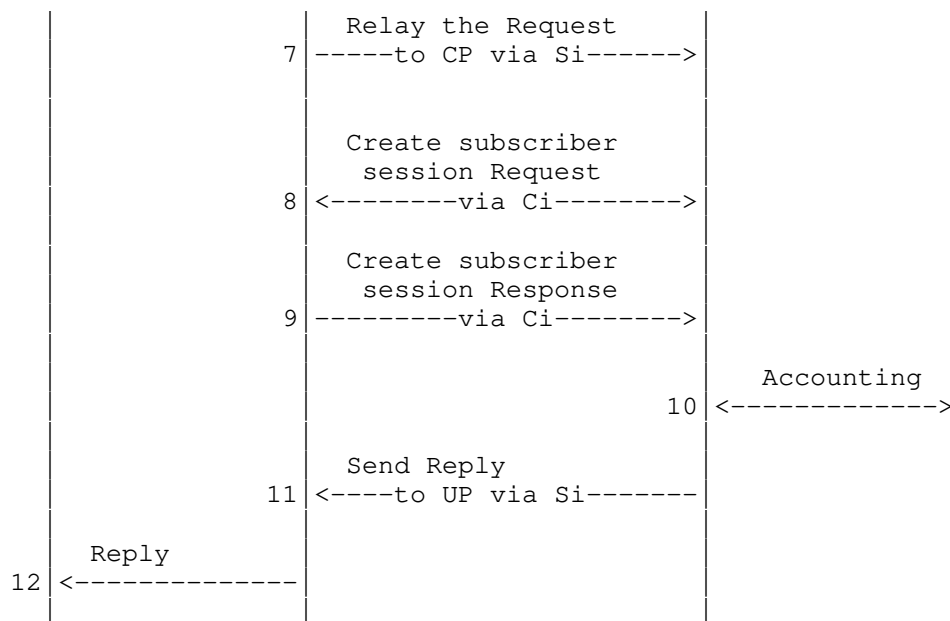


Figure 15: DHCPv6 Access

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.

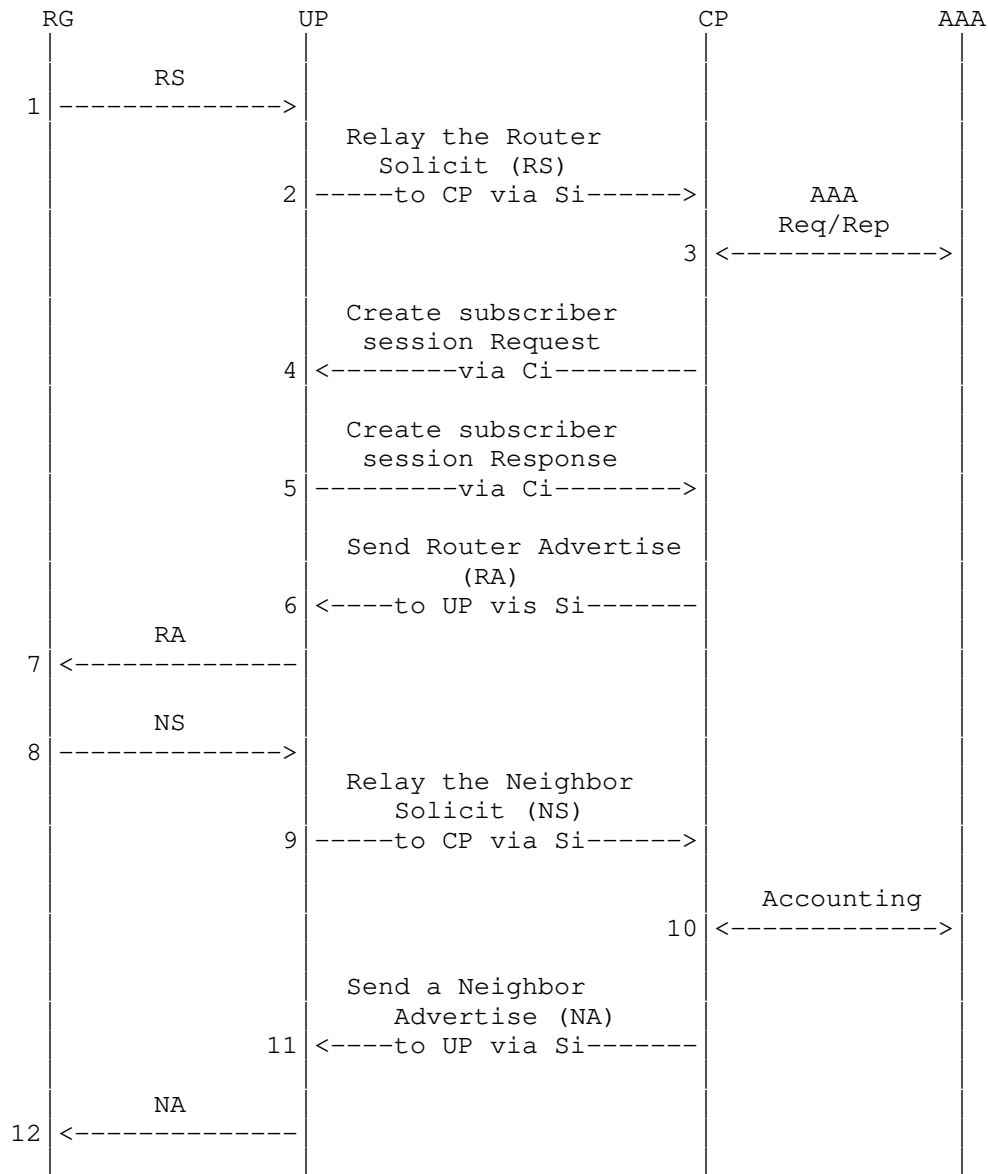


Figure 16: IPv6 SLAAC Access

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:

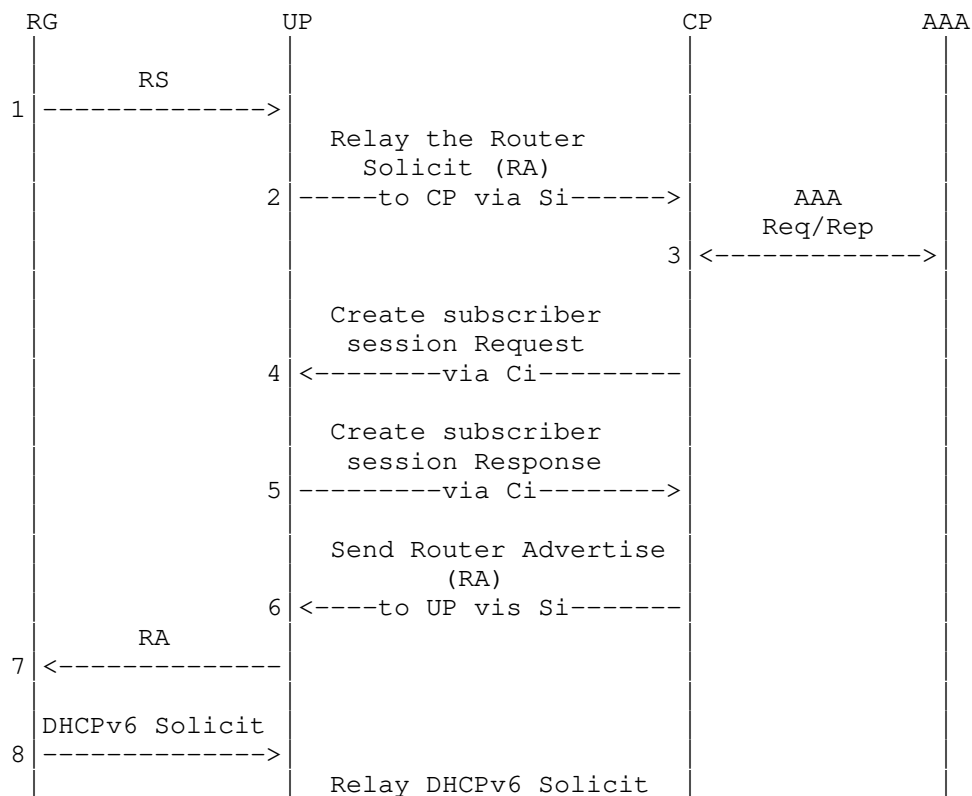
```
<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]
```

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.





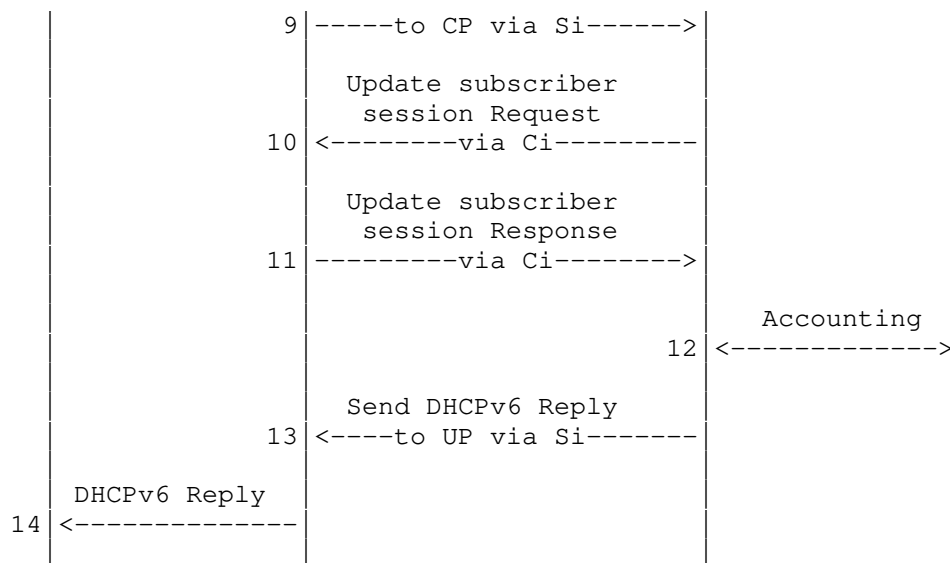


Figure 17: DHCPv6 + SLAAC Access

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:

```

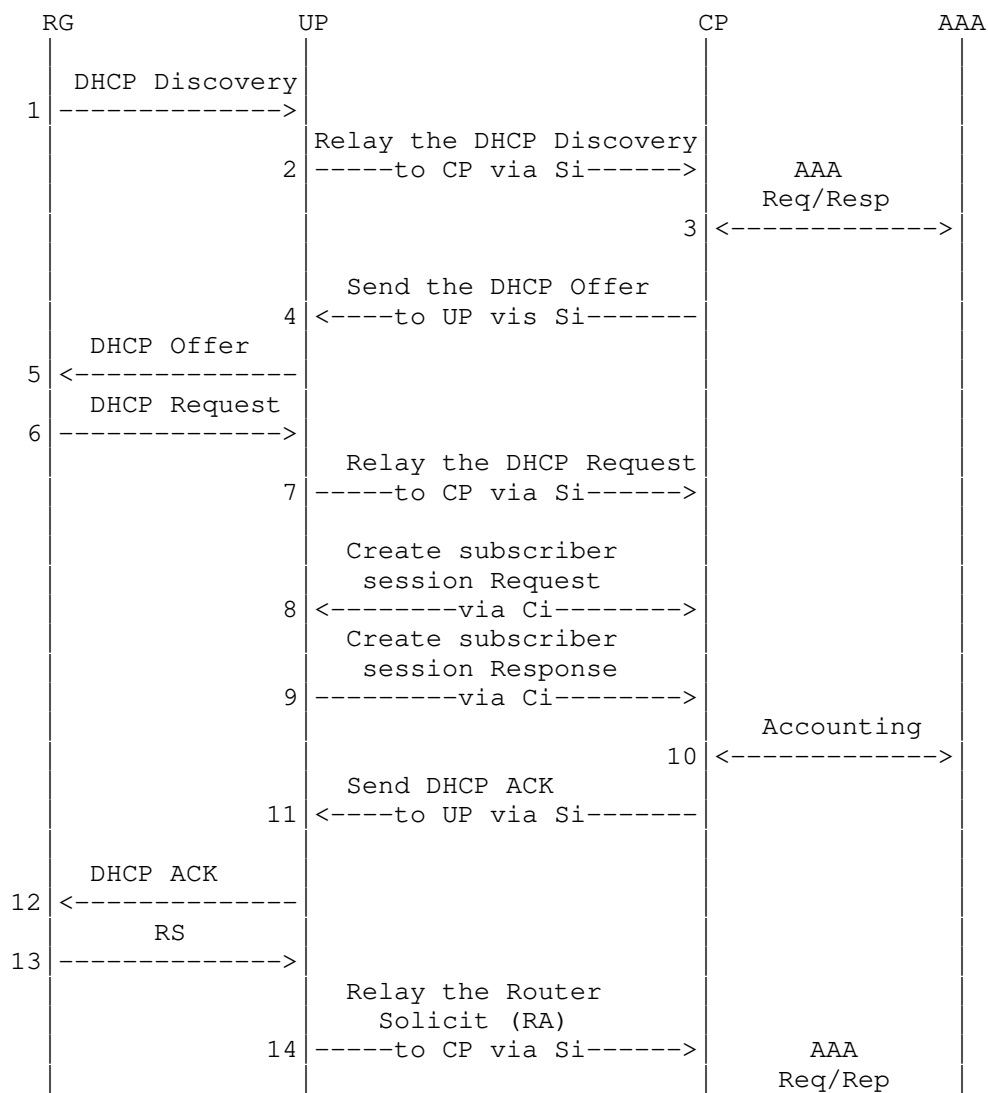
<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv6 Subscriber TLV>
                             <IPv6 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

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.



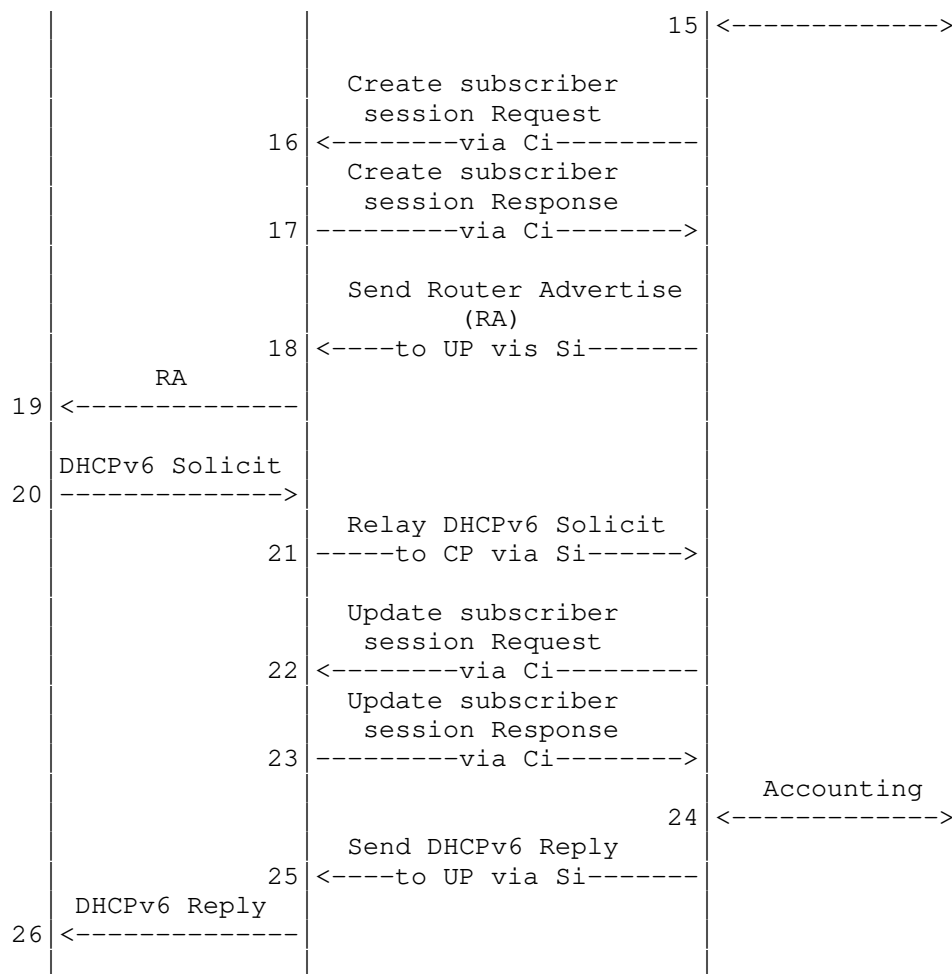


Figure 18: DHCP Dual Stack Access

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>]

```

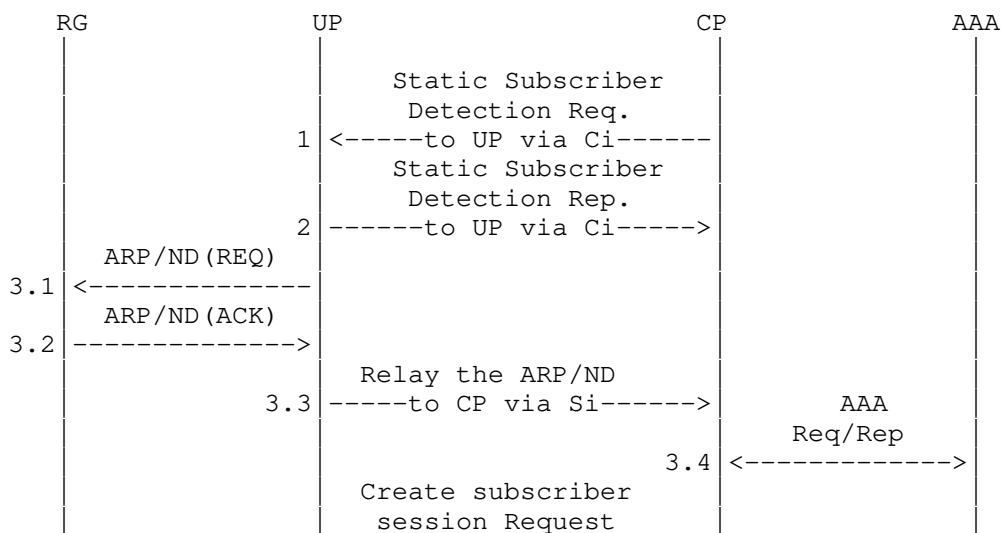
```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

### 5.1.6 L2 Static Subscriber Access

L2 static subscriber access processes are as follows:



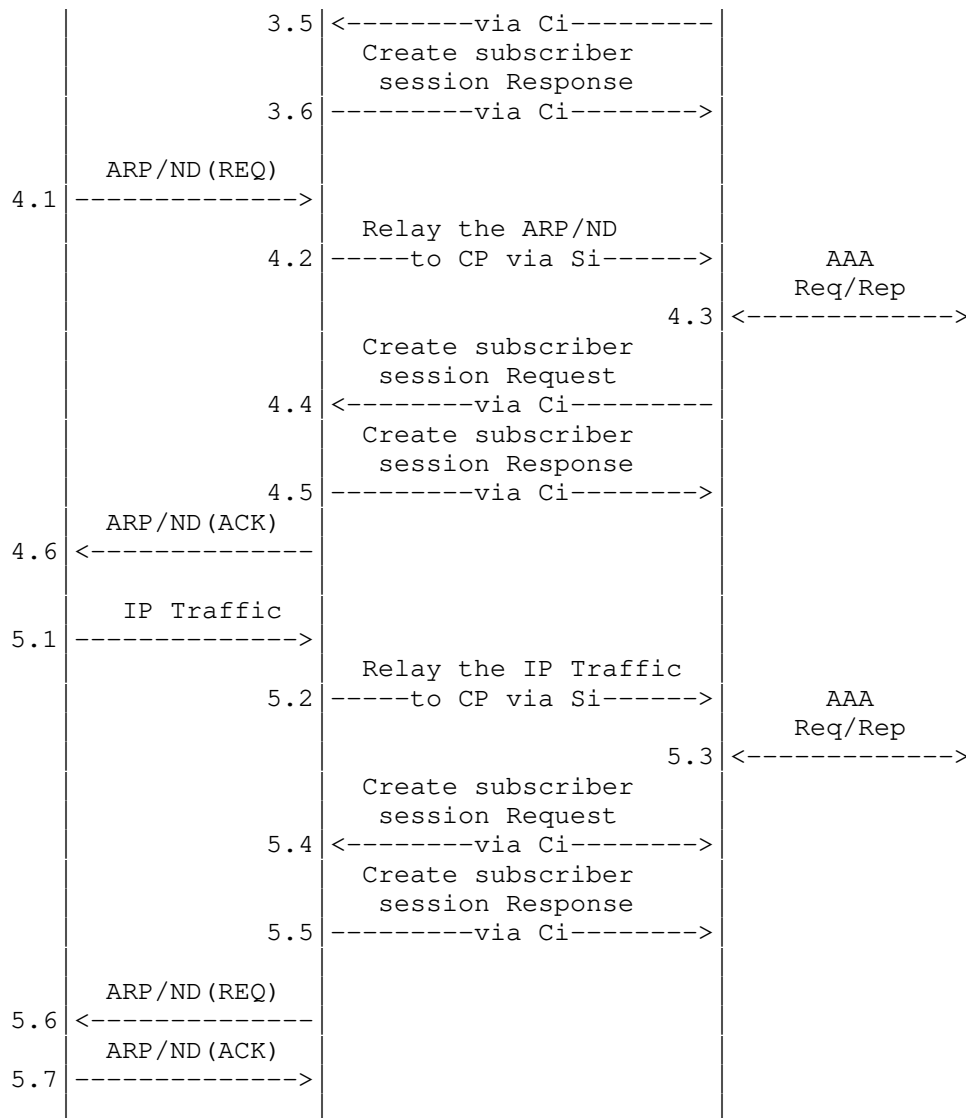


Figure 19: L2 Static Subscriber Access

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.

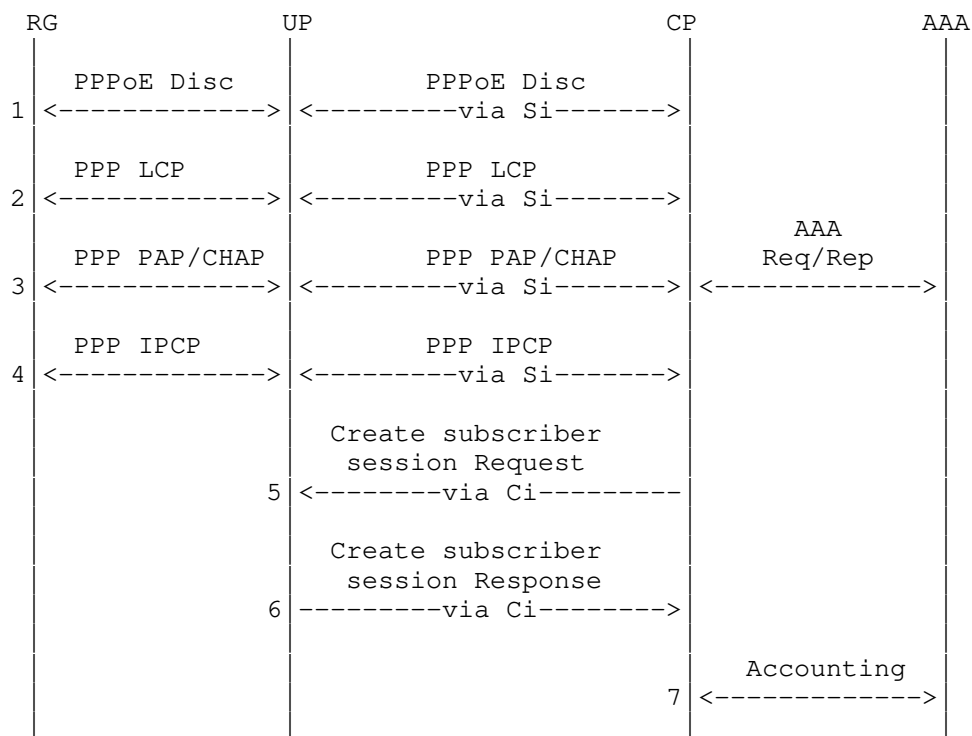


Figure 20: IPv4 PPPoE Access

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>]

```

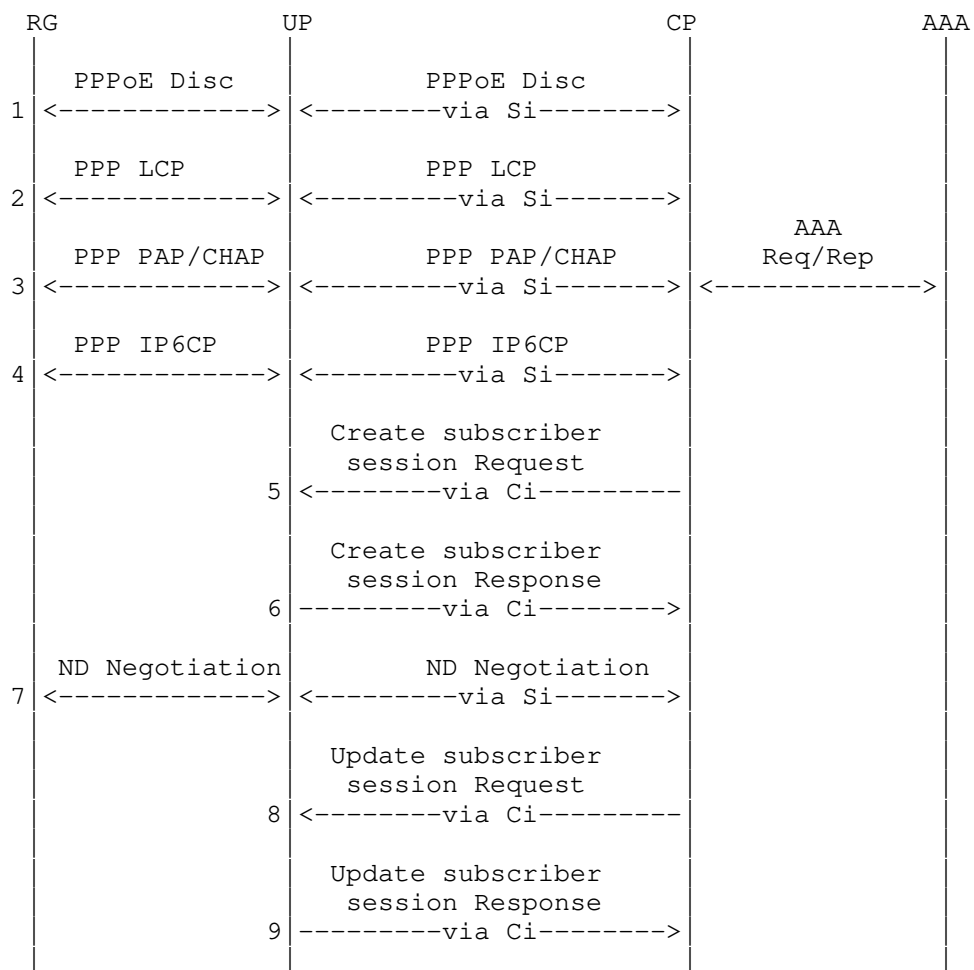
```

<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.





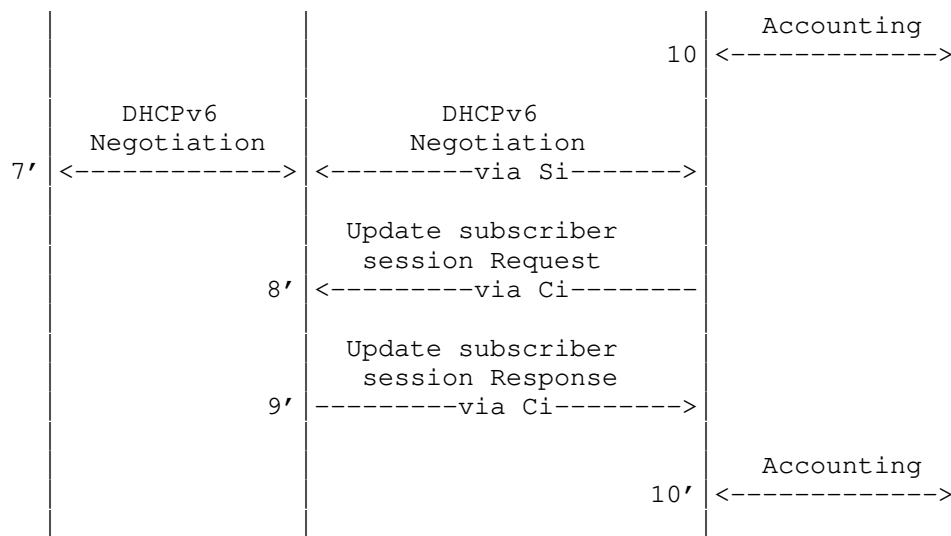


Figure 21: IPv6 PPPoE Access

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>]

```

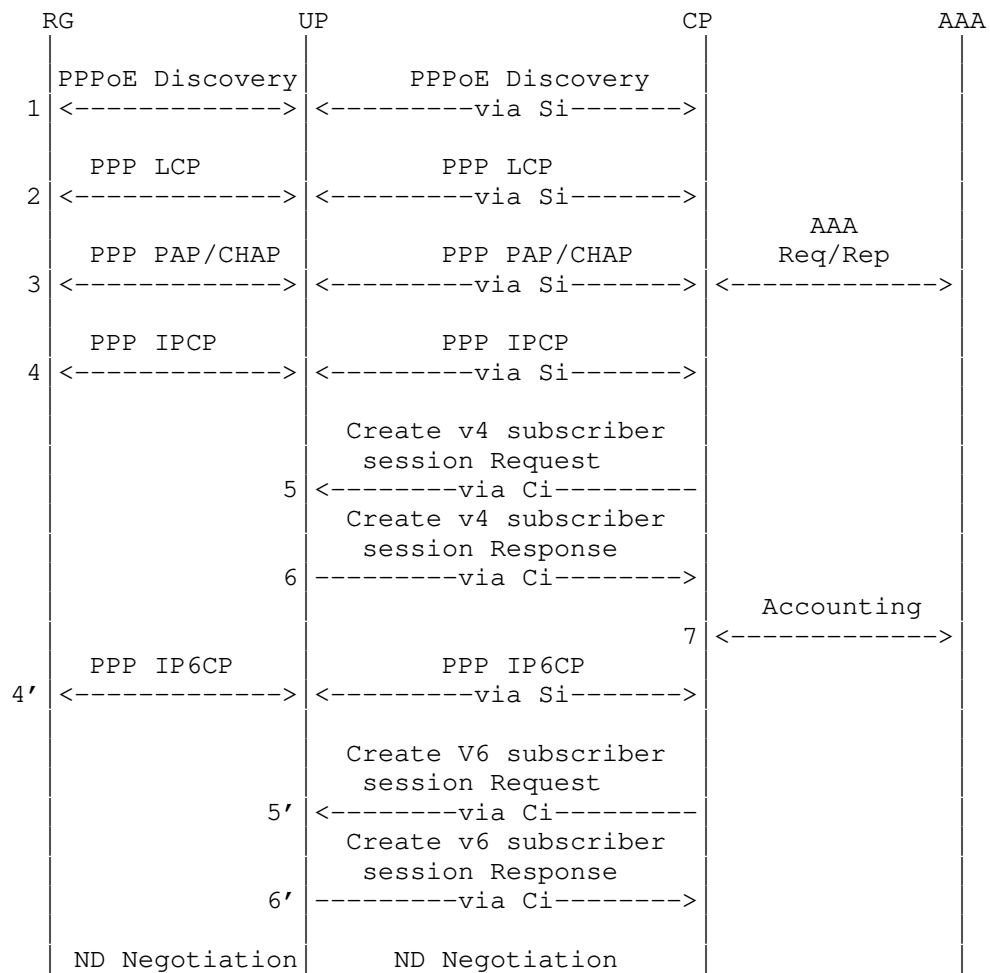
```

<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.



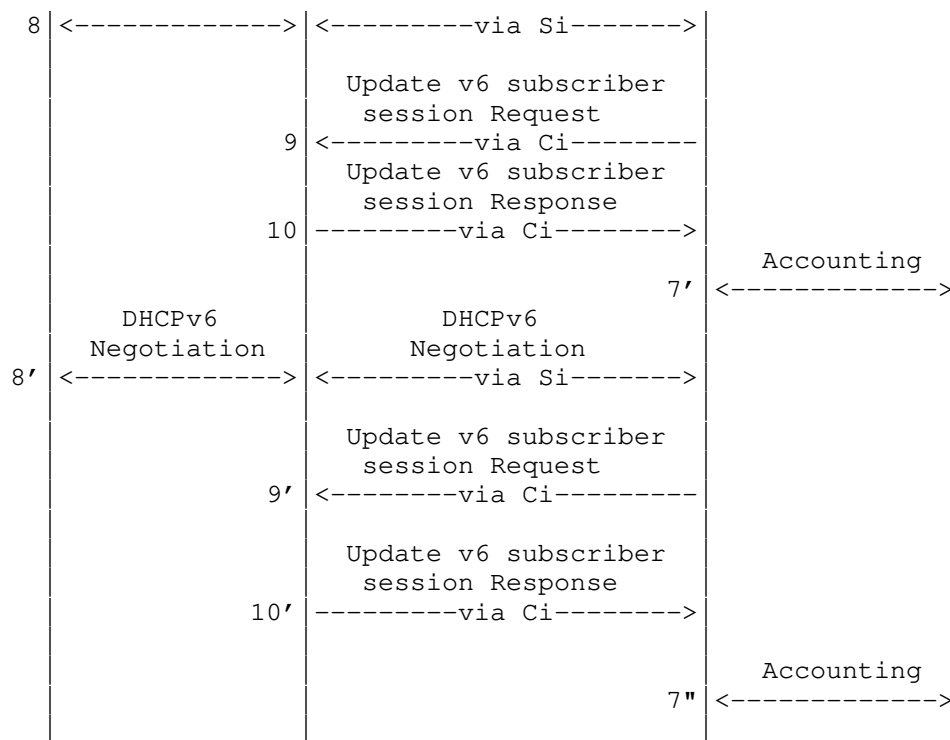


Figure 22: PPPoE Dual Stack Access

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>]

```

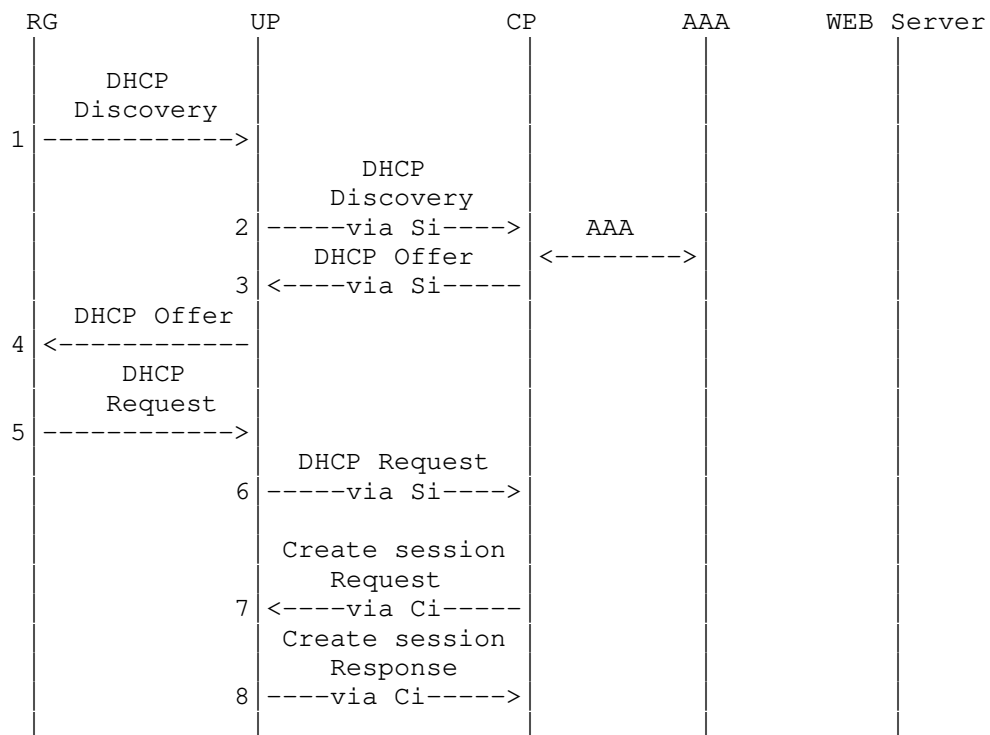
```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>

```

## 5.3 WLAN Access

The following figure shows the WLAN access call flow.



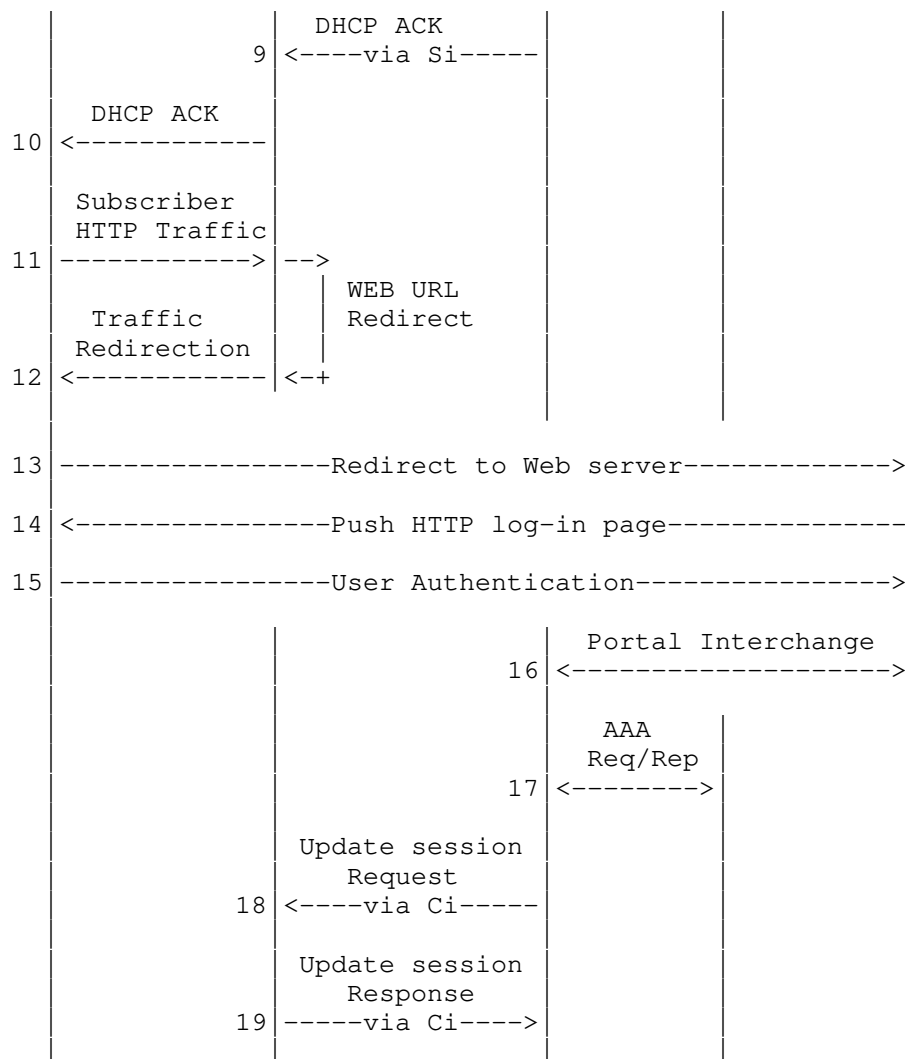


Figure 23: WLAN Access

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>]

```

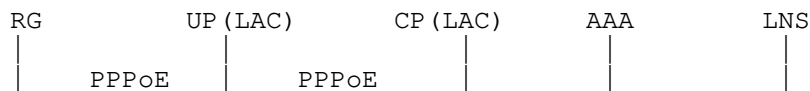
```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

## 5.4 L2TP

### 5.4.1 L2TP LAC Access



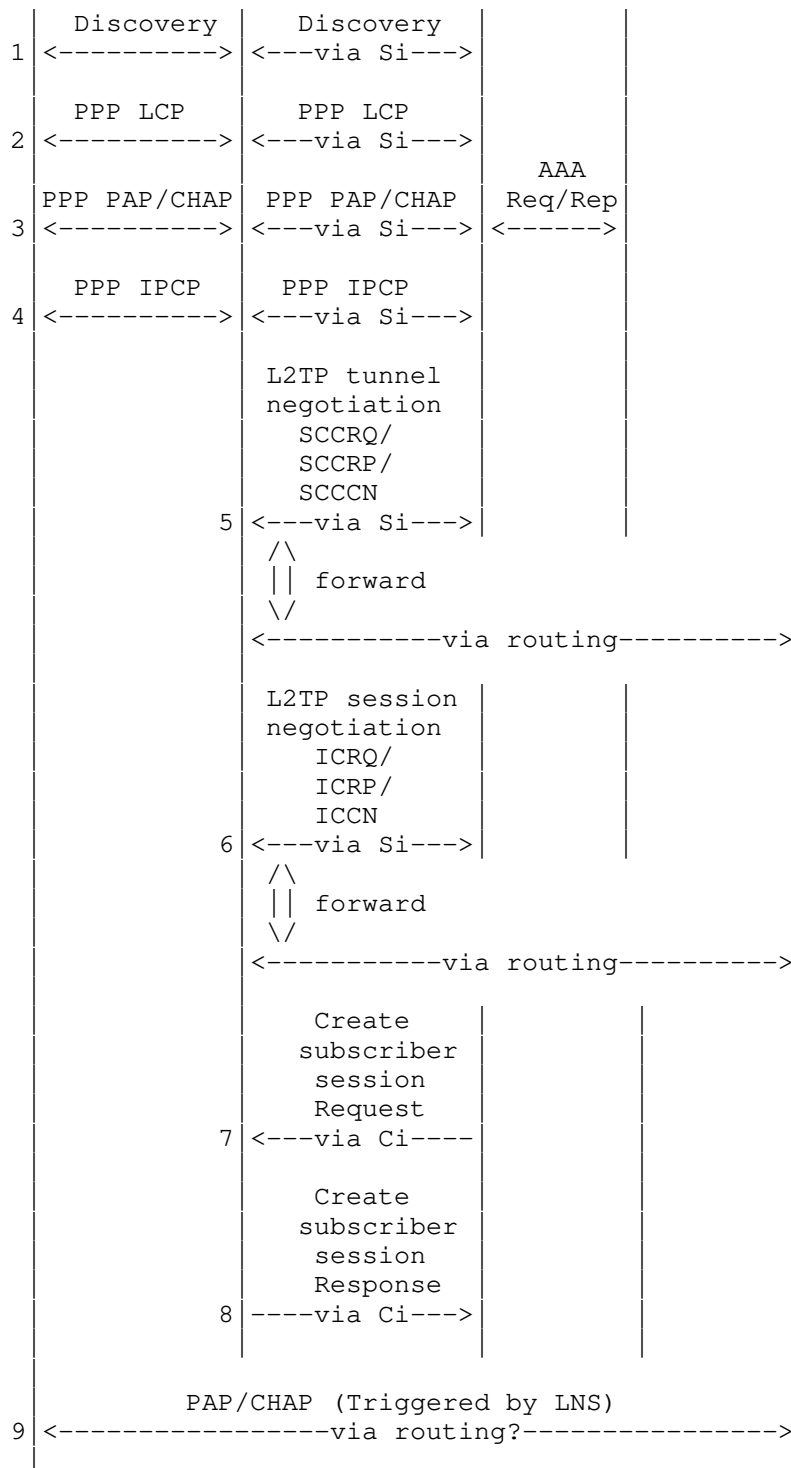


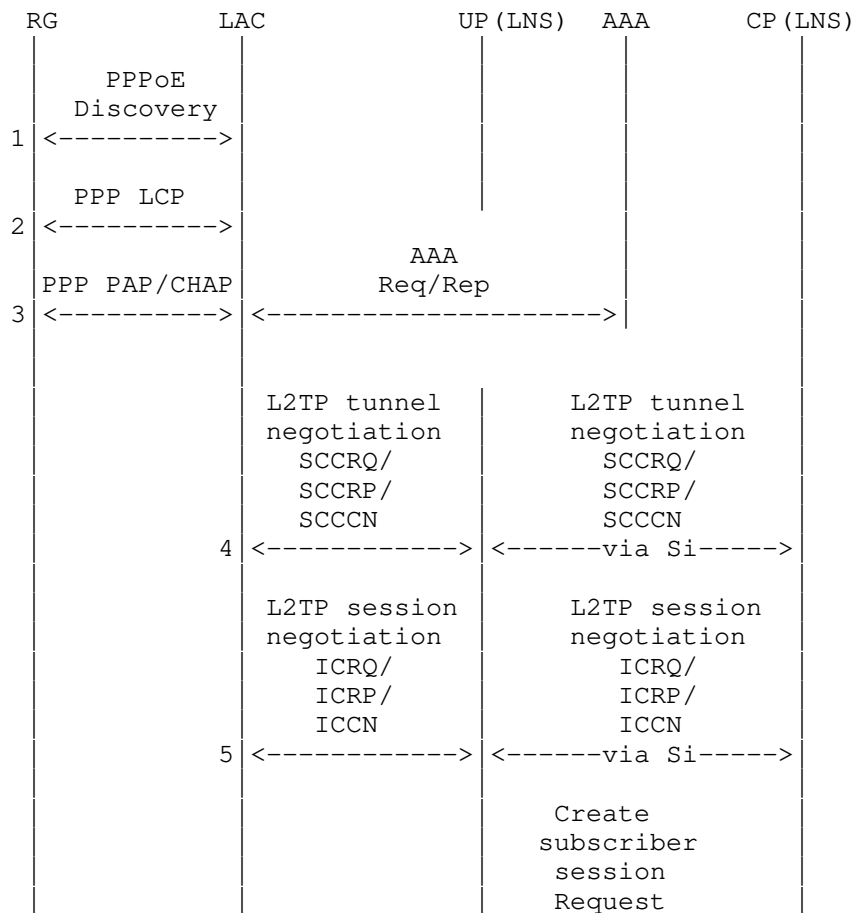
Figure 24: L2TP-LAC Access

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





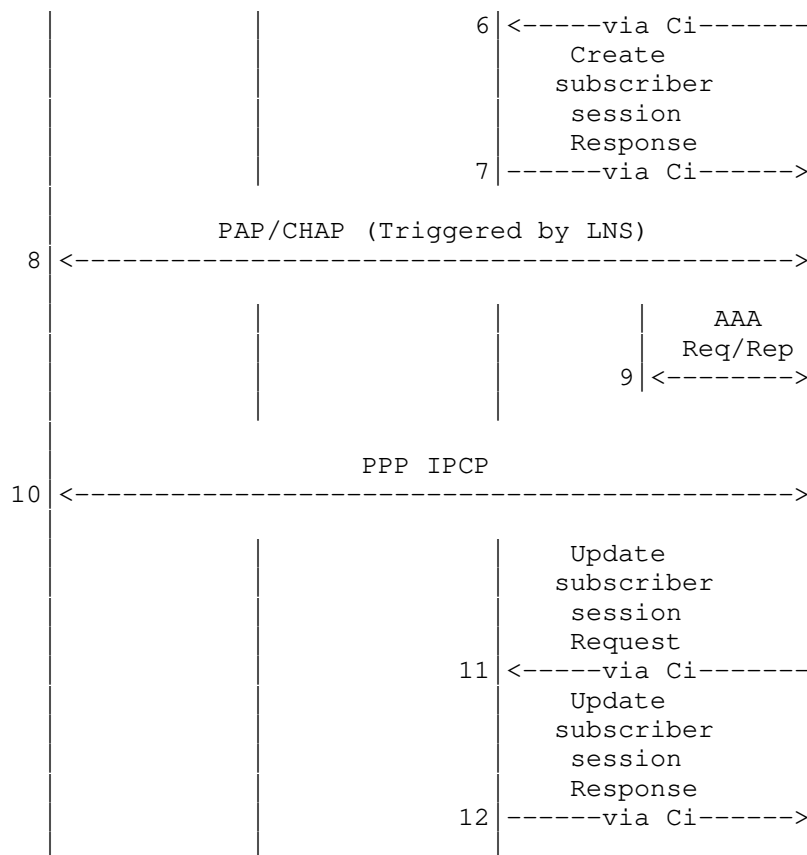


Figure 25: IPv4 L2TP-LNS Access

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>]
  
```

```

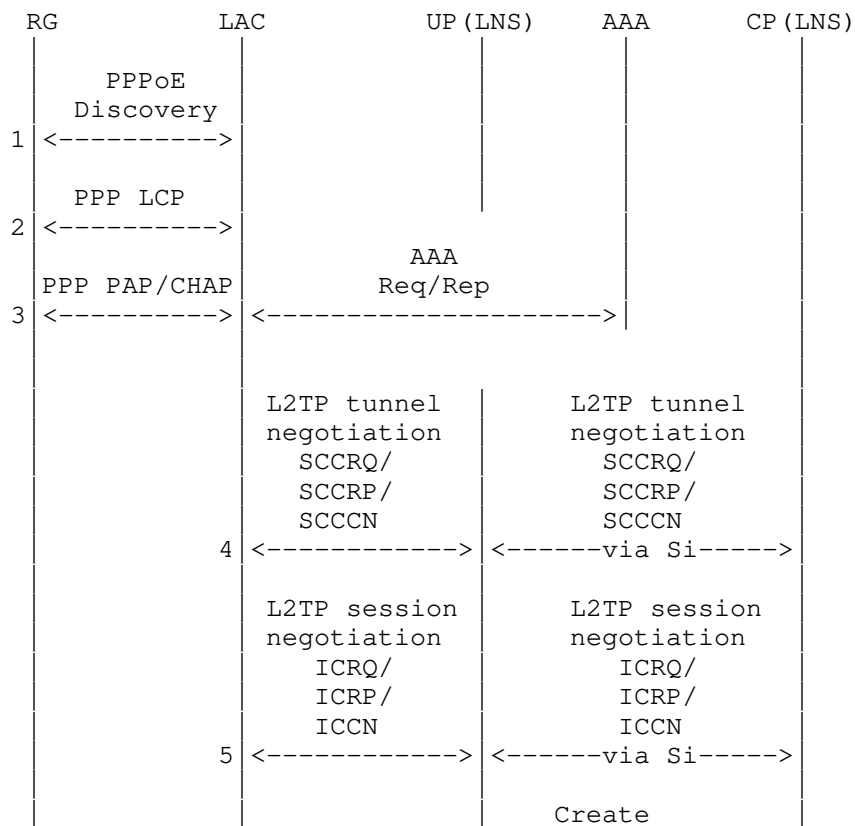
<Update_Response Message> ::= <Common Header>
                             <Update Response TLV>
                             [<Subscriber CGN Port Range TLV>]
  
```

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



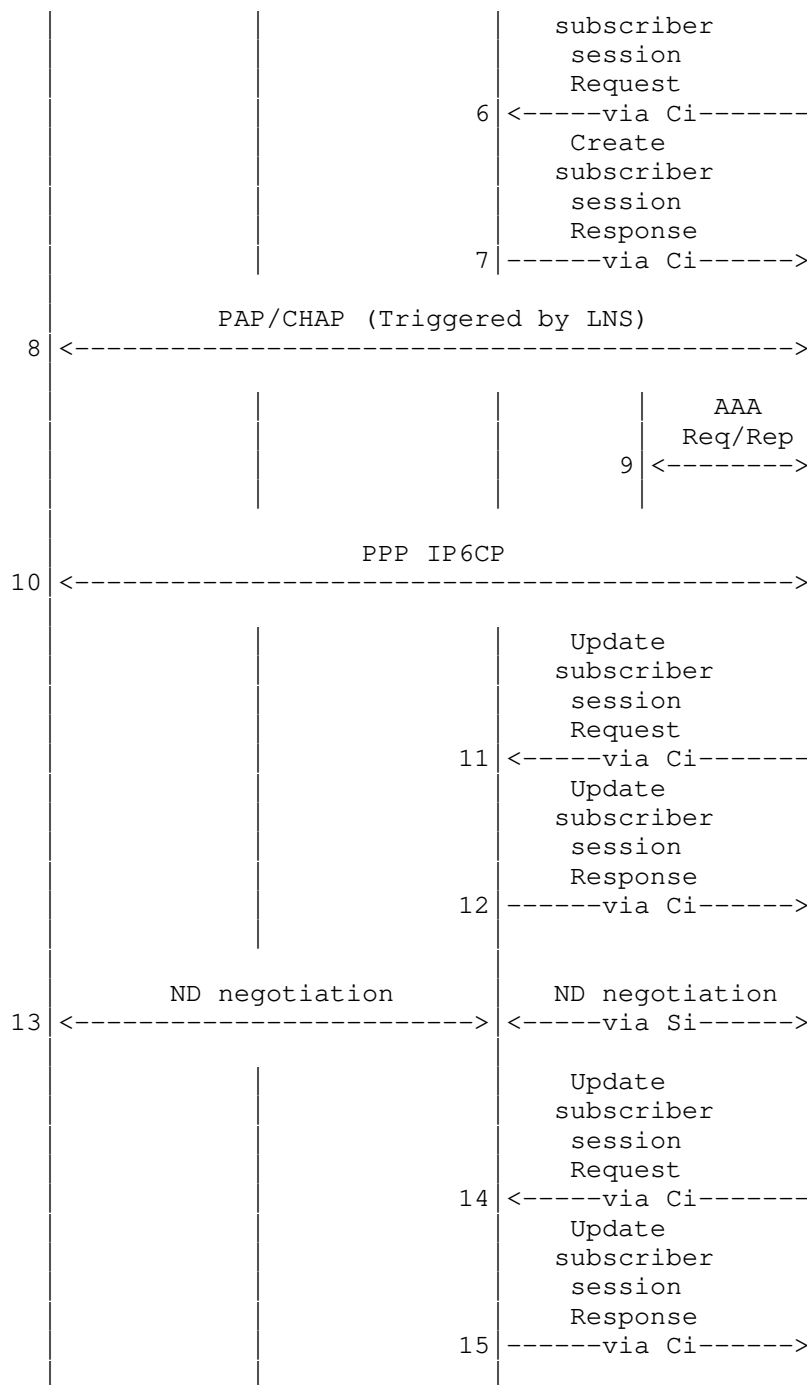


Figure 26: L2TP-LNS IPv6 Access

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)

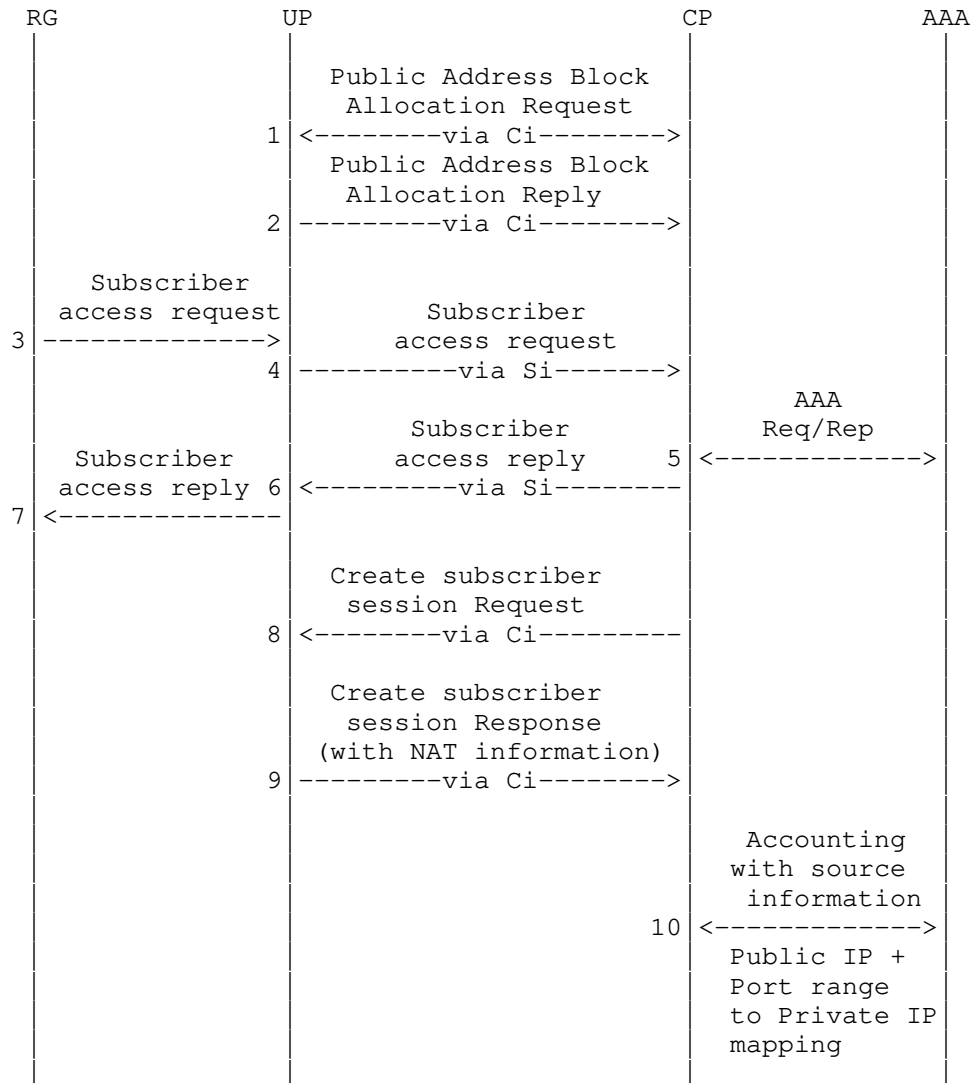


Figure 27: CGN Access

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.

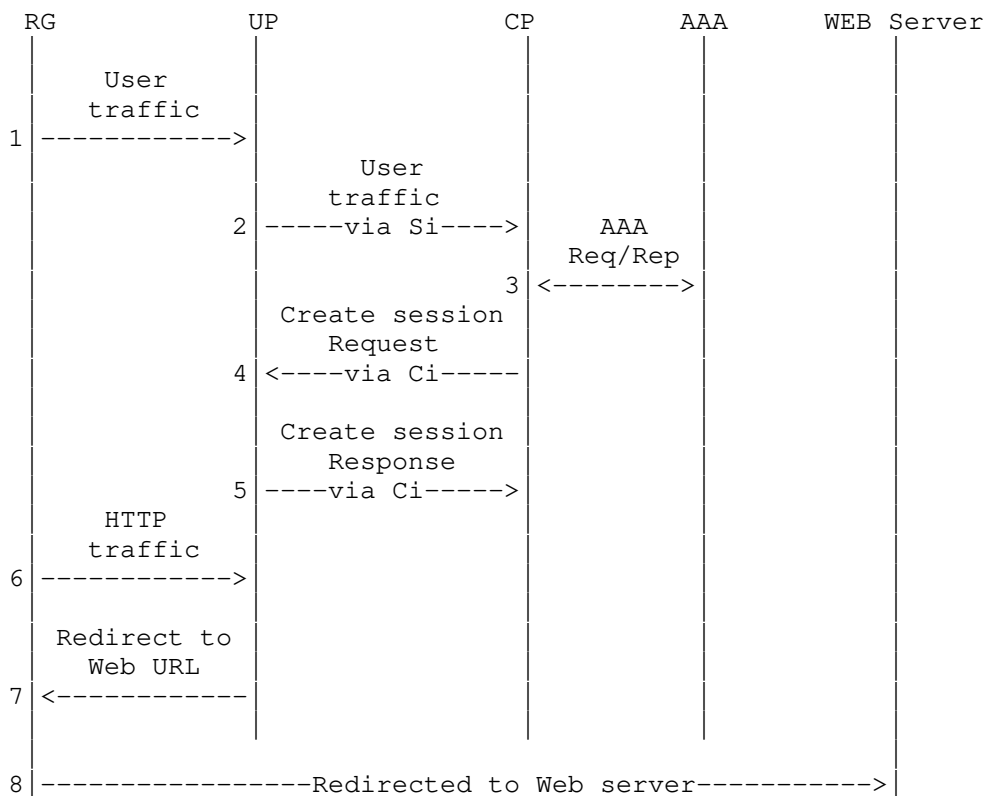
[illegible][illegible]

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



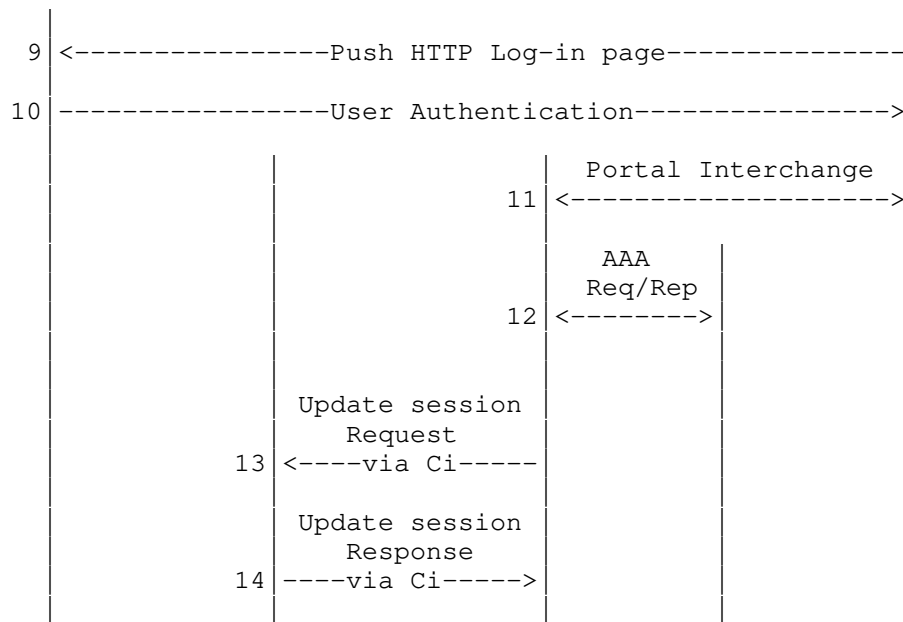


Figure 28: Web Authentication based L3 Leased Line Access

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>]
```

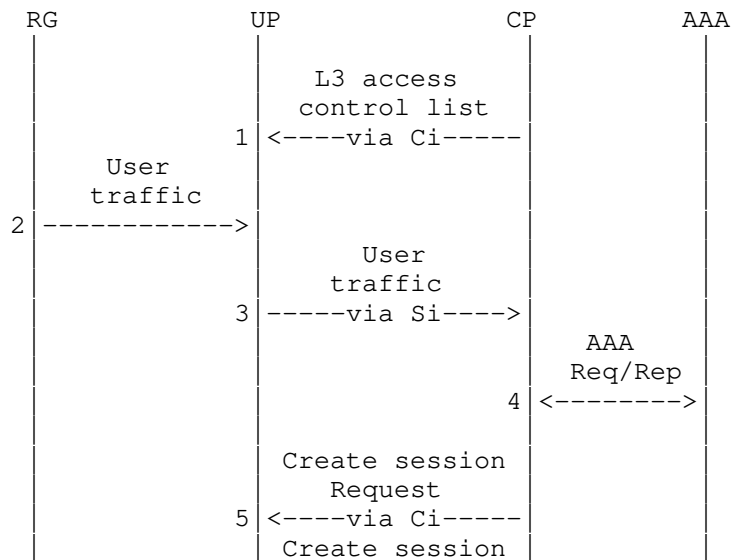
```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
                               [<Subscriber CGN Port Range TLV>]
```

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





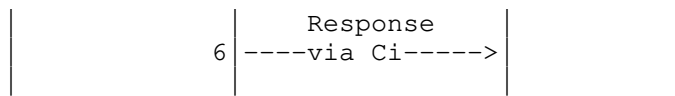


Figure 29: User Traffic Triggered L3 Leased Line Access

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:

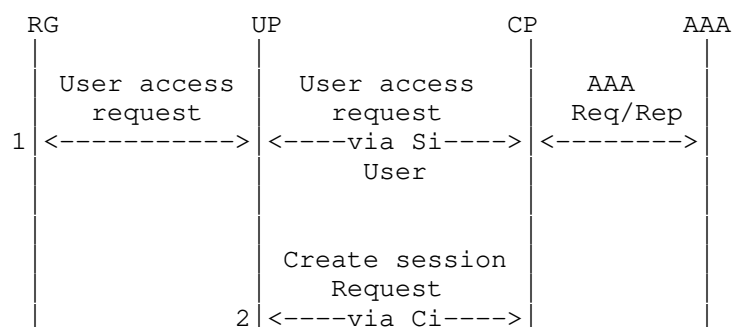
```

<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



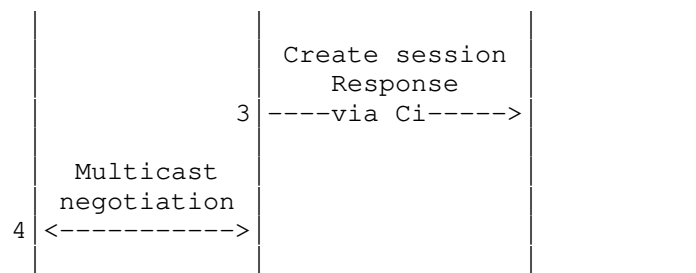


Figure 30: Multicast Access

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:

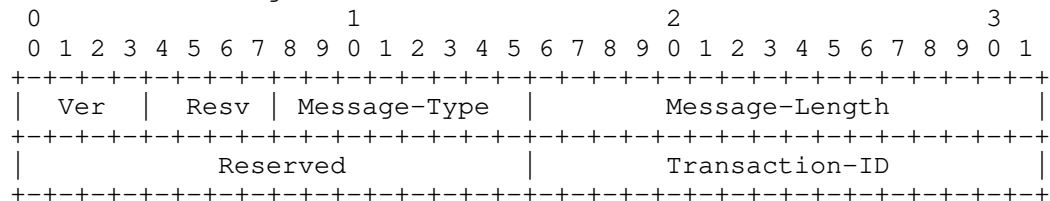


Figure 6.1: S-CUSP Message Common Header

- 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.

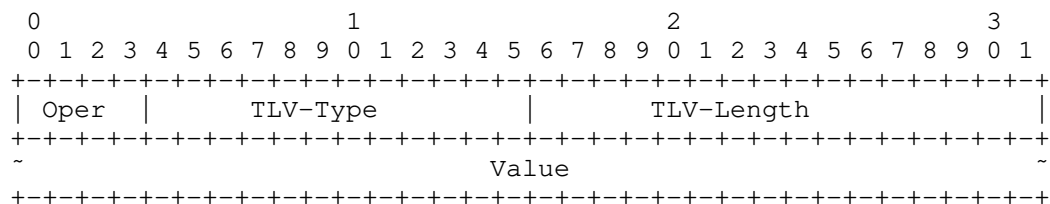


Figure 32: Common TLV Header

- 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:

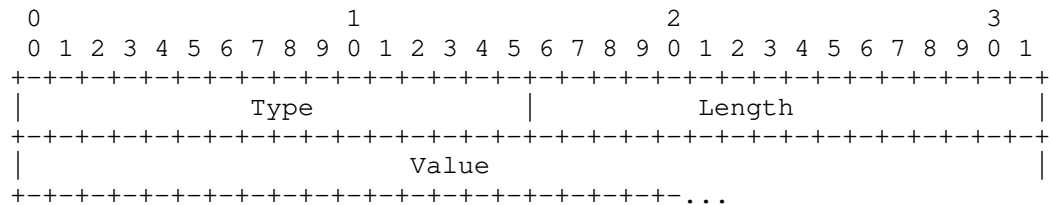


Figure 33: Sub-TLV Header

- 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.

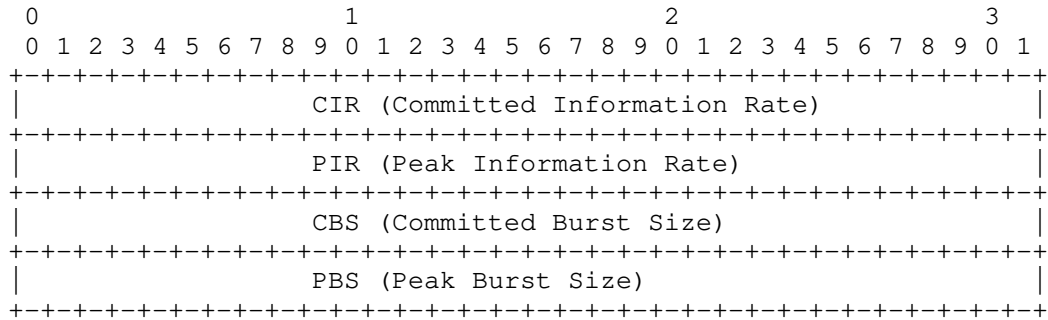


Figure 34: Ingress-CAR sub-TLV

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.

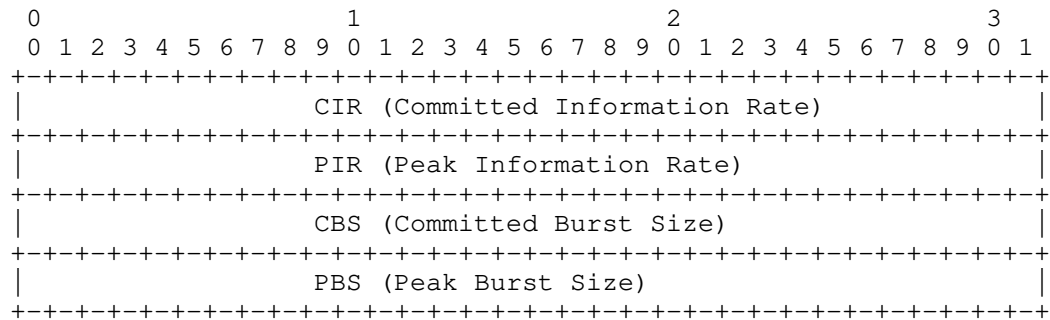


Figure 35: Egress-CAR sub-TLV

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:



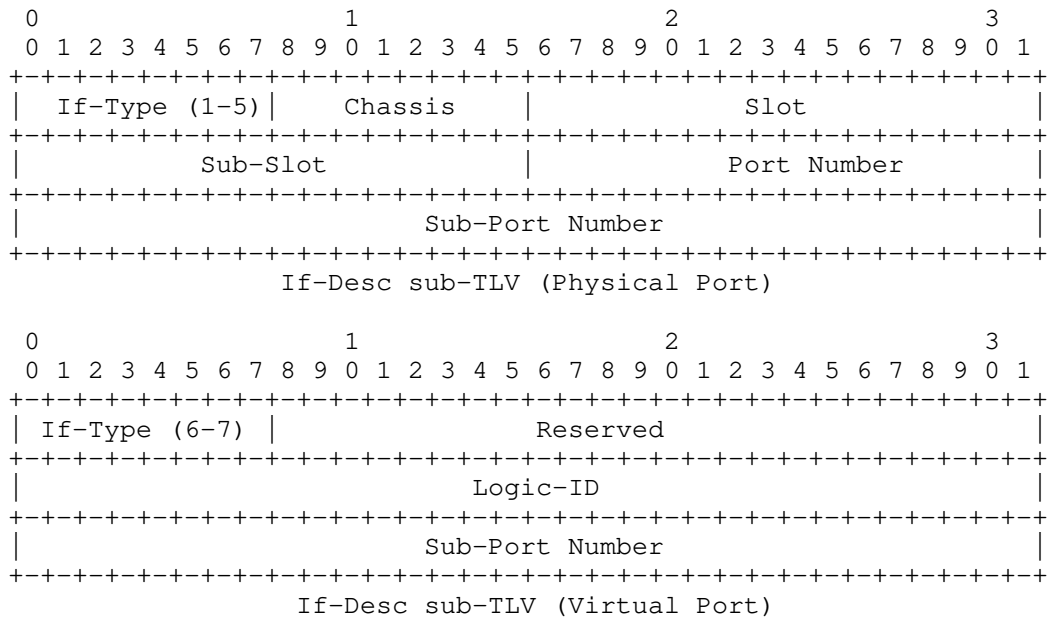


Figure 36: If-Desc sub-TLV Formats

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.

Logic-ID (32 bits): An identifier of a virtual interface (e.g.,  
If-Type: 6-7)

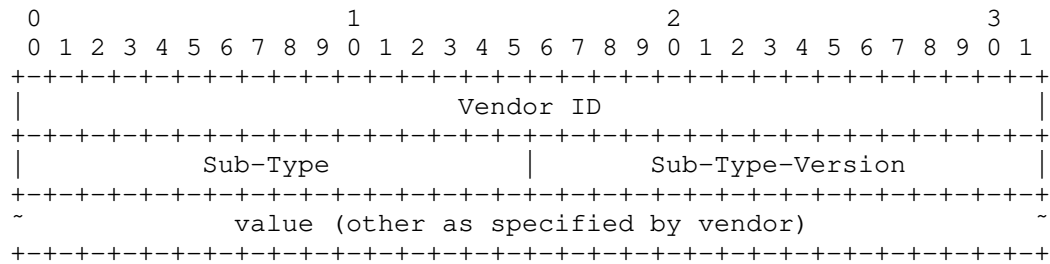


Figure 38: Vendor sub-TLV

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:

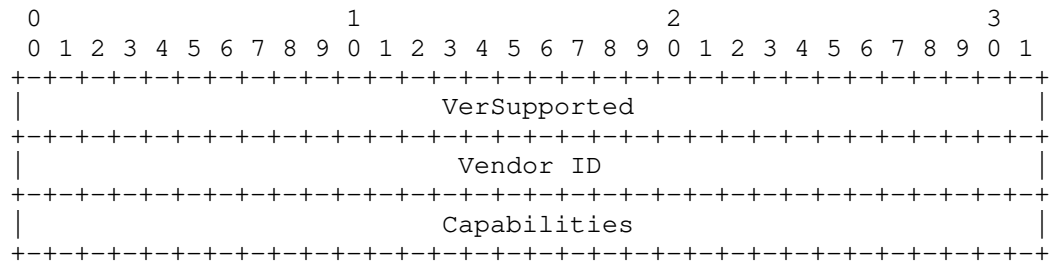


Figure 39: Hello TLV

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:

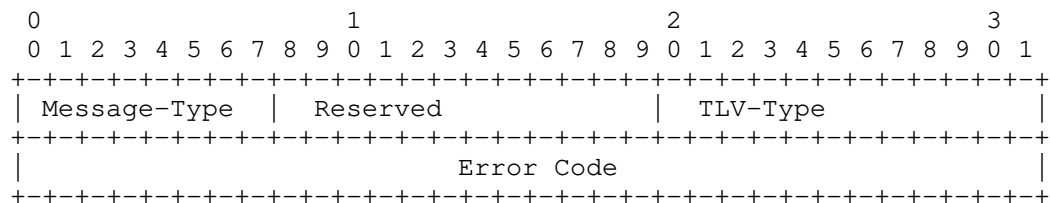


Figure 41: Error Information TLV

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:

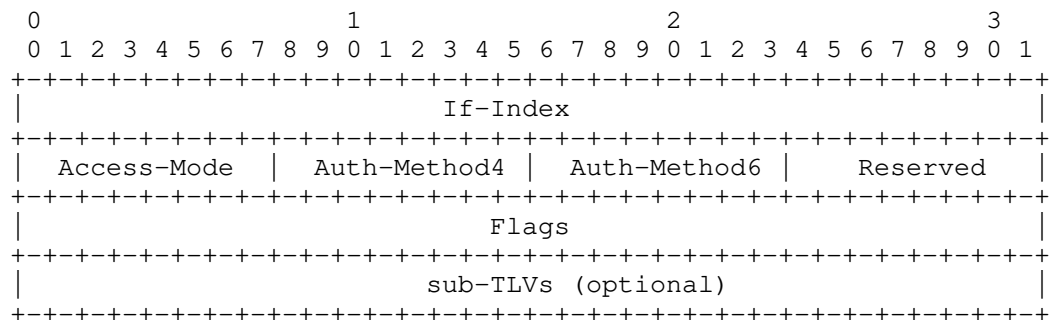


Figure 42: BAS Function TLV

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:

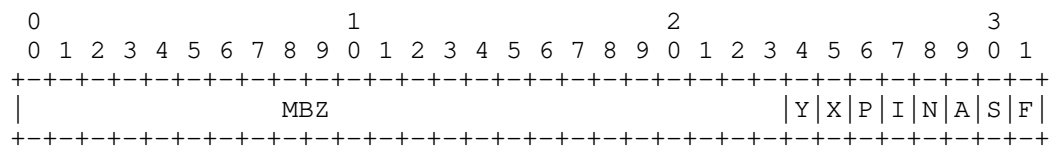


Figure 43: Interface Flags

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:

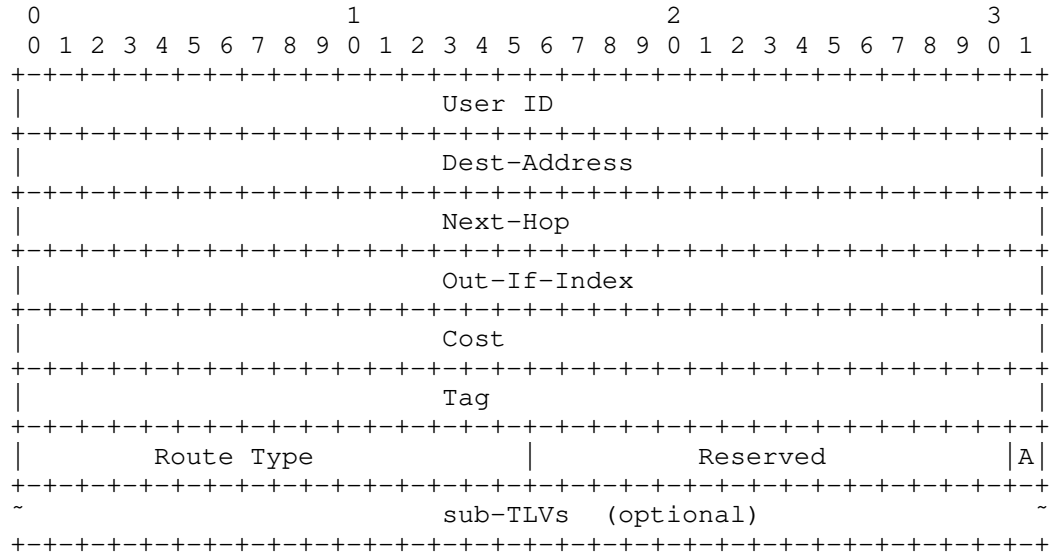


Figure 44: IPv4 Routing TLV

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:

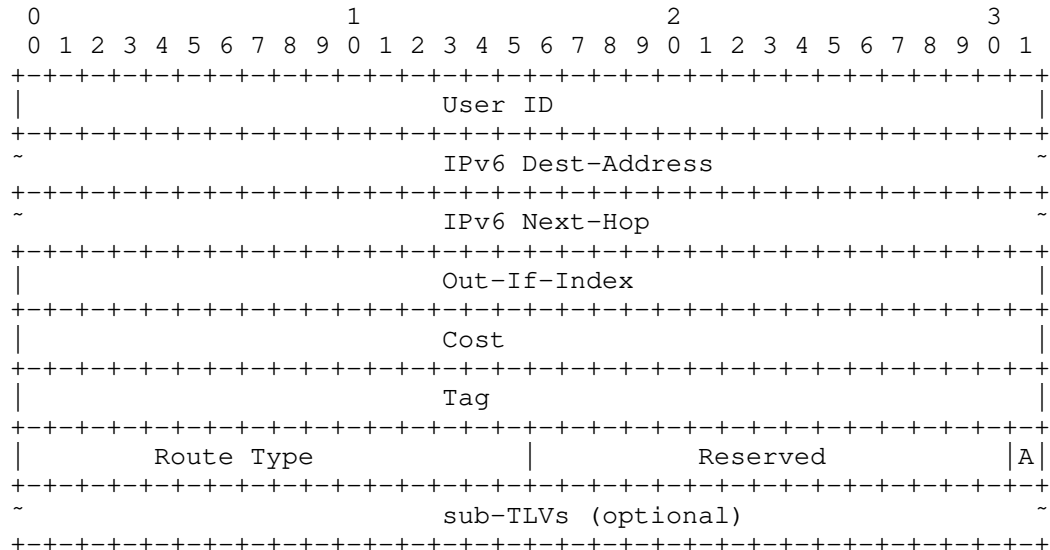


Figure 45: IPv6 Routing TLV

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:

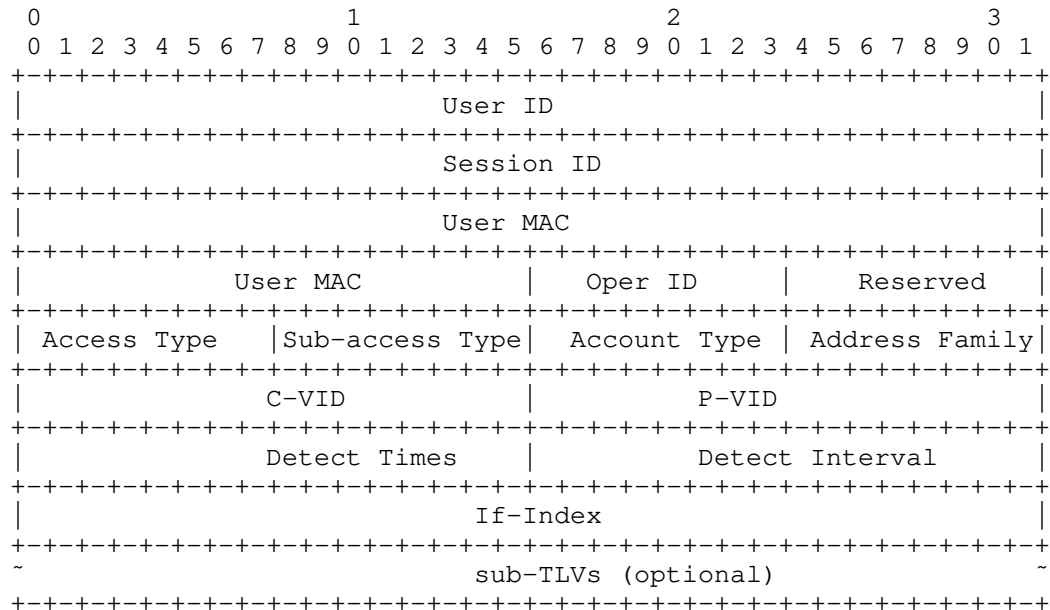


Figure 46: Basic Subscriber TLV

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:

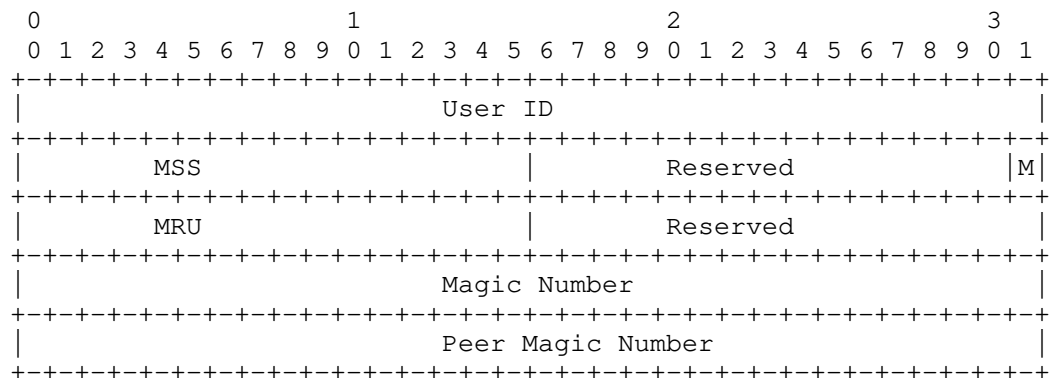


Figure 47: PPP Subscriber TLV

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:

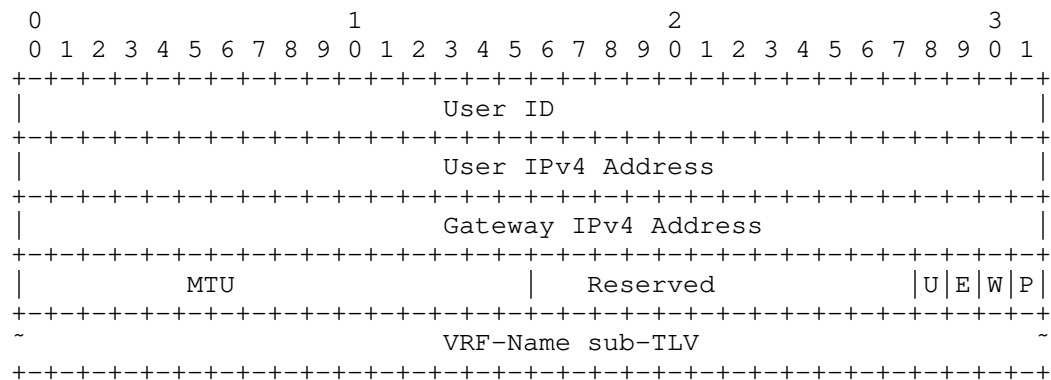


Figure 48: IPv4 Subscriber TLV

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:

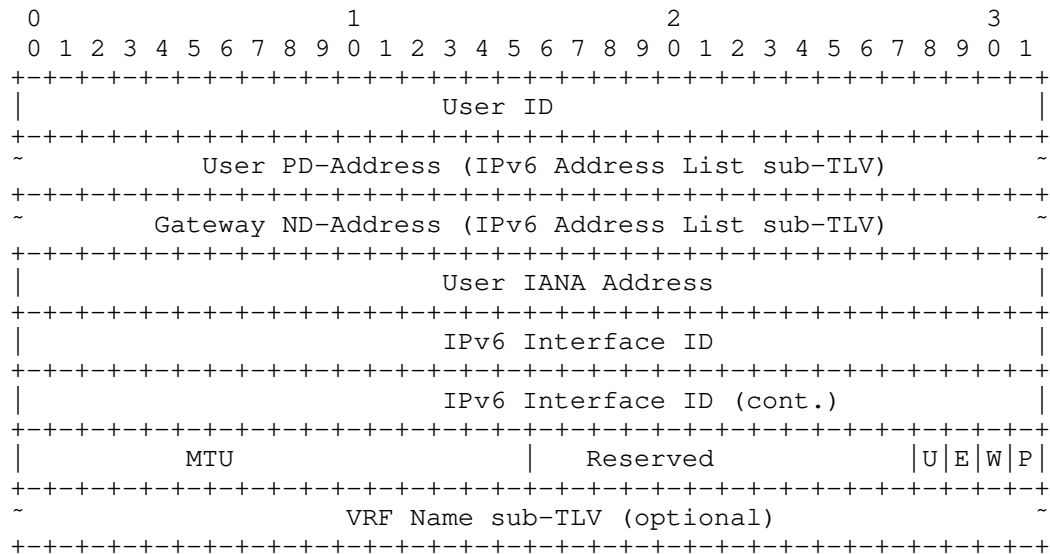


Figure 49: IPv6 Subscriber TLV

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:

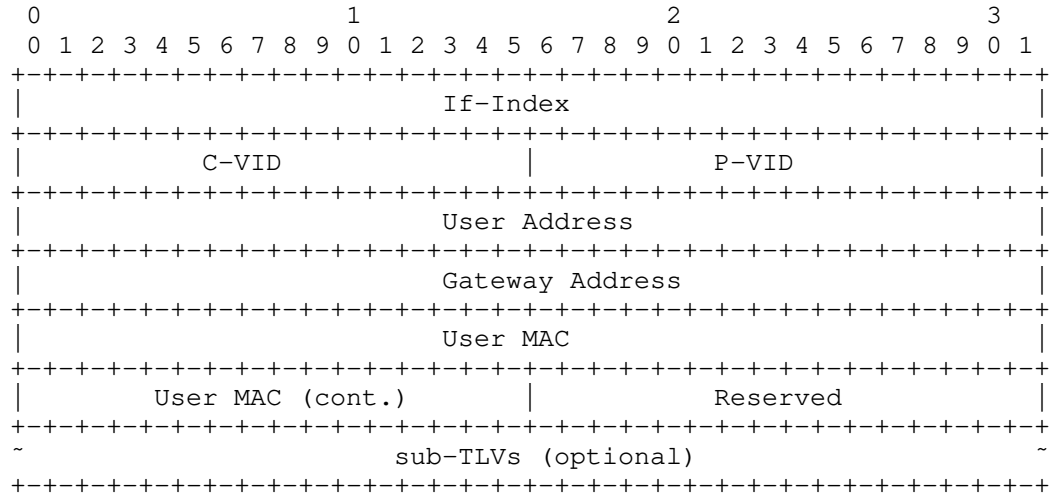


Figure 50: IPv4 Static Subscriber TLV

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:

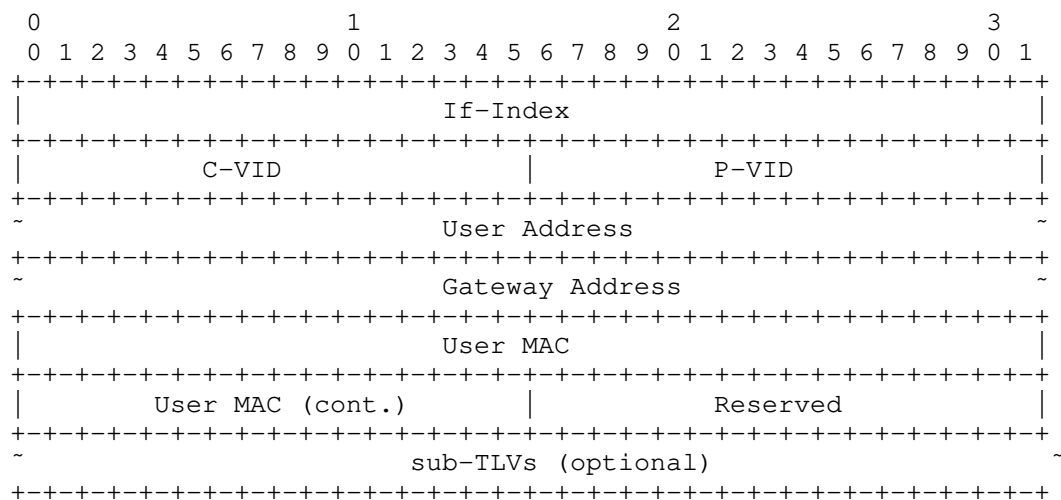


Figure 51: IPv6 Static Subscriber Detect TLV

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:

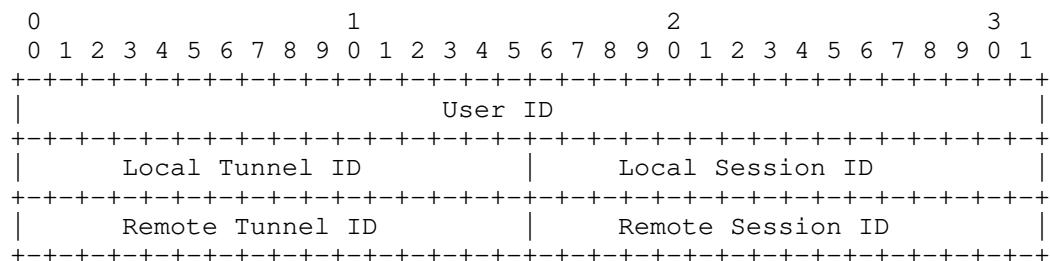


Figure 52: L2TP-LAC Subscriber TLV

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:

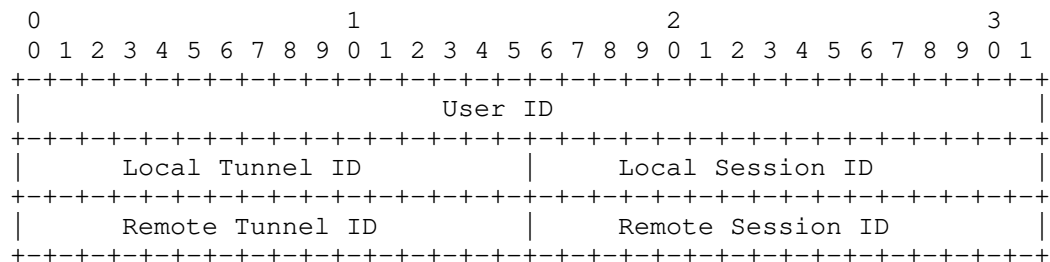


Figure 53: L2TP-LNS Subscriber TLV

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:

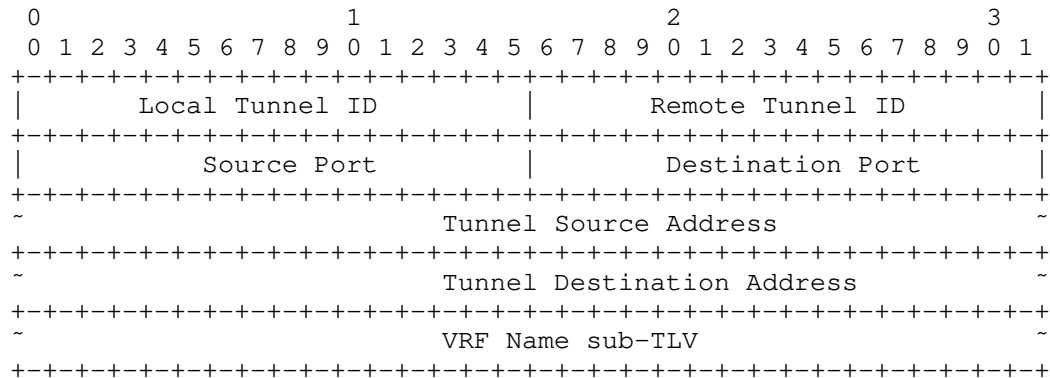


Figure 54: L2TP-LAC Tunnel TLV

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:

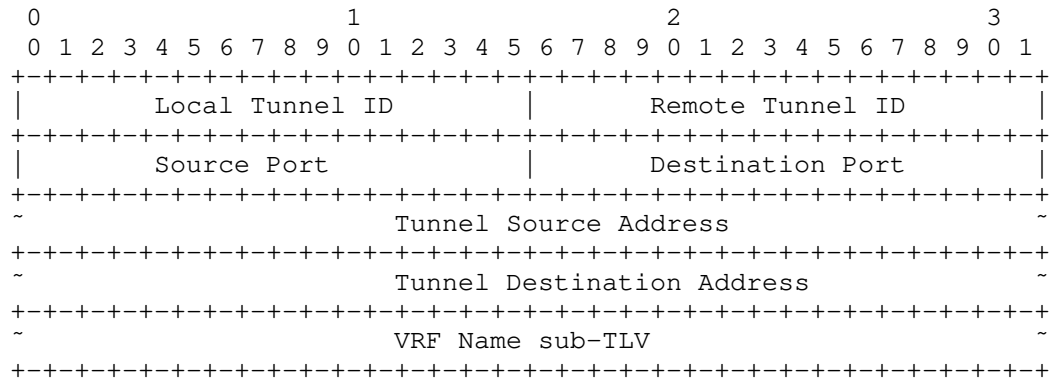


Figure 55: L2TP-LNS Tunnel TLV

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:

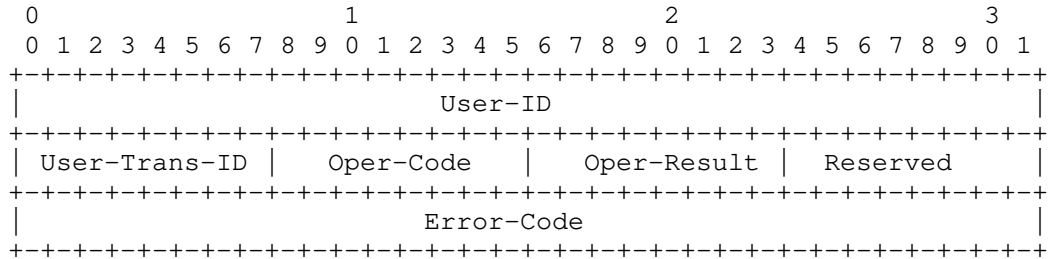


Figure 56: Update Response TLV

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:

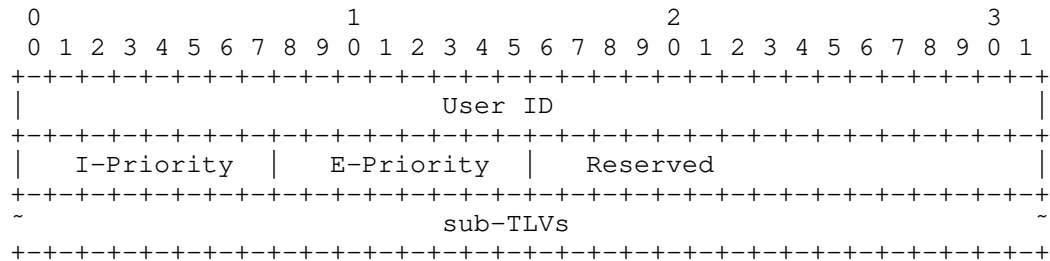


Figure 57: User QoS Policy Information TLV

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:

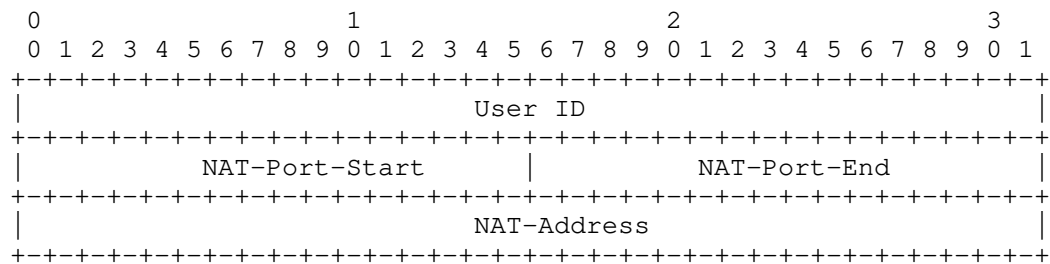


Figure 58: Subscriber CGN Port Range TLV

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:

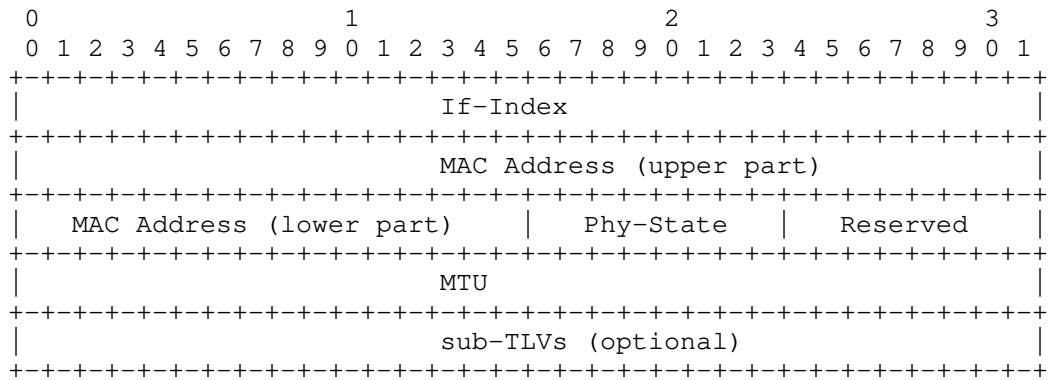


Figure 59: Interface Status TLV

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:

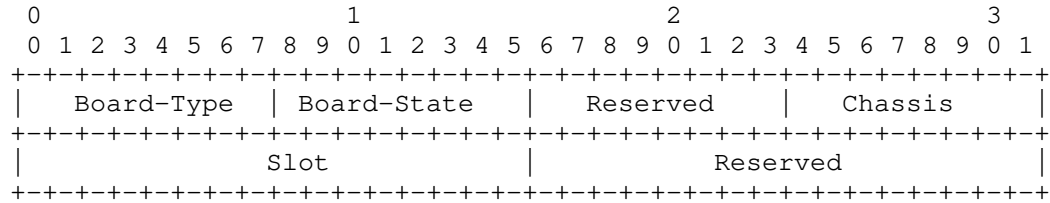


Figure 60: Interface Resource TLV

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:

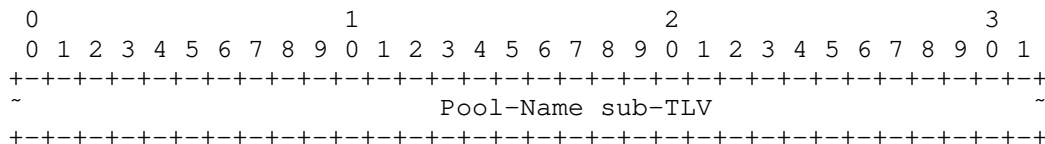


Figure 61: Address Allocation Request TLV

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:

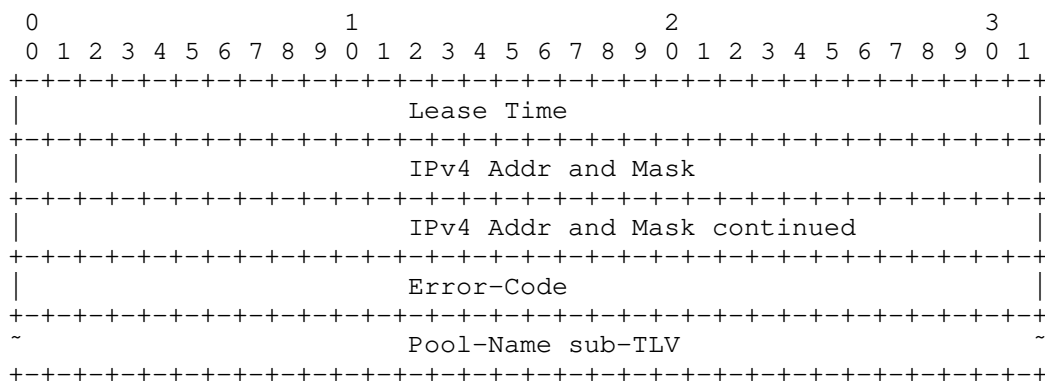


Figure 62: Address Assignment Response TLV

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:

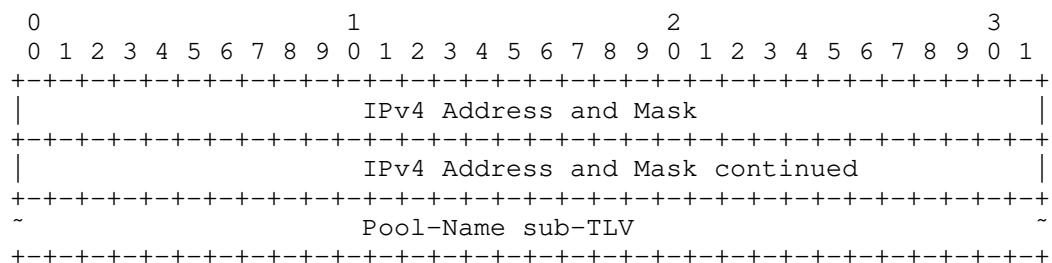


Figure 63: Address Renewal Request TLV

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:

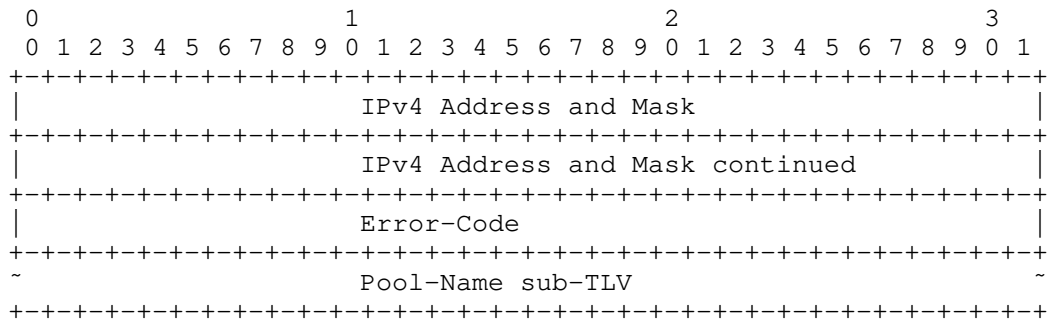


Figure 64: Address Renewal Response TLV

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:

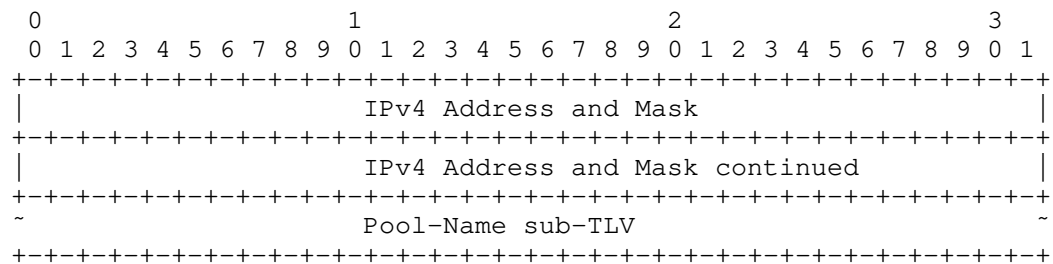


Figure 65: Address Release Request TLV

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:

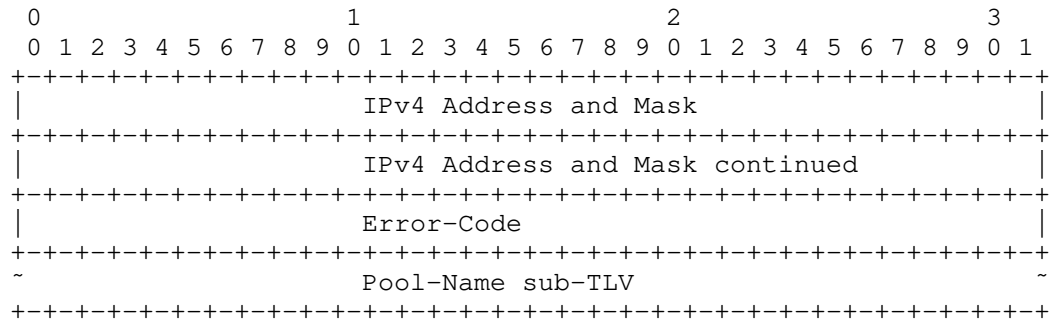


Figure 66: Address Renewal Response TLV

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:

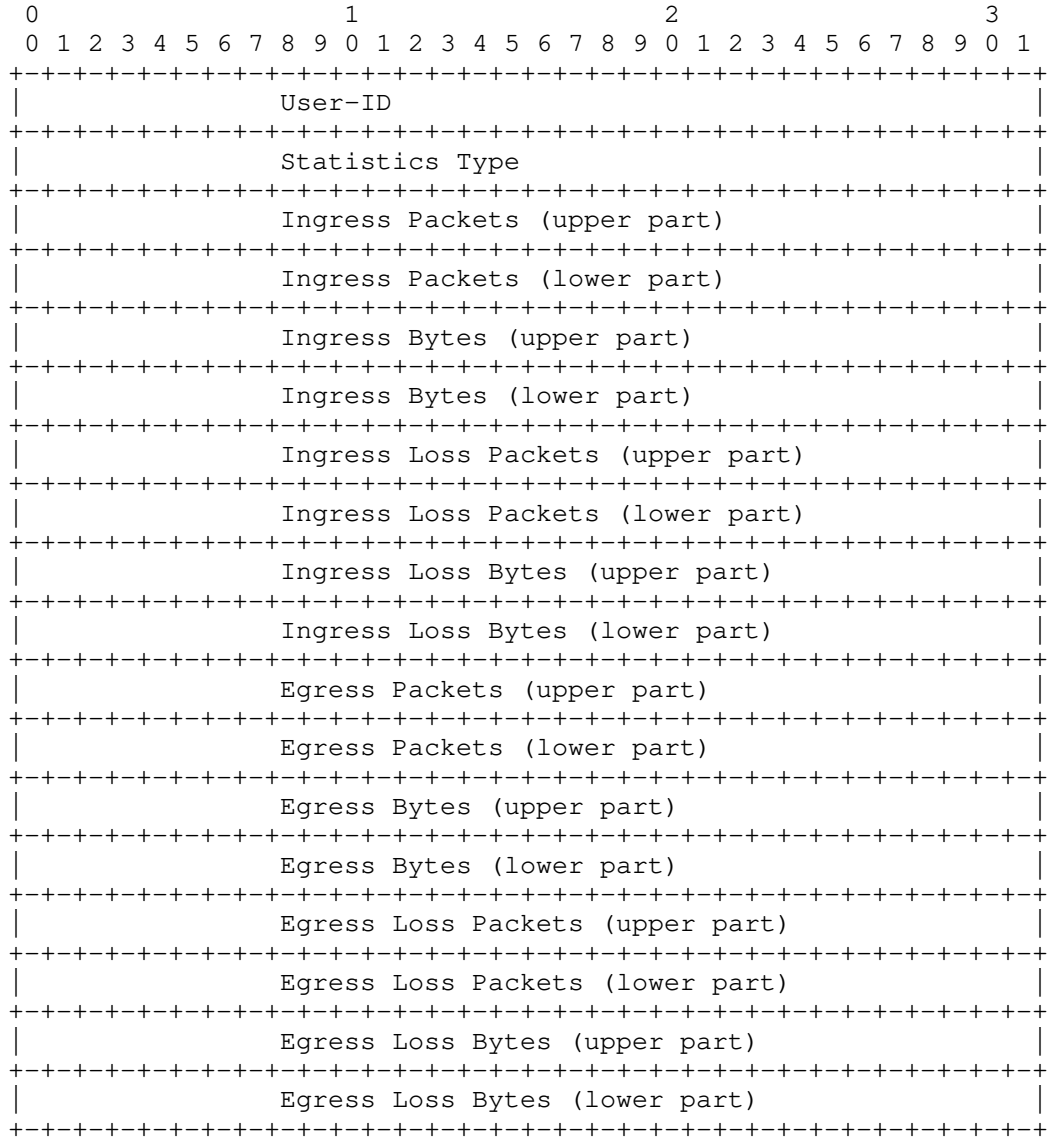


Figure 67: Subscriber Traffic Statistics TLV

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:

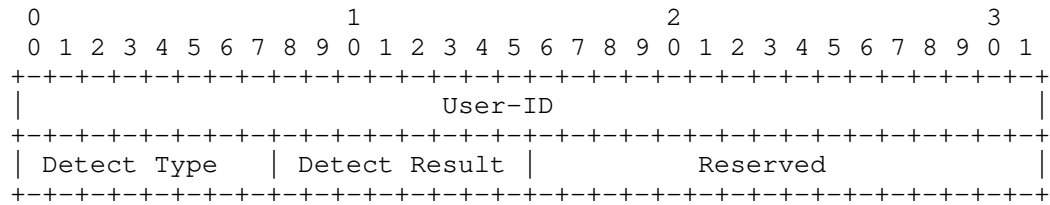


Figure 68: Subscriber Detection Result TLV

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:

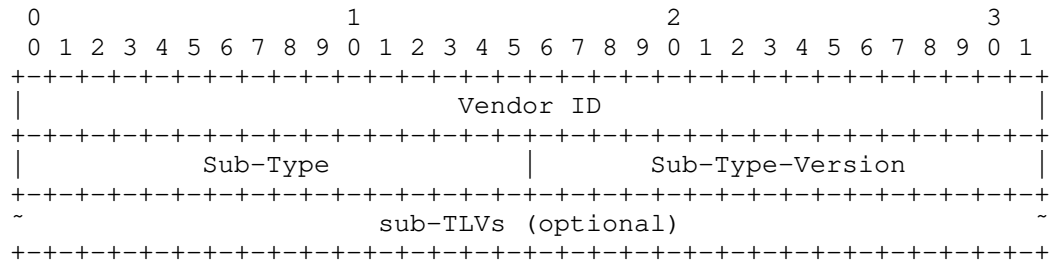


Figure 69: Vendor TLV

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.



RTGWG  
Internet-Draft  
Intended status: Standards Track  
Expires: December 30, 2018

X. Ding  
F. Zheng  
Huawei  
R. Wilton  
Cisco Systems  
June 28, 2018

YANG Data Model for ARP  
draft-ding-rtgwg-arp-yang-model-02

Abstract

This document defines a YANG data model to describe Address Resolution Protocol (ARP) configurations. The data model performs as a guideline for configuring ARP capabilities on a system. It is intended this model be used by service providers who manipulate devices from different vendors in a standard way.

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 December 30, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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. Terminology . . . . .	2
1.2. Tree Diagrams . . . . .	3
2. Problem Statement . . . . .	3
3. Design of the Data Model . . . . .	4
3.1. ARP Caching . . . . .	4
3.2. proxy ARP . . . . .	4
3.3. gratuitous ARP . . . . .	4
3.4. ietf-arp Module . . . . .	5
4. ARP YANG Module . . . . .	5
5. Data Model Examples . . . . .	12
5.1. Static ARP Entries . . . . .	12
5.2. ARP Dynamic Learning . . . . .	12
6. Security Considerations . . . . .	13
7. Acknowledgments . . . . .	14
8. References . . . . .	14
8.1. Normative References . . . . .	14
8.2. Informative References . . . . .	14
Authors' Addresses . . . . .	15

## 1. Introduction

This document defines a YANG [RFC7950] data model for Address Resolution Protocol [RFC826] implementation and identification of some common properties within a device. Devices have common properties that need to be configured and monitored in a standard way. This document is intended to present universal ARP protocol configuration and many vendors can implement it.

The data model covers configuration of system parameters of ARP, such as static ARP entries, timeout for dynamic ARP entries, interface ARP, proxy ARP, and so on. It also provides information about running state of ARP implementations.

### 1.1. 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].

The following terms are defined in [RFC6241] and are not redefined here:

- o client
- o configuration data
- o server
- o state data

## 1.2. Tree Diagrams

A simplified graphical representation of the data model is presented in Section 3.

- o Brackets "[" and "]" enclose list keys.
- o Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- o Symbols after data node names: "?" means an optional node, "!" means a presence container, and "\*" denotes a list and leaf-list.
- o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- o Ellipsis ("...") stands for contents of subtrees that are not shown.

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

## 2. Problem Statement

This document defines a YANG [RFC7950] configuration data model that may be used to configure the ARP feature running on a system. Data model "ietf-ip" [I-D.ietf-netmod-rfc7277bis] covers the address mapping functionality. However, this functionality is strictly dependent on IPv4 networks, and many ARP related functionalities are missing, e.g. device global ARP entries and control, configuration related to dynamic ARP learning, proxy ARP, gratuitous ARP, etc.

The data model makes use of the YANG "feature" construct which allows implementations to support only those ARP features that lie within their capabilities. It is intended this model be used by service providers who manipulate devices from different vendors in a standard way.

This model can be used to configure the ARP applications for discovering the link layer address associated with a given Internet layer address.

### 3. Design of the Data Model

This data model intends to describe the processing that a protocol finds the hardware address, also known as Media Access Control (MAC) address, of a host from its known IP address. These tasks include, but are not limited to, adding a static entry in the ARP cache, configuring dynamic ARP learning, proxy ARP, gratuitous ARP. There are two kind of ARP configurations: global ARP configuration, which is across all interfaces on the device, and per interface ARP configuration.

#### 3.1. ARP Caching

ARP caching is the method of storing network addresses and the associated data-link addresses in memory for a period of time as the addresses are learned. This minimizes the use of valuable network resources to broadcast for the same address each time a datagram is sent.

There are static ARP cache entries and dynamic ARP cache entries. Static entries are manually configured and kept in the cache table on a permanent basis. Dynamic entries are added by vendor software, kept for a period of time, and then removed. We can specify how long an entry remains in the ARP cache. If we specify a timeout of 0 seconds, entries are never cleared from the ARP cache.

#### 3.2. proxy ARP

Proxy ARP [RFC1027] can be configured to enable the switch to respond to ARP queries for network addresses by offering its own Ethernet media access control (MAC) address. With proxy ARP enabled, the switch captures and routes traffic to the intended destination.

#### 3.3. gratuitous ARP

Gratuitous ARP requests help detect duplicate IP addresses. A gratuitous ARP is a broadcast request for a router's own IP address. If a router or switch sends an ARP request for its own IP address and no ARP replies are received, the router- or switch-assigned IP address is not being used by other nodes. However, if a router or switch sends an ARP request for its own IP address and an ARP reply is received, the router- or switch-assigned IP address is already being used by another node.

### 3.4. ietf-arp Module

This module has one top level container, ARP, which consists of two second level containers, which are used for static entries configuration and global parameters control.

```

module: ietf-arp
  +--rw arp
    +--rw global-static-entries {global-static-entries}?
      | +--rw static-entry* [ip-address]
      |   +--rw ip-address      inet:ipv4-address-no-zone
      |   +--rw mac-address     yang:mac-address
    +--rw global-control
      +--rw enable-learning?   boolean
      +--rw enable-proxy?     boolean
  augment /if:interfaces/if:interface:
    +--rw arp-dynamic-learning
      +--rw expire-time?      yang:timeticks
      +--rw learn-disable?    boolean
      +--rw proxy
        | +--rw mode?         enumeration
      +--rw probe
        | +--rw interval?     uint8
        | +--rw times?        uint8
        | +--rw unicast?      boolean
      +--rw gratuitous
        | +--rw enable?       boolean
        | +--rw interval?     uint32
        | +--rw drop?         boolean
      +--ro statistics
        +--ro in-requests-pkts?  uint16
        +--ro in-replies-pkts?   uint16
        +--ro in-gratuitous-pkts? uint16
        +--ro out-requests-pkts?  uint16
        +--ro out-replies-pkts?   uint16
        +--ro out-gratuitous-pkts? uint16
  augment /if:interfaces/if:interface/ip:ipv4/ip:neighbor:
    +--ro remaining-expire-time?  uint32

```

### 4. ARP YANG Module

This section presents the ARP YANG module defined in this document. This YANG module imports typedefs from [RFC6991].

<CODE BEGINS>file "ietf-arp@2018-01-27.yang"

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

  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991: INET Types Model";
  }

  import ietf-yang-types {
    prefix yang;
    reference "RFC 6991: yang Types Model";
  }

  import ietf-interfaces {
    prefix if;
    description
      "A Network Management Datastore Architecture (NMDA)
       compatible version of the ietf-interfaces module
       is required.";
  }
  import ietf-ip {
    prefix ip;
    description
      "A Network Management Datastore Architecture (NMDA)
       compatible version of the ietf-ip module is
       required.";
  }

  organization
    "IETF Routing Area Working Group (rtgwg)";
  contact
    "WG Web: <http://tools.ietf.org/wg/rtgwg/>
     WG List: <mailto:rtgwg@ietf.org>
     Editor: Xiaojian Ding
             dingxiaojian1@huawei.com
     Editor: Feng Zheng
             habby.zheng@huawei.com
     Editor: Robert Wilton
             rwilton@cisco.com";
  description
    "Address Resolution Protocol (ARP) management, which includes
     static ARP configuration, dynamic ARP learning, ARP entry query,
     and packet statistics collection.";

  revision 2018-01-27 {
    description
```

```
    "Init revision";
    // NOTE TO RFC EDITOR:
    // Please replace the following reference
    // to draft-ding-rtgwg-arp-yang-model-02 with
    // RFC number when published (i.e. RFC xxxx).
reference
    "draft-ding-rtgwg-arp-yang-model-02";
}

/*
 * Features
 */

feature global-static-entries {
description
    "This feature indicates that the device allows static entries
    to be configured globally.";
}

container arp {
description
    "Address Resolution Protocol (ARP) management, which includes
    static ARP configuration, dynamic ARP learning, ARP entry
    query, and packet statistics collection.";

container global-static-entries {
    if-feature "global-static-entries";
description
    "Set a global static ARP entry, which is independent of the interface.";
    list static-entry {
        key "ip-address";
description
        "List of ARP static entries that can be configured globally.";
        leaf ip-address {
            type inet:ipv4-address-no-zone;
description
                "IP address, in dotted decimal notation.";
        }
        leaf mac-address {
            type yang:mac-address;
            mandatory true;
description
                "MAC address in the format of H-H-H, in which H is
                a hexadecimal number of 1 to 4 bits.";
        }
    }
}
}
```

```
container global-control {
  description
    "Set global control parameters, which are independent of interface.";
  leaf enable-learning {
    type boolean;
    default "true";
    description
      "Enables or disables global dynamic ARP learning.
       If 'true', then enforcement is enabled.
       If 'false', then enforcement is disabled.";
  }
  leaf enable-proxy {
    type boolean;
    default "true";
    description
      "Proxy ARP is enabled by default; perform this
       task to globally disable proxy ARP on all interfaces.";
  }
}

augment "/if:interfaces/if:interface" {
  description
    "Augment interface configuration with parameters of ARP.";
  container arp-dynamic-learning {
    description
      "Support for ARP configuration on interfaces.";
    leaf expire-time {
      type yang:timeticks {
        range "60..86400";
      }
      units "second";
      description
        "Aging time of a dynamic ARP entry.";
    }
    leaf learn-disable {
      type boolean;
      default "false";
      description
        "Whether dynamic ARP learning is disabled on an interface.
         If the value is True, dynamic ARP learning is disabled.
         If the value is False, dynamic ARP learning is enabled.";
    }
  }

  container proxy {
    description
      "Configuration parameters for proxy ARP";
    leaf mode {
      type enumeration {
```

```

        enum DISABLE {
            description
                "The system should not respond to ARP requests t
hat
                do not specify an IP address configured on the l
ocal
                subinterface as the target address.";
        }
        enum REMOTE_ONLY {
            description
                "The system responds to ARP requests only when t
he
                sender and target IP addresses are in different
                subnets.";
        }
        enum ALL {
            description
                "The system responds to ARP requests where the s
ender
                and target IP addresses are in different subnets
, as well
                as those where they are in the same subnet.";
        }
    }
    default "DISABLE";
    description
        "When set to a value other than DISABLE, the local syste
m should
        respond to ARP requests that are for target addresses ot
her than
        those that are configured on the local subinterface usin
g its own
        MAC address as the target hardware address. If the REMOT
E_ONLY
        value is specified, replies are only sent when the targe
t address
        falls outside the locally configured subnets on the inte
rface,
        whereas with the ALL value, all requests, regardless of
their
        target address are replied to.";
    reference "RFC1027: Using ARP to Implement Transparent Subnet
Gateways";
}
}

container probe {
    description
        "Common configuration parameters for all ARP probe.";
    leaf interval {
        type uint8 {
            range "1..5";
        }
        units "second";
        description
            "Interval for detecting dynamic ARP entries.";
    }
    leaf times {
        type uint8 {
            range "0..10";
        }
    }
}

```





```
    description
      "Number of aging probe attempts for a dynamic ARP entry.
      If a device does not receive an ARP reply message after
      the number of aging probe attempts reaches a specified
      number, the dynamic ARP entry is deleted.";
  }
  leaf unicast {
    type boolean;
    default "false";
    description
      "Send unicast ARP aging probe messages for a dynamic ARP
      entry.";
  }
}

container gratuitous {
  description
    "Configure gratuitous ARP.";
  leaf enable {
    type boolean;
    default "false";
    description
      "Enable or disable sending gratuitous-arp packet on
      interface.";
  }
  leaf interval {
    type uint32 {
      range "1..86400";
    }
    units "second";
    description
      "The interval of sending gratuitous-arp packet on the
      interface.";
  }
  leaf drop {
    type boolean;
    default "false";
    description
      "Drop the receipt of gratuitous ARP packets on the interface.";
  }
}

container statistics {
  config false;
  description
    "IP ARP Statistics information on interfaces";
  leaf in-requests-pkts {
    type uint16;
```

```
        description
            "Total ARP requests received";
    }
    leaf in-replies-pkts {
        type uint16;
        description
            "Total ARP replies received";
    }
    leaf in-gratuitous-pkts {
        type uint16;
        description
            "Total gratuitous ARP received";
    }
    leaf out-requests-pkts {
        type uint16;
        description
            "Total ARP requests sent";
    }
    leaf out-replies-pkts {
        type uint16;
        description
            "Total ARP replies sent";
    }
    leaf out-gratuitous-pkts {
        type uint16;
        description
            "Total gratuitous ARP sent";
    }
    }
}

augment "/if:interfaces/if:interface/ip:ipv4/ip:neighbor" {
    description
        "Augment neighbor list with parameters of ARP,
        eg., support for remaining expire time query on interfaces.";
    leaf remaining-expire-time {
        type uint32;
        config false;
        description
            "Remaining expire time of a dynamic ARP entry. ";
    }
}

}
```

## 5. Data Model Examples

This section presents a simple but complete example of configuring static ARP entries and dynamic learning, based on the YANG modules specified in Section 4.

### 5.1. Static ARP Entries

Requirement:

Enable static ARP entry global configuration (not rely on interface).

```
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <arp xmlns="urn:ietf:params:xml:ns:yang:ietf-arp">
    <static-tables>
      <ip-address> 10.2.2.3 </ip-address>
      <mac-address> 00e0-fc01-0000 </mac-address>
    </static-tables>
  </arp>
```

Requirement:

Enable static ARP entry configuration on interface (defined in draft [I-D.ietf-netmod-rfc7277bis]).

```
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
    <neighbor>
      <ip-address> 10.2.2.3 </ip-address>
      <mac-address> 00e0-fc01-0000 </mac-address>
      <if-name> GE1/0/1 </if-name>
    </neighbor>
  </ipv4>
```

### 5.2. ARP Dynamic Learning

## Requirement:

Enable ARP dynamic learning configuration.

```
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <arp-dynamic-learning xmlns="urn:ietf:params:xml:ns:yang:ietf-arp-dynamic-
learning">
    <if-name> GE1/0/1 </if-name>
    <expire-time>1200</expire-time>
    <learn-disable>false</learn-disable>
    <proxy>
      <mode>DISABLE</mode>
    </proxy>
    <probe>
      <interval>5</interval>
      <times>3</times>
      <unicast>false</unicast>
    </probe>
    <gratuitous>
      <gratuitous-enable>false</gratuitous-enable>
      <interval>60</interval>
      <drop>false</drop>
    </gratuitous>
  </arp-dynamic-learning>
```

## 6. Security Considerations

The YANG module defined in this document is designed to be accessed via YANG based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. Both of these protocols have mandatory-to-implement secure transport layers (e.g., SSH, TLS) with mutual authentication.

The NETCONF access control model (NACM) [RFC6536] provides the means to restrict access for particular users to a pre-configured subset of all available protocol operations and content.

These are the subtrees and data nodes and their sensitivity/vulnerability:

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

## 7. Acknowledgments

The authors wish to thank Alex Campbell and Reshad Rahman, Qin Wu, many others for their helpful comments.

## 8. References

### 8.1. Normative References

- [I-D.ietf-netmod-rfc7223bis]  
Bjorklund, M., "A YANG Data Model for Interface Management", draft-ietf-netmod-rfc7223bis-03 (work in progress), January 2018.
- [I-D.ietf-netmod-rfc7277bis]  
Bjorklund, M., "A YANG Data Model for IP Management", draft-ietf-netmod-rfc7277bis-03 (work in progress), January 2018.
- [RFC1027] Carl-Mitchell, S. and J. Quarterman, "Using ARP to implement transparent subnet gateways", RFC 1027, DOI 10.17487/RFC1027, October 1987, <<https://www.rfc-editor.org/info/rfc1027>>.
- [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>>.
- [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>>.

### 8.2. Informative References

- [RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <<https://www.rfc-editor.org/info/rfc826>>.

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

## Authors' Addresses

Xiaojian Ding  
Huawei  
101 Software Avenue, Yuhua District  
Nanjing, Jiangsu 210012  
China

Email: [dingxiaojian1@huawei.com](mailto:dingxiaojian1@huawei.com)

Feng Zheng  
Huawei  
101 Software Avenue, Yuhua District  
Nanjing, Jiangsu 210012  
China

Email: [habby.zheng@huawei.com](mailto:habby.zheng@huawei.com)

Robert Wilton  
Cisco Systems

Email: [rwilton@cisco.com](mailto:rwilton@cisco.com)

Network Working Group  
Internet Draft  
Intended status: Informational  
Expires: August 6, 2019

L. Dunbar  
A. Malis  
Huawei  
C. Jacquenet  
Orange  
February 6, 2019

Gap Analysis of Interconnecting Underlay with Cloud Overlay  
draft-dm-net2cloud-gap-analysis-04

Abstract

This document analyzes the technological gaps when using SD-WAN to interconnect workloads & apps hosted in various locations, especially cloud data centers when the network service providers do not have or have limited physical infrastructure to reach the locations [Net2Cloud-problem].

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 July 6, 2019.



## Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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. Conventions used in this document.....	3
3. Gap Analysis of C-PEs Registration Protocol.....	4
4. Gap Analysis in aggregating VPN paths and Internet paths.....	5
4.1. Gap analysis of Using BGP to cover SD-WAN paths.....	6
4.2. Gaps in preventing attacks from Internet facing ports.....	9
5. Gap analysis of CPEs not directly connected to VPN PEs.....	10
5.1. Gap Analysis of Floating PEs to connect to Remote CPEs...	11
5.2. NAT Traversal.....	12
5.3. Complication of using BGP between PEs and remote CPEs via Internet.....	12
5.4. Designated Forwarder to the remote edges.....	13
5.5. Traffic Path Management.....	13
6. Manageability Considerations.....	14
7. Security Considerations.....	14
8. IANA Considerations.....	14
9. References.....	14
9.1. Normative References.....	15
9.2. Informative References.....	15
10. Acknowledgments.....	16

## 1. Introduction

[Net2Cloud-Problem] describes the problems that enterprises face today in transitioning their IT infrastructure to support digital

economy, such as connecting enterprises' branch offices to dynamic workloads in different Cloud DCs.

This document analyzes the technological gaps to interconnect dynamic workloads & apps hosted in various locations and in Cloud DCs that the enterprise existing VPN service provider might not have or have limited the physical infrastructure to reach. When enterprise' VPN service providers do not have or have insufficient bandwidth to reach a location, SD-WAN is emerged as way to aggregate bandwidth of multiple networks, such as MPLS VPN, Public Internet, etc. This document primarily focuses on the technological gaps of SD-WAN.

For ease of description, SD-WAN edge, SD-WAN end points, C-PE, or CPE are used interchangeably throughout this document.

## 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 SD-WAN controller to manage SD-WAN overlay path creation/deletion and monitor the path conditions between sites.

**CPE-Based VPN:** Virtual Private Secure network formed among CPEs. This is to differentiate from most commonly used PE-based VPNs a la RFC 4364.

**OnPrem:** On Premises data centers and branch offices

**SD-WAN:** Software Defined Wide Area Network. In this document, "SD-WAN" refers to the solutions specified by ONUG (Open Network User Group), <https://www.onug.net/software-defined-wide-area-network-sd-wan/>, which is about pooling WAN bandwidth from multiple underlay networks to get better WAN bandwidth management, visibility & control. When the underlay networks are private

networks, traffic can traverse without additional encryption; when the underlay networks are public, such as Internet, some traffic needs to be encrypted when traversing through (depending on user provided policies).

### 3. Gap Analysis of C-PEs Registration Protocol

SD-WAN, conceived in ONUG (Open Network User Group) a few years ago as a means to aggregate multiple connections between any two points, has emerged as an on-demand technology to securely interconnect the OnPrem branches with the workloads instantiated in Cloud DCs that do not connect to BGP/MPLS VPN PEs or have very limited bandwidth.

Some SD-WAN networks use the NHRP protocol [RFC2332] to register SD-WAN edges with a "Controller" (or NHRP server), which then has the ability to map a private VPN address to a public IP address of the destination node. DSVPN [DSVPN] or DMVPN [DMVPN] are used to establish tunnels among SD-WAN edge nodes.

NHRP was originally intended for ATM address resolution, and as a result, it misses many attributes that are necessary for dynamic endpoint C-PE registration to controller, such as:

- Interworking with MPLS VPN control plane. A SD-WAN edge can have some ports facing MPLS VPN network and some ports facing public Internet that requires encryption for some sensitive data to traverse.
- Scalability. NHRP/DSVPN/DMVPN works fine with small number of edge nodes. When a network has more than 100 nodes, the protocol does not work well.
- NHRP does not have the IPsec attributes, which are needed for peers to build Security Associations over public internet.
- NHRP does not have field to indicate C-PE supported encapsulation types, such as IPsec-GRE, IPsec-VxLAN, or others.
- NHRP does not have field to indicate C-PE Location identifier, such as Site Identifier, System ID, and/or Port ID.
- NHRP does not have field to describe the gateway to which the C-PE is attached. When a C-PE is instantiated in a Cloud DC, to

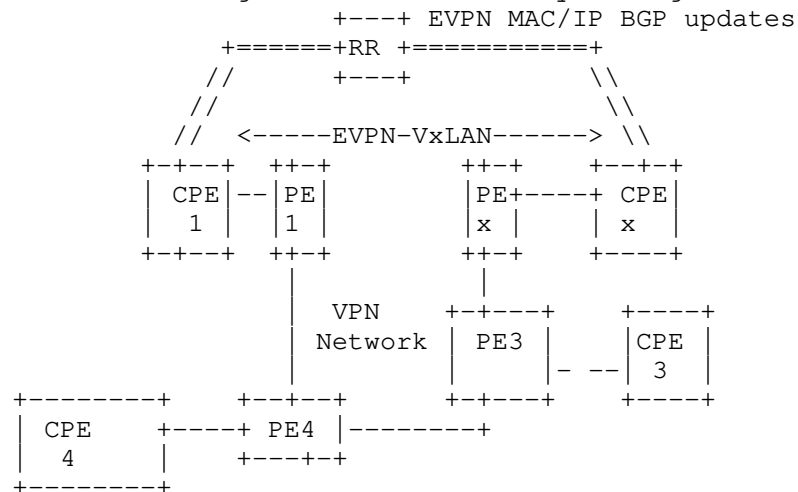
establish connection to the C-PE, it is necessary to know the Cloud DC operator's Gateway to which the CPE is attached.

- NHRP does not have field to describe C-PE's NAT properties if the C-PE is using private addresses, such as the NAT type, Private address, Public address, Private port, Public port, etc.

[BGP-SDWAN-EXT] describes how to use BGP for SD-WAN edge nodes to register its properties to SD-WAN controller, which then disseminates the information to other SD-WAN edge nodes that are authenticated to communicate.

#### 4. Gap Analysis in aggregating VPN paths and Internet paths

Most likely, enterprises especially large ones already have their CPEs interconnected by providers' VPNs, such as EVPN, L2VPN, or L3VPN. The VPN can be PE based or CPE based as shown in the following diagram. The commonly used CPE-based VPNs have CPE directly attached to PEs, therefore the communication is considered as secure. BGP are used to distribute routes among CPEs, even though sometimes routes among CPEs are statically configured.



=== or \\ indicates control plane communications

Figure 1: L2 or L3 VPNs over IP WAN

To use SD-WAN to aggregate Internet routes with the VPN routes, the C-PEs need to have some ports connected to PEs and other ports connected to the Internet. It is necessary to have a registration protocol for C-PEs to register with their SD-WAN Controllers to establish secure tunnels among relevant C-PEs.

If using NHRP for registration, C-PEs need to participate in two separate control planes: EVPN&BGP for CPE-based VPNs via links directly attached to PEs and NHRP & DSVPN/DMVPN for ports connected to internet. Two separate control planes not only add complexity to C-PEs, but also increase operational cost.

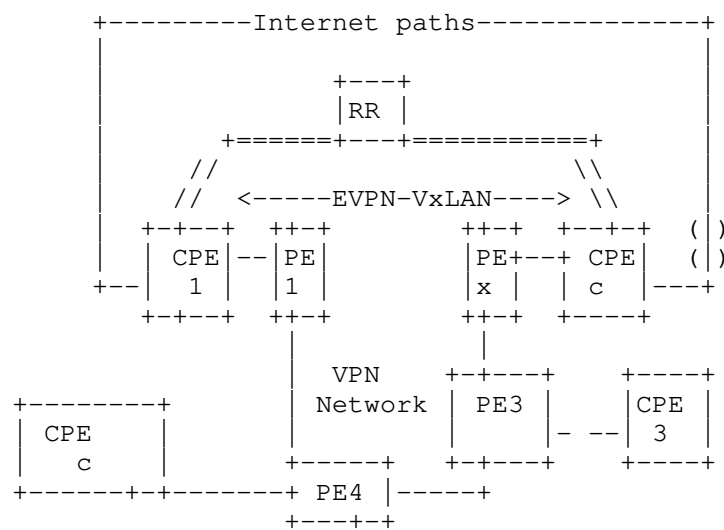


Figure 2: CPEs interconnected by VPN paths and Internet Paths

#### 4.1. Gap analysis of Using BGP to cover SD-WAN paths

Since C-PE already supports BGP, it is desirable to consider using BGP to control the SD-WAN instead of two separate control planes. This section analyzes the gaps of using BGP to control SD-WAN.

As described [BGP-SDWAN-Usage], SD-WAN Overlay Control Plane has three distinct functional tiers:

- SD-WAN node's private address and WAN Ports/Addresses registration to the SD-WAN Controller.

- o It is for informing the SD-WAN controller and potential peers of the underlay networks to which the C-PE is connected.
- Controller Facilitated IPsec SA management and NAT information distribution
  - o It is for SD-WAN controller to facilitate or manage the IPsec configurations and peer authentications for all IPsec tunnels terminated at the SDWAN nodes. For some scenarios where the WAN ports are private addresses, this step is for informing the type of NAT translating the private addresses to public ones.
- Establishing and Managing the topology and reachability for services attached to the client ports of SD-WAN nodes.
  - o This is for the overlay layer's routes distribution, so that a C-PE can establish the overlay routing table that identifies the next hop for reaching a specific route/service attached to remote nodes. [SECURE-EVPN] describes EVPN and other options.

RFC5512 and [Tunnel-Encap] describe methods for endpoints to advertise tunnel information and to trigger tunnel establishment. RFC5512 & [Tunnel-Encap] have the Endpoint Address to indicate IPv4 or IPv6 address format, the Tunnel Encapsulation attribute to indicate different encapsulation formats, such as L2TPv3, GRE, VxLAN, IP-in-IP, etc. There are sub-TLVs to describe the detailed tunnel information for each of the encapsulations.

[Tunnel-Encap] removed SAFI =7 (which was specified by RFC5512) for distributing encapsulation tunnel information. [Tunnel-Encap] require Tunnels being associated with routes.

There is also the Color sub-TLV to describe customer-specified information about the tunnels (which can be creatively used for SD-WAN)

Here are some of the gaps using [Tunnel-Encap] to control SD-WAN:

- Lacking C-PE Registration functionality
- Lacking IPsec Tunnel type

- [Tunnel-Encap] has Remote Address SubTLV, but does not have any field to indicate the Tunnel originating interface, which was in RFC5512.
- The mechanisms described by [Tunnel-Encap] cannot be effectively used for SD-WAN overlay network because a SD-WAN Tunnel can be between Internet facing WAN ports of two C-PEs and needs to be established before data arrival because the tunnel establishment can fail, e.g. two end points supporting different encryption algorithms.
- Client traffic (e.g. an EVPN route) can have option of going through MPLS network natively without encryption, or going through the IPsec tunnels between the internet facing WAN ports of two C-PEs.
- There is no routes to be associated with the SD-WAN Tunnel between two C-PE's internet facing WAN ports, unless consider using the interface facing WAN Port addresses assigned by ISP (Internet Service Providers) as the route for the Tunnel.  
There is a suggestion on using a "Fake Route" for a SD-WAN node to use [Tunnel-Encap] to advertise its SD-WAN tunnel end-points properties. However, using "Fake Route" can create deployment complexity for large SD-WAN networks with many tunnels. For example, for a SD-WAN network with hundreds of nodes, with each node having many ports & many end-points to establish SD-WAN tunnels to their corresponding peers, the node would need many "fake addresses". For large SD-WAN networks (such as has more than 10000 nodes), each node might need 10's thousands of "fake addresses", which is very difficult to manage and needs lots of configuration to get the nodes provisioned.
- Does not have fields to carry detailed information of the remote C-PE: such as Site-ID, System-ID, Port-ID
- Does not have the proper field to express IPsec attributes among the SD-WAN edge nodes to establish proper IPsec Security Associations.
- Does not have proper way for two peer CPEs to negotiate IPsec keys, based on the configuration sent by the Controller.
- Does not have field to indicate the UDP NAT private address <-> public address mapping
- C-PEs tend to communicate with a few other CPEs, not all the C-PEs need to form mesh connections. Without any BGP extension, many

nodes can get dumped with too much information of other nodes that they never need to communicate with.

[VPN-over-Internet] describes a way to securely interconnect C-PEs via IPsec using BGP. This method is useful, however, it still misses some aspects to aggregate CPE-based VPN routes with internet routes that interconnect the CPEs. In addition:

- The draft does not have options of C-PE having both MPLS ports and Internet ports.
- The draft assumes that 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. In SD-WAN, Zero Touch Provisioning is expected. It is not acceptable to require manual configuration.
- For RR communication with CPE, this draft only mentioned IPsec. Missing TLS/DTLS.
- The draft assumes that CPEs and RR are connected with an IPsec tunnel. With zero touch provisioning, we need an automatic way to synchronize the IPsec SA between CPE and RR. The draft assumes:
  - 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 draft hasn't addressed how to synchronize the refreshment among multiple nodes.
- IPsec usually only sends configuration parameters to two endpoints and let the two endpoints negotiate the KEY. Now we assume that RR is responsible for creating the KEY for all endpoints. When one endpoint is compromised, all other connections are impacted.

#### 4.2. Gaps in preventing attacks from Internet facing ports

When C-PEs have ports facing Internet, it brings in the security risks of potential DDoS attacks to the C-PEs from the ports facing internet.

To mitigate security risks, in addition to requiring Anti-DDoS features on C-PEs to prevent major DDoS attacks, it is necessary to have ways for C-PEs to validate traffic from remote peers to prevent spoofed traffic.



## 5. Gap analysis of CPEs not directly connected to VPN PEs

Because of the ephemeral property of the selected Cloud DCs, an enterprise or its network service provider may not have the direct links to the Cloud DCs that are optimal for hosting the enterprise's specific workloads/Apps. Under those circumstances, SD-WAN is a very flexible choice to interconnect the enterprise on-premises data centers & branch offices to its desired Cloud DCs.

However, SD-WAN paths over public Internet can have unpredictable performance, especially over long distances and across domains. Therefore, it is highly desirable to place as much as possible the portion of SD-WAN paths over service provider VPN (e.g., enterprise's existing VPN) that have guaranteed SLA to minimize the distance/segments over public Internet.

MEF Cloud Service Architecture [MEF-Cloud] also describes a use case of network operators needing to use SD-WAN over LTE or public Internet for last mile accesses where they are not present.

Under those scenarios, one or both of the SD-WAN endpoints may not directly be attached to the PEs of a VPN Domain.

Using SD-WAN to connect the enterprise existing sites with the workloads in Cloud DC, the enterprise existing sites' CPEs have to be upgraded to support SD-WAN. If the workloads in Cloud DC need to be connected to many sites, the upgrade process can be very expensive.

[Net2Cloud-Problem] describes a hybrid network approach that integrates SD-WAN with traditional MPLS-based VPNs, to extend the existing MPLS-based VPNs to the Cloud DC Workloads over the access paths that are not under the VPN provider control. To make it work properly, a small number of the PEs of the MPLS VPN can be designated to connect to the remote workloads via SD-WAN 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 in Cloud DC can be reached by the enterprise's VPN without upgrading all of the enterprise's existing CPEs. The only CPE that needs to support SD-WAN would be a virtualized CPE instantiated within the cloud DC.

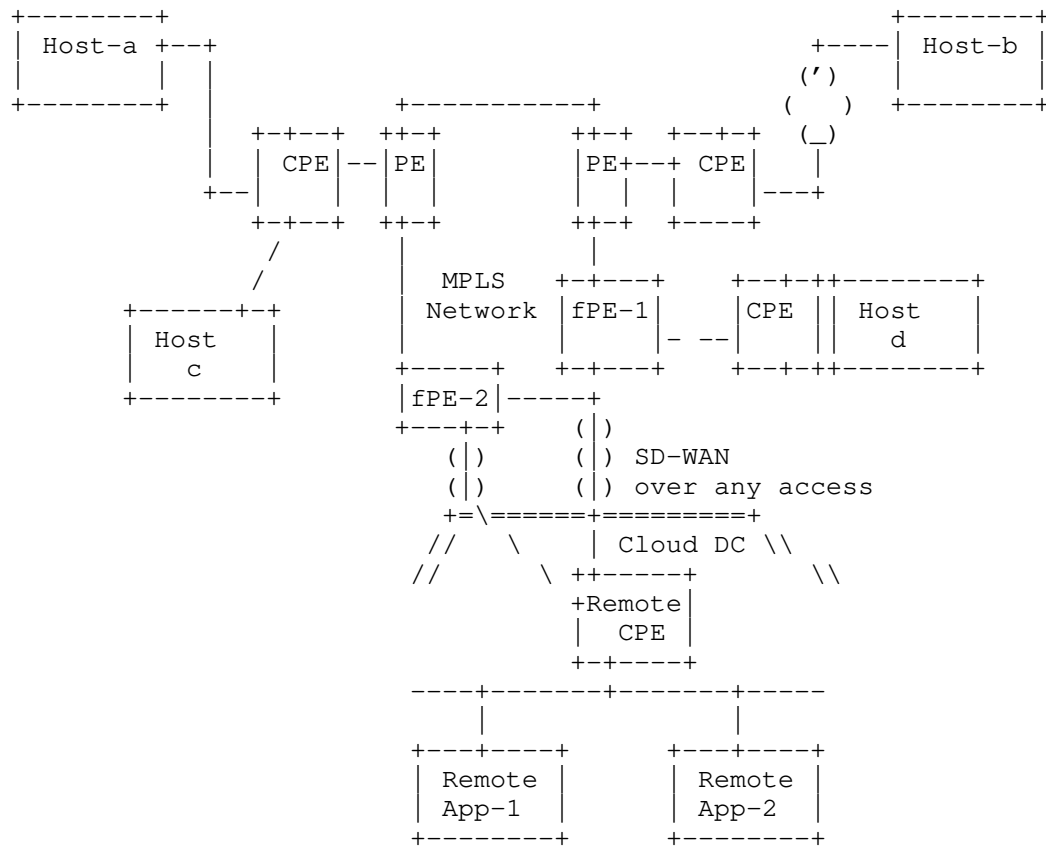


Figure 3: VPN Extension to Cloud DC

In Figure 3, the optimal Cloud DC to host the workloads (due to proximity, capacity, pricing, or other criteria chosen by the enterprises) does not happen to have a direct connection to the PEs of the MPLS VPN that interconnects the enterprise's existing sites.

#### 5.1. Gap Analysis of Floating PEs to connect to Remote CPEs

To extend MPLS VPNs to remote CPEs, it is necessary to establish secure tunnels (such as IPsec tunnels) between the Floating PEs and the remote CPEs.

Gap:

Even though a set of PEs can be manually selected to act as the floating PEs for a specific cloud data center, there are no standard protocols for those PEs to interact with the remote CPEs (most likely virtualized) instantiated in the third party cloud data centers (such as exchanging performance or route information).

When there is more than one fPE available for use (as there should be for resiliency or the ability to support multiple cloud DCs scattered geographically), it is not straightforward to designate an egress fPE to remote CPEs based on applications. There is too much applications' traffic traversing PEs, and it is not feasible for PEs to recognize applications from the payload of packets.

## 5.2. NAT Traversal

Most cloud DCs only assign private addresses to the workloads instantiated. Therefore, traffic to/from the workload usually need to traverse NAT.

A SD-WAN edge node can inquire STUN (Session Traversal of UDP Through Network Address Translation RFC 3489) Server to get the NAT property, the public IP address and the Public Port number to pass to peers.

## 5.3. Complication of using 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 complications/gaps in extending BGP from MPLS VPN PEs to remote CPEs via any access paths (e.g., Internet).

The path between the remote CPEs and VPN PE can traverse untrusted nodes.

EBGP Multi-hop scheme requires static configuration on both peers. To use EBGp between a PE and remote CPEs, the PE has to be manually configured with the "next-hop" set to the IP addresses of the CPEs. When remote CPEs, especially remote virtualized CPEs are dynamically instantiated or removed, the configuration on the PE Multi-Hop EBGp has to be changed accordingly.

Gap:

Egress peering engineering (EPE) is not enough. Running BGP on virtualized CPEs in Cloud DC requires GRE tunnels being established first, which in turn requires address and key management for the remote CPEs. RFC 7024 (Virtual Hub & Spoke) and Hierarchical VPN is not enough.

Also there is a need for a method to automatically trigger configuration changes on PE when remote CPEs' are instantiated or moved (leading to an IP address change) or deleted.

EBGP Multi-hop scheme does not have an embedded security mechanism. The PE and remote CPEs need secure communication channels when connecting via the public Internet.

Remote CPEs, if instantiated in Cloud DC, might have to traverse NAT to reach PE. It is not clear how BGP can be used between devices outside the NAT and the entities behind the NAT. It is not clear how to configure the Next Hop on the PEs to reach private IPv4 addresses.

#### 5.4. Designated Forwarder to the remote edges

Among multiple floating PEs available for a remote CPE, multicast traffic from the remote CPE towards the MPLS VPN can be forwarded back to the remote CPE due to the PE receiving the multicast data frame forwarding the multicast/broadcast frame to other PEs that in turn send to all attached CPEs. This process may cause traffic loop.

Therefore, it is necessary to designate one floating PE as the CPE's Designated Forwarder, similar to TRILL's Appointed Forwarders [RFC6325].

Gap: the MPLS VPN does not have features like TRILL's Appointed Forwarders.

#### 5.5. Traffic Path Management

When there are multiple floating PEs that have established IPsec tunnels to the remote CPE, the remote CPE can forward the outbound traffic to the Designated Forwarder PE, which in turn forwards the

traffic to egress PEs to the destinations. However, it is not straightforward for the egress PE to send back the return traffic to the Designated Forwarder PE.

Example of Return Path management using Figure 3 above.

- fPE-1 is DF for communication between App-1 <-> Host-a due to latency, pricing or other criteria.
- fPE-2 is DF for communication between App-1 <-> Host-b.

## 6. Manageability Considerations

Zero touch provisioning of SD-WAN edge nodes is expected in SD-WAN deployment. It is necessary for a newly powered up SD-WAN edges to establish a secure connection (such as TLS, DTLS, etc.) to its controller.

## 7. Security Considerations

The intention of this draft is to identify the gaps in current and proposed SD-WAN approaches that can address requirements identified in [Net2Cloud-problem].

Several of these approaches have gaps in meeting enterprise security requirements when tunneling their traffic over the Internet, as is the general intention of SD-WAN. See the individual sections above for further discussion of these security gaps.

## 8. IANA Considerations

This document requires no IANA actions. RFC Editor: Please remove this section before publication.

## 9. References

## 9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 9.2. 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-SDWAN-EXT] L. Dunbar, "BGP Extension for SDWAN Overlay Networks", draft-dunbar-idr-bgp-sdwan-overlay-ext-00, Oct 2018.
- [BGP-SDWAN-Usage] L. Dunbar, et al, "Framework of Using BGP for SDWAN Overlay Networks", draft-dunbar-idr-sdwan-framework-00, work-in-progress, Feb 2019.
- [Tunnel-Encap] E. Rosen, et al, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-10, July 2018.
- [VPN-over-Internet] 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

## 10. Acknowledgments

Acknowledgements to xxx for his review and contributions.

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Linda Dunbar  
Huawei  
Email: Linda.Dunbar@huawei.com

Andrew G. Malis  
Huawei  
Email: agmalis@gmail.com

Christian Jacquenet  
Orange  
Rennes, 35000  
France  
Email: Christian.jacquenet@orange.com





Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 3, 2019

Z. Hu  
G. Yan  
J. Yao  
Huawei Technologies  
July 2, 2018

Network Automatic Optimization Based on Dynamic Adjustment of IGP  
Metrics  
draft-hu-lsr-network-automatic-optimization-00

## Abstract

The centralized traffic engineering technology adopts centralized calculation, PCE/SDN performs centralized calculation based on the collected link-status and TE information collected by BGP-LS/IGP, so that the traffic in the network is optimized to the optimization goal.

The distributed traffic engineering technology uses IGP to calculate the shortest path. Each node in the network calculates the next hop to a destination address based on the same algorithm and network topology. Each algorithm can calculate a shortest path tree in the entire network, which can reduce the amount of calculation, so that the hard convergence time of TE can be close to the convergence time of IGP.

The distributed computing and management methods can not co-ordinate management of network bandwidth resources. As a result, network bandwidth resources can not be planned in an integrated manner, bandwidth resources are wasted, and even tunnels cannot be established. This document attempts to discuss a automatic network optimization function of the distributed traffic engineering technology by the method of dynamically adjusting the link Metric.

## 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 January 3, 2019.

#### Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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. Automatic Network Optimization Method . . . . .	3
3. Optimization Rules . . . . .	5
3.1. Threshold Regulation . . . . .	5
3.2. Optimization Rules . . . . .	5
4. Controller Adjustment Scenario . . . . .	6
5. IANA Considerations . . . . .	8
6. Security Considerations . . . . .	8
7. Acknowledgements . . . . .	8
8. References . . . . .	8
8.1. Normative References . . . . .	8
8.2. Informative References . . . . .	8
Authors' Addresses . . . . .	9

#### 1. Introduction

The concept of IGP Metric defined in [RFC1195] for ISIS, and defined in [RFC2328] and [RFC5340] for OSPF, indicating the cost of using this router link.

Segment Routing (SR) described in [I-D.ietf-spring-segment-routing] leverages the source routing paradigm. Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane [I-D.ietf-spring-segment-routing-mpls] and applied to the IPv6 architecture with a new type of routing header called the SR header (SRH) [I-D.ietf-6man-segment-routing-header]. Segment Routing uses the IGP protocol as the control protocol. The distributed traffic engineering technology has unique advantages in solving the problem of SRv6's label stack depth, in rapid convergence and self-healing.

The distributed traffic engineering technology adopts the distributed route calculation method. However, because the IGP calculates the shortest path based on the SPF algorithm and the metric of the link is constant, it is possible that a certain segment of the optimal path of multiple services is the same segment of the link. In a multi-service scenario, it is easy to cause congestion on some network links while the bandwidth resources on other links are idle. The resources of the entire network cannot be fully utilized.

This document proposes a method for automatic optimization when traffic flow is congested by dynamically adjusting the link metric.

Section 2 describes the implementation of automatic network optimization in distributed traffic engineering technology. Section 3 defines the thresholds and optimization rules of automatic network optimization. In Section 4, based on the dynamic adjustment of link metric technology, the network optimization method for controller weak control management is introduced.

## 2. Automatic Network Optimization Method

The distributed traffic engineering technology is based on the IGP protocol. Each node uses the SPF algorithm to calculate the shortest path to the destination node based on the entire network topology. Each link calculates the shortest path based on the same metric for all services. As the services in the network increase, some links in the network may be seriously congested, but some links are still underutilized. This causes a waste of network resources.

Consider a method that, when a link is congested, dynamically adjusts the metric of the link so that traffic of a part of the service can be switched to other paths for transmission, thereby reducing the load of the congested link. Through the adjustment of metrics of different links in the network, different priority services calculate different shortest paths based on different metrics of the same link in the same network, thereby avoiding network congestion on certain links and increasing Utilization of network resources.

The automatic network optimization method divides the metric of the link according to the SLA type into multiple types corresponding to the SLA type (For example divided into Bandwidth Metric, Latency Metric...etc.), Each type of Metric is divided into several metrics with different priorities. In the initial case, each type of metric is defined as the priority of user static planning, and the values of the priority metrics are the same. When traffic is forwarded, the services carried on the network are first mapped to different types of different priority metrics according to the SLA type and the service priority. When the network device calculates the forwarding path of the service, the shortest path is calculated using the mapped metric of the service as a metric, and the service is forwarded according to the calculated shortest path.

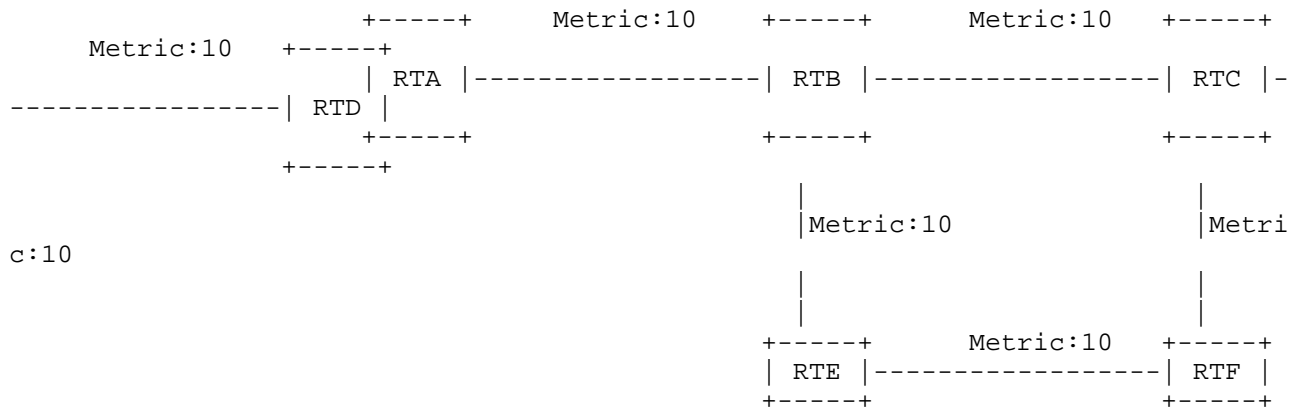


Figure 1. Network Topology

Assume that there are four types of services of the same type and different priorities: service 1, service 2, service 3, service 4 (decrease of priority), Business traffic needs to be transmitted from device RTA to RTD. The four services are mapped to different types of different priority metrics according to the SLA type and service priorities. In this example, four services are mapped to BW Metric 1, BW Metric 2, BW Metric 3, and BW Metric 4 according to the SLA type and priority. In the initial case, the four metrics corresponding to the same link have the same value. According to the distributed traffic engineering technology, all four service traffics are forwarded according to the shortest path calculated by the SPF algorithm (RTA->RTB->RTC->RTD). If the RTB->RTC link is congested, the device RTB identifies the attributes of all services forwarded on the link RTB->RTC and starts network optimization from the lowest priority service. Specifically, the value of the BW Metric 4 mapped by the low-priority service 4 may be dynamically incrementally adjusted, for example, to 1000. At this time, the shortest path for the device RTB to calculate the service 4 to the device RTC according to the SPF algorithm is changed to: RTB->RTE->RTF->RTC. The traffic

of service 4 is forwarded from the link RTB->RTE->RTF->RTC, thereby reducing the congestion of the link RTB->RTC.

### 3. Optimization Rules

In order to prevent frequent traffic switching between the primary path and backup path, the network status may oscillate for a long period of time and cannot converge. Taking the topology of Figure 1 as an example, this paper presents a threshold regulation method and some optimization rules.

#### 3.1. Threshold Regulation

In order to prevent network oscillation caused by frequent adjustment of Metric, we SHOULD set a threshold for network bandwidth utilization and maintenance time.

The bandwidth occupancy rate is set two thresholds respectively: Overlimit threshold V1 and recovery threshold V2.

To prevent network turbulence, we also SHOULD set two time thresholds : Overtime maintenance time T1 and recovery maintenance time T2.

Overlimit threshold V1 and recovery threshold V2 can be set according to the actual situation of the network. For example, the Overlimit threshold V1 is set to 80%, and the recovery threshold V2 is set to 20%, which prevents congestion and idle of network links. The overtime maintenance time T1 and the recovery maintenance time T2 are configured according to the actual situation of the network, which can be set to the same value, and can be different by the network planner.

#### 3.2. Optimization Rules

According to the threshold defined in Section 3.1, the following conditions may be encountered in the process of network optimization:

1. The device RTB determines that the link traffic of RTB->RTC is congested, the bandwidth occupancy rate is higher than the threshold V1, and the congestion maintenance time exceeds the threshold time T1. The RTB adjusts the value of the BW Metric 4 mapped by the service 4 (for example, adjusting to 1000, Adjusting the Metric to a large enough value to switch traffic directly to the backup path) from the lowest priority service 4 by identifying the priority order of service traffic of the current link until the RTA->RTB link Bandwidth usage starts to fall.

2. If service 4 has been adjusted to the backup path (RTA->RTB->RTE->RTF->RTC->RTB), but the bandwidth usage of the A->B link is still higher than the overrun threshold V1, and has experienced another Maintain time T1, RTB again recognizes the priority order of the service traffic of the RTB->RTC link, and continues to dynamically adjust the value of the BW Metric 3 mapped by the second-lowest-priority service 3 until the bandwidth occupancy rate is lower than the overrun threshold V1.

3. If the BW Metric 3 value corresponding to the next-low-priority service 3 is incrementally adjusted, the bandwidth occupancy rate of the link RTB->RTC is lower than the restoration threshold V2, and after the maintenance time T2, the next-low-priority service 3 is established. The corresponding metric returns to its initial value.

4. If the BW Metric 3 value corresponding to the next lower priority service 3 is adjusted, the bandwidth occupancy rate of the backup path RTB->RTE->RTF->RTC exceeds the threshold V1 and maintains the time T1. Then, the BW Metric 3 corresponding to the next lower priority service 3 is restored to the initial value, and an alarm is printed, indicating that all the links are on the verge of overreach and need to be expanded.

5. If the Metric value of a business is adjusted, it is found that the traffic has not been switched to the backup path according to the intended purpose. It shows that there is no backup path in the link, the Metric value is restored to the initial value, and the service is waiver to adjust the business and continue to adjust the Metric value of the next business.

6. If the RTB finds that the service that has been tuned to the backup path goes back to the RTB->RTC link after experiencing a period of time, then RTB no longer adjusts the service, and keeps the service forwarded on the RTB->RTC link. This situation is usually caused because the backup path is also congested and the network optimization method is performed on the backup path.

#### 4. Controller Adjustment Scenario

Based on the distributed network optimization method for automatic adjustment of metrics of different services, the network itself has a certain ability of automatic optimization. However, If multiple services are high-priority services and have the same priority, services with the same priority are mapped to the same priority metric. With the above-mentioned automatic adjustment method, dynamic network optimization cannot be achieved. This situation requires the help of the controller. The controller only needs to intervene if the network itself cannot be adjusted.

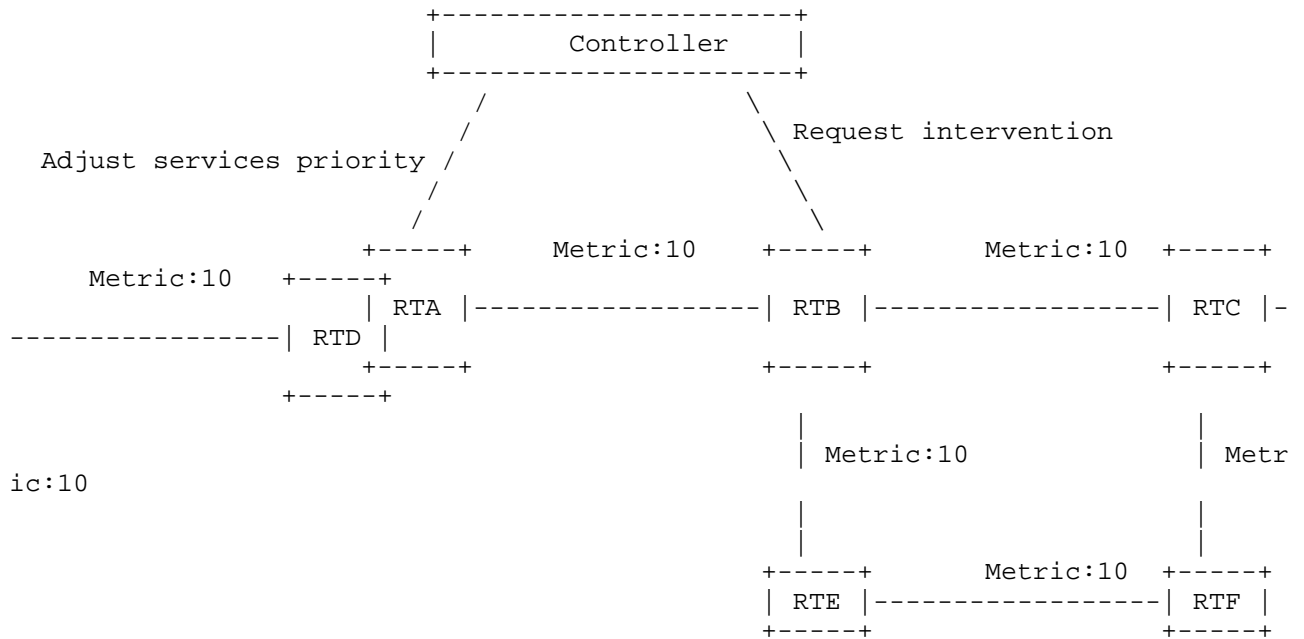


Figure 2. Controller Weak Control Management Network Optimization Method

Assume that there are four types of services of the same type and different priorities: Service 1, Service 2, Service 3, and Service 4 (the same priority). Traffic needs to be transmitted from the device RTA to the RTD. If the link RTB->RTC is congested, the device RTB recognizes that the four services have the same priority and cannot use the methods in Section 3 to dynamically adjust the network traffic. This scenario follows the following controller adjustment scenario:

1. RTB distinguishes congestion and initiates the network optimization;
2. The RTB obtains the priority of all services that congest the link RTB->RTC, and determines that the priorities are the same;
3. RTB requests intervention from the controller to adjust preference of the services;
4. The controller adjusts preference of the services and delivers it to the ingress node;
5. RTB enables the distributed network optimization method again.



## 5. IANA Considerations

TBD.

## 6. Security Considerations

TBD.

## 7. Acknowledgements

The authors of this document would like to thank Zhenbin Li, Jie Dong, Gang Yan and Peng Wu for their comments and review of this document.

## 8. References

### 8.1. Normative References

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

### 8.2. Informative References

[I-D.ietf-6man-segment-routing-header]  
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-14 (work in progress), June 2018.

[I-D.ietf-spring-segment-routing]  
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

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

[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>>.

- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.

Authors' Addresses

Zhibo Hu  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [huzhibo@huawei.com](mailto:huzhibo@huawei.com)

Gang Yan  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [yangang@huawei.com](mailto:yangang@huawei.com)

Junda Yao  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [yaojunda@huawei.com](mailto:yaojunda@huawei.com)

RTGWG  
Internet-Draft  
Intended status: Standards Track  
Expires: February 13, 2022

Y. Qu  
Futurewei  
J. Tantsura  
Microsoft  
A. Lindem  
Cisco  
X. Liu  
Volta Networks  
August 12, 2021

A YANG Data Model for Routing Policy  
draft-ietf-rtgwg-policy-model-31

Abstract

This document defines a YANG data model for configuring and managing routing policies in a vendor-neutral way. The model provides a generic routing policy framework which can be extended for specific routing protocols using the YANG 'augment' mechanism.

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 February 13, 2022.

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 . . . . .	2
1.1. Goals and approach . . . . .	3
2. Terminology and Notation . . . . .	3
2.1. Tree Diagrams . . . . .	4
2.2. Prefixes in Data Node Names . . . . .	4
3. Model overview . . . . .	5
4. Route policy expression . . . . .	6
4.1. Defined sets for policy matching . . . . .	6
4.2. Policy conditions . . . . .	7
4.3. Policy actions . . . . .	8
4.4. Policy subroutines . . . . .	9
5. Policy evaluation . . . . .	10
6. Applying routing policy . . . . .	10
7. YANG Module and Tree . . . . .	11
7.1. Routing Policy Model Tree . . . . .	11
7.2. Routing policy model . . . . .	12
8. Security Considerations . . . . .	32
9. IANA Considerations . . . . .	34
10. Acknowledgements . . . . .	34
11. References . . . . .	34
11.1. Normative references . . . . .	34
11.2. Informative references . . . . .	36
Appendix A. Routing protocol-specific policies . . . . .	36
Appendix B. Policy examples . . . . .	39
Authors' Addresses . . . . .	41

## 1. Introduction

This document describes a YANG [RFC7950] data model for routing policy configuration based on operational usage and best practices in a variety of service provider networks. The model is intended to be vendor-neutral, to allow operators to manage policy configuration consistently in environments with routers supplied by multiple vendors.

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

### 1.1. Goals and approach

This model does not aim to be feature complete -- it is a subset of the policy configuration parameters available in a variety of vendor implementations, but supports widely used constructs for managing how routes are imported, exported, and modified across different routing protocols. The model development approach has been to examine actual policy configurations in use across several operator networks. Hence, the focus is on enabling policy configuration capabilities and structure that are in wide use.

Despite the differences in details of policy expressions and conventions in various vendor implementations, the model reflects the observation that a relatively simple condition-action approach can be readily mapped to several existing vendor implementations, and also gives operators a familiar and straightforward way to express policy. A side effect of this design decision is that other methods for expressing policies are not considered.

Consistent with the goal to produce a data model that is vendor neutral, only policy expressions that are deemed to be widely available in existing major implementations are included in the model. Those configuration items that are only available from a single implementation are omitted from the model with the expectation they will be available in separate vendor-provided modules that augment the current model.

## 2. Terminology and Notation

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.

**Routing policy:** A routing policy defines how routes are imported, exported, modified, and advertised between routing protocol instances or within a single routing protocol instance.

**Policy chain:** A policy chain is a sequence of policy definitions. They can be referenced from different contexts.

**Policy statement:** Policy statements consist of a set of conditions and actions (either of which may be empty).

The following terms are defined in [RFC8342]:

- o client

- o server
- o configuration
- o system state
- o operational state
- o intended configuration

The following terms are defined in [RFC7950]:

- o action
- o augment
- o container
- o container with presence
- o data model
- o data node
- o feature
- o leaf
- o list
- o mandatory node
- o module
- o schema tree
- o RPC (Remote Procedure Call) operation

## 2.1. Tree Diagrams

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

## 2.2. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise,

names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
if	ietf-interfaces	[RFC8343]
rt	ietf-routing	[RFC8349]
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and Corresponding YANG Modules

### 3. Model overview

The routing policy module has three main parts:

- o A generic framework is provided to express policies as sets of related conditions and actions. This includes match sets and actions that are useful across many routing protocols.
- o A structure that allows routing protocol models to add protocol-specific policy conditions and actions through YANG augmentations is also provided. There is a complete example of this for BGP [RFC4271] policies in the proposed vendor-neutral BGP data model [I-D.ietf-idr-bgp-model]. Appendix A provides an example of how an augmentation for BGP policies might be accomplished. Note that this section is not normative as the BGP model is still evolving.
- o Finally, a reusable grouping is defined for attaching import and export rules in the context of routing configuration for different protocols, VRFs, etc. This also enables creation of policy chains and expressing default policy behavior. In this document, policy chains are sequences of policy definitions that are applied in order (described in Section 4).

The module makes use of the standard Internet types, such as IP addresses, autonomous system numbers, etc., defined in RFC 6991 [RFC6991].

#### 4. Route policy expression

Policies are expressed as a sequence of top-level policy definitions each of which consists of a sequence of policy statements. Policy statements in turn consist of simple condition-action tuples. Conditions may include multiple match or comparison operations, and similarly, actions may include multiple changes to route attributes, or indicate a final disposition of accepting or rejecting the route. This structure is shown below.

```

+--rw routing-policy
  +--ro match-modified-attributes?  boolean
  +--rw policy-definitions
    +--rw policy-definition* [name]
      +--rw name                string
      +--rw statements
        +--rw statement* [name]
          +--rw name              string
          +--rw conditions
          |   ...
          +--rw actions
          ...

```

##### 4.1. Defined sets for policy matching

The model provides a collection of generic sets that can be used for matching in policy conditions. These sets are applicable for route selection across multiple routing protocols. They may be further augmented by protocol-specific models which have their own defined sets. The defined sets include:

- o prefix sets - Each prefix set defines a set of IP prefixes, each with an associated IP prefix and netmask range (or exact length).
- o neighbor sets - Each neighbor set defines a set of neighboring nodes by their IP addresses. A neighbor set is used for selecting routes based on the neighbors advertising the routes.
- o tag set - Each tag set defines a set of generic tag values that can be used in matches for filtering routes.

The model structure for defined sets is shown below.



```

+--rw routing-policy
+--rw defined-sets
|   +--rw prefix-sets
|   |   +--rw prefix-set* [name]
|   |   |   +--rw name          string
|   |   |   +--rw mode?        enumeration
|   |   |   +--rw prefixes
|   |   |   |   +--rw prefix-list* [ip-prefix mask-length-lower
|   |   |   |   |   mask-length-upper]
|   |   |   |   |   +--rw ip-prefix          inet:ip-prefix
|   |   |   |   |   +--rw mask-length-lower  uint8
|   |   |   |   |   +--rw mask-length-upper  uint8
|   |   +--rw neighbor-sets
|   |   |   +--rw neighbor-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw address*      inet:ip-address
|   |   +--rw tag-sets
|   |   |   +--rw tag-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw tag-value*    tag-type

```

#### 4.2. Policy conditions

Policy statements consist of a set of conditions and actions (either of which may be empty). Conditions are used to match route attributes against a defined set (e.g., a prefix set), or to compare attributes against a specific value. The default action is to reject-route.

Match conditions may be further modified using the match-set-options configuration which allows network operators to change the behavior of a match. Three options are supported:

- o ALL - match is true only if the given value matches all members of the set.
- o ANY - match is true if the given value matches any member of the set.
- o INVERT - match is true if the given value does not match any member of the given set.

Not all options are appropriate for matching against all defined sets (e.g., match ALL in a prefix set does not make sense). In the model, a restricted set of match options is used where applicable.

Comparison conditions may similarly use options to change how route attributes should be tested, e.g., for equality or inequality, against a given value.

While most policy conditions will be added by individual routing protocol models via augmentation, this routing policy model includes several generic match conditions and the ability to test which protocol or mechanism installed a route (e.g., BGP, IGP, static, etc.). The conditions included in the model are shown below.

```

+--rw routing-policy
  +--rw policy-definitions
    +--rw policy-definition* [name]
      +--rw name                string
      +--rw statements
        +--rw statement* [name]
          +--rw conditions
            +--rw call-policy?
            +--rw source-protocol?
            +--rw match-interface
            |   +--rw interface?
            +--rw match-prefix-set
            |   +--rw prefix-set?
            |   +--rw match-set-options?
            +--rw match-neighbor-set
            |   +--rw neighbor-set?
            +--rw match-tag-set
            |   +--rw tag-set?
            |   +--rw match-set-options?
            +--rw match-route-type* identityref
            +--rw route-type*

```

#### 4.3. Policy actions

When policy conditions are satisfied, policy actions are used to set various attributes of the route being processed, or to indicate the final disposition of the route, i.e., accept or reject.

Similar to policy conditions, the routing policy model includes generic actions in addition to the basic route disposition actions. These are shown below.

```

+--rw routing-policy
  +--rw policy-definitions
    +--rw policy-definition* [name]
      +--rw statements
        +--rw statement* [name]
          +--rw actions
            +--rw policy-result?    policy-result-type
            +--rw set-metric
              | +--rw metric-modification?
              | | metric-modification-type
              | +--rw metric?      uint32
            +--rw set-metric-type
              | +--rw metric-type?  identityref
            +--rw set-route-level
              | +--rw route-level?  identityref
            +--rw set-route-preference?  uint16
            +--rw set-tag?              tag-type
            +--rw set-application-tag?  tag-type

```

#### 4.4. Policy subroutines

Policy 'subroutines' (or nested policies) are supported by allowing policy statement conditions to reference other policy definitions using the call-policy configuration. Called policies apply their conditions and actions before returning to the calling policy statement and resuming evaluation. The outcome of the called policy affects the evaluation of the calling policy. If the called policy results in an accept-route, then the subroutine returns an effective Boolean true value to the calling policy. For the calling policy, this is equivalent to a condition statement evaluating to a true value and evaluation of the policy continues (see Section 5). Note that the called policy may also modify attributes of the route in its action statements. Similarly, a reject-route action returns false and the calling policy evaluation will be affected accordingly. When the end of the subroutine policy statements is reached, the default route disposition action is returned (i.e., Boolean false for reject-route). Consequently, a subroutine cannot explicitly accept or reject a route. Rather, the called policy returns Boolean true if its outcome is accept-route or Boolean false if its outcome is reject-route. Route acceptance or rejection is solely determined by the top-level policy.

Note that the called policy may itself call other policies (subject to implementation limitations). The model does not prescribe a nesting depth because this varies among implementations. For example, an implementation may only support a single level of subroutine recursion. As with any routing policy construction, care must be taken with nested policies to ensure that the effective

return value results in the intended behavior. Nested policies are a convenience in many routing policy constructions but creating policies nested beyond a small number of levels (e.g., 2-3) is discouraged. Also, implementations MUST validate to ensure that there is no recursion among nested routing policies.

## 5. Policy evaluation

Evaluation of each policy definition proceeds by evaluating its individual policy statements in order that they are defined. When all the condition statements in a policy statement are satisfied, the corresponding action statements are executed. If the actions include either accept-route or reject-route actions, evaluation of the current policy definition stops, and no further policy statement is evaluated. If there are multiple policies in the policy chain, subsequent policies are not evaluated. Policy chains are sequences of policy definitions (as described in Section 4).

If the conditions are not satisfied, then evaluation proceeds to the next policy statement. If none of the policy statement conditions are satisfied, then evaluation of the current policy definition stops, and the next policy definition in the chain is evaluated. When the end of the policy chain is reached, the default route disposition action is performed (i.e., reject-route unless an alternate default action is specified for the chain).

Whether the route's pre-policy attributes are used for testing policy statement conditions is dependent on the implementation specific value of the match-modified-attributes leaf. If match-modified-attributes is false and actions modify route attributes, these modifications are not used for policy statement conditions. Conversely, if match-modified-attributes is true and actions modify the policy application-specific attributes, the attributes as modified by the policy are used for policy condition statements.

## 6. Applying routing policy

Routing policy is applied by defining and attaching policy chains in various routing contexts. Policy chains are sequences of policy definitions (described in Section 4). They can be referenced from different contexts. For example, a policy chain could be associated with a routing protocol and used to control its interaction with its protocol peers. Or it could be used to control the interaction between a routing protocol and the local routing information base. A policy chain has an associated direction (import or export), with respect to the context in which it is referenced.

The routing policy model defines an apply-policy grouping that can be imported and used by other models. As shown below, it allows definition of import and export policy chains, as well as specifying the default route disposition to be used when no policy definition in the chain results in a final decision.

```
+--rw apply-policy
|   +--rw import-policy*
|   +--rw default-import-policy?    default-policy-type
|   +--rw export-policy*
|   +--rw default-export-policy?    default-policy-type
```

The default policy defined by the model is to reject the route for both import and export policies.

## 7. YANG Module and Tree

### 7.1. Routing Policy Model Tree

The tree of the routing policy model is shown below.

```
module: ietf-routing-policy
rw routing-policy
+--rw defined-sets
|   +--rw prefix-sets
|   |   +--rw prefix-set* [name mode]
|   |   |   +--rw name          string
|   |   |   +--rw mode          enumeration
|   |   |   +--rw prefixes
|   |   |   |   +--rw prefix-list* [ip-prefix mask-length-lower
|   |   |   |   |   mask-length-upper]
|   |   |   |   |   +--rw ip-prefix          inet:ip-prefix
|   |   |   |   |   +--rw mask-length-lower    uint8
|   |   |   |   |   +--rw mask-length-upper    uint8
|   |   +--rw neighbor-sets
|   |   |   +--rw neighbor-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw address*      inet:ip-address
|   |   +--rw tag-sets
|   |   |   +--rw tag-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw tag-value*    tag-type
|   +--rw policy-definitions
|   |   +--ro match-modified-attributes?    boolean
|   |   +--rw policy-definition* [name]
|   |   |   +--rw name          string
|   |   |   +--rw statements
|   |   |   |   +--rw statement* [name]
```

```

+--rw name          string
+--rw conditions
|   +--rw call-policy?      -> ../../../../..
|   |                               /policy-definitions
|   |                               /policy-definition/name
|   +--rw source-protocol?  identityref
|   +--rw match-interface
|   |   +--rw interface?    -> /if:interfaces/interface
|   |   |                               /name
|   +--rw match-prefix-set
|   |   +--rw prefix-set?   -> ../../../../..
|   |   |                               /defined-sets/prefix-sets
|   |   |                               /prefix-set/name
|   |   +--rw match-set-options? match-set-options-type
|   +--rw match-neighbor-set
|   |   +--rw neighbor-set? -> ../../../../..
|   |   |                               /defined-sets/neighbor-sets
|   |   |                               /neighbor-set/name
|   +--rw match-tag-set
|   |   +--rw tag-set?      -> ../../../../..
|   |   |                               /defined-sets/tag-sets
|   |   |                               /tag-set/name
|   |   +--rw match-set-options? match-set-options-type
|   +--rw match-route-type*  identityref
+--rw actions
|   +--rw policy-result?      policy-result-type
|   +--rw set-metric
|   |   +--rw metric-modification? metric-modification-type
|   |   +--rw metric?         uint32
|   +--rw set-metric-type
|   |   +--rw metric-type?    identityref
|   +--rw set-route-level
|   |   +--rw route-level?    identityref
|   +--rw set-route-preference? uint16
|   +--rw set-tag?            tag-type
|   +--rw set-application-tag? tag-type

```

## 7.2. Routing policy model

The following RFCs are not referenced in the document text but are referenced in the `ietf-routing-policy.yang` module: [RFC2328], [RFC3101], [RFC5130], [RFC5302], [RFC6991], and [RFC8343].

```

<CODE BEGINS> file "ietf-routing-policy@2021-08-12.yang"
module ietf-routing-policy {

  yang-version "1.1";

```

```
namespace "urn:ietf:params:xml:ns:yang:ietf-routing-policy";
prefix rt-pol;

import ietf-inet-types {
  prefix "inet";
  reference
    "RFC 6991: Common YANG Data Types";
}

import ietf-yang-types {
  prefix "yang";
  reference
    "RFC 6991: Common YANG Data Types";
}

import ietf-interfaces {
  prefix "if";
  reference
    "RFC 8343: A YANG Data Model for Interface
      Management (NMDA Version)";
}

import ietf-routing {
  prefix "rt";
  reference
    "RFC 8349: A YANG Data Model for Routing
      Management (NMDA Version)";
}

organization
  "IETF RTGWG - Routing Area Working Group";
contact
  "WG Web:    <https://datatracker.ietf.org/wg/rtgwg/>
  WG List:    <mailto: rtgwg@ietf.org>

  Editor:     Yingzhen Qu
               <mailto: yingzhen.qu@futurewei.com>
               Jeff Tantsura
               <mailto: jefftant.ietf@gmail.com>
               Acee Lindem
               <mailto: acee@cisco.com>
               Xufeng Liu
               <mailto: xufeng.liu.ietf@gmail.com>";

description
  "This module describes a YANG model for routing policy
  configuration. It is a limited subset of all of the policy
  configuration parameters available in the variety of vendor
```

implementations, but supports widely used constructs for managing how routes are imported, exported, modified and advertised across different routing protocol instances or within a single routing protocol instance. This module is intended to be used in conjunction with routing protocol configuration modules (e.g., BGP) defined in other models.

This YANG module conforms to the Network Management Datastore Architecture (NMDA), as described in RFC 8342.

Copyright (c) 2021 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; 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.";

reference "RFC XXXX: A YANG Data Model for Routing Policy.";

```
revision "2021-08-12" {
  description
    "Initial revision.";
  reference
    "RFC XXXX: A YANG Data Model for Routing Policy Management.";
}
```

```
/* Identities */
```

```
identity metric-type {
  description
    "Base identity for route metric types.";
}
```

```
identity ospf-type-1-metric {
  base metric-type;
  description
```



```
        "Identity for the OSPF type 1 external metric types. It
        is only applicable to OSPF routes.";
    reference
        "RFC 2328: OSPF Version 2";
}

identity ospf-type-2-metric {
    base metric-type;
    description
        "Identity for the OSPF type 2 external metric types. It
        is only applicable to OSPF routes.";
    reference
        "RFC 2328: OSPF Version 2";
}

identity isis-internal-metric {
    base metric-type;
    description
        "Identity for the IS-IS internal metric types. It is only
        applicable to IS-IS routes.";
    reference
        "RFC 5302: Domain-Wide Prefix Distribution with
        Two-Level IS-IS";
}

identity isis-external-metric {
    base metric-type;
    description
        "Identity for the IS-IS external metric types. It is only
        applicable to IS-IS routes.";
    reference
        "RFC 5302: Domain-Wide Prefix Distribution with
        Two-Level IS-IS";
}

identity route-level {
    description
        "Base identity for route import level.";
}

identity ospf-normal {
    base route-level;
    description
        "Identity for OSPF importation into normal areas
        It is only applicable to routes imported
        into the OSPF protocol.";
    reference
        "RFC 2328: OSPF Version 2";
}
```

```
}

identity ospf-nssa-only {
  base route-level;
  description
    "Identity for the OSPF Not-So-Stubby Area (NSSA) area
    importation. It is only applicable to routes imported
    into the OSPF protocol.";
  reference
    "RFC 3101: The OSPF Not-So-Stubby Area (NSSA) Option";
}

identity ospf-normal-nssa {
  base route-level;
  description
    "Identity for OSPF importation into both normal and NSSA
    areas, it is only applicable to routes imported into
    the OSPF protocol.";
  reference
    "RFC 3101: The OSPF Not-So-Stubby Area (NSSA) Option";
}

identity isis-level-1 {
  base route-level;
  description
    "Identity for IS-IS Level 1 area importation. It is only
    applicable to routes imported into the IS-IS protocol.";
  reference
    "RFC 5302: Domain-Wide Prefix Distribution with
    Two-Level IS-IS";
}

identity isis-level-2 {
  base route-level;
  description
    "Identity for IS-IS Level 2 area importation. It is only
    applicable to routes imported into the IS-IS protocol.";
  reference
    "RFC 5302: Domain-Wide Prefix Distribution with
    Two-Level IS-IS";
}

identity isis-level-1-2 {
  base route-level;
  description
    "Identity for IS-IS importation into both Level 1 and Level 2
    areas. It is only applicable to routes imported into the IS-IS
    protocol.";
```

```
reference
  "RFC 5302: Domain-Wide Prefix Distribution with
    Two-Level IS-IS";
}

identity proto-route-type {
  description
    "Base identity for route type within a protocol.";
}

identity isis-level-1-type {
  base proto-route-type;
  description
    "Identity for IS-IS Level 1 route type. It is only
      applicable to IS-IS routes.";
  reference
    "RFC 5302: Domain-Wide Prefix Distribution with
      Two-Level IS-IS";
}

identity isis-level-2-type {
  base proto-route-type;
  description
    "Identity for IS-IS Level 2 route type. It is only
      applicable to IS-IS routes.";
  reference
    "RFC 5302: Domain-Wide Prefix Distribution with
      Two-Level IS-IS";
}

identity ospf-internal-type {
  base proto-route-type;
  description
    "Identity for OSPF intra-area or inter-area route type.
      It is only applicable to OSPF routes.";
  reference
    "RFC 2328: OSPF Version 2";
}

identity ospf-external-type {
  base proto-route-type;
  description
    "Identity for OSPF external type 1/2 route type.
      It is only applicable to OSPF routes.";
  reference
    "RFC 2328: OSPF Version 2";
}
```

```
identity ospf-external-t1-type {
  base ospf-external-type;
  description
    "Identity for OSPF external type 1 route type.
    It is only applicable to OSPF routes.";
  reference
    "RFC 2328: OSPF Version 2";
}

identity ospf-external-t2-type {
  base ospf-external-type;
  description
    "Identity for OSPF external type 2 route type.
    It is only applicable to OSPF routes.";
  reference
    "RFC 2328: OSPF Version 2";
}

identity ospf-nssa-type {
  base proto-route-type;
  description
    "Identity for OSPF NSSA type 1/2 route type.
    It is only applicable to OSPF routes.";
  reference
    "RFC 3101: The OSPF Not-So-Stubby Area (NSSA) Option";
}

identity ospf-nssa-t1-type {
  base ospf-nssa-type;
  description
    "Identity for OSPF NSSA type 1 route type.
    It is only applicable to OSPF routes.";
  reference
    "RFC 3101: The OSPF Not-So-Stubby Area (NSSA) Option";
}

identity ospf-nssa-t2-type {
  base ospf-nssa-type;
  description
    "Identity for OSPF NSSA type 2 route type.
    It is only applicable to OSPF routes.";
  reference
    "RFC 3101: The OSPF Not-So-Stubby Area (NSSA) Option";
}

identity bgp-internal {
  base proto-route-type;
  description
```

```
    "Identity for routes learned from internal BGP (IBGP).  
    It is only applicable to BGP routes.";  
reference  
    "RFC 4271: A Border Gateway Protocol 4 (BGP-4)";  
}  
  
identity bgp-external {  
    base proto-route-type;  
    description  
        "Identity for routes learned from external BGP (EBGP).  
        It is only applicable to BGP routes.";  
    reference  
        "RFC 4271: A Border Gateway Protocol 4 (BGP-4)";  
}  
  
/* Type Definitions */  
  
typedef default-policy-type {  
    type enumeration {  
        enum accept-route {  
            description  
                "Default policy to accept the route.";  
        }  
        enum reject-route {  
            description  
                "Default policy to reject the route.";  
        }  
    }  
    description  
        "Type used to specify route disposition in  
        a policy chain. This typedef is used in  
        the default import and export policy.";  
}  
  
typedef policy-result-type {  
    type enumeration {  
        enum accept-route {  
            description  
                "Policy accepts the route.";  
        }  
        enum reject-route {  
            description  
                "Policy rejects the route.";  
        }  
    }  
    description  
        "Type used to specify route disposition in  
        a policy chain.";
```

```
}

typedef tag-type {
  type union {
    type uint32;
    type yang:hex-string;
  }
  description
    "Type for expressing route tags on a local system,
    including IS-IS and OSPF; may be expressed as either decimal
    or hexadecimal integer.";
  reference
    "RFC 2328: OSPF Version 2
    RFC 5130: A Policy Control Mechanism in IS-IS Using
    Administrative Tags";
}

typedef match-set-options-type {
  type enumeration {
    enum any {
      description
        "Match is true if given value matches any member
        of the defined set.";
    }
    enum all {
      description
        "Match is true if given value matches all
        members of the defined set.";
    }
    enum invert {
      description
        "Match is true if given value does not match any
        member of the defined set.";
    }
  }
  default any;
  description
    "Options that govern the behavior of a match statement. The
    default behavior is any, i.e., the given value matches any
    of the members of the defined set.";
}

typedef metric-modification-type {
  type enumeration {
    enum set-metric {
      description
        "Set the metric to the specified value.";
    }
  }
}
```

```
enum add-metric {
    description
        "Add the specified value to the existing metric.
        If the result overflows the maximum metric
        (0xffffffff), set the metric to the maximum.";
}
enum subtract-metric {
    description
        "Subtract the specified value from the existing metric. If
        the result is less than 0, set the metric to 0.";
}
}
description
    "Type used to specify how to set the metric given the
    specified value.";
}

/* Groupings */

grouping prefix {
    description
        "Configuration data for a prefix definition.

        The combination of mask-length-lower and mask-length-upper
        define a range for the mask length, or single 'exact'
        length if mask-length-lower and mask-length-upper are
        equal.

        Example: 192.0.2.0/24 through 192.0.2.0/26 would be
        expressed as prefix: 192.0.2.0/24,
                mask-length-lower=24,
                mask-length-upper=26

        Example: 192.0.2.0/24 (an exact match) would be
        expressed as prefix: 192.0.2.0/24,
                mask-length-lower=24,
                mask-length-upper=24

        Example: 2001:DB8::/32 through 2001:DB8::/64 would be
        expressed as prefix: 2001:DB8::/32,
                mask-length-lower=32,
                mask-length-upper=64";

    leaf ip-prefix {
        type inet:ip-prefix;
        mandatory true;
        description
            "The IP prefix represented as an IPv6 or IPv4 network
```

```
        number followed by a prefix length with an intervening
        slash character as a delimiter. All members of the
        prefix-set MUST be of the same address family as the
        prefix-set mode.";
    }

    leaf mask-length-lower {
        type uint8 {
            range "0..128";
        }
        description
            "Mask length range lower bound. It MUST NOT be less than
            the prefix length defined in ip-prefix.";
    }
    leaf mask-length-upper {
        type uint8 {
            range "1..128";
        }
        must "../mask-length-upper >= ../mask-length-lower" {
            error-message "The upper bound MUST NOT be less"
                + "than lower bound.";
        }
        description
            "Mask length range upper bound. It MUST NOT be less than
            lower bound.";
    }
}

grouping match-set-options-group {
    description
        "Grouping containing options relating to how a particular set
        will be matched.";

    leaf match-set-options {
        type match-set-options-type;
        description
            "Optional parameter that governs the behavior of the
            match operation.";
    }
}

grouping match-set-options-restricted-group {
    description
        "Grouping for a restricted set of match operation
        modifiers.";

    leaf match-set-options {
        type match-set-options-type {
```



```
    enum any {
      description
        "Match is true if given value matches any
        member of the defined set.";
    }
    enum invert {
      description
        "Match is true if given value does not match
        any member of the defined set.";
    }
  }
  description
    "Optional parameter that governs the behavior of the
    match operation. This leaf only supports matching on
    'any' member of the set or 'invert' the match.
    Matching on 'all' is not supported.";
}

grouping apply-policy-group {
  description
    "Top level container for routing policy applications. This
    grouping is intended to be used in routing models where
    needed.";

  container apply-policy {
    description
      "Anchor point for routing policies in the model.
      Import and export policies are with respect to the local
      routing table, i.e., export (send) and import (receive),
      depending on the context.";

    leaf-list import-policy {
      type leafref {
        path "/rt-pol:routing-policy/rt-pol:policy-definitions/" +
          "rt-pol:policy-definition/rt-pol:name";
        require-instance true;
      }
      ordered-by user;
      description
        "List of policy names in sequence to be applied on
        receiving redistributed routes from another routing protocol
        or receiving a routing update in the current context, e.g.,
        for the current peer group, neighbor, address family, etc.";
    }

    leaf default-import-policy {
      type default-policy-type;
    }
  }
}
```

```
    default reject-route;
    description
        "Explicitly set a default policy if no policy definition
        in the import policy chain is satisfied.";
}

leaf-list export-policy {
    type leafref {
        path "/rt-pol:routing-policy/rt-pol:policy-definitions/" +
            "rt-pol:policy-definition/rt-pol:name";
        require-instance true;
    }
    ordered-by user;
    description
        "List of policy names in sequence to be applied on
        redistributing routes from one routing protocol to another
        or sending a routing update in the current context, e.g.,
        for the current peer group, neighbor, address family, etc.";
}

leaf default-export-policy {
    type default-policy-type;
    default reject-route;
    description
        "Explicitly set a default policy if no policy definition
        in the export policy chain is satisfied.";
}
}

container routing-policy {
    description
        "Top-level container for all routing policy.";

    container defined-sets {
        description
            "Predefined sets of attributes used in policy match
            statements.";

        container prefix-sets {
            description
                "Data definitions for a list of IPv4 or IPv6
                prefixes which are matched as part of a policy.";
            list prefix-set {
                key "name mode";
                description
                    "List of the defined prefix sets";
            }
        }
    }
}
```

```
leaf name {
  type string;
  description
    "Name of the prefix set -- this is used as a label to
    reference the set in match conditions.";
}

leaf mode {
  type enumeration {
    enum ipv4 {
      description
        "Prefix set contains IPv4 prefixes only.";
    }
    enum ipv6 {
      description
        "Prefix set contains IPv6 prefixes only.";
    }
  }
  description
    "Indicates the mode of the prefix set, in terms of
    which address families (IPv4 or IPv6) are present.
    The mode provides a hint, all prefixes MUST be of
    the indicated type. The device MUST validate that
    all prefixes and reject the configuration if there
    is a discrepancy.";
}

container prefixes {
  description
    "Container for the list of prefixes in a policy
    prefix list. Since individual prefixes do not have
    unique actions, the order in which the prefix in
    prefix-list are matched has no impact on the outcome
    and is left to the implementation. A given prefix-set
    condition is satisfied if the input prefix matches
    any of the prefixes in the prefix-set.";

  list prefix-list {
    key "ip-prefix mask-length-lower mask-length-upper";
    description
      "List of prefixes in the prefix set.";

    uses prefix;
  }
}
}
```

```
container neighbor-sets {
  description
    "Data definition for a list of IPv4 or IPv6
    neighbors which can be matched in a routing policy.";

  list neighbor-set {
    key "name";
    description
      "List of defined neighbor sets for use in policies.";

    leaf name {
      type string;
      description
        "Name of the neighbor set -- this is used as a label
        to reference the set in match conditions.";
    }

    leaf-list address {
      type inet:ip-address;
      description
        "List of IP addresses in the neighbor set.";
    }
  }
}

container tag-sets {
  description
    "Data definitions for a list of tags which can
    be matched in policies.";

  list tag-set {
    key "name";
    description
      "List of tag set definitions.";

    leaf name {
      type string;
      description
        "Name of the tag set -- this is used as a label to
        reference the set in match conditions.";
    }

    leaf-list tag-value {
      type tag-type;
      description
        "Value of the tag set member.";
    }
  }
}
```

```
    }  
  }  
  
  container policy-definitions {  
    description  
      "Enclosing container for the list of top-level policy  
      definitions.";  
  
    leaf match-modified-attributes {  
      type boolean;  
      config false;  
      description  
        "This boolean value dictates whether matches are performed  
        on the actual route attributes or route attributes  
        modified by policy statements preceding the match.";  
    }  
  
    list policy-definition {  
      key "name";  
      description  
        "List of top-level policy definitions, keyed by unique  
        name. These policy definitions are expected to be  
        referenced (by name) in policy chains specified in  
        import or export configuration statements.";  
  
      leaf name {  
        type string;  
        description  
          "Name of the top-level policy definition -- this name  
          is used in references to the current policy.";  
      }  
  
      container statements {  
        description  
          "Enclosing container for policy statements.";  
  
        list statement {  
          key "name";  
          ordered-by user;  
          description  
            "Policy statements group conditions and actions  
            within a policy definition. They are evaluated in  
            the order specified.";  
  
          leaf name {  
            type string;  
            description  
              "Name of the policy statement.";  
          }  
        }  
      }  
    }  
  }  
}
```

```
}

container conditions {
  description
    "Condition statements for the current policy
    statement.";

  leaf call-policy {
    type leafref {
      path "../..../..../.." +
        "rt-pol:policy-definitions/" +
        "rt-pol:policy-definition/rt-pol:name";
      require-instance true;
    }
    description
      "Applies the statements from the specified policy
      definition and then returns control to the current
      policy statement. Note that the called policy
      may itself call other policies (subject to
      implementation limitations). This is intended to
      provide a policy 'subroutine' capability. The
      called policy SHOULD contain an explicit or a
      default route disposition that returns an
      effective true (accept-route) or false
      (reject-route), otherwise the behavior may be
      ambiguous.";
  }

  leaf source-protocol {
    type identityref {
      base rt:control-plane-protocol;
    }
    description
      "Condition to check the protocol / method used to
      install the route into the local routing table.";
  }

  container match-interface {
    leaf interface {
      type leafref {
        path "/if:interfaces/if:interface/if:name";
      }
      description
        "Reference to a base interface.";
    }
    description
      "Container for interface match conditions";
  }
}
```

```
container match-prefix-set {
  leaf prefix-set {
    type leafref {
      path "../../../../../defined-sets/" +
        "prefix-sets/prefix-set/name";
    }
    description
      "References a defined prefix set.";
  }
  uses match-set-options-restricted-group;

  description
    "Match a referenced prefix-set according to the
    logic defined in the match-set-options leaf.";
}

container match-neighbor-set {
  leaf neighbor-set {
    type leafref {
      path "../../../../../defined-sets/" +
        "neighbor-sets/neighbor-set/name";
      require-instance true;
    }
    description
      "References a defined neighbor set.";
  }

  description
    "Match a referenced neighbor set.";
}

container match-tag-set {
  leaf tag-set {
    type leafref {
      path "../../../../../defined-sets/" +
        "tag-sets/tag-set/name";
      require-instance true;
    }
    description
      "References a defined tag set.";
  }
  uses match-set-options-group;

  description
    "Match a referenced tag set according to the logic
    defined in the match-set-options leaf.";
}
```

```
    container match-route-type {
      description
        "This container provides route-type match condition";

      leaf-list route-type {
        type identityref {
          base proto-route-type;
        }
        description
          "Condition to check the protocol-specific type
           of route. This is normally used during route
           importation to select routes or to set protocol
           specific attributes based on the route type.";
      }
    }
  }

  container actions {
    description
      "Top-level container for policy action
       statements.";
    leaf policy-result {
      type policy-result-type;
      default reject-route;
      description
        "Select the final disposition for the route,
         either accept or reject.";
    }
    container set-metric {
      leaf metric-modification {
        type metric-modification-type;
        description
          "Indicates how to modify the metric.";
      }
      leaf metric {
        type uint32;
        description
          "Metric value to set, add, or subtract.";
      }
      description
        "Set the metric for the route.";
    }
    container set-metric-type {
      leaf metric-type {
        type identityref {
          base metric-type;
        }
        description

```





```
}  
}  
<CODE ENDS>
```

## 8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a pre-configured subset of all available NETCONF or RESTCONF protocol operations and content.

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

```
/routing-policy/defined-sets/prefix-sets -- Modification to  
prefix-sets could result in a Denial-of-Service (DoS) attack. An  
attacker may try to modify prefix-sets and redirect or drop  
traffic. Redirection of traffic could be used as part of a more  
elaborate attack to either collect sensitive information or  
masquerade a service. Additionally, a control-plane DoS attack  
could be accomplished by allowing a large number of routes to be  
leaked into a routing protocol domain (e.g., BGP).
```

```
/routing-policy/defined-sets/neighbor-sets -- Modification to the  
neighbor-sets could be used to mount a DoS attack or more  
elaborate attack as with prefix-sets. For example, a DoS attack  
could be mounted by changing the neighbor-set from which routes  
are accepted.
```

```
/routing-policy/defined-sets/tag-sets -- Modification to the tag-  
sets could be used to mount a DoS attack. Routes with certain  
tags might be redirected or dropped. The implications are similar  
to prefix-sets and neighbor-sets. However, the attack may be more  
difficult to detect as the routing policy usage of route tags and
```

intent must be understood to recognize the breach. Conversely, the implications of prefix-set or neighbor set modification are easier to recognize.

```
/routing-policy/policy-definitions/policy-definition
/statements/statement/conditions -- Modification to the conditions
could be used to mount a DoS attack or other attack. An attacker
may change a policy condition and redirect or drop traffic. As
with prefix-sets, neighbor-sets, or tag-sets, traffic redirection
could be used as part of a more elaborate attack.
```

```
/routing-policy/policy-definitions/policy-definition
/statements/statement/actions -- Modification to actions could be
used to mount a DoS attack or other attack. Traffic may be
redirected or dropped. As with prefix-sets, neighbor-sets, or
tag-sets, traffic redirection could be used as part of a more
elaborate attack. Additionally, route attributes may be changed
to mount a second-level attack that is more difficult to detect.
```

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

```
/routing-policy/defined-sets/prefix-sets -- Knowledge of these
data nodes can be used to ascertain which local prefixes are
susceptible to a Denial-of-Service (DoS) attack.
```

```
/routing-policy/defined-sets/prefix-sets -- Knowledge of these
data nodes can be used to ascertain local neighbors against whom
to mount a Denial-of-Service (DoS) attack.
```

```
/routing-policy/policy-definitions/policy-definition /statements/
-- Knowledge of these data nodes can be used to attack the local
router with a Denial-of-Service (DoS) attack. Additionally,
policies and their attendant conditions and actions should be
considered proprietary and disclosure could be used to ascertain
partners, customers, and supplies. Furthermore, the policies
themselves could represent intellectual property and disclosure
could diminish their corresponding business advantage.
```

Routing policy configuration has a significant impact on network operations, and, as such, other YANG models that reference routing policies are also susceptible to vulnerabilities relating the YANG data nodes specified above.

## 9. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made:

```
URI: urn:ietf:params:xml:ns:yang:ietf-routing-policy
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
```

This document registers a YANG module in the YANG Module Names registry [RFC6020].

```
name: ietf-routing-policy
namespace: urn:ietf:params:xml:ns:yang:ietf-routing-policy
prefix: rt-pol
reference: RFC XXXX
```

## 10. Acknowledgements

The routing policy module defined in this document is based on the OpenConfig route policy model. The authors would like to thank to OpenConfig for their contributions, especially Anees Shaikh, Rob Shakir, Kevin D'Souza, and Chris Chase.

The authors are grateful for valuable contributions to this document and the associated models from: Ebben Aires, Luyuan Fang, Josh George, Stephane Litkowski, Ina Minei, Carl Moberg, Eric Osborne, Steve Padgett, Juergen Schoenwaelder, Jim Uttaro, Russ White, and John Heasley.

Thanks to Mahesh Jethanandani, John Scudder, Chris Bowers and Tom Petch for their reviews and comments.

## 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>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.

- [RFC3101] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, DOI 10.17487/RFC3101, January 2003, <<https://www.rfc-editor.org/info/rfc3101>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC5130] Previdi, S., Shand, M., Ed., and C. Martin, "A Policy Control Mechanism in IS-IS Using Administrative Tags", RFC 5130, DOI 10.17487/RFC5130, February 2008, <<https://www.rfc-editor.org/info/rfc5130>>.
- [RFC5302] Li, T., Smit, H., and T. Przygienda, "Domain-Wide Prefix Distribution with Two-Level IS-IS", RFC 5302, DOI 10.17487/RFC5302, October 2008, <<https://www.rfc-editor.org/info/rfc5302>>.
- [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>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [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>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.

- [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>>.
- [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>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [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>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
- [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>>.

## 11.2. Informative references

- [I-D.ietf-idr-bgp-model]  
Jethanandani, M., Patel, K., Hares, S., and J. Haas, "BGP YANG Model for Service Provider Networks", draft-ietf-idr-bgp-model-11 (work in progress), July 2021.

## Appendix A. Routing protocol-specific policies

Routing models that require the ability to apply routing policy may augment the routing policy model with protocol or other specific policy configuration. The routing policy model assumes that additional defined sets, conditions, and actions may all be added by other models.

The example below provides an illustration of how another data model can augment parts of this routing policy data model. It uses

specific examples from draft-ietf-idr-bgp-model-09 to show in a concrete manner how the different pieces fit together. This example is not normative with respect to [I-D.ietf-idr-bgp-model]. The model similarly augments BGP-specific conditions and actions in the corresponding sections of the routing policy model. In the example below, the XPath prefix "bp:" specifies import from the ietf-bgp-policy sub-module and the XPath prefix "bt:" specifies import from the ietf-bgp-types sub-module [I-D.ietf-idr-bgp-model].

```

module: ietf-routing-policy
+--rw routing-policy
|   +--rw defined-sets
|   |   +--rw prefix-sets
|   |   |   +--rw prefix-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw mode?         enumeration
|   |   |   |   +--rw prefixes
|   |   |   |   |   +--rw prefix-list* [ip-prefix mask-length-lower
|   |   |   |   |   |   mask-length-upper]
|   |   |   |   |   +--rw ip-prefix          inet:ip-prefix
|   |   |   |   |   +--rw mask-length-lower  uint8
|   |   |   |   |   +--rw mask-length-upper  uint8
|   |   +--rw neighbor-sets
|   |   |   +--rw neighbor-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw address*     inet:ip-address
|   |   +--rw tag-sets
|   |   |   +--rw tag-set* [name]
|   |   |   |   +--rw name          string
|   |   |   |   +--rw tag-value*   tag-type
|   |   +--rw bp:bp-defined-sets
|   |   |   +--rw bp:community-sets
|   |   |   |   +--rw bp:community-set* [name]
|   |   |   |   |   +--rw bp:name      string
|   |   |   |   |   +--rw bp:member*  union
|   |   |   +--rw bp:ext-community-sets
|   |   |   |   +--rw bp:ext-community-set* [name]
|   |   |   |   |   +--rw bp:name      string
|   |   |   |   |   +--rw bp:member*  union
|   |   +--rw bp:as-path-sets
|   |   |   +--rw bp:as-path-set* [name]
|   |   |   |   +--rw bp:name      string
|   |   |   |   +--rw bp:member*  string
|   +--rw policy-definitions
|   |   +--ro match-modified-attributes?  boolean
|   |   +--rw policy-definition* [name]
|   |   |   +--rw name          string
|   |   |   +--rw statements

```

```

+--rw statement* [name]
  +--rw name          string
  +--rw conditions
    +--rw call-policy?
    +--rw source-protocol?          identityref
    +--rw match-interface
    |   +--rw interface?
    +--rw match-prefix-set
    |   +--rw prefix-set?          prefix-set/name
    |   +--rw match-set-options?  match-set-options-type
    +--rw match-neighbor-set
    |   +--rw neighbor-set?
    +--rw match-tag-set
    |   +--rw tag-set?
    |   +--rw match-set-options?  match-set-options-type
    +--rw match-route-type*  identityref
    +--rw bp:bgp-conditions
      +--rw bp:med-eq?          uint32
      +--rw bp:origin-eq?      bt:bgp-origin-attr-type
      +--rw bp:next-hop-in*    inet:ip-address-no-zone
      +--rw bp:afi-safi-in*    identityref
      +--rw bp:local-pref-eq?  uint32
      +--rw bp:route-type?    enumeration
      +--rw bp:community-count
      +--rw bp:as-path-length
      +--rw bp:match-community-set
      |   +--rw bp:community-set?
      |   +--rw bp:match-set-options?
      +--rw bp:match-ext-community-set
      |   +--rw bp:ext-community-set?
      |   +--rw bp:match-set-options?
      +--rw bp:match-as-path-set
      |   +--rw bp:as-path-set?
      |   +--rw bp:match-set-options?
    +--rw actions
      +--rw policy-result?      policy-result-type
      +--rw set-metric
      |   +--rw metric-modification?
      |   +--rw metric?          uint32
      +--rw set-metric-type
      |   +--rw metric-type?    identityref
      +--rw set-route-level
      |   +--rw route-level?    identityref
      +--rw set-route-preference?  uint16
      +--rw set-tag?              tag-type
      +--rw set-application-tag?  tag-type
      +--rw bp:bgp-actions
      |   +--rw bp:set-route-origin?bt:bgp-origin-attr-type

```



```

+--rw bp:set-local-pref?    uint32
+--rw bp:set-next-hop?      bgp-next-hop-type
+--rw bp:set-med?           bgp-set-med-type
+--rw bp:set-as-path-prepend
|   +--rw bp:repeat-n?      uint8
+--rw bp:set-community
|   +--rw bp:method?        enumeration
|   +--rw bp:options?
|   +--rw bp:inline
|   |   +--rw bp:communities*  union
|   +--rw bp:reference
|   |   +--rw bp:community-set-ref?
+--rw bp:set-ext-community
|   +--rw bp:method?        enumeration
|   +--rw bp:options?
|   +--rw bp:inline
|   |   +--rw bp:communities*  union
|   +--rw bp:reference
|   |   +--rw bp:ext-community-set-ref?

```

## Appendix B. Policy examples

Below we show examples of XML-encoded configuration data using the routing policy and BGP models to illustrate both how policies are defined, and how they can be applied. Note that the XML has been simplified for readability.

The following example shows how prefix-set and tag-set can be defined. The policy condition is to match a prefix-set and a tag-set, and the action is to accept routes that match the condition.

```

<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <routing-policy
    xmlns="urn:ietf:params:xml:ns:yang:ietf-routing-policy">

    <defined-sets>
      <prefix-sets>
        <prefix-set>
          <name>prefix-set-A</name>
          <mode>ipv4</mode>
          <prefixes>
            <prefix-list>
              <ip-prefix>192.0.2.0/24</ip-prefix>
              <mask-length-lower>24</mask-length-lower>
              <mask-length-upper>32</mask-length-upper>
            </prefix-list>
            <prefix-list>
              <ip-prefix>198.51.100.0/24</ip-prefix>
            </prefix-list>
          </prefixes>
        </prefix-set>
      </prefix-sets>
    </defined-sets>
  </routing-policy>
</config>

```

```
        <mask-length-lower>24</mask-length-lower>
        <mask-length-upper>32</mask-length-upper>
    </prefix-list>
</prefixes>
</prefix-set>
<prefix-set>
  <name>prefix-set-B</name>
  <mode>ipv6</mode>
  <prefixes>
    <prefix-list>
      <ip-prefix>2001:DB8::/32</ip-prefix>
      <mask-length-lower>32</mask-length-lower>
      <mask-length-upper>64</mask-length-upper>
    </prefix-list>
  </prefixes>
</prefix-set>
</prefix-sets>
<tag-sets>
  <tag-set>
    <name>cust-tag1</name>
    <tag-value>10</tag-value>
  </tag-set>
</tag-sets>
</defined-sets>

<policy-definitions>
  <policy-definition>
    <name>export-tagged-BGP</name>
    <statements>
      <statement>
        <name>term-0</name>
        <conditions>
          <match-prefix-set>
            <prefix-set>prefix-set-A</prefix-set>
          </match-prefix-set>
          <match-tag-set>
            <tag-set>cust-tag1</tag-set>
          </match-tag-set>
        </conditions>
        <actions>
          <policy-result>accept-route</policy-result>
        </actions>
      </statement>
    </statements>
  </policy-definition>
</policy-definitions>

</routing-policy>
```

```
</config>
```

In the following example, all routes in the RIB that have been learned from OSPF advertisements corresponding to OSPF intra-area and inter-area route types should get advertised into ISIS level-2 advertisements.

```
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <routing-policy
    xmlns="urn:ietf:params:xml:ns:yang:ietf-routing-policy">
    <policy-definitions>
      <policy-definition>
        <name>export-all-OSPF-prefixes-into-ISIS-level-2</name>
        <statements>
          <statement>
            <name>term-0</name>
            <conditions>
              <match-route-type>ospf-internal-type</match-route-type>
            </conditions>
            <actions>
              <set-route-level>
                <route-level>isis-level-2</route-level>
              </set-route-level>
              <policy-result>accept-route</policy-result>
            </actions>
          </statement>
        </statements>
      </policy-definition>
    </policy-definitions>
  </routing-policy>
</config>
```

#### Authors' Addresses

Yingzhen Qu  
Futurewei  
2330 Central Expressway  
Santa Clara CA 95050  
USA

Email: yingzhen.qu@futurewei.com

Jeff Tantsura  
Microsoft

Email: [jefftant.ietf@gmail.com](mailto:jefftant.ietf@gmail.com)

Acee Lindem  
Cisco  
301 Midenhall Way  
Cary, NC 27513  
US

Email: [acee@cisco.com](mailto:acee@cisco.com)

Xufeng Liu  
Volta Networks

Email: [xufeng.liu.ietf@gmail.com](mailto:xufeng.liu.ietf@gmail.com)

RTGWG Working Group  
Internet Draft  
Intended status: Informational  
Expires January 3, 2019

Daniel King  
Lancaster University

Young Lee (Editor)  
Huawei

Jeff Tansura  
Nuage

Qin Wu  
Huawei

Daniele Ceccarelli  
Ericsson

July 2, 2018

Applicability of Abstraction and Control of Traffic Engineered  
Networks (ACTN) to Enhanced VPN

draft-lee-rtgwg-actn-applicability-enhanced-vpn-03

Abstract

The Abstraction and Control of Traffic Engineered Networks (ACTN) defines an SDN-based architecture that relies on the concepts of network and service abstraction to detach network and service control from the underlying data plane.

This document outlines the overview of ACTN capability and the applicability of ACTN to Enhanced VPN. In particular, this document also discusses how ACTN features can fulfill some of the requirements of the enhanced VPN, which is also known as VPN+ [VPN+].

Status of this Memo

This Internet-Draft is submitted to IETF 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 2, 2019.

#### Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Introduction.....	3
2. ACTN Overview.....	3
2.1. ACTN Virtual Network.....	4
2.2. Examples of ACTN Delivering Types of Virtual Networks.....	5
2.2.1. ACTN Used for Virtual Private Line Model.....	6
2.2.2. ACTN Used for VPN Delivery Model.....	7
2.2.3. ACTN Used to Deliver a Virtual Customer Network.....	8
2.3. Service Mapping from TE to ACTN VN Models.....	9
2.4. ACTN VN KPI telemetry Models.....	10
3. Enhanced VPN and ACTN.....	10
3.1. Isolation between VPNs.....	11
3.2. Guaranteed Performance.....	11
3.3. Integration.....	13
3.4. Dynamic Configuration.....	13
3.5. Customized Control Plane.....	14
3.6. The Gaps.....	15
4. Security Considerations.....	16
5. IANA Considerations.....	16
6. Acknowledgements.....	17

7. References.....	17
7.1. Informative References.....	17
8. Contributors.....	18
Authors' Addresses.....	18

## 1. Introduction

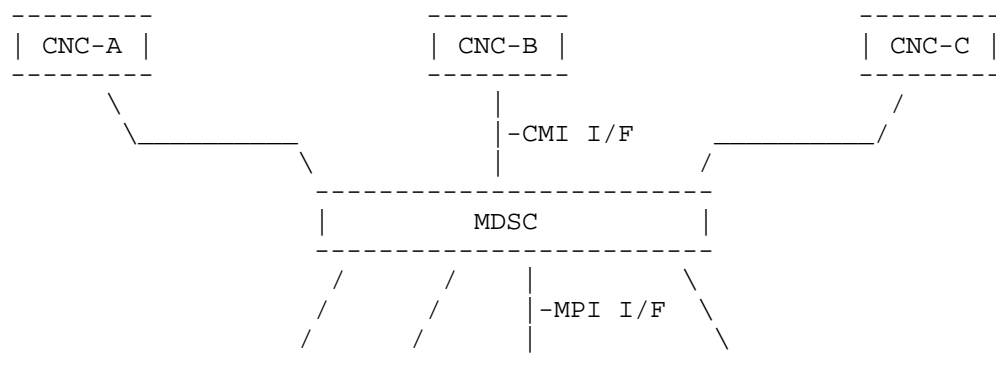
The Abstraction and Control of Traffic Engineered Networks (ACTN) defines an SDN-based architecture that relies on the concepts of network and service abstraction to detach network and service control from the underlying data plane.

This document outlines the overview of ACTN capability and the applicability of ACTN to Enhanced VPN. In particular, this document also discusses how ACTN features can fulfill some of the key requirements of the enhanced VPN, which is also known as VPN+ [VPN+].

## 2. ACTN Overview

The framework for ACTN [actn-framework] includes a reference architecture that has been adapted for Figure 1 in this document, it describes the functional entities and methods for the coordination of resources across multiple domains, to provide end-to-end services, components include:

- o Customer Network Controller (CNC);
- o Multi-domain Service Coordinator (MDSC);
- o Provisioning Network Controller (PNC).



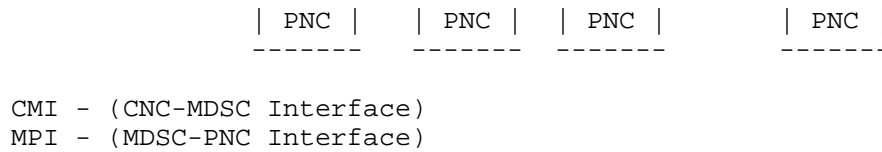


Figure 1: ACTN Hierarchy

ACTN facilitates end-to-end connections and provides them to the user. The ACTN framework highlights how:

- o Abstraction of the underlying network resources are provided to higher-layer applications and customers;
- o Virtualization of underlying resources, whose selection criterion is the allocation of those resources for the customer, application, or service;
- o Creation of a virtualized environment allowing operators to view and control multi-domain networks as a single virtualized network;
- o The presentation to customers of networks as a virtual network via open and programmable interfaces.

The ACTN managed infrastructure are traffic engineered network resources, which may include:

- o Statistical packet bandwidth;
- o Physical forwarding plane sources, such as: wavelengths and time slots;
- o Forwarding and cross connect capabilities.

The ACTN type of network virtualization provides customers and applications (tenants) to utilize and independently control allocated virtual network resources as if resources as if they were physically their own resource.

## 2.1. ACTN Virtual Network

To support multiple clients each with its own view of and control of the server network, a network operator needs to partition the network resources. The resulting partition can be assigned to each client for guaranteed usage which is a step further than shared use



of common network resources. See [actn-vn] for detailed ACTN VN and VNS.

An ACTN Virtual Network (VN) is a client view of the ACTN managed infrastructure, and is presented by the ACTN provider as a set of abstracted resources.

Depending on the agreement between client and provider various VN operations and VN views are possible.

- o Virtual Network Creation: A VN could be pre-configured and created via static or dynamic request and negotiation between customer and provider. It must meet the specified SLA attributes which satisfy the customer's objectives.
- o Virtual Network Operations: The virtual network may be further modified and deleted based on customer request to request changes in the network resources reserved for the customer, and used to construct the network slice. The customer can further act upon the virtual network to manage traffic flow across the virtual network.
- o Virtual Network View: The VN topology from a customer point of view. These may be a variety of tunnels, or an entire VN topology. Such connections may comprise of customer end points, access links, intra domain paths and inter-domain links.

Dynamic VN Operations allow a customer to modify or delete the VN. The customer can further act upon the virtual network to create/modify/delete virtual links and nodes. These changes will result in subsequent tunnel management in the operator's networks.

Primitives (capabilities and messages) have been provided to support the different ACTN network control functions that will enable virtual network. These include: topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options. [actn-info]

## 2.2. Examples of ACTN Delivering Types of Virtual Networks

In examples below the ACTN framework is used to provide control, management and orchestration for the virtual network life-cycle, and the connectivity. These dynamic and highly flexible, end-to-end and

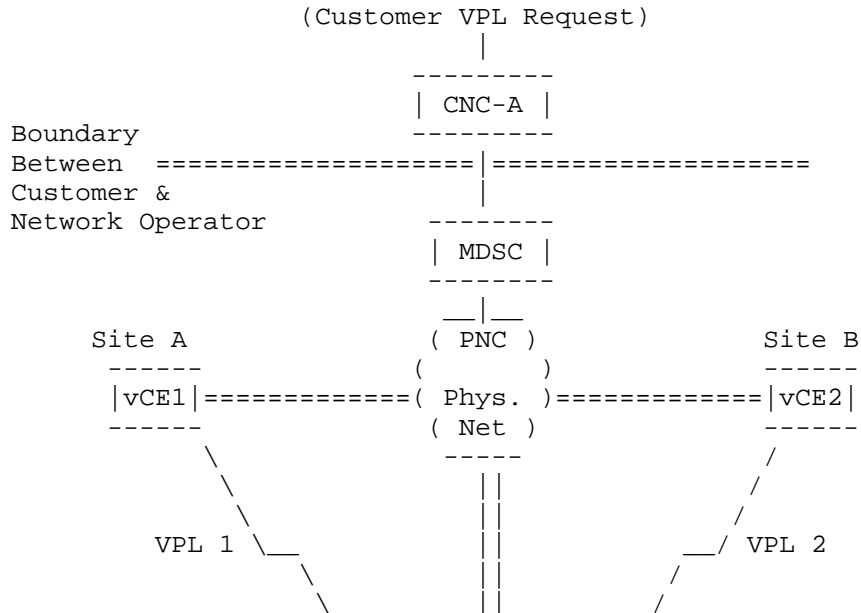
dedicated virtual network utilizing common physical infrastructure, and according to vertical-specific requirements.

The rest of this section provides three examples of using ACTN to achieve different scenarios of ACTN for virtual network. All three scenarios can be scaled up in capacity or be subject to topology changes as well as changes from customer requirements perspective.

### 2.2.1. ACTN Used for Virtual Private Line Model

ACTN provides virtual connections between multiple customer locations, requested via Virtual Private Line (VPL) requester (CNC-A). Benefits of this model include:

- o Automated: the service set-up and operation is network provider managed;
- o Virtual: the private line is seamlessly extended from customers Site A (vCE1 to vCE3) and Site B (vCE2 to vCE3) across the ACTN-managed WAN to Site C;
- o Agile: on-demand where the customer needs connectivity and fully adjustable bandwidth.



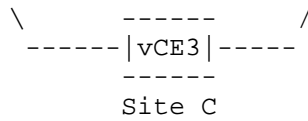
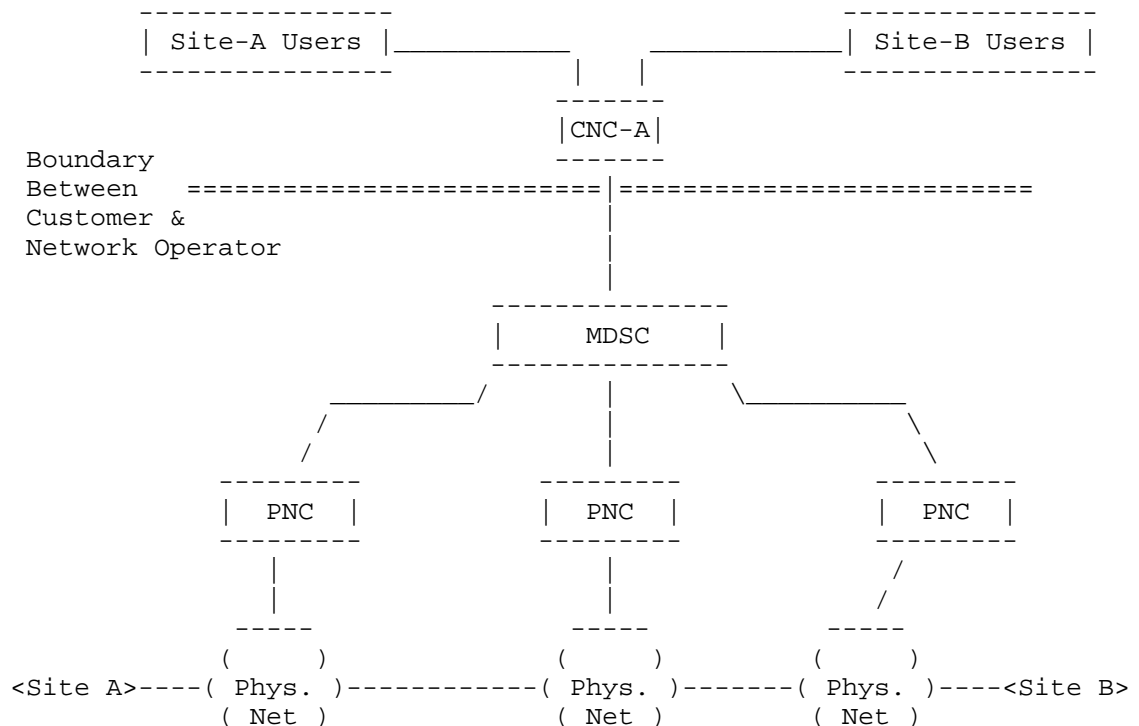


Figure 2: Virtual Private Line Model

### 2.2.2. ACTN Used for VPN Delivery Model

ACTN provides VPN connections between multiple sites, requested Via a VPN requestor (CNC-A), which is managed by the customer themselves. The CNC will then interact with the network provider's MDSC. Benefits of this model include:

- o Provides edge-to-edge VPN multi-access connection;
- o Mostly network provider managed, with some flexibility delegated to the customer managed CNC.



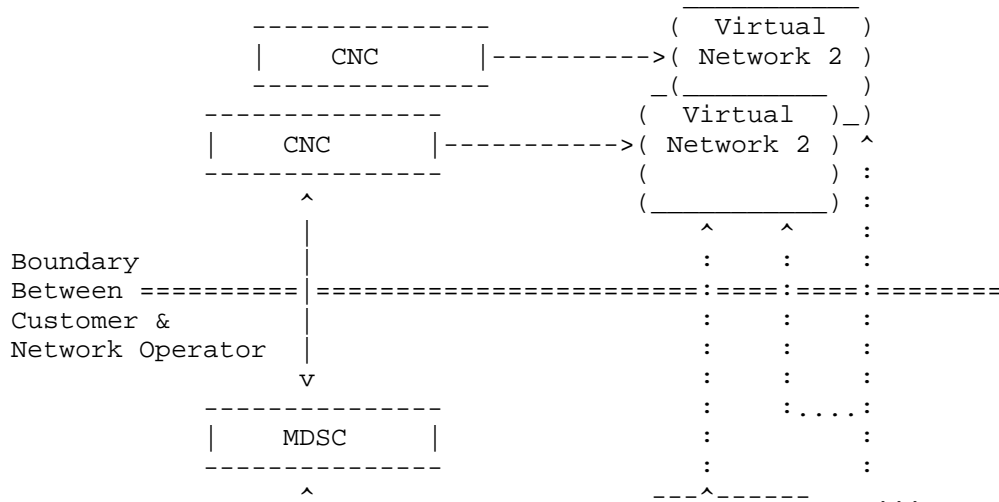
-----

Figure 3: VPN Model

### 2.2.3. ACTN Used to Deliver a Virtual Customer Network

In this example ACTN provides a virtual network resource to the customer. This resource is customer managed. Empowering the tenant to control allocated VN (recursively). Benefits of this model include:

- o The MDSC provides the topology as part of the customer view so that the customer can control their network slice to fit their needs;
- o Resource isolation, each customer network slice is fixed and will not be affected by changes to other customer network slices;
- o Applications can interact with their assigned network slice directly, the customer may implement their own network control method and traffic prioritization, manage their own addressing scheme, and further slice their assigned network resource;
- o The network slice may also include specific capability nodes, delivered as Physical Network Functions (PNFs) or Virtual Network Functions (VNFs).



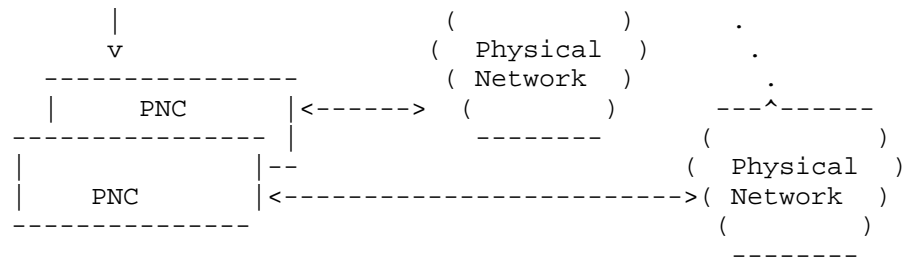


Figure 4: Virtual Customer Networks

### 2.3. Service Mapping from TE to ACTN VN Models

The role of TE-service mapping model [te-service-mapping] is to create a binding relationship across a Layer-3 Service Model [L3sm], Layer-2 Service Model [L2SM], Layer-1 Service Model [L1CSM], and TE Tunnel model [TE-tunnel], via a generic ACTN Virtual Network (VN) model [actn-vn].

The ACTN VN YANG model [actn-vn] is a generic virtual network service model that allows customers (internal or external) to create a VN that meets the customer's service objective with various constraints.

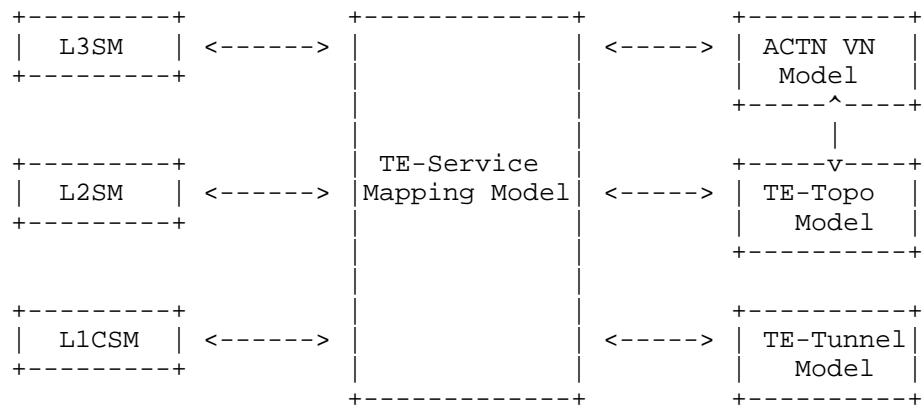


Figure 5: TE-Service Mapping ([te-service-mapping])

The TE-service mapping model [te-service-map] is needed to bind L1/2/3 VPN specific service requirements and policies pertaining to TE-specific parameters. For example, the model can express the

isolation requirement for each VPN service instance. Some VPN service would require a strict hard isolation with deterministic characteristic. In such case the underlay TE networks has to find end-to-end tunnels/LSPs that satisfy this particular isolation requirement.

This binding will facilitate a seamless service operation with underlay-TE network visibility. The TE-service model developed in this document can also be extended to support other services including L2SM, and L1CSM.

#### 2.4. ACTN VN KPI telemetry Models

The role of ACTN VN KPI telemetry model [actn-pm-telemetry] is to provide YANG models so that customer can define key performance monitoring data relevant for its VN via the YANG subscription model.

Key characteristics of [actn-pm-telemetry] include:

- o an ability to provide scalable VN-level telemetry aggregation based on customer-subscription model for key performance parameters defined by the customer;
- o an ability to facilitate proactive re-optimization and reconfiguration of VNs based on network autonomic traffic engineering scaling configuration mechanism.

#### 3. Enhanced VPN and ACTN

This section discusses how the advanced features of ACTN discussed in Section 3 can fulfill the enhanced VPN requirements defined in [vpn+]. Key requirements of the enhanced VPN include:

1. Isolation between VPNs
2. Guaranteed Performance
3. Integration
4. Dynamic Configuration
5. Customized Control Plane

Simple creation, deletion and modification of the services.  
Control over VPN Seamless integration of both physical and virtual network and service functions

In the subsequent sections, we discuss how each requirement can be fulfilled by the ACTN features and the gaps that remain to be solved if applicable.

### 3.1. Isolation between VPNs

The ACTN VN YANG model [actn-vn] and the TE-service mapping model [te-service-mapping] fulfill the isolation requirement by providing the features.

- o Each VN is identified with a unique identifier (vn-id and vn-name) and so is each VN member that belongs to the VN (vn-member-id).
- o Each instantiated VN is managed and controlled independent of other VNs in the network with proper protection level (protection)
- o Each VN is instantiated with proper isolation requirement mapping introduced by the TE-service mapping model [te-service-mapping]. This mapping can support:
  - o hard isolation with deterministic characteristics (e.g., this case may need optical bypass tunnel or DetNet/TSN tunnel to guarantee latency with no jitter);
  - o hard isolation (i.e., dedicated TE resources in all layers (e.g., packet and optical));
  - o soft isolation (i.e., optical layer may be shared while packet layer is dedicated);
  - o no isolation (i.e., sharing with other VN).

### 3.2. Guaranteed Performance

Performance objectives of a VN need first to be expressed in order to assure the performance guarantee. [actn-vn] and [te-topo] allow configuration of several parameters that may affect the VN performance objectives. Among the performance-related parameters per a VN level provided by [actn-vn] and [te-topo] are as follows:

- o Bandwidth
- o Objective function (e.g., min cost path, min load path, etc.)

- o Metric Types and their threshold:
  - o TE cost, IGP cost, Hop count, or Unidirectional Delay (e.g., can set all path delay <= threshold)

See the below actn-vn tree structure for the pointer for the connectivity matrix identifier for each vn member in which the configuration parameters listed above is provisioned using [te-topo] model together with [te-tunnel] model in the network.

```

+--rw vn
  +--rw vn-list* [vn-id]
    +--rw vn-id                uint32
    +--rw vn-name?             string
    +--rw vn-topology-id?      te-types:te-topology-id
    +--rw abstract-node?       -> /nw:networks/network/node/tet:te-nod
e-id
  +--rw vn-member-list* [vn-member-id]
    | +--rw vn-member-id      uint32
    | +--rw src
    | | +--rw src?            -> /actn/ap/access-point-list/access-po
int-id
    | | +--rw src-vn-ap-id?   -> /actn/ap/access-point-list/vn-ap/vn-
ap-id
    | | +--rw multi-src?      boolean {multi-src-dest}?
    | +--rw dest
    | | +--rw dest?           -> /actn/ap/access-point-list/access-p
oint-
id
    | | +--rw dest-vn-ap-id?  -> /actn/ap/access-point-list/vn-ap/vn
-ap-id
    | | +--rw multi-dest?     boolean {multi-src-dest}?
    | +--rw connetivity-matrix-id? -> /nw:networks/network/node/tet:
te/te-
node-attributes/connectivity-matrices/connectivity-matrix/id
    | +--ro oper-status?      identityref
    +--ro if-selected?        boolean {multi-src-dest}?
    +--rw admin-status?       identityref
    +--ro oper-status?        identityref
    +--rw vn-level-diversity?  vn-disjointness

```

Once these requests are instantiated, the resources are committed and guaranteed through the life cycle of the VN.

[actn-pm-telemetry] provides models that allow for key performance telemetry configuration mechanisms per VN level, VN member level as well as path/link level.

The following tree structure from [actn-pm-telemetry] illustrates how performance data (e.g., delay, delay-variation, utilization, etc.) can be subscribed per VN need and monitored via YANG push streaming mechanism.



```

module: ietf-actn-te-kpi-telemetry
...
augment /vn:actn/vn:vn/vn:vn-list/vn:vn-member-list:
  +--ro vn-member-telemetry
    +--ro unidirectional-delay?          uint32
    +--ro unidirectional-min-delay?      uint32
    +--ro unidirectional-max-delay?      uint32
    +--ro unidirectional-delay-variation? uint32
    +--ro unidirectional-packet-loss?    decimal64
    +--ro unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro unidirectional-available-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro bidirectional-delay?          uint32
    +--ro bidirectional-min-delay?      uint32
    +--ro bidirectional-max-delay?      uint32
    +--ro bidirectional-delay-variation? uint32
    +--ro bidirectional-packet-loss?    decimal64
    +--ro bidirectional-residual-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro bidirectional-available-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro bidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-fl
oat32
    +--ro utilized-percentage?          uint8
    +--ro te-grouped-params*            -> /te:te/tunnels/tunnel/t
e-
  kpi:te-telemetry/id
    +--ro grouping-operation?           grouping-operation

```

### 3.3. Integration

ACTN provides mechanisms to correlate customer's VN and the actual TE tunnels instantiated in the provider's network. Specifically,

- o Link each VN member to actual TE tunnel
- o Each VN can be monitored on a various level such as VN level, VN member level, TE-tunnel level, and link/node level.

Service function integration with network topology (L3 and TE topology) is in progress in [sf-topology]. Specifically, [sf-topology] addresses a number of use-cases that how TE topology supports various service functions.

### 3.4. Dynamic Configuration

ACTN provides an architecture that allows the customer network controller (CNC) interacts with the MDSC which is network provider's SDN controller in such a way that customer is given the control of their VNs.

Specifically, the ACTN VN model [actn-vn] allows the following capabilities:

- o Dynamic control over VN the customer creates.
- o Create, Modify, Delete

See the following tree structure from [actn-vn] as an example for the dynamic configuration capability (write) VN creation, modify and delete. VN can be dynamically created/modified/deleted with constraints such as metric types (e.g., delay), bandwidth, protection, etc.

```

+--rw vn
  +--rw vn-list* [vn-id]
    +--rw vn-id                uint32
    +--rw vn-name?             string
    +--rw vn-topology-id?      te-types:te-topology-id
    +--rw abstract-node?      -> /nw:networks/network/node/tet:te-node-i
d
    +--rw vn-member-list* [vn-member-id]
      | +--rw vn-member-id      uint32
      | +--rw src
      | | +--rw src?           -> /actn/ap/access-point-list/access-point
-id
      | | +--rw src-vn-ap-id?   -> /actn/ap/access-point-list/vn-ap/vn-ap-
id
      | | +--rw multi-src?      boolean {multi-src-dest}?
      | +--rw dest
      | | +--rw dest?          -> /actn/ap/access-point-list/access-poin
t-id
      | | +--rw dest-vn-ap-id?  -> /actn/ap/access-point-list/vn-ap/vn-ap
-id
      | | +--rw multi-dest?      boolean {multi-src-dest}?
      | +--rw connetivity-matrix-id? -> /nw:networks/network/node/tet:te/
te-
node-attributes/connectivity-matrices/connectivity-matrix/id
      | +--ro oper-status?      identityref
      +--ro if-selected?        boolean {multi-src-dest}?
      +--rw admin-status?       identityref
      +--ro oper-status?        identityref
      +--rw vn-level-diversity?  vn-disjointness

```

### 3.5. Customized Control Plane

ACTN provides a YANG model that allows the customer network controller (CNC) to control VN via type 2 operation. Type 2 VN allows the customer to provision pertinent LSPs that connect their endpoints over the customized VN topology dynamically.

See the following tree structure from [actn-vn] as an example for the provisioning of LSPs over the VN topology via TE-topology's [TE-Topo] Connectivity Matrix's construct.

For some VN members of a VN, the customers are allowed to configure the actual path (i.e., detailed virtual nodes and virtual links) over the VN/abstract topology agreed mutually between CNC and MDSC prior to or a topology created by the MDSC as part of VN instantiation. Type 2 VN is always built on top of a Type 1 VN. If a Type 2 VN is desired for some or all of VN members of a type 1 VN (see the example in Section 2.1 of [ACTN-VN]), the TE-topology model can provide the following abstract topology (that consists of virtual nodes and virtual links) which is built on top of the Type 1 VN so that customers can configure path over this topology.

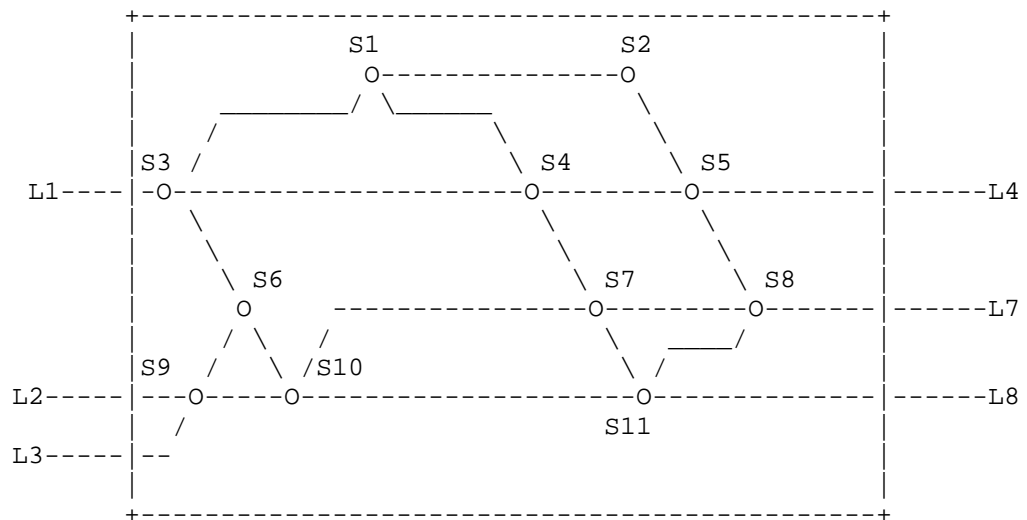


Figure 3. Type 2 topology

### 3.6. The Gaps

ACTN allows the customers/users to subscribe and monitor VN/Tunnel level performance data such as latency. The low level latency and isolation characteristics that are sought by some VPN+ users such as steering packets through specific queues resources are not in the scope of ACTN.

This implies that the device-level performance data such as queuing delay caused by various queuing mechanisms needs to be characterized and monitored by a device level YANG PM model. Then the Domain SDN controller (PNC) will need to estimate Domain LSP level PM data from

device-level PM data. Finally, the MDSC will need to derive VN/Tunnel level PM data and present to the customers.

Another gap that needs to be filled up is how to coordinate non-TE element from the routing and signaling standpoints. Currently, ACTN is limited to TE elements. From an end-to-end network standpoint, the scope of VPN+ may encompass non-TE elements in some segments/domains as well as TE elements. How to seamlessly provide end-to-end tunnel management and the operations of abstraction of resources across non-TE and TE elements of the network will need to be worked out further.

#### 4. Security Considerations

Virtual network instantiation involves the control of network resources in order to meet the service requirements of consumers. In some deployment models, the consumer is able to directly request modification in the behaviour of resources owned and operated by a service provider. Such changes could significantly affect the service provider's ability to provide services to other consumers. Furthermore, the resources allocated for or consumed by a consumer will normally be billable by the service provider.

Therefore, it is crucial that the mechanisms used in any virtual network system allow for authentication of requests, security of those requests, and tracking of resource allocations.

It should also be noted that while the partitioning of resources is virtual, the consumers expect and require that there is no risk of leakage of data from one slice to another, no transfer of knowledge of the structure or even existence of other virtual networks, and that changes to one virtual network (under the control of one consumer) should not have detrimental effects on the operation of other virtual networks (whether under control of different or the same consumers) beyond the limits allowed within the SLA. Thus, virtual networks are assumed to be private and to provide the appearance of genuine physical connectivity.

ACTN operates using the [netconf] or [restconf] protocols and assumes the security characteristics of those protocols. Deployment models for ACTN should fully explore the authentication and other security aspects before networks start to carry live traffic.

#### 5. IANA Considerations

This document has no actions for IANA.

## 6. Acknowledgements

Thanks to James Guichard, Stewart Bryant, Dong Jie for their insight and useful discussions about VPN+.

## 7. References

### 7.1. Informative References

- [actn-framework] Ceccarelli, D. and Y. Lee, "Framework for Abstraction and Control of Traffic Engineered Networks", draft-ietf-teas-actn-framework, work in progress, February 2017.
- [te-service-map] Y. Lee, D. Dhody, and D. Ceccarelli, "Traffic Engineering and Service Mapping Yang Model", draft-lee-teas-te-service-mapping-yang, work in progress.
- [actn-vn] Y. Lee (Editor), "A Yang Data Model for ACTN VN Operation", draft-lee-teas-actn-vn-yang, work in progress.
- [actn-info] Y. Lee, S. Belotti (Editors), "Information Model for Abstraction and Control of TE Networks (ACTN)", draft-ietf-teas-actn-info-model, work in progress.
- [actn-pm-elemetry] Y. Lee, et al, "YANG models for ACTN TE Performance Monitoring Telemetry and Network Autonomics", draft-lee-teas-actn-pm-telemetry-autonomics, work in progress.
- [vpn+] S. Bryant, and D. Jie, "Enhanced Virtual Private Networks (VPN+)", draft-bryant-rtgwg-enhanced-vpn, work in progress.
- [TE-Tunnel] T. Saad (Editor), "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te, work in progress.
- [TE-topo] X. Liu, et. al, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo, work in progress.
- [L3SM] S. Litkowski, L.Tomotaki, and K. Ogaki, "YANG Data Model for L3VPN service delivery", draft-ietf-l3sm-l3vpn-service-model, work in progress.

[L2SM] B. Wen, et al, "A YANG Data Model for L2VPN Service Delivery", draft-ietf-l2sm-l2vpn-service-model, work in progress.

[L1CSM] G. Fioccola, et al, "A Yang Data Model for L1 Connectivity Service Model (L1CSM)", draft-fioccola-ccamp-l1csm-yang, work in progress.

## 8. Contributors

### Authors' Addresses

Daniel King  
Lancaster University  
Email: d.king@lancaster.ac.uk

Young Lee (Editor)  
Huawei  
Phone: (469)277-5838  
Email: leeyoung@huawei.com

Jeff Tansura  
Futurewei  
Email: jefftant.ietf@gmail.com

Qin Wu  
Huawei Technologies Co.,Ltd.  
Email: bill.wu@huawei.com

Daniele Ceccarelli  
Ericsson  
Email: daniele.ceccarelli@ericsson.com



RTGWG Working Group  
Internet-Draft  
Intended status: Informational  
Expires: August 29, 2020

G. Mirsky  
ZTE Corp.  
February 26, 2020

Identification of Overlay Operations, Administration, and Maintenance  
(OAM)  
draft-mirsky-rtgwg-oam-identify-04

Abstract

This document analyzes how the presence of Operations, Administration, and Maintenance (OAM) control command and/or special data is identified in some overlay networks and an impact on the choice of identification may have on OAM functionality.

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 29, 2020.

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 (<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. Conventions used in this document . . . . .	2
2.1. Terminology . . . . .	2
2.2. Keywords . . . . .	3
3. A Control Channel in an Overlay Network . . . . .	3
4. Overlay Network Encapsulations . . . . .	4
4.1. Encapsulations with Meta-data . . . . .	4
4.1.1. Available Solutions . . . . .	6
4.2. Fixed-size Encapsulations . . . . .	6
4.3. Source Information Availability . . . . .	7
4.4. On-path OAM . . . . .	7
5. Conclusions . . . . .	8
6. IANA Considerations . . . . .	8
7. Security Considerations . . . . .	8
8. Acknowledgment . . . . .	8
9. References . . . . .	8
9.1. Normative References . . . . .	9
9.2. Informational References . . . . .	9
Author's Address . . . . .	11

## 1. Introduction

Operations, Administration, and Maintenance (OAM) protocols are used to detect, localize defects in the network, and monitor network performance. Some OAM functions, e.g., failure detection, work in the network proactively, while others, e.g., defect localization, usually performed on-demand. These tasks achieved by a combination of active, passive, and hybrid OAM methods, as defined in [RFC7799].

This document analyzes how the presence of Operations, Administration, and Maintenance (OAM) control command and/or special data, i.e., OAM packet, is identified in some overlay networks, and an impact the choice of identification may have on OAM functionality of active and hybrid OAM methods for the respective overlay network encapsulation.

## 2. Conventions used in this document

## 2.1. Terminology

AMM Alternate Marking method

BIER Bit Indexed Explicit Replication

DetNet Deterministic Networks

GUE Generic UDP Encapsulation

HTS Hybrid Two-step

NSH Network Service Header

NVO3 Network Virtualization Overlays

OAM Operations, Administration and Maintenance

SFC Service Function Chaining

TLV Type-Length-Value

VXLAN-GPE Generic Protocol Extension for VXLAN

ACH Associated Channed Header

Underlay Network or Underlay Layer: The network that provides connectivity between the DetNet nodes. MPLS network that provides LSP connectivity between DetNet nodes is an example of an underlay layer.

## 2.2. 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.

## 3. A Control Channel in an Overlay Network

There's a need for a general control channel between the endpoints of an overlay network for OAM protocols that can be used for fault detection, diagnostics, maintenance, and other functions. Such a control tunnel is dedicated to carrying only control and management data between tunnel endpoints. In other words, the control channel of an overlay network SHOULD NOT carry the client's data. And the endpoint node SHOULD NOT forward a packet received over the control channel. The identification of the control channel might be using different methods. For example, Virtual Network Identifier might be used to identify the control channel in VXLAN and Geneve.

#### 4. Overlay Network Encapsulations

New overlay network encapsulations analyzed in two groups:

- o encapsulations that support optional meta-data;
- o fixed-size encapsulations.

##### 4.1. Encapsulations with Meta-data

Number of the new encapsulation protocols (e.g., Geneve [I-D.ietf-nvo3-geneve], GUE [I-D.ietf-intarea-gue], and SFC NSH [RFC8300]) support use of Type-Length-Value (TLV) encoding to include optional information into the header. The identification of OAM in these protocols is as the following:

Geneve:

O (1 bit): after the WGLC discussion, the interpretation of the O field has changed. The O field now identifies a control packet. This packet contains a control message. Control messages are sent between tunnel endpoints. Tunnel Endpoints MUST NOT forward the payload and transit devices MUST NOT attempt to interpret it. Since these are infrequent control messages, it is RECOMMENDED that tunnel endpoints direct these packets to a high priority control queue (for example, to direct the packet to a general purpose CPU from a forwarding ASIC or to separate out control traffic on a NIC). Transit devices MUST NOT alter forwarding behavior on the basis of this bit, such as ECMP link selection.

[I-D.mmabb-nvo3-geneve-oam] defines the Geneve encapsulation for active OAM. Initially, four options have been presented:

- + with IP/UDP header demultiplexing active OAM protocols, e.g., Fault Management and Performance Monitoring, can be done using the destination UDP port number.
- + demultiplex active OAM protocols by the value of the Protocol Type field in the Geneve header.
- + with using MPLS Generic Associated Channel Label [RFC5586] and Associated Channel Header (ACH) [RFC4385]. Active OAM protocols are demultiplexed using the value of the Channel Type field.

- + using the new EtherType to identify Geneve OAM and the ACH. Active OAM protocols will be demultiplexed based on the Channel Type field's value.

#### GUE:

C-bit provides the separate namespace to carry formatted data that are implicitly addressed to the decapsulator to monitor or control the state or behavior of a tunnel. The payload is interpreted as a control message with the type specified in the proto/ctype field. The format and contents of the control message are indicated by the type and can be variable length.

#### SFC NSH:

0 bit: Setting this bit indicates an OAM packet.

Common between Geneve and NSH is the use of the dedicated flag to identify the OAM packet and, at the same time, the presence of the field that identifies the protocol of the payload that immediately follows after the encapsulation header. [RFC8393] points out that if the value of that field interpreted as none, i.e., no payload follows the header, then OAM may be included in TLVs, thus creating an active OAM packet. The problem with this mechanism to support active OAM methods may be a limitation of the size of data that can be included in a TLV. For example, the maximum size of data in an NSH Meta-data Type 2, as defined in section 2.5.1 [RFC8300], is 512 octets. The maximum length of data in Geneve Option, per section 3.5 [I-D.ietf-nvo3-geneve], is 128 octets. Thus, using one TLV as active OAM packet, would not allow creating test packets of larger size, which is useful when measuring packet loss and latency with synthetic traffic as part of the service activation procedure.

[I-D.ietf-sfc-oam-framework] suggests that the 0 bit used to identify OAM packet and the Next Protocol field identifies the OAM function:

While the presence of OAM marker in the overlay header (e.g., 0 bit in the NSH header) indicates it as OAM packet, it is not sufficient to signal for which OAM function the packet is intended.

At the same time, some of in-situ OAM proposals, e.g., [I-D.ietf-sfc-ioam-nsh], suggest using TLV to communicate hybrid OAM commands and data. The proposed resolution of using the combination of 0 bit and the Next Protocol field:

... the O bit MUST NOT be set for regular customer traffic which also carries IOAM data and the O bit MUST be set for OAM packets which carry only IOAM data without any regular data payload.

implies that the O bit only identifies the active OAM packet and not set when hybrid OAM methods used.

#### 4.1.1. Available Solutions

One of the possible solutions for encapsulations with meta-data has been specified in [I-D.ietf-sfc-multi-layer-oam]:

To identify the active OAM message the value on the Next Protocol field MUST be set to Active SFC OAM. The rules of interpreting the values of O bit and the Next Protocol field are as follows:

- o O bit set and the Next Protocol value is not one of identifying active or hybrid OAM protocol (per [RFC7799] definitions), e.g., defined in this specification Active SFC OAM - a Fixed-Length Context Header or Variable-Length Context Header(s) contain OAM command or data and the type of payload determined by the Next Protocol field;
- o O bit set and the Next Protocol value is one of identifying active or hybrid OAM protocol - the payload that immediately follows SFC NSH contains OAM command or data;
- o O bit is clear - no OAM in a Fixed-Length Context Header or Variable-Length Context Header(s) and the payload determined by the value of the Next Protocol field;
- o O bit is clear, and the Next Protocol value is one of identifying active or hybrid OAM protocol MUST be identified and reported as the erroneous combination. An implementation MAY have control to enable processing of the OAM payload.

From the above-listed rules follows the recommendation to avoid the combination of OAM in a Fixed-Length Context Header or Variable-Length Context Header(s) and in the payload immediately following the SFC NSH because there is no unambiguous way to identify such combination using the O bit and the Next Protocol field.

#### 4.2. Fixed-size Encapsulations

Number of the new encapsulation protocols (e.g., VXLAN-GPE [I-D.ietf-nvo3-vxlan-gpe], BIER [RFC8296]) use fixed-size header. The identification of OAM in these protocols is as the following:

**VXLAN-GPE:**

OAM Flag Bit (O bit): The O bit is set to indicate that the packet is an OAM packet.

**BIER:**

OAM packet identified by the value of the Next Protocol field. IANA BIER Next Protocol Identifiers registry includes the identifier for OAM (5).

The use of a combination of OAM Flag Bit and the Next Protocol field in VXLAN-GPE requires clarification of the header interpretation when the OAM Flag Bit is set, and the value of the Next Protocol field is one of defined in section 3.2 of [I-D.ietf-nvo3-vxlan-gpe].

BIER encapsulation, defined in [RFC8296], identifies OAM message immediately following the BIER header by the value of the Next Protocol field.

**4.3. Source Information Availability**

Availability of the packet originator's source information is required for active two-way OAM, e.g., echo request/reply. In cases when the underlay network is IPv4/IPv6 the source information will be derived from the underlay. But when using MPLS underlay network encapsulation of an active OAM packet have to follow specific rules:

- o if available, use Sender ID in the overlay domain (example BFIR ID in BIER [RFC8296];
- o use IP/UDP encapsulation of an OAM packet in the overlay (similar to Section 4.3 [RFC8029]).

**4.4. On-path OAM**

In addition to active methods, OAM toolset may include methods that don't use specially constructed and injected in the network test packets. [RFC7799] defines OAM methods that are neither entirely active nor passive but are a combination of both as hybrid methods.

One of the examples of the hybrid OAM methods, in-situ OAM, mentioned in Section 4.1. Another example, Alternate Marking method (AMM) [RFC8321], enables on-path OAM functions, e.g., delay and loss measurements, using the data traffic. Because AMM impact on the network can be minimized, measured metrics can be correlated to the network conditions experienced by the specific service. Of all listed in Section 4, BIER allocated the field that may be used for

AMM, as discussed in [I-D.ietf-bier-pmmm-oam]. Applicability of AMM to other overlay protocols, i.e., SFC NSH discussed in [I-D.mirsky-sfc-pmamm], Geneve [I-D.fmm-nvo3-pm-alt-mark], and in IPv6 networks [I-D.fioccola-v6ops-ipv6-alt-mark], been actively discussed.

Hybrid Two-step (HTS), defined in [I-D.mirsky-ippm-hybrid-two-step], provides on-path collection and transport of the telemetry information. HTS enables accurate and consistent measurements by separating the measurement action from the transporting data while ensuring that the follow-up packet that carries the telemetry information does follow the data packet that had triggered the measurement.

## 5. Conclusions

OAM control commands and data may be present as part of the overlay encapsulation header or as a payload that follows the overlay network header. The recommendations:

- o OAM in the overlay header, if supported by the overlay network, identified by the dedicated flag. Use of this method as active OAM is possible, but functionality is limited.
- o OAM that follows the overlay header identified as payload type, e.g., by the value of the Next Protocol field.

## 6. IANA Considerations

This document does not propose any IANA consideration. This section may be removed.

## 7. Security Considerations

This document lists the OAM requirements for a DetNet domain and does not raise any security concerns or issues in addition to ones common to networking.

## 8. Acknowledgment

TBD

## 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>>.
- [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>>.

### 9.2. Informational References

- [I-D.fioccola-v6ops-ipv6-alt-mark]  
Fioccola, G., Velde, G., Cociglio, M., and P. Muley, "IPv6 Performance Measurement with Alternate Marking Method", draft-fioccola-v6ops-ipv6-alt-mark-01 (work in progress), June 2018.
- [I-D.fmm-nvo3-pm-alt-mark]  
Fioccola, G., Mirsky, G., and T. Mizrahi, "Performance Measurement (PM) with Alternate Marking in Network Virtualization Overlays (NVO3)", draft-fmm-nvo3-pm-alt-mark-03 (work in progress), October 2018.
- [I-D.ietf-bier-pmmm-oam]  
Mirsky, G., Zheng, L., Chen, M., and G. Fioccola, "Performance Measurement (PM) with Marking Method in Bit Index Explicit Replication (BIER) Layer", draft-ietf-bier-pmmm-oam-07 (work in progress), January 2020.
- [I-D.ietf-intarea-gue]  
Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", draft-ietf-intarea-gue-09 (work in progress), October 2019.
- [I-D.ietf-nvo3-geneve]  
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-14 (work in progress), September 2019.
- [I-D.ietf-nvo3-vxlan-gpe]  
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-09 (work in progress), December 2019.



- [I-D.ietf-sfc-ioam-nsh]  
Brockners, F. and S. Bhandari, "Network Service Header (NSH) Encapsulation for In-situ OAM (IOAM) Data", draft-ietf-sfc-ioam-nsh-02 (work in progress), September 2019.
- [I-D.ietf-sfc-multi-layer-oam]  
Mirsky, G., Meng, W., Khasnabish, B., and C. Wang, "Active OAM for Service Function Chains in Networks", draft-ietf-sfc-multi-layer-oam-04 (work in progress), November 2019.
- [I-D.ietf-sfc-oam-framework]  
Aldrin, S., Pignataro, C., Kumar, N., Krishnan, R., and A. Ghanwani, "Service Function Chaining (SFC) Operations, Administration and Maintenance (OAM) Framework", draft-ietf-sfc-oam-framework-11 (work in progress), September 2019.
- [I-D.mirsky-ippm-hybrid-two-step]  
Mirsky, G., Lingqiang, W., and G. Zhui, "Hybrid Two-Step Performance Measurement Method", draft-mirsky-ippm-hybrid-two-step-04 (work in progress), October 2019.
- [I-D.mirsky-sfc-pmamm]  
Mirsky, G., Fioccola, G., and T. Mizrahi, "Performance Measurement (PM) with Alternate Marking Method in Service Function Chaining (SFC) Domain", draft-mirsky-sfc-pmamm-09 (work in progress), December 2019.
- [I-D.mmbb-nvo3-geneve-oam]  
Mirsky, G., Xiao, M., Boutros, S., and D. Black, "OAM for use in GENEVE", draft-mmbb-nvo3-geneve-oam-01 (work in progress), January 2020.
- [RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, DOI 10.17487/RFC4385, February 2006, <<https://www.rfc-editor.org/info/rfc4385>>.
- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed., "MPLS Generic Associated Channel", RFC 5586, DOI 10.17487/RFC5586, June 2009, <<https://www.rfc-editor.org/info/rfc5586>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.

- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [RFC8393] Farrel, A. and J. Drake, "Operating the Network Service Header (NSH) with Next Protocol "None"", RFC 8393, DOI 10.17487/RFC8393, May 2018, <<https://www.rfc-editor.org/info/rfc8393>>.

## Author's Address

Greg Mirsky  
ZTE Corp.

Email: [gregimirsky@gmail.com](mailto:gregimirsky@gmail.com)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: January 3, 2019

H. Song, Ed.  
T. Zhou  
ZB. Li  
Huawei  
G. Fioccola  
Telecom Italia  
ZQ. Li  
China Mobile  
P. Martinez-Julia  
NICT  
L. Ciavaglia  
Nokia  
A. Wang  
China Telecom  
July 2, 2018

Toward a Network Telemetry Framework  
draft-song-ntf-02

Abstract

This document suggests the necessity of an architectural framework for network telemetry in order to meet the current and future network operation requirements. The defining characteristics of network telemetry shows a clear distinction from the conventional network OAM concept; hence the network telemetry demands new techniques and protocols. This document clarifies the terminologies and classifies the categories and components of a network telemetry framework. The requirements, challenges, existing solutions, and future directions are discussed for each category. The network telemetry framework and the taxonomy help to set a common ground for the collection of related works and put future technique and standard developments into perspective.

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 January 3, 2019.

#### Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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. Motivation . . . . .	3
1.1. Use Cases . . . . .	3
1.2. Challenges . . . . .	5
1.3. Glossary . . . . .	5
1.4. Network Telemetry . . . . .	6
2. The Necessity of a Network Telemetry Framework . . . . .	8
3. Network Telemetry Framework . . . . .	9
3.1. Existing Works Mapped in the Framework . . . . .	11
3.2. Management Plane Telemetry . . . . .	12
3.2.1. Requirements and Challenges . . . . .	12
3.2.2. Push Extensions for NETCONF . . . . .	13
3.2.3. gRPC Network Management Interface . . . . .	13
3.3. Control Plane Telemetry . . . . .	14
3.3.1. Requirements and Challenges . . . . .	14
3.3.2. BGP Monitoring Protocol . . . . .	14
3.4. Data Plane Telemetry . . . . .	15
3.4.1. Requirements and Challenges . . . . .	15
3.4.2. Technique Classification . . . . .	16
3.4.3. The IPFPM technology . . . . .	16
3.4.4. Dynamic Network Probe . . . . .	18
3.4.5. IP Flow Information Export (IPFIX) protocol . . . . .	18

3.4.6. In-Situ OAM . . . . .	18
3.5. External Data and Event Telemetry . . . . .	19
3.5.1. Requirements and Challenges . . . . .	19
4. Security Considerations . . . . .	20
5. IANA Considerations . . . . .	20
6. Contributors . . . . .	20
7. Acknowledgments . . . . .	20
8. References . . . . .	20
8.1. Normative References . . . . .	20
8.2. Informative References . . . . .	20
Authors' Addresses . . . . .	23

## 1. Motivation

The advance of AI/ML technologies gives networks an unprecedented opportunity to realize network autonomy with closed control loops. An intent-driven autonomous network is the logical next step for network evolution following SDN, aiming to reduce (or even eliminate) human labor, make the most efficient use of network resources, and provide better services more aligned with customer requirements. Although we still have a long way to reach the ultimate goal, the journey has started nevertheless.

The storage and computing technologies are already mature enough to be able to retain and process a huge amount of data and make real-time inference. Tools based on machine learning technologies and big data analytics are powerful in detecting and reacting on network faults, anomalies, and policy violations. In turn, the network policy updates for planning, intrusion prevention, optimization, and self-healing can be applied. Some tools can even predict future events based on historical data.

However, the networks fail to keep pace with such data need. The current network architecture, protocol suite, and system design are not ready yet to provide enough quality data. In the remaining of this section, first we identify a few key network operation use cases that network operators need the most. These use cases are also the essential functions of the future autonomous networks. Next, we show why the current network OAM techniques and protocols are not sufficient to meet the requirements of these use cases. The discussion underlines the need of a new brood of techniques and protocols which we put under an umbrella term - network telemetry.

### 1.1. Use Cases

All these use cases involves the data extracted from the network data plane and sometimes from the network control plane and management plane.

**Intent and Policy Compliance:** Network policies are the rules that constraint the services for network access, provide differentiate within a service, or enforce specific treatment on the traffic. For example, a service function chain is a policy that requires the selected flows to pass through a set of network functions in order. An intents is a high-level abstract policy which requires a complex translation and mapping process before being applied on networks. While a policy is enforced, the compliance needs to be verified and monitored continuously.

**SLA Compliance:** A Service-Level Agreement (SLA) defines the level of service a user expects from a network operator, which include the metrics for the service measurement and remedy/penalty procedures when the service level misses the agreement. Users need to check if they get the service as promised and network operators need to evaluate how they can deliver the services that can meet the SLA.

**Root Cause Analysis:** Network failure often involves a sequence of chained events and the source of the failure is not straightforward to identify, especially when the failure is sporadic. While machine learning or other data analytics technologies can be used for root cause analysis, it up to the network to provide all the relevant data for analysis.

**Load Balancing, Traffic Engineering, and Network Planning:** Network operators are motivated to optimize their network utilization for better ROI or lower CAPEX, as well as differentiation across services and/or users of a given service. The first step is to know the real-time network conditions before applying policies to steer the user traffic or adjust the load balancing algorithm. In some cases network micro-bursts need to be detected in a very short time-frame so that fine grained traffic control can be applied to avoid possible network congestion. The long term network capacity planning and topology augmentation also rely on the accumulated data of the network operation.

**Event Tracking and Prediction:** Network visibility is critical for a healthy network operation. Numerous network events are of interest to network operators. For example, Network operators always want to learn where and why packets are dropped for an application flow. They also want to be warned by some early signs that some component is going to fail so the proper fix or replacement can be made in time.

## 1.2. Challenges

The conventional OAM techniques, as described in [RFC7276], are not sufficient to support the above use cases for the following reasons:

- o Most use cases need to continuously monitor the network and dynamically refine the data collection in real-time and interactively. The poll-based low-frequency data collection is ill-suited for these applications. Streaming data directly pushed from the data source is preferred.
- o Various data is needed from any place ranging from the packet processing engine to the QoS traffic manager. Traditional data plane devices cannot provide the necessary probes. An open and programmable data plane is therefore needed.
- o Many application scenarios need to correlate data from multiple sources (e.g., from distributed nodes or from different network plane). A piecemeal solution is often lacking the capability to consolidate the data from multiple sources. The composition of a complete solution, as partly proposed by ARCA [I-D.pedro-nmrg-anticipated-adaptation], will be empowered and guided by a comprehensive framework.
- o The passive measurement techniques can either consume too much network resources and render too much redundant data, or lead to inaccurate results. The active measurement techniques are indirect, and they can interfere with the user traffic. We need techniques that can collect direct and on-demand data from user traffic.

## 1.3. Glossary

Before further discussion, we list some key terminology and acronyms used in this documents. We make an intended distinction between network telemetry and network OAM.

AI: Artificial Intelligence. Use machine-learning based technologies to automate network operation.

BMP: BGP Monitoring Protocol

DNP: Dynamic Network Probe

DPI: Deep Packet Inspection

gNMI: gPRC Network Management Interface

gRPC: gRPC Remote Procedure Call

IDN: Intent-Driven Network

IPFIX: IP Flow Information Export Protocol

IPFPM: IP Flow Performance Measurement

IOAM: In-situ OAM

NETCONF: Network Configuration Protocol

Network Telemetry: A general term for a new brood of network visibility techniques and protocols, with the characteristics defined in this document. Network telemetry enables smooth evolution toward intent-driven autonomous networks.

NMS: Network Management System

OAM: Operations, Administration, and Maintenance. A group of network management functions that provide network fault indication, fault localization, performance information, and data and diagnosis functions. Most conventional network monitoring techniques and protocols belong to network OAM.

SNMP: Simple Network Management Protocol

YANG: A data modeling language for NETCONF

YANG FSM: A YANG model to define device side finite state machine

YANG PUSH: A method to subscribe pushed data from remote YANG datastore

#### 1.4. Network Telemetry

For a long time, network operators have relied upon protocols such as SNMP [RFC1157] to monitor the network. SNMP can only provide limited information about the network. Since SNMP is poll-based, it incurs low data rate and high processing overhead. Such drawbacks make SNMP unsuitable for today's automatic network applications.

Network telemetry has emerged as a mainstream technical term to refer to the newer techniques of data collection and consumption, distinguishing itself from the convention techniques for network OAM. It is expected that network telemetry can provide the necessary network visibility for autonomous networks, address the shortcomings



of conventional OAM techniques, and allow for the emergence of new techniques bearing certain characteristics.

One key difference between the network telemetry and the network OAM is that the network telemetry assumes an intelligent machine in the center of a closed control loop, while the network OAM assumes the human network operators in the middle of an open control loop. The network telemetry can directly trigger the automated network operation; The conventional OAM tools only help human operators to monitor and diagnose the networks and guide manual network operations. The different assumptions lead to very different techniques.

Although the network telemetry techniques are just emerging and subject to continuous evolution, several defining characteristics of network telemetry have been well accepted:

- o Push and Streaming: Instead of polling data from network devices, the telemetry collector subscribes to the streaming data pushed from the data source in network devices.
- o Volume and Velocity: The telemetry data is intended to be consumed by machine rather than by human. Therefore, the data volume is huge and the processing is often in realtime.
- o Normalization and Unification: Telemetry aims to address the overall network automation needs. The piecemeal solutions offered by the conventional OAM approach are no longer suitable. Efforts need to be made to normalize the data representation and unify the protocols.
- o Model-based: The data is model-based which allows applications to configure and consume data with ease.
- o Data Fusion: The data for a single application can come from multiple data sources (e.g., cross domain, cross device, and cross layer) and needs to be correlated to take effect.
- o Dynamic and Interactive: Since the network telemetry means to be used in a closed control loop for network automation, it needs to run continuously and adapt to the dynamic and interactive queries from the network operation controller.

In addition, the ideal network telemetry solution should also support the following features:

- o In-Network Customization: The data can be customized in network at run-time to cater to the specific need of applications. This

needs the support of a programmable data plane which allows probes to be deployed at flexible locations.

- o Direct Data Plane Export: The data originated from data plane can be directly exported to the data consumer for efficiency, especially when the data bandwidth is large and the real-time processing is required.
- o In-band Data Collection: In addition to the passive and active data collection approaches, the new hybrid approach allows to directly collect data for any target flow on its entire forwarding path.
- o Non-intrusive: The telemetry system should not fall into the trap of the "observer effect". That is, it should not change the network behavior or affect the forwarding performance.

## 2. The Necessity of a Network Telemetry Framework

Big data analytics and machine-learning based AI technologies are applied for network operation automation, relying on abundant data from networks. The single-sourced and static data acquisition cannot meet the data requirements. It is desirable to have a framework that integrates multiple telemetry approaches from different layers, and allows flexible combinations for different applications. The framework will benefit application development for the following reasons.

- o The future autonomous networks will require a holistic view on network visibility. All the use cases and applications need to be supported uniformly and coherently under a single intelligent agent. Therefore, the protocols and mechanisms should be consolidated into a minimum yet comprehensive set. A telemetry framework can help to normalize the technique developments.
- o Network visibility presents multiple viewpoints. For example, the device viewpoint takes the network infrastructure as the monitoring object from which the network topology and device status can be acquired; the traffic viewpoint takes the flows or packets as the monitoring object from which the traffic quality and path can be acquired. An application may need to switch its viewpoint during operation. It may also need to correlate a service and its network experience to acquire the comprehensive information.
- o Applications require network telemetry to be elastic in order to efficiently use the network resource and reduce the performance impact. Routine network monitoring covers the entire network with

low data sampling rate. When issues arise or trends emerge, the telemetry data source can be modified and the data rate can be boosted.

- o Efficient data fusion is critical for applications to reduce the overall quantity of data and improve the accuracy of analysis.

So far, some telemetry related work has been done within IETF. However, this work is fragmented and scattered in different working groups. The lack of coherence makes it difficult to assemble a comprehensive network telemetry system and causes repetitive and redundant work.

A formal network telemetry framework is needed for constructing a working system. The framework should cover the concepts and components from the standardization perspective. This document clarifies the layers on which the telemetry is exerted and decomposes the telemetry system into a set of distinct components that the existing and future work can easily map to.

### 3. Network Telemetry Framework

Telemetry can be applied on the data plane, the control plane, and the management plane in a network, as well as other sources out of the network, as shown in Figure 1.

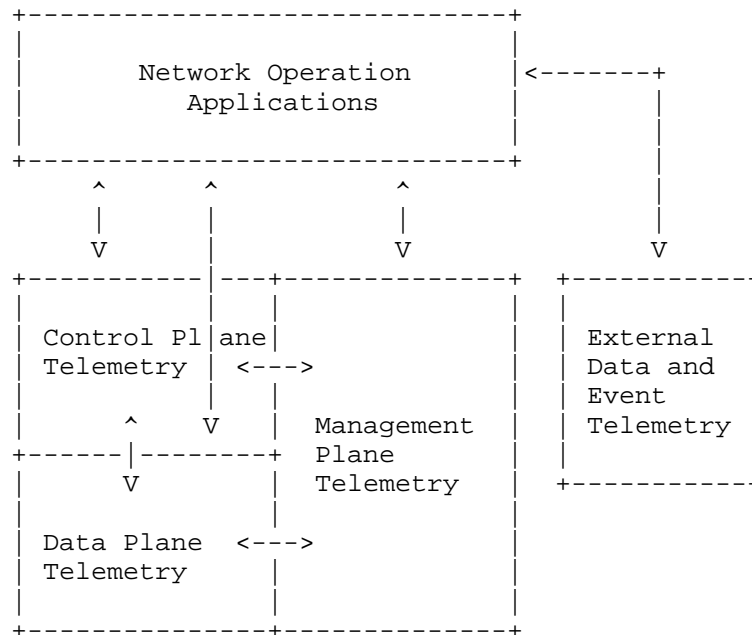


Figure 1: Layer Category of the Network Telemetry Framework

Note that the interaction with the network operation applications can be indirect. For example, in the management plane telemetry, the management plane may need to acquire data from the data plane. On the other hand, an application may involve more than one plane simultaneously. For example, an SLA compliance application may require both the data plane telemetry and the control plane telemetry.

At each plane, the telemetry can be further partitioned into five distinct components:

**Data Source:** Determine where the original data is acquired. The data source usually just provides raw data which needs further processing. A data source can be considered a probe. A probe can be statically installed or dynamically installed.

**Data Subscription:** Determine the protocol and channel for applications to acquire desired data. Data subscription is also responsible to define the desired data that might not be directly available from data sources. The subscription data can be described by a model. The model can be statically installed or dynamically installed.

**Data Generation:** The original data needs to be processed, encoded, and formatted in network devices to meet application subscription requirements. This may involve in-network computing and processing on either the fast path or the slow path in network devices.

**Data Export:** Determine how the ready data are delivered to applications.

**Data Analysis and Storage:** In this final step, data is consumed by applications or stored for future reference. Data analysis can be interactive. It may initiate further data subscription.

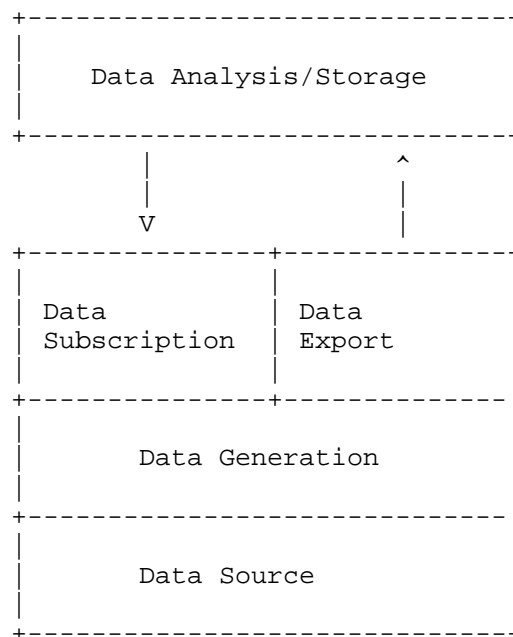


Figure 2: Components in the Network Telemetry Framework

Since most existing standard-related work belongs to the first four components, in the remainder of the document, we focus on these components only.

### 3.1. Existing Works Mapped in the Framework

The following table provides a non-exhaustive list of existing works (mainly published in IETF and with the emphasis on the latest new technologies) and shows their positions in the framework.

	Management Plane	Control Plane	Data Plane
Data Source	YANG Data Store	Control Proto. Network State	Flow/Packet Statistics States DPI
Data Subscribe	gPRC YANG PUSH	NETCONF/YANG BGP	NETCONF/YANG YANG FSM
Data Generation	Soft DNP	Soft DNP	In-situ OAM IPFPM Hard DNP
Data Export	gRPC YANG PUSH UDP	BMP	IPFIX UDP

Figure 3: Existing Work

### 3.2. Management Plane Telemetry

#### 3.2.1. Requirements and Challenges

The management plane of the network element interacts with the Network Management System (NMS), and provides information such as performance data, network logging data, network warning and defects data, and network statistics and state data. Some legacy protocols are widely used for the management plane, such as SNMP and Syslog, but these protocols do not meet the requirements of the automatic network operation applications.

New management plane telemetry protocols should consider the following requirements:

**Convenient Data Subscription:** An application should have the freedom to choose the data export means such as the data types and the export frequency.

Structured Data: For automatic network operation, machines will replace human for network data comprehension. The schema languages such as YANG can efficiently describe structured data and normalize data encoding and transformation.

High Speed Data Transport: In order to retain the information, a server needs to send a large amount of data at high frequency. Compact encoding formats are needed to compress the data and improve the data transport efficiency. The push mode, by replacing the poll mode, can also reduce the interactions between clients and servers, which help to improve the server's efficiency.

### 3.2.2. Push Extensions for NETCONF

NETCONF [RFC6241] is one popular network management protocol, which is also recommended by IETF. Although it can be used for data collection, NETCONF is good at configurations. YANG Push [I-D.ietf-netconf-yang-push] extends NETCONF and enables subscriber applications to request a continuous, customized stream of updates from a YANG datastore. Providing such visibility into changes made upon YANG configuration and operational objects enables new capabilities based on the remote mirroring of configuration and operational state. Moreover, distributed data collection mechanism [I-D.zhou-netconf-multi-stream-originators] via UDP based publication channel [I-D.ietf-netconf-udp-pub-channel] provides enhanced efficiency for the NETCONF based telemetry.

### 3.2.3. gRPC Network Management Interface

gRPC Network Management Interface (gNMI) [I-D.openconfig-rtgwg-gnmi-spec] is a network management protocol based on the gRPC [I-D.kumar-rtgwg-grpc-protocol] RPC (Remote Procedure Call) framework. With a single gRPC service definition, both configuration and telemetry can be covered. gRPC is an HTTP/2 [RFC7540] based open source micro service communication framework. It provides a number of capabilities that makes it well-suited for network telemetry, including:

- o Full-duplex streaming transport model combined with a binary encoding mechanism provided further improved telemetry efficiency.
- o gRPC provides higher-level features consistency across platforms that common HTTP/2 libraries typically do not. This characteristic is especially valuable for the fact that telemetry data collectors normally reside on a large variety of platforms.
- o The built-in load-balancing and failover mechanism.

### 3.3. Control Plane Telemetry

#### 3.3.1. Requirements and Challenges

The control plane telemetry refers to the health condition monitoring of different network protocols, which covers Layer 2 to Layer 7. Keeping track of the running status of these protocols is beneficial for detecting, localizing, and even predicting various network issues, as well as network optimization, in real-time and in fine granularities.

One of the most challenging problems for the control plane telemetry is how to correlate the E2E Key Performance Indicators (KPI) to a specific layer's KPIs. For example, an IPTV user may describe his User Experience (UE) by the video fluency and definition. Then in case of an unusually poor UE KPI or a service disconnection, it is non-trivial work to delimit and localize the issue to the responsible protocol layer (e.g., the Transport Layer or the Network Layer), the responsible protocol (e.g., ISIS or BGP at the Network Layer), and finally the responsible device(s) with specific reasons.

Traditional OAM-based approaches for control plane KPI measurement include PING (L3), Tracert (L3), Y.1731 (L2) and so on. One common issue behind these methods is that they only measure the KPIs instead of reflecting the actual running status of these protocols, making them less effective or efficient for control plane troubleshooting and network optimization. An example of the control plane telemetry is the BGP monitoring protocol (BMP), it is currently used to monitoring the BGP routes and enables rich applications, such as BGP peer analysis, AS analysis, prefix analysis, security analysis, and so on. However, the monitoring of other layers, protocols and the cross-layer, cross-protocol KPI correlations are still in their infancies (e.g., the IGP monitoring is missing), which require substantial further research.

#### 3.3.2. BGP Monitoring Protocol

BGP Monitoring Protocol (BMP) [RFC7854] is used to monitor BGP sessions and intended to provide a convenient interface for obtaining route views.

The BGP routing information is collected from the monitored device(s) to the BMP monitoring station by setting up the BMP TCP session. The BGP peers are monitored by the BMP Peer Up and Peer Down Notifications. The BGP routes (including Adjacency\_RIB\_In [RFC7854], Adjacency\_RIB\_out [I-D.ietf-grow-bmp-adj-rib-out], and Local\_Rib [I-D.ietf-grow-bmp-local-rib] are encapsulated in the BMP Route Monitoring Message and the BMP Route Mirroring Message, in the form



of both initial table dump and real-time route update. In addition, BGP statistics are reported through the BMP Stats Report Message, which could be either timer triggered or event driven. More BMP extensions can be explored to enrich the applications of BGP monitoring.

### 3.4. Data Plane Telemetry

#### 3.4.1. Requirements and Challenges

An effective data plane telemetry system relies on the data that the network device can expose. The data's quality, quantity, and timeliness must meet some stringent requirements. This raises some challenges to the network data plane devices where the first hand data originate.

- o A data plane device's main function is user traffic processing and forwarding. While supporting network visibility is important, the telemetry is just an auxiliary function and it should not impede normal traffic processing and forwarding (i.e., the performance is not lowered and the behavior is not altered due to the telemetry functions).
- o The network operation applications requires end-to-end visibility from various sources, which results in a huge volume of data. However, the sheer data quantity should not stress the network bandwidth, regardless of the data delivery approach (i.e., through in-band or out-of-band channels).
- o The data plane devices must provide the data in a timely manner with the minimum possible delay. Long processing, transport, storage, and analysis delay can impact the effectiveness of the control loop and even render the data useless.
- o The data should be structured and labeled, and easy for applications to parse and consume. At the same time, the data types needed by applications can vary significantly. The data plane devices need to provide enough flexibility and programmability to support the precise data provision for applications.
- o The data plane telemetry should support incremental deployment and work even though some devices are unaware of the system. This challenge is highly relevant to the standards and legacy networks.

The industry has agreed that the data plane programmability is essential to support network telemetry. Newer data plane chips are

all equipped with advanced telemetry features and provide flexibility to support customized telemetry functions.

### 3.4.2. Technique Classification

There can be multiple possible dimensions to classify the data plane telemetry techniques.

**Active and Passive:** The active and passive methods (as well as the hybrid types) are well documented in [RFC7799]. The passive methods include TCPDUMP, IPFIX [RFC7011], sflow, and traffic mirror. These methods usually have low data coverage. The bandwidth cost is very high in order to improve the data coverage. On the other hand, the active methods include Ping, Traceroute, OWAMP [RFC4656], and TWAMP [RFC5357]. These methods are intrusive and only provide indirect network measurement results. The hybrid methods, including in-situ OAM [I-D.brockners-inband-oam-requirements], IPFPM [RFC8321], and Multipoint Alternate Marking [I-D.fioccola-ippm-multipoint-alt-mark], provide a well-balanced and more flexible approach. However, these methods are also more complex to implement.

**In-Band and Out-of-Band:** The telemetry data, before being exported to some collector, can be carried in user packets. Such methods are considered in-band (e.g., in-situ OAM [I-D.brockners-inband-oam-requirements]). If the telemetry data is directly exported to some collector without modifying the user packets, Such methods are considered out-of-band (e.g., postcard-based INT). It is possible to have hybrid methods. For example, only the telemetry instruction or partial data is carried by user packets (e.g., IPFPM [RFC8321]).

**E2E and In-Network:** Some E2E methods start from and end at the network end hosts (e.g., Ping). The other methods work in networks and are transparent to end hosts. However, if needed, the in-network methods can be easily extended into end hosts.

**Flow, Path, and Node:** Depending on the telemetry objective, the methods can be flow-based (e.g., in-situ OAM [I-D.brockners-inband-oam-requirements]), path-based (e.g., Traceroute), and node-based (e.g., IPFIX [RFC7011]).

### 3.4.3. The IPFPM technology

The Alternate Marking method is efficient to perform packet loss, delay, and jitter measurements both in an IP and Overlay Networks, as

presented in IPFPM [RFC8321] and [I-D.fioccola-ippm-multipoint-alt-mark].

This technique can be applied to point-to-point and multipoint-to-multipoint flows. Alternate Marking creates batches of packets by alternating the value of 1 bit (or a label) of the packet header. These batches of packets are unambiguously recognized over the network and the comparison of packet counters for each batch allows the packet loss calculation. The same idea can be applied to delay measurement by selecting ad hoc packets with a marking bit dedicated for delay measurements.

Alternate Marking method needs two counters each marking period for each flow under monitor. For instance, by considering  $n$  measurement points and  $m$  monitored flows, the order of magnitude of the packet counters for each time interval is  $n*m*2$  (1 per color).

Since networks offer rich sets of network performance measurement data (e.g packet counters), traditional approaches run into limitations. One reason is the fact that the bottleneck is the generation and export of the data and the amount of data that can be reasonably collected from the network. In addition, management tasks related to determining and configuring which data to generate lead to significant deployment challenges.

Multipoint Alternate Marking approach, described in [I-D.fioccola-ippm-multipoint-alt-mark], aims to resolve this issue and makes the performance monitoring more flexible in case a detailed analysis is not needed.

An application orchestrates network performance measurements tasks across the network to allow an optimized monitoring and it can calibrate how deep can be obtained monitoring data from the network by configuring measurement points roughly or meticulously.

Using Alternate Marking, it is possible to monitor a Multipoint Network without examining in depth by using the Network Clustering (subnetworks that are portions of the entire network that preserve the same property of the entire network, called clusters). So in case there is packet loss or the delay is too high the filtering criteria could be specified more in order to perform a detailed analysis by using a different combination of clusters up to a per-flow measurement as described in IPFPM [RFC8321].

In summary, an application can configure initially an end to end monitoring between ingress points and egress points of the network. If the network does not experiment issues, this approximate monitoring is good enough and is very cheap in terms of network

resources. But, in case of problems, the application becomes aware of the issues from this approximate monitoring and, in order to localize the portion of the network that has issues, configures the measurement points more exhaustively. So a new detailed monitoring is performed. After the detection and resolution of the problem the initial approximate monitoring can be used again.

#### 3.4.4. Dynamic Network Probe

Hardware based Dynamic Network Probe (DNP) [I-D.song-opsawg-dnp4iq] provides a programmable means to customize the data that an application collects from the data plane. A direct benefit of DNP is the reduction of the exported data. A full DNP solution covers several components including data source, data subscription, and data generation. The data subscription needs to define the custom data which can be composed and derived from the raw data sources. The data generation takes advantage of the moderate in-network computing to produce the desired data.

While DNP can introduce unforeseeable flexibility to the data plane telemetry, it also faces some challenges. It requires a flexible data plane that can be dynamically reprogrammed at run-time. The programming API is yet to be defined.

#### 3.4.5. IP Flow Information Export (IPFIX) protocol

Traffic on a network can be seen as a set of flows passing through network elements. IP Flow Information Export (IPFIX) [RFC7011] provides a means of transmitting traffic flow information for administrative or other purposes. A typical IPFIX enabled system includes a pool of Metering Processes collects data packets at one or more Observation Points, optionally filters them and aggregates information about these packets. An Exporter then gathers each of the Observation Points together into an Observation Domain and sends this information via the IPFIX protocol to a Collector.

#### 3.4.6. In-Situ OAM

Traditional passive and active monitoring and measurement techniques are either inaccurate or resource-consuming. It is preferable to directly acquire data associated with a flow's packets when the packets pass through a network. In-situ OAM (ioAM) [I-D.brockners-inband-oam-requirements], a data generation technique, embeds a new instruction header to user packets and the instruction directs the network nodes to add the requested data to the packets. Thus, at the path end the packet's experience on the entire forwarding path can be collected. Such firsthand data is invaluable to many network OAM applications.

However, iOAM also faces some challenges. The issues on performance impact, security, scalability and overhead limits, encapsulation difficulties in some protocols, and cross-domain deployment need to be addressed.

### 3.5. External Data and Event Telemetry

Events that occur outside the boundaries of the network system are another important source of telemetry information. Correlating both internal telemetry data and external events with the requirements of network systems, as presented in Exploiting External Event Detectors to Anticipate Resource Requirements for the Elastic Adaptation of SDN/NFV Systems [I-D.pedro-nmrg-anticipated-adaptation], provides a strategic and functional advantage to management operations.

#### 3.5.1. Requirements and Challenges

As with other sources of telemetry information, the data and events must meet strict requirements, especially in terms of timeliness, which is essential to properly incorporate external event information to management cycles. Thus, the specific challenges are described as follows:

- o The role of external event detector can be played by multiple elements, including hardware (e.g. physical sensors, such as seismometers) and software (e.g. Big Data sources that analyze streams of information, such as Twitter messages). Thus, the transmitted data must support different shapes but, at the same time, follow a common but extensible ontology.
- o Since the main function of the external event detectors is actually to perform the notifications, their timeliness is assumed. However, once messages have been dispatched, they must be quickly collected and inserted into the control plane with variable priority, which will be high for important sources and/or important events and low for secondary ones.
- o The ontology used by external detectors must be easily adopted by current and future devices and applications. Therefore, it must be easily mapped to current information models, such as in terms of YANG.

Organizing together both internal and external telemetry information will be key for the general exploitation of the management possibilities of current and future network systems, as reflected in the incorporation of cognitive capabilities to new hardware and software (virtual) elements.

#### 4. Security Considerations

TBD

#### 5. IANA Considerations

This document includes no request to IANA.

#### 6. Contributors

The other main contributors of this document are listed as follows.

- o James N. Guichard, Huawei
- o Yunan Gu, Huawei

#### 7. Acknowledgments

We would like to thank Victor Liu and others who have provided helpful comments and suggestions to improve this document.

#### 8. References

##### 8.1. Normative References

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

##### 8.2. Informative 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., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

- [I-D.fioccola-ippm-multipoint-alt-mark]  
Fioccola, G., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate Marking method for passive and hybrid performance monitoring", draft-fioccola-ippm-multipoint-alt-mark-04 (work in progress), June 2018.

- [I-D.ietf-grow-bmp-adj-rib-out]  
Evens, T., Bayraktar, S., Lucente, P., Mi, K., and S. Zhuang, "Support for Adj-RIB-Out in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-adj-rib-out-01 (work in progress), March 2018.
- [I-D.ietf-grow-bmp-local-rib]  
Evens, T., Bayraktar, S., Bhardwaj, M., and P. Lucente, "Support for Local RIB in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-local-rib-01 (work in progress), February 2018.
- [I-D.ietf-netconf-udp-pub-channel]  
Zheng, G., Zhou, T., and A. Clemm, "UDP based Publication Channel for Streaming Telemetry", draft-ietf-netconf-udp-pub-channel-03 (work in progress), July 2018.
- [I-D.ietf-netconf-yang-push]  
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", draft-ietf-netconf-yang-push-17 (work in progress), July 2018.
- [I-D.kumar-rtgwg-grpc-protocol]  
Kumar, A., Kolhe, J., Ghemawat, S., and L. Ryan, "gRPC Protocol", draft-kumar-rtgwg-grpc-protocol-00 (work in progress), July 2016.
- [I-D.openconfig-rtgwg-gnmi-spec]  
Shakir, R., Shaikh, A., Borman, P., Hines, M., Lebsack, C., and C. Morrow, "gRPC Network Management Interface (gNMI)", draft-openconfig-rtgwg-gnmi-spec-01 (work in progress), March 2018.
- [I-D.pedro-nmrg-anticipated-adaptation]  
Martinez-Julia, P., "Exploiting External Event Detectors to Anticipate Resource Requirements for the Elastic Adaptation of SDN/NFV Systems", draft-pedro-nmrg-anticipated-adaptation-02 (work in progress), June 2018.
- [I-D.song-opsawg-dnp4iq]  
Song, H. and J. Gong, "Requirements for Interactive Query with Dynamic Network Probes", draft-song-opsawg-dnp4iq-01 (work in progress), June 2017.

- [I-D.zhou-netconf-multi-stream-originators]  
Zhou, T., Zheng, G., Voit, E., Clemm, A., and A. Bierman,  
"Subscription to Multiple Stream Originators", draft-zhou-  
netconf-multi-stream-originators-02 (work in progress),  
May 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>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M.  
Zekauskas, "A One-way Active Measurement Protocol  
(OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006,  
<<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J.  
Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)",  
RFC 5357, DOI 10.17487/RFC5357, October 2008,  
<<https://www.rfc-editor.org/info/rfc5357>>.
- [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>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken,  
"Specification of the IP Flow Information Export (IPFIX)  
Protocol for the Exchange of Flow Information", STD 77,  
RFC 7011, DOI 10.17487/RFC7011, September 2013,  
<<https://www.rfc-editor.org/info/rfc7011>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y.  
Weingarten, "An Overview of Operations, Administration,  
and Maintenance (OAM) Tools", RFC 7276,  
DOI 10.17487/RFC7276, June 2014,  
<<https://www.rfc-editor.org/info/rfc7276>>.
- [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>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with  
Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799,  
May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.



- [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>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

## Authors' Addresses

Haoyu Song (editor)  
Huawei  
2330 Central Expressway  
Santa Clara  
USA

Email: [haoyu.song@huawei.com](mailto:haoyu.song@huawei.com)

Tianran Zhou  
Huawei  
156 Beiqing Road  
Beijing, 100095  
P.R. China

Email: [zhoutianran@huawei.com](mailto:zhoutianran@huawei.com)

Zhenbin Li  
Huawei  
156 Beiqing Road  
Beijing, 100095  
P.R. China

Email: [lizhenbin@huawei.com](mailto:lizhenbin@huawei.com)

Giuseppe Fioccola  
Telecom Italia  
Via Reiss Romoli, 274  
Torino 10148  
Italy

Email: [giuseppe.fioccola@telecomitalia.it](mailto:giuseppe.fioccola@telecomitalia.it)

Zhenqiang Li  
China Mobile  
No. 32 Xuanwumenxi Ave., Xicheng District  
Beijing, 100032  
P.R. China

Email: lizhenqiang@chinamobile.com

Pedro Martinez-Julia  
NICT  
4-2-1, Nukui-Kitamachi  
Koganei, Tokyo 184-8795  
Japan

Phone: +81 42 327 7293  
Email: pedro@nict.go.jp

Laurent Ciavaglia  
Nokia  
Villardeaux 91460  
France

Email: laurent.ciavaglia@nokia.com

Aijun Wang  
China Telecom  
Beiqijia Town, Changping District  
Beijing, 102209  
P.R. China

Email: wangaj.bri@chinatelecom.cn

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: February 17, 2019

F. Templin, Ed.  
G. Saccone  
Boeing Research & Technology  
G. Dawra  
LinkedIn  
A. Lindem  
V. Moreno  
Cisco Systems, Inc.  
August 16, 2018

A Simple BGP-based Mobile Routing System for the Aeronautical  
Telecommunications Network  
draft-templin-atn-bgp-08.txt

Abstract

The International Civil Aviation Organization (ICAO) is investigating mobile routing solutions for a worldwide Aeronautical Telecommunications Network with Internet Protocol Services (ATN/IPS). The ATN/IPS will eventually replace existing communication services with an IPv6-based service supporting pervasive Air Traffic Management (ATM) for Air Traffic Controllers (ATC), Airline Operations Controllers (AOC), and all commercial aircraft worldwide. This informational document describes a simple and extensible mobile routing service based on industry-standard BGP to address the ATN/IPS requirements.

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 February 17, 2019.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	5
3. ATN/IPS Routing System . . . . .	6
4. ATN/IPS Radio Access Network (RAN) Model . . . . .	9
5. ATN/IPS Route Optimization . . . . .	11
6. BGP Protocol Considerations . . . . .	13
7. Implementation Status . . . . .	14
8. IANA Considerations . . . . .	14
9. Security Considerations . . . . .	14
10. Acknowledgements . . . . .	15
11. References . . . . .	15
11.1. Normative References . . . . .	15
11.2. Informative References . . . . .	16
Appendix A. Change Log . . . . .	17
Authors' Addresses . . . . .	17

## 1. Introduction

The worldwide Air Traffic Management (ATM) system today uses a service known as Aeronautical Telecommunications Network based on Open Systems Interconnection (ATN/OSI). The service is used to augment controller to pilot voice communications with rudimentary short text command and control messages. The service has seen successful deployment in a limited set of worldwide ATM domains.

The International Civil Aviation Organization [ICAO] is now undertaking the development of a next-generation replacement for ATN/OSI known as Aeronautical Telecommunications Network with Internet Protocol Services (ATN/IPS). ATN/IPS will eventually provide an IPv6-based [RFC8200] service supporting pervasive ATM for Air Traffic Controllers (ATC), Airline Operations Controllers (AOC), and all

commercial aircraft worldwide. As part of the ATN/IPS undertaking, a new mobile routing service will be needed. This document presents an approach based on the Border Gateway Protocol (BGP) [RFC4271].

Aircraft communicate via wireless aviation data links that typically support much lower data rates than terrestrial wireless and wired-line communications. For example, some Very High Frequency (VHF)-based data links only support data rates on the order of 32Kbps and an emerging L-Band data link that is expected to play a key role in future aeronautical communications only supports rates on the order of 1Mbps. Although satellite data links can provide much higher data rates during optimal conditions, like any other aviation data link they are subject to errors, delay, disruption, signal intermittence, degradation due to atmospheric conditions, etc. The well-connected ground domain ATN/IPS network should therefore treat each safety-of-flight critical packet produced by (or destined to) an aircraft as a precious commodity and strive for an optimized service that provides the highest possible degree of reliability.

The ATN/IPS is an IPv6-based overlay network that assumes a worldwide connected Internetworking underlay for carrying tunneled ATM communications. The Internetworking underlay could be manifested as a private collection of long-haul backbone links (e.g., fiber optics, copper, SATCOM, etc.) interconnected by high-performance networking gear such as bridges, switches, and routers. Such a private network would need to connect all ATN/IPS participants worldwide, and could therefore present a considerable cost for a large-scale deployment of new infrastructure. Alternatively, the ATN/IPS could be deployed as a secured overlay over the existing global public Internet. For example, ATN/IPS nodes could be deployed as part of an SD-WAN or an MPLS-WAN that rides over the public Internet via secured tunnels. Further details of the Internetworking underlay design are out of scope for this document.

The ATN/IPS further assumes that each aircraft will receive an IPv6 Mobile Network Prefix (MNP) that accompanies the aircraft wherever it travels. ICAO is further proposing to assign each aircraft an entire /56 MNP for numbering its on-board networks. ATCs and AOCs will likewise receive IPv6 prefixes, but they would typically appear in static (not mobile) deployments such as air traffic control towers, airline headquarters, etc. Throughout the rest of this document, we therefore use the term "MNP" when discussing an IPv6 prefix that is delegated to any ATN/IPS end system, including ATCs, AOCs, and aircraft. We also use the term Mobility Service Prefix (MSP) to refer to an aggregated prefix assigned to the ATN/IPS by an Internet assigned numbers authority, and from which all MNPs are delegated (e.g., up to  $2^{32}$  IPv6 /56 MNPs could be delegated from an IPv6 /24 MSP).

Connexion By Boeing [CBB] was an early aviation mobile routing service based on dynamic updates in the global public Internet BGP routing system. Practical experience with the approach has shown that frequent injections and withdrawals of MNPs in the Internet routing system can result in excessive BGP update messaging, slow routing table convergence times, and extended outages when no route is available. This is due to both conservative default BGP protocol timing parameters (see Section 6) and the complex peering interconnections of BGP routers within the global Internet infrastructure. The situation is further exacerbated by frequent aircraft mobility events that each result in BGP updates that must be propagated to all BGP routers in the Internet that carry a full routing table.

We therefore consider an approach using a BGP overlay network routing system where a private BGP routing protocol instance is maintained between ATN/IPS Autonomous System (AS) Border Routers (ASBRs). The private BGP instance does not interact with the native BGP routing system in the connected Internetworking underlay, and BGP updates are unidirectional from "stub" ASBRs (s-ASBRs) to a small set of "core" ASBRs (c-ASBRs) in a hub-and-spokes topology. No extensions to the BGP protocol are necessary.

The s-ASBRs for each stub AS connect to a small number of c-ASBRs via dedicated high speed links and/or tunnels across the Internetworking underlay using industry-standard encapsulations (e.g., Generic Routing Encapsulation (GRE) [RFC2784], IPsec [RFC4301], etc.). In particular, tunneling must be used when neighboring ASBRs are separated by many Internetworking underlay hops.

The s-ASBRs engage in external BGP (eBGP) peerings with their respective c-ASBRs, and only maintain routing table entries for the MNPs currently active within the stub AS. The s-ASBRs send BGP updates for MNP injections or withdrawals to c-ASBRs but do not receive any BGP updates from c-ASBRs. Instead, the s-ASBRs maintain default routes with their c-ASBRs as the next hop, and therefore hold only partial topology information.

The c-ASBRs connect to other c-ASBRs using internal BGP (iBGP) peerings over which they collaboratively maintain a full routing table for all active MNPs currently in service. Therefore, only the c-ASBRs maintain a full BGP routing table and never send any BGP updates to s-ASBRs. This simple routing model therefore greatly reduces the number of BGP updates that need to be synchronized among peers, and the number is reduced further still when intradomain routing changes within stub ASes are processed within the AS instead of being propagated to the core. BGP Route Reflectors (RRs) [RFC4456] can also be used to support increased scaling properties.

The remainder of this document discusses the proposed BGP-based ATN/IPS mobile routing service.

## 2. Terminology

The terms Autonomous System (AS) and Autonomous System Border Router (ASBR) are the same as defined in [RFC4271].

The following terms are defined for the purposes of this document:

**Air Traffic Management (ATM)**

The worldwide service for coordinating safe aviation operations.

**Air Traffic Controller (ATC)**

A government agent responsible for coordinating with aircraft within a defined operational region via voice and/or data Command and Control messaging.

**Airline Operations Controller (AOC)**

An airline agent responsible for tracking and coordinating with aircraft within their fleet.

**Aeronautical Telecommunications Network with Internet Protocol Services (ATN/IPS)**

A future aviation network for ATCs and AOCs to coordinate with all aircraft operating worldwide. The ATN/IPS will be an IPv6-based overlay network service that connects access networks via tunneling over an Internetworking underlay.

**Internetworking underlay** A connected wide-area network that supports overlay network tunneling and connects Radio Access Networks to the rest of the ATN/IPS.

**Radio Access Network (RAN)**

An aviation radio data link service provider's network, including radio transmitters and receivers as well as supporting ground-domain infrastructure needed to convey a customer's data packets to the outside world. The term RAN is intended in the same spirit as for cellular operator networks and other radio-based Internet service provider networks. For simplicity, we also use the term RAN to refer to ground-domain networks that connect AOCs and ATCs without any aviation radio communications.

**Core Autonomous System Border Router (c-ASBR)** A BGP router located in the hub of the ATN/IPS hub-and-spokes overlay network topology.

**Core Autonomous System** The "hub" autonomous system maintained by all c-ASBRs.

**Stub Autonomous System Border Router (s-ASBR)** A BGP router configured as a spoke in the ATN/IPS hub-and-spokes overlay network topology.

**Stub Autonomous System** A logical grouping that includes all Clients currently associated with a given s-ASBR.

**Client** An ATC, AOC or aircraft that connects to the ATN/IPS as a leaf node. The Client could be a singleton host, or a router that connects a mobile network.

**Proxy** A node at the edge of a RAN that acts as an intermediary between Clients and s-ASBRs. From the Client's perspective, the Proxy presents the appearance that the Client is communicating directly with the s-ASBR. From the s-ASBR's perspective, the Proxy presents the appearance that the s-ASBR is communicating directly with the Client.

**Mobile Network Prefix (MNP)** An IPv6 prefix that is delegated to any ATN/IPS end system, including ATCs, AOCs, and aircraft.

**Mobility Service Prefix (MSP)** An aggregated prefix assigned to the ATN/IPS by an Internet assigned numbers authority, and from which all MNPs are delegated (e.g., up to  $2^{32}$  IPv6 /56 MNPs could be delegated from a /24 MSP).

### 3. ATN/IPS Routing System

The proposed ATN/IPS routing system comprises a private BGP instance coordinated in an overlay network via tunnels between neighboring ASBRs over the Internetworking underlay. The overlay does not interact with the native BGP routing system in the connected underlying Internetwork, and each c-ASBR advertises only a small and unchanging set of MSPs into the Internetworking underlay routing system instead of the full dynamically changing set of MNPs. (For example, when the Internetworking underlay is the global public Internet the c-ASBRs advertise the MSPs in the public BGP Internet routing system.)

In a reference deployment, one or more s-ASBRs connect each stub AS to the overlay using a shared stub AS Number (ASN). Each s-ASBR further uses eBGP to peer with one or more c-ASBRs. All c-ASBRs are members of the same core AS, and use a shared core ASN. Globally-unique public ASNs could be assigned, e.g., either according to the standard 16-bit ASN format or the 32-bit ASN scheme defined in [RFC6793].



The c-ASBRs use iBGP to maintain a synchronized consistent view of all active MNPs currently in service. Figure 1 below represents the reference deployment. (Note that the figure shows details for only two s-ASBRs (s-ASBR1 and s-ASBR2) due to space constraints, but the other s-ASBRs should be understood to have similar Stub AS, MNP and eBGP peering arrangements.) The solution described in this document is flexible enough to extend to these topologies.

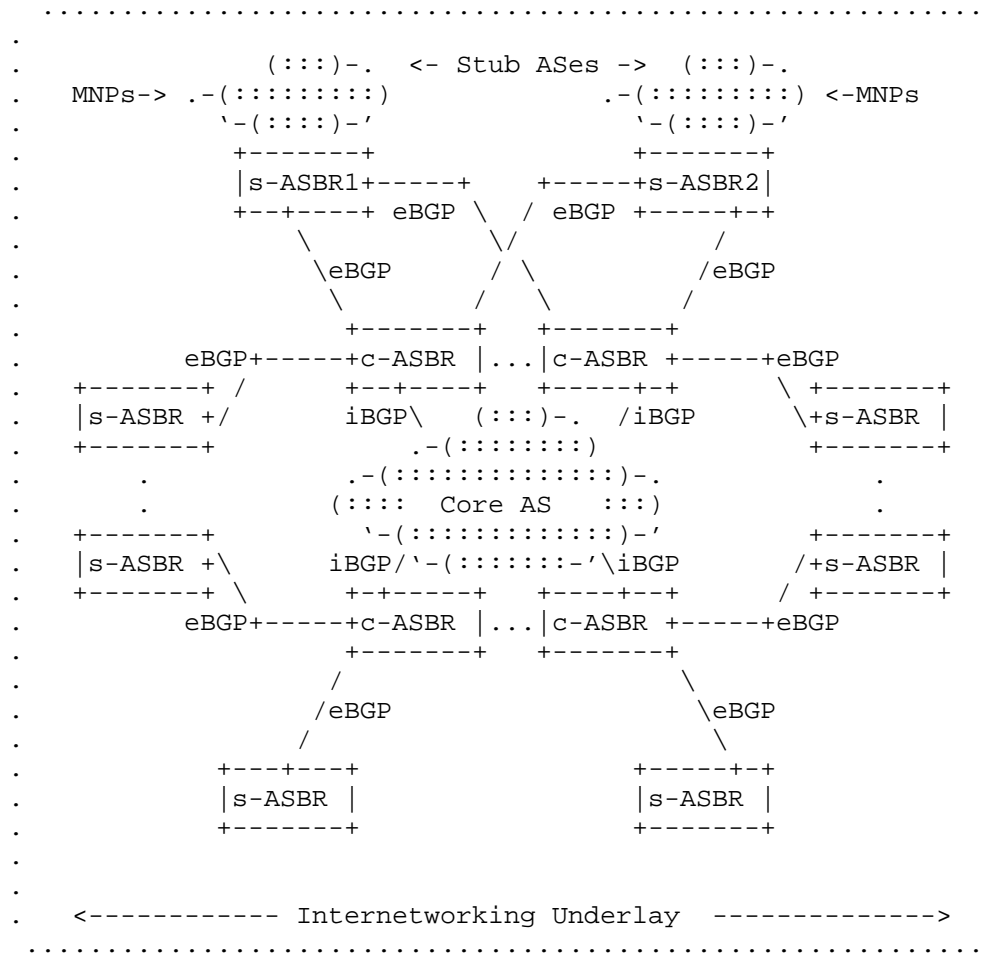


Figure 1: Reference Deployment

In the reference deployment, each s-ASBR maintains routes for active MNPs that currently belong to its stub AS. In response to "Inter-domain" mobility events, each s-ASBR will dynamically announces new MNPs and withdraws departed MNPs in its eBGP updates to c-ASBRs.

Since ATN/IPS end systems are expected to remain within the same stub AS for extended timeframes, however, intra-domain mobility events (such as an aircraft handing off between cell towers) are handled within the stub AS instead of being propagated as inter-domain eBGP updates.

Each c-ASBR configures a black-hole route for each of its MSPs. By black-holing the MSPs, the c-ASBR will maintain forwarding table entries only for the MNPs that are currently active, and packets destined to all other MNPs will correctly incur ICMPv6 Destination Unreachable messages [RFC4443] due to the black hole route. (This is the same behavior as for ordinary BGP routers in the Internet when they receive packets for which there is no route available.) The c-ASBRs do not send eBGP updates for MNPs to s-ASBRs, but instead originate a default route. In this way, s-ASBRs have only partial topology knowledge (i.e., they know only about the active MNPs currently within their stub ASes) and they forward all other packets to c-ASBRs which have full topology knowledge.

Scaling properties of this ATN/IPS routing system are limited by the number of BGP routes that can be carried by the c-ASBRs. A 2015 study showed that BGP routers in the global public Internet at that time carried more than 500K routes with linear growth and no signs of router resource exhaustion [BGP]. A more recent network emulation study also showed that a single c-ASBR can accommodate at least 1M dynamically changing BGP routes even on a lightweight virtual machine. Commercially-available high-performance dedicated router hardware can support many millions of routes.

Therefore, assuming each c-ASBR can carry 1M or more routes, this means that at least 1M ATN/IPS end system MNPs can be serviced by a single set of c-ASBRs and that number could be further increased by using RRs and/or more powerful routers. Another means of increasing scale would be to assign a different set of c-ASBRs for each set of MSPs. In that case, each s-ASBR still peers with one or more c-ASBRs from each set of c-ASBRs, but the s-ASBR institutes route filters so that it only sends BGP updates to the specific set of c-ASBRs that aggregate the MSP. In this way, each set of c-ASBRs maintains separate routing and forwarding tables so that scaling is distributed across multiple c-ASBR sets instead of concentrated in a single c-ASBR set. For example, a first c-ASBR set could aggregate an MSP segment A::/32, a second set could aggregate B::/32, a third could aggregate C::/32, etc. The union of all MSP segments would then constitute the collective MSP(s) for the entire ATN/IPS.

In this way, each set of c-ASBRs services a specific set of MSPs that they inject into the Internetworking underlay native routing system, and each s-ASBR configures MSP-specific routes that list the correct

set of c-ASBRs as next hops. This design also allows for natural incremental deployment, and can support initial medium-scale deployments followed by dynamic deployment of additional ATN/IPS infrastructure elements without disturbing the already-deployed base. For example, a few more c-ASBRs could be added if the MNP service demand ever outgrows the initial deployment.

#### 4. ATN/IPS Radio Access Network (RAN) Model

Radio Access Networks (RANs) connect end system Clients such as aircraft, ATCs, AOCs etc. to the ATN/IPS routing system. Clients may connect to multiple RANs at once, for example, when they have both satellite and cellular data links activated simultaneously. Clients may further move between RANs in a manner that is perceived as a network layer mobility event. Clients could therefore employ a multilink/mobility routing service such as that discussed in [I-D.templin-aerolink].

Clients register all of their active data link connections with their serving s-ASBRs as discussed in Section 3. Clients may connect to s-ASBRs either directly, or via a Proxy at the edge of the RAN.

Figure 2 shows the ATN/IPS RAN model where Clients connect to RANs via aviation data links. Clients register their RAN addresses with a nearby s-ASBR, where the registration process may be brokered by a Proxy at the edge of the RAN.

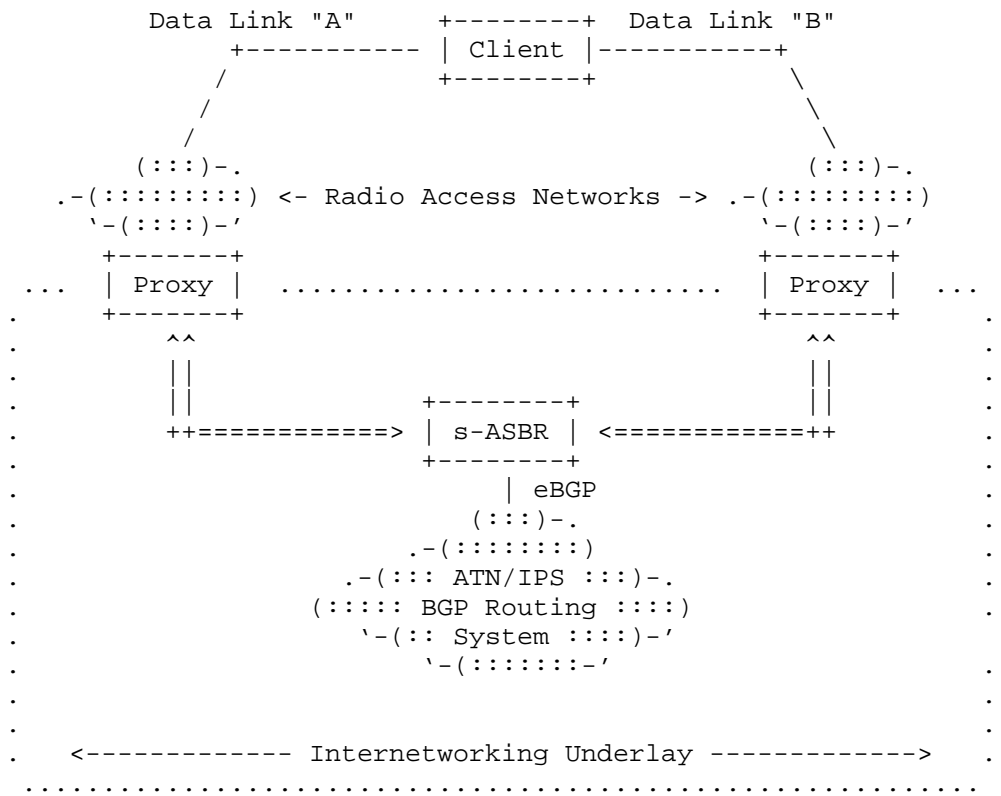


Figure 2: ATN/IPS RAN Architecture

When a Client logs into a RAN, it specifies a nearby s-ASBR that it has selected to connect to the ATN/IPS. The login process is brokered by a Proxy at the border of the RAN, which then conveys the connection request to the s-ASBR via tunneling across the Internetworking underlay. The s-ASBR then registers the address of the Proxy as the address for the Client, and the Proxy forwards the s-ASBR's reply to the Client. If the Client connects to multiple RANs, the s-ASBR will register the addresses of all Proxies as addresses through which the Client can be reached.

The s-ASBR represents all of its active Clients as MNP routes in the ATN/IPS BGP routing system. The s-ASBR's stub AS therefore consists of the set of all of its active Clients (i.e., the stub AS is a logical construct and not a physical construct). The s-ASBR injects the MNPs of its active Clients and withdraws the MNPs of its departed Clients via BGP updates to c-ASBRs. Since Clients are expected to remain associated with their current s-ASBR for extended periods, the level of MNP injections and withdrawals in the BGP routing system

will be on the order of the numbers of network joins, leaves and s-ASBR handovers for aircraft operations (see: Section 6). It is important to observe that fine-grained events such as Client mobility and Quality of Service (QoS) signaling are coordinated only by Proxies and the Client's current s-ASBRs, and do not involve other ASBRs in the routing system. In this way, intradomain routing changes within the stub AS are not propagated into the rest of the ATN/IPS BGP routing system.

## 5. ATN/IPS Route Optimization

ATN/IPS end systems will frequently need to communicate with correspondents associated with other s-ASBRs. In the BGP peering topology discussed in Section 3, this can initially only be accommodated by including multiple tunnel segments in the forwarding path. In many cases, it would be desirable to eliminate extraneous tunnel segments from this "dogleg" route so that packets can traverse a minimum number of tunneling hops across the Internetworking underlay. ATN/IPS end systems could therefore employ a route optimization service such as that discussed in [I-D.templin-aerolink].

A route optimization example is shown in Figure 3 and Figure 4 below. In the first figure, multiple tunneled segments between Proxys and ASBRs are necessary to convey packets between Clients associated with different s-ASBRs. In the second figure, the optimized route tunnels packets directly between Proxys without involving the ASBRs.

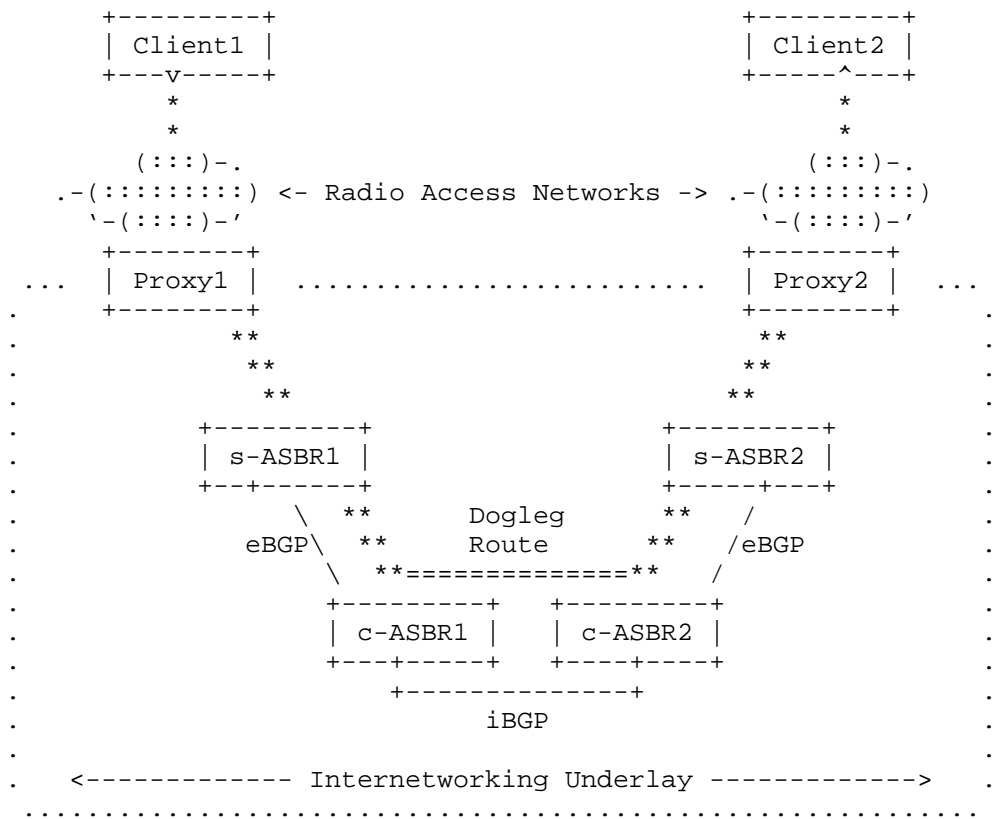


Figure 3: Dogleg Route Before Optimization

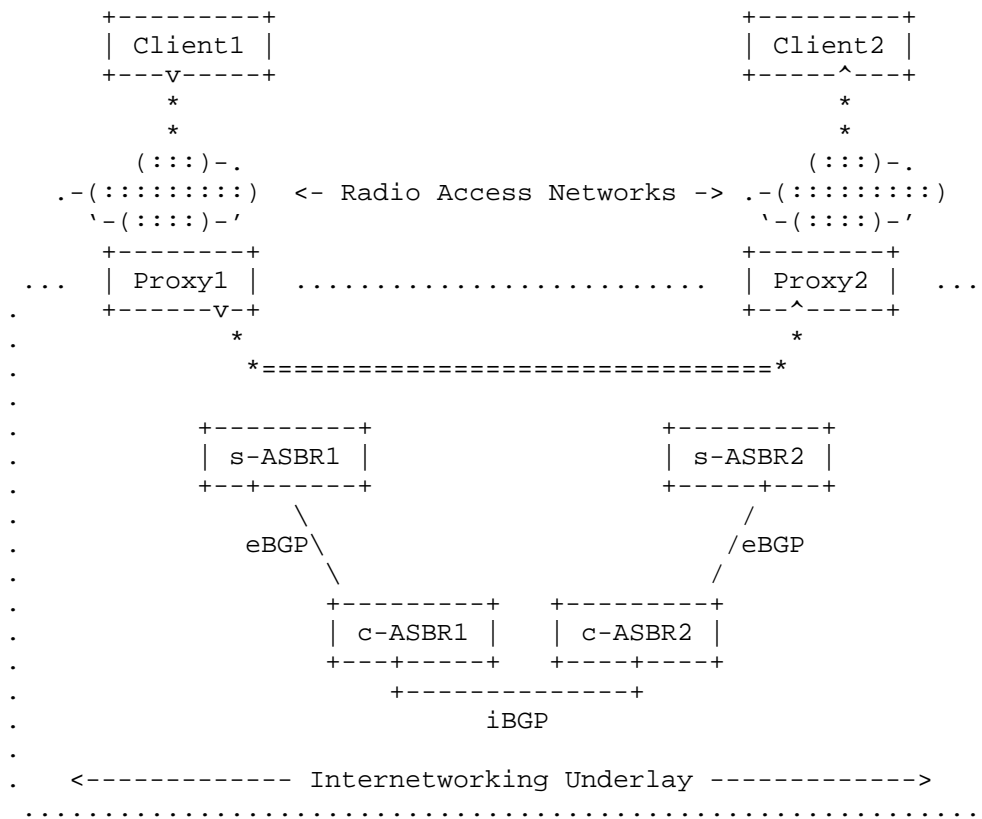


Figure 4: Optimized Route

## 6. BGP Protocol Considerations

The number of eBGP peering sessions that each c-ASBR must service is proportional to the number of s-ASBRs in the system. Network emulations with lightweight virtual machines have shown that a single c-ASBR can service at least 100 eBGP peerings from s-ASBRs that each advertise 10K MNP routes (i.e., 1M total). It is expected that robust c-ASBRs can service many more peerings than this - possibly by multiple orders of magnitude. But even assuming a conservative limit, the number of s-ASBRs could be increased by also increasing the number of c-ASBRs. Since c-ASBRs also peer with each other using iBGP, however, larger-scale c-ASBR deployments may need to employ an adjunct facility such as BGP Route Reflectors (RRs)[RFC4456].

The number of aircraft in operation at a given time worldwide is likely to be significantly less than 1M, but we will assume this number for a worst-case analysis. Assuming a worst-case average 1

hour flight profile from gate-to-gate with 10 service region transitions per flight, the entire system will need to service at most 10M BGP updates per hour (2778 updates per second). This number is within the realm of the peak BGP update messaging seen in the global public Internet today [BGP2]. Assuming a BGP update message size of 100 bytes (800bits), the total amount of BGP control message traffic to a single c-ASBR will be less than 2.5Mbps which is a nominal rate for modern data links.

Industry standard BGP routers provide configurable parameters with conservative default values. For example, the default hold time is 90 seconds, the default keepalive time is 1/3 of the hold time, and the default MinRouteAdvertisementInterval is 30 seconds for eBGP peers and 5 seconds for iBGP peers (see Section 10 of [RFC4271]). For the simple mobile routing system described herein, these parameters can and should be set to more aggressive values to support faster neighbor/link failure detection and faster routing protocol convergence times. For example, a hold time of 3 seconds and a MinRouteAdvertisementInterval of 0 seconds for both iBGP and eBGP.

Each c-ASBR will be using eBGP both in the ATN/IPS and the Internetworking Underlay with the ATN/IPS unicast IPv6 routes resolving over Internetworking Underlay routes. Consequently, c-ASBRs and potentially s-ASBRs will need to support separate local ASes for the two BGP routing domains and routing policy or assure routes are not propagated between the two BGP routing domains. From a conceptual and operational standpoint, the implementation should provide isolation between the two BGP routing domains (e.g., separate BGP instances).

## 7. Implementation Status

The BGP routing topology described in this document has been modeled in realistic network emulations showing that at least 1 million MNPs can be propagated to each c-ASBR even on lightweight virtual machines. No BGP routing protocol extensions need to be adopted.

## 8. IANA Considerations

This document does not introduce any IANA considerations.

## 9. Security Considerations

ATN/IPS ASBRs on the open Internet are susceptible to the same attack profiles as for any Internet nodes. For this reason, ASBRs should employ physical security and/or IP securing mechanisms such as IPsec [RFC4301], TLS [RFC5246], etc.



ATN/IPS ASBRs present targets for Distributed Denial of Service (DDoS) attacks. This concern is no different than for any node on the open Internet, where attackers could send spoofed packets to the node at high data rates. This can be mitigated by connecting ATN/IPS ASBRs over dedicated links with no connections to the Internet and/or when ASBR connections to the Internet are only permitted through well-managed firewalls.

ATN/IPS s-ASBRs should institute rate limits to protect low data rate aviation data links from receiving DDoS packet floods.

This document does not include any new specific requirements for mitigation of DDoS.

## 10. Acknowledgements

This work is aligned with the FAA as per the SE2025 contract number DTFAWA-15-D-00030.

This work is aligned with the NASA Safe Autonomous Systems Operation (SASO) program under NASA contract number NNA16BD84C.

This work is aligned with the Boeing Information Technology (BIT) MobileNet program.

## 11. References

### 11.1. Normative References

- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <<https://www.rfc-editor.org/info/rfc4456>>.

- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

## 11.2. Informative References

- [BGP] Huston, G., "BGP in 2015, <http://potaroo.net>", January 2016.
- [BGP2] Huston, G., "BGP Instability Report, <http://bgpupdates.potaroo.net/instability/bgpupd.html>", May 2017.
- [CBB] Dul, A., "Global IP Network Mobility using Border Gateway Protocol (BGP), [http://www.quark.net/docs/Global\\_IP\\_Network\\_Mobility\\_using\\_BGP.pdf](http://www.quark.net/docs/Global_IP_Network_Mobility_using_BGP.pdf)", March 2006.
- [I-D.templin-aerolink] Templin, F., "Asymmetric Extended Route Optimization (AERO)", draft-templin-aerolink-82 (work in progress), May 2018.
- [ICAO] ICAO, I., "<http://www.icao.int/Pages/default.aspx>", February 2017.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <<https://www.rfc-editor.org/info/rfc2784>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet Autonomous System (AS) Number Space", RFC 6793, DOI 10.17487/RFC6793, December 2012, <<https://www.rfc-editor.org/info/rfc6793>>.

## Appendix A. Change Log

<< RFC Editor - remove prior to publication >>

Changes from -07 to -08:

- o Removed suggestion to use private ASNs
- o Ran spelling checker and corrected errors
- o Re-worked Section 3 final two paragraphs on scaling
- o Stated Internetwork underlay as being out of scope for this document

## Authors' Addresses

Fred L. Templin (editor)  
Boeing Research & Technology  
P.O. Box 3707  
Seattle, WA 98124  
USA

Email: fltemplin@acm.org

Greg Saccone  
Boeing Research & Technology  
P.O. Box 3707  
Seattle, WA 98124  
USA

Email: gregory.t.saccone@boeing.com

Gaurav Dawra  
LinkedIn  
USA

Email: gdawra.ietf@gmail.com

Acee Lindem  
Cisco Systems, Inc.  
USA

Email: acee@cisco.com

Victor Moreno  
Cisco Systems, Inc.  
USA

Email: [vimoreno@cisco.com](mailto:vimoreno@cisco.com)

gwg  
Internet Draft  
Intended status: Informational  
Expires: September 11, 2019

S. Wadhwa  
K. DeSmedt  
Nokia  
R. Shinde  
Reliance Jio  
J. Newton  
Vodafone  
R. Hoffman  
TELUS  
P. Muley  
Nokia  
Subrat Pani  
Juniper Networks  
Mar 11, 2019

Architecture for Control and User Plane Separation on BNG  
draft-wadhwa-rtgwg-bng-cups-03.txt

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 11, 2019.

## Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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.

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.

## Abstract

This document discusses separation of subscriber-management control plane and data-plane for BNG. Traditionally, the BNG provides aggregation of fixed access nodes (such as DSLAM and OLTs) over Ethernet and provides subscriber management and traffic management functions for residential subscribers. The BNG has however evolved to become a multi-access edge device that also provides termination of subscribers over fixed-wireless and hybrid access. Therefore, this document proposes interfaces between control and user-plane of a BNG that can support multi-access BNG.

## Table of Contents

1. Introduction.....	3
1.1. Requirements Language.....	3
2. CUPS for BNG.....	3
2.1. Convergence.....	5
3. Interfaces for CUPS.....	6
3.1. In-band Signaling Channel.....	7
3.2. State Control Interface.....	8
3.2.1. Session level state management.....	8
3.2.2. Session level event notifications.....	14
3.2.3. Node level management.....	15

3.2.4. Node level event notifications.....	16
3.3. Management Interface.....	17
4. Protocol Selection for CUPS Interfaces.....	17
5. Address Pool Management.....	19
6. Security Considerations.....	19
7. IANA Considerations.....	19
8. References.....	20
8.1. Normative References.....	20
8.2. Informative References.....	20

## 1. Introduction

This document describes requirements and architecture for separation of subscriber management control plane and user plane for the BNG. In rest of the document the control plane is referred to as CP, user plane as UP, and the separation is referred to as CUPS (control and user plane separation). The draft describes the functional decomposition between CP and UP, and applicability of CUPS to a BNG that can support multiple access technologies such as fixed (DSL or Fiber), fixed-wireless (LTE,5G) and hybrid access i.e. simultaneous fixed and wireless access described in BBF [WT378]. The subsequent sections of the draft also define the interfaces required between CP and UP and briefly discusses a candidate base protocol for these interfaces.

### 1.1. Requirements Language

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

## 2. CUPS for BNG

In a CUPS architecture, signaling to setup subscriber sessions CP terminates signaling to setup subscriber sessions, and interfaces with the UP to create forwarding state for these sessions on the UP.

For fixed access subscribers, the CP terminates the signaling protocols (e.g. DHCP, PPPoE, SLAAC) from the customer premise, performs authorization/authentication with AAA Server, participates in address assignment, and then interfaces with the UP to create state related to forwarding and SLA management for the subscriber sessions on the UP. A subscriber session is a single IP connection, such as an IPoE or PPPoE session. The session can be single-stack (IPv4 or IPv6 only), or dual-stack (both IPv4 and IPv6). A CPE can

have multiple sessions, if multiple IP connections are required (e.g. on per service, or one per device behind the CPE).

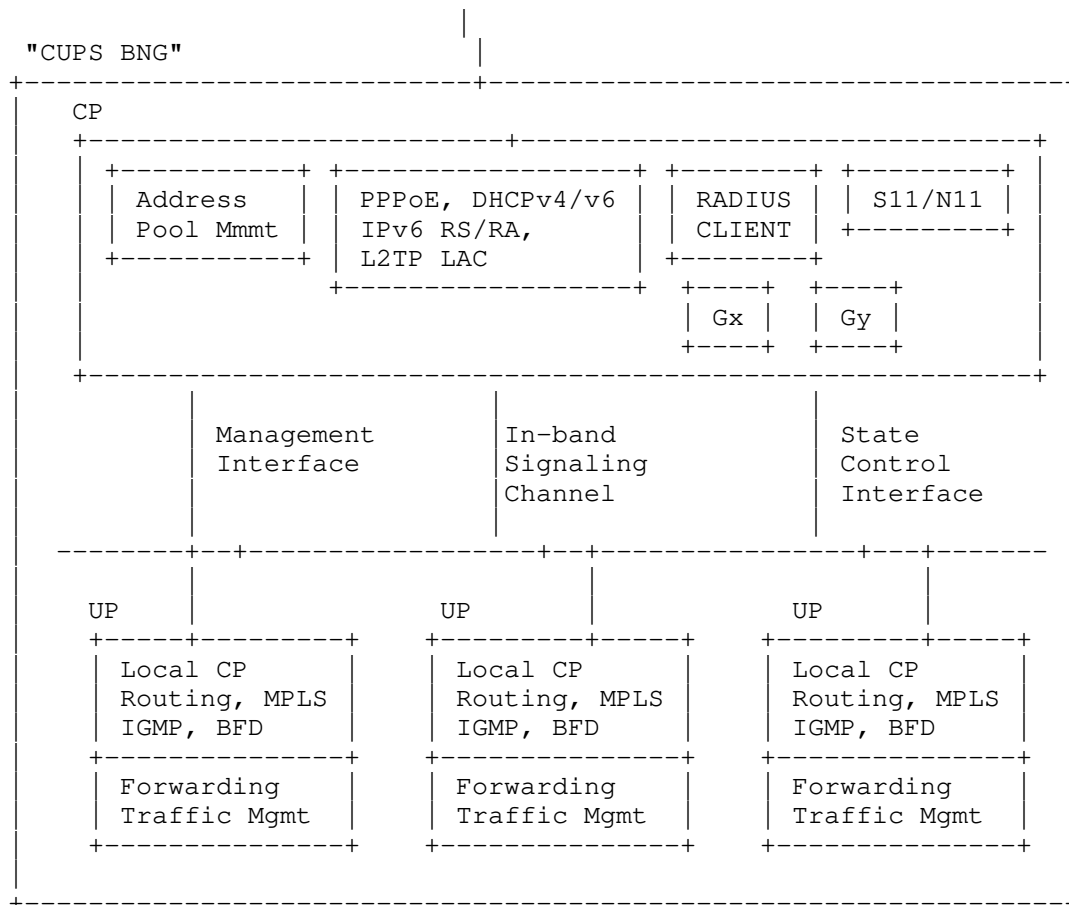
The CP also processes solicited or unsolicited event notifications from the UP e.g. periodic accounting updates, usage reports, or session inactivity notifications. The interface between CP and UP that is used by the CP to manage session related forwarding state on the UP is being referred to as "state control interface". Asynchronous event notifications from UP to CP are also part of this interface.

In typical fixed access deployments, signaling (e.g. DHCPv4/v6, PPPoE, ICMPv6 RS/RA) to setup the subscriber sessions is in-band, and hence the UP receives the signaling messages from the customer premise. The UP should transparently forward (unmodified) in-band control messages as received from the customer premise to the CP and return messages from CP to the customer premise. Therefore, an in-band signaling channel is required between UP and CP. With a typical "CUPS BNG" deployment, the CP and UP are connected over a network, and the in-band signaling channel must be over a tunnel.

The UP performs forwarding and traffic management for the subscriber sessions. The infrastructure routing and signaling is done on the local control plane of the UP for fast convergence on network topology changes. In rest of the document the term "UP" is used generically for both functions performed by the local control plane on the UP and the data-plane.

A typical deployment architecture for CUPS includes a centralized CP running as a VNF interacting with multiple BNG UP instances that may be more distributed than the CP and could run as VNF or PNF. In this model, the CP and UP association is 1:N. This composite system containing CP VNF and one or more UP instances is referred to as a "CUPS BNG" in rest of the document. For operational ease, the CP MUST provide a single point for control and management for the entire "CUPS BNG". It MUST expose a single interface on behalf of the "CUPS BNG" to external systems such as AAA servers, OSS/BSS, Policy and charging servers. The CP VNF MUST support scale-out in order to cope with growth in number of subscriber sessions and/or increase in number of UP instances in the "CUPS BNG". Figure 1 below shows the functional components and interfaces for a "CUPS BNG".



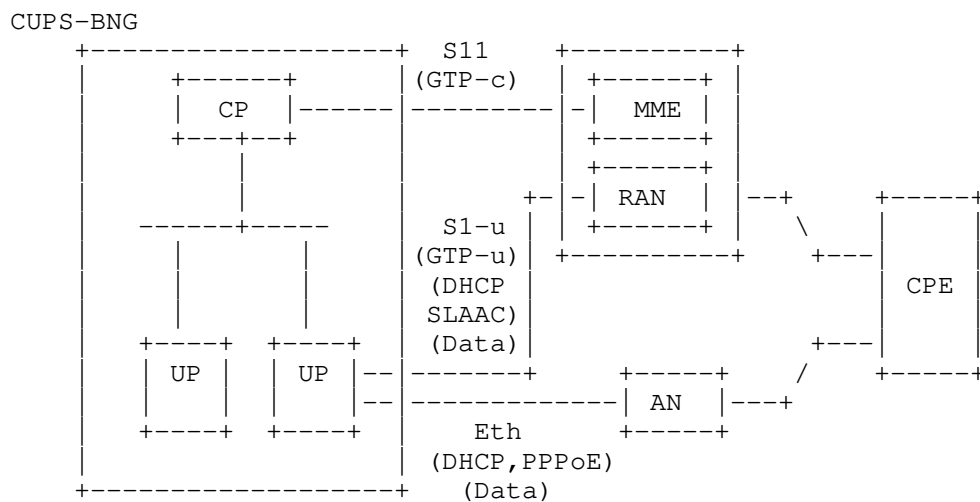


CUPS BNG System

### 2.1. Convergence

A single BNG can support subscribers over fixed, "fixed-wireless" or hybrid access. When a residential gateway has fixed-wireless access (LTE or 5G), then the BNG participates in 3GPP signaling with an MME

or AMF (i.e. support 3GPP S11 and N11 interfaces) to setup connections from (NG)RAN. With Hybrid access the customer premise initiates both fixed and wireless connections. The BNG in this case aggregates subscribers over Ethernet from fixed access nodes (DSLAMs and OLTs), but simultaneously terminates connections from (NG)RAN by participating in signaling with MME or AMF (S11/N11 interface). These deployment models are drivers for fixed-mobile convergence. It is important to ensure that the interfaces between CP and UP for CUPS can support not only fixed L2 access, but also the converged access scenario shown in Figure 2. One key requirement on the CP in these cases is the need to participate in 3GPP signaling (which is out-of-band) to setup the data-path. The data-path is a GTP-u (GPRS Tunneling protocol - User Plane) tunnel from the RAN (i.e. S1-u interface for LTE) as described in 3GPP [TS29281], and it terminates on the UP. It carries data traffic but also subscriber signaling messages (e.g. DHCPv4, DHCPv6, SLAAC) from the customer premise. The UP therefore still requires an in-band signaling channel to transport these protocol messages to the CP.



"CUPS BNG" with Converged Access

### 3. Interfaces for CUPS

A "CUPS BNG" MUST support the following interfaces between CP and UP, as shown in the figure in section 2.

### 3.1. In-band Signaling Channel

Section 2 describes the need for a signaling channel between CP and UP to transport in-band control messages between CP and the customer premise. Following are some key requirements for this interface.

- . The UP MUST pass the access circuit identifier over which the signaling messages are received as meta-data to the CP. This includes port, VLAN tags, tunnel endpoint IPs, any tunnel identifiers such as GTP TEID, MPLS labels, L2TP tunnel-id etc. The UP MUST also pass the L2 or L3 transport service that the access circuit is associated with. In case the control message PDU is carried in an Ethernet frame, then the UP SHOULD pass the received Ethernet frame to the CP. Both access circuit identifier and information in the Ethernet header are required by the CP to construct successful response packet (control message) back towards the customer premise. The access circuit identifier MUST be reflected from CP to UP, so UP can identify the access circuit over which it needs to send the CP's response packet. In the control message sent from UP to CP, the UP MUST also include the local MAC address associated with access circuit. This is because certain control messages from the customer premise are destined to a broadcast MAC (e.g. DHCP DISCOVER) or multicast (e.g. ICMPv6 RS), so CP cannot infer the local MAC from these messages. Certain messages also require the local MAC address to be inserted in the message (e.g. Link-Layer address in ICMPv6 RA messages)
- . The CP MUST be able to control the UP to forward only specific control messages to the CP.
- . The CP MUST be able to control the UP to block certain control messages received on a particular access circuit.
- . The CP MUST be able to control the UP to limit the rate of control messages (of specified type) to be sent by the UP.
- . The CP MUST be able to prioritize reception of certain control messages over others in a granular manner (e.g. prioritize DHCP RENEWS over DISCOVERS or prioritize PPP Keepalive over other messages).
- . The in-band signaling channel MUST support both fixed and converged access as described in section 2.1. The tunnel used for transporting these messages should therefore support both Ethernet and IP payloads.

### 3.2. State Control Interface

The CP and UP can exchange state at two levels using the "state control interface". One is at the node level and includes node-level information such as supported features, software releases, available resources, and operational state (e.g. active, failed, or overloaded). The other is at the subscriber session level. Subscriber session is described in section 2. The session level state includes basic forwarding and traffic management rules per session, that need to be provided by the CP to the UP in order to control per session forwarding and traffic management on the UP. It also includes state that triggers routing related actions on the UP. The session level state can include asynchronous event notifications from UP to CP, such as notifications to report per session usage (periodically or based on thresholds), notification to report session inactivity, and session liveness.

The interactions between CP and UP over "state control interface" can be categorized as:

- o Session level state management
- o Session level event notifications
- o Node level management
- o Node level event notifications

Following sub-sections provide more details on these interactions. The interactions between CP and UP over "state control interface" are modeled via abstract request/response messages between CP and UP. These messages will need to be defined as part of the protocol specification for this interface.

The protocol selected to implement this interface MUST support both fixed access and converged access (described in section 2.1) on BNG

#### 3.2.1. Session level state management

Once the CP has successfully authorized and/or authenticated the subscriber session, and completed address assignment, it uses the "state control interface" to install forwarding and related state for the session on the forwarding path of the UP. This is abstracted as a "session create request" call from CP to UP, as shown in the figure below. The UP MUST ack or NACK via a response back to CP.

Since BNG can support different access types (e.g. fixed L2 access, or tunneled L3 in case of fixed-wireless, or a combination in case of hybrid access), it is important that the forwarding state information for the subscriber sessions, sent from CP to UP, can be specified as flexible packet matching rules and set of actions related to forwarding and traffic management. The UP should be able to use these match rules and actions to derive various lookup tables and processing in the forwarding path to forward traffic to and from the CPE.

The basic forwarding state in upstream direction (i.e. access to network) and downstream direction (i.e. network to access) fundamentally consists of session identification and one or more actions. Following shows a logical representation of a directive from CP to UP to install basic forwarding state on the UP for fixed L2 access (i.e. access from DSLAM or OLTs over Ethernet).

Direction Upstream - Access to Network:

Subscriber-identification: Port/VLAN-tag(s) + subscriber-MAC

Action: remove encapsulation, IP FIB lookup, forward to network.

Direction Downstream - Network to Access:

Subscriber-identification: IP address

Action: lookup IP DA, build encapsulation using Port/VLAN-tag(s)+ subscriber-MAC, forward to access.

Optionally, the IP address assigned to the CPE can also be provided for subscriber-identification (e.g. for anti-spoofing) in the upstream direction.

In case of PPPoE sessions, the subscriber-identification for upstream direction and encapsulation for downstream direction also includes the PPPoE session-id.

Based on the directive from CP to UP (as shown in the example above), the UP can then populate appropriate tables in the forwarding path, e.g. subscriber lookup tables, IP-FIB, and ARP or IPv6 Neighbor discovery table. It can also program the packet processing in both upstream and downstream direction based on the specified actions.

In case of "fixed-wireless" access, the access circuit is a GTP-u tunnel. In this case there is no physical interface (or port), and hence the CP MUST provide a tunnel definition to the UP to use as access circuit in upstream direction, and encapsulation in downstream direction. The tunnel definition will include the tunnel endpoint IP, and TEID that is established via out-of-band signaling

between the CP and the customer premise. It can also include the routing context for transporting the tunnel.

In addition to setting up the forwarding state as directed by the CP, the UP also needs to announce in routing the aggregate prefixes from which the CP assigns IPv4 and IPv6 addresses (or prefixes) to the CPEs. The CP SHOULD provide these aggregate prefixes to the UP as part session state. In case the aggregate prefixes are not provided, the UP MUST announce individual CPE addresses in routing, or it MAY try to aggregate in case addresses for multiple CPEs are from a contiguous address space.

The CPE can have a routed subnet behind it (aka framed-route). CP can learn the framed-routes during authentication/authorization. The CP should provide the framed-route to the UP as part of session state. The UP MUST install this route in the forwarding path and associate it with the forwarding state of the corresponding subscriber session. It should also announce this in routing towards the Network.

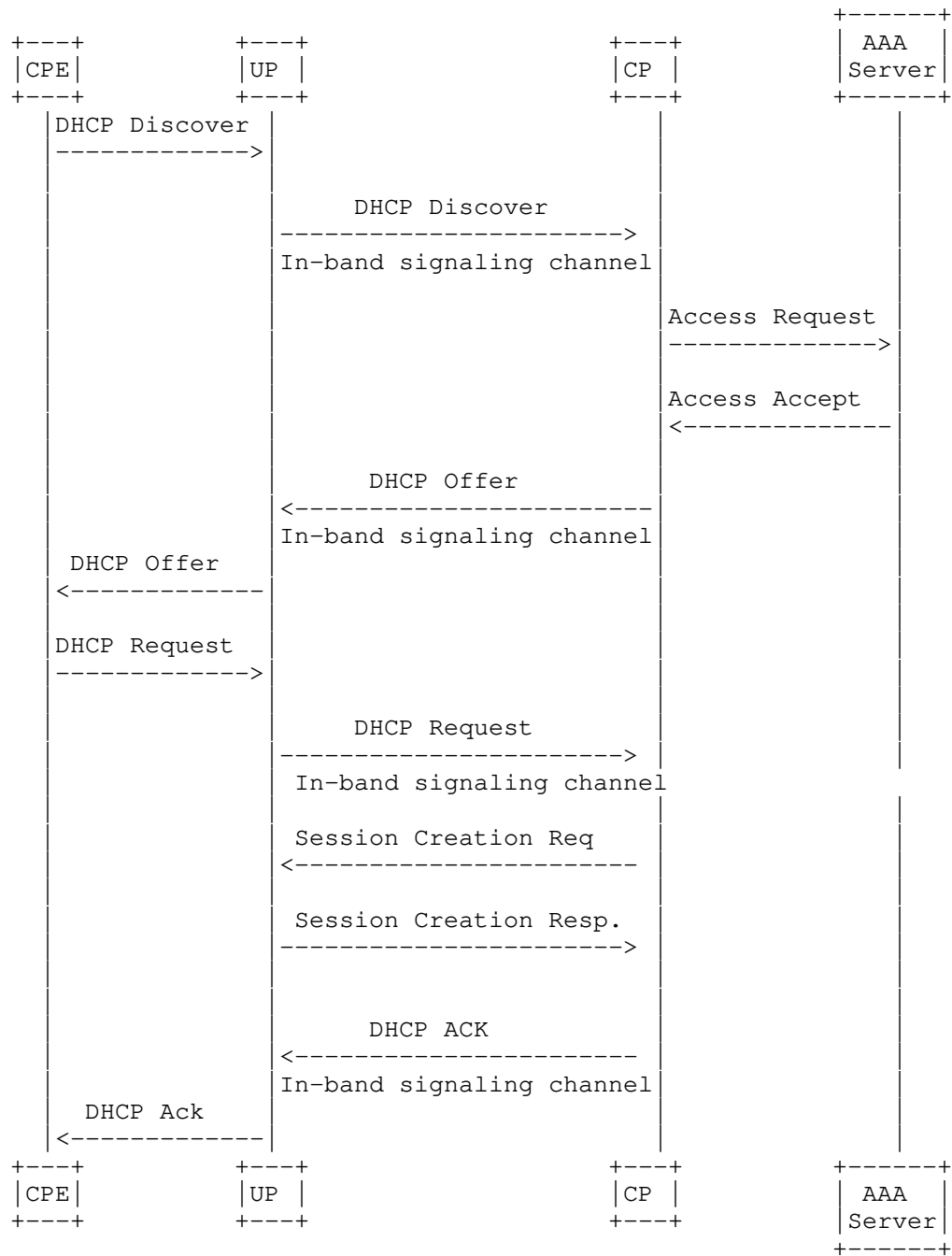
The CP MUST also provide to the UP the address assigned as IP gateway address to the CPEs in DHCP. The UP MUST locally configure this address appropriately, such that it can respond to ARP requests for this address from the CPEs.

The session state on the UP is always controlled by the CP i.e. the UP just follows the directive from the CP to install, modify and delete the session state. In addition to the basic forwarding state, the CP can also associate, update and disassociate other related state with the session e.g. state related to:

- . Filtering
- . SLA management
- . Statistics collection
- . Credit control
- . Traffic mirroring
- . Traffic Steering
- . NAT
- . Application aware policies

BNG deployments use hierarchical QoS (H-QoS) models which follows from a combination of link-layer over-subscription, multi-service networks and multiple layers of aggregation. For example, a common hierarchy exists of at least a QoS layer per access-node, and per

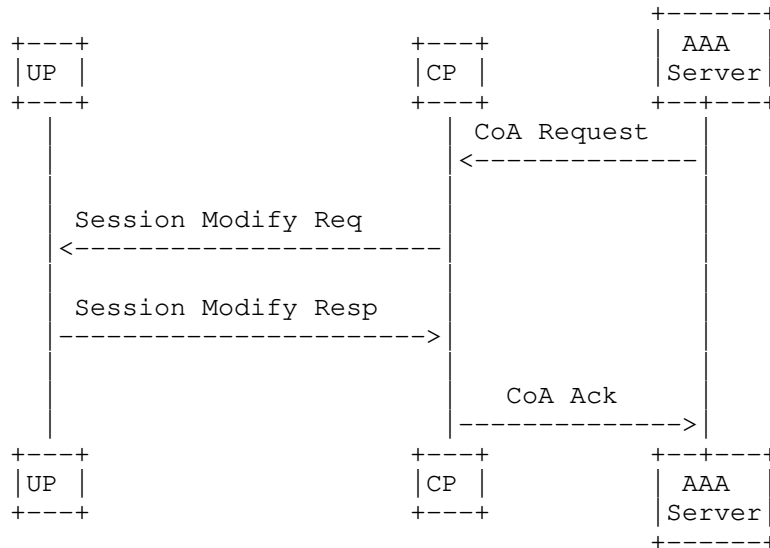
CPE. The CP MUST provide SLA management information to the UP per CPE. This includes applicable QoS parameters (e.g. rates, queues, markings) and the QoS hierarchy to which the CPE belongs. The CP may choose to signal this via a QoS policy that is locally pre-configured on the UP.



Session Creation Sequence

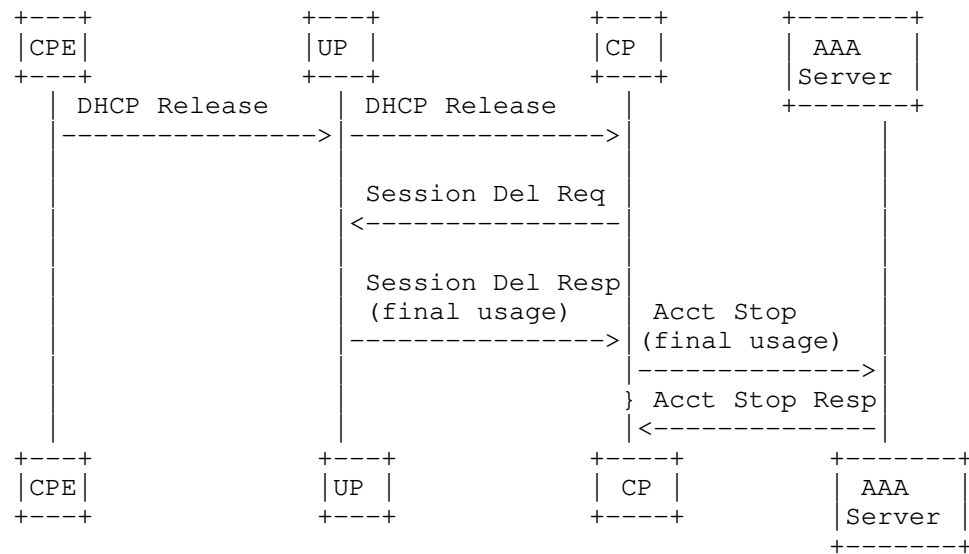


CP can trigger update of session state on the UP, triggered by re-authentication or CoA from AAA or policy-server, as show in the figure below.



Session Modification

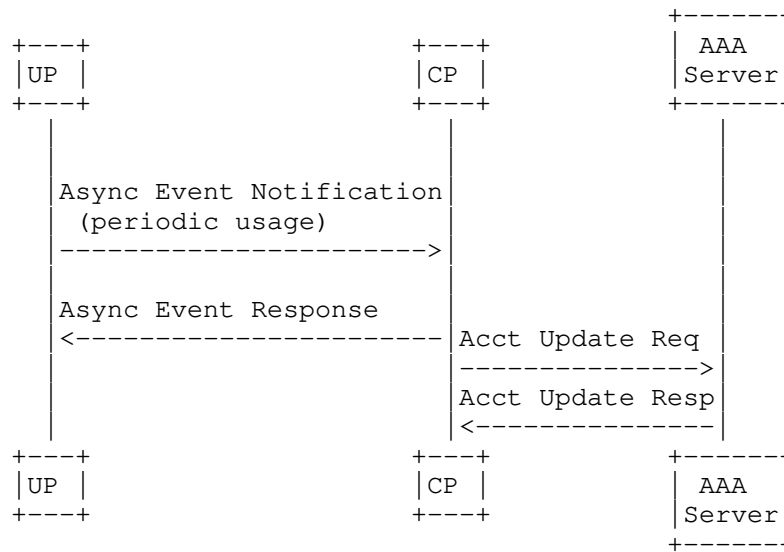
CP can trigger the deletion of session state based on signaling messages (as shown in the figure below), administrative action or disconnect-message initiated from the AAA server.



Session Deletion

### 3.2.2. Session level event notifications

UP can asynchronously generate Session level event notifications to the CP. An example of asynchronous notification is periodic usage reporting from UP to the CP, so that the CP can report the usage to a AAA server via interim accounting-updates. The CP can set the periodicity of this notification on the UP based on interim accounting interval configured by the operator on the CP.



Async Event Notification for periodic usage

Following are some other examples requiring asynchronous notifications from UP to CP.

- o Threshold based usage reporting
- o Inactivity timeout
- o Subscriber unreachability detection

The protocol for "state control interface" MUST support asynchronous notifications from UP to CP.

### 3.2.3. Node level management

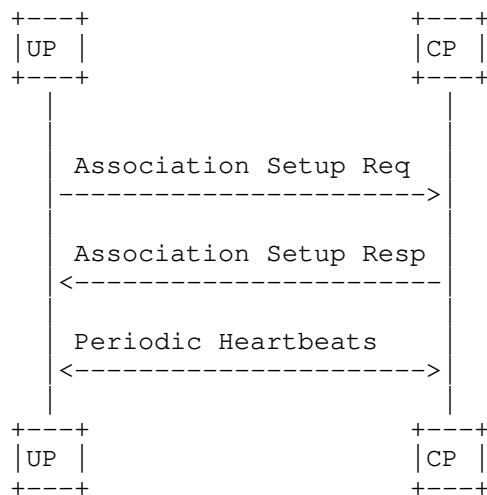
There needs to exist a concept of association between CP and UP. When the CP or UP comes online it should setup an association with configured or discovered peers via a message exchange. In association setup, the nodes should be able to exchange supported capabilities, version of software, load/overload information, and resource information. Also, any node-wide parameters can be exchanged during association setup.

No session state related messages should be accepted from the peer by either CP or UP unless an association exists.

Either node should be able to update the association to report changed feature capabilities, overload condition, resource exhaustion or any other node-wide parameters.

The UP should be able to request a graceful association release from the CP. In this case the CP should delete all sessions from that UP and process the final stats report for each session and send it in accounting-stop to the AAA server. During this process the CP MUST not create new sessions on the UP. Once all sessions are successfully deleted, the CP should release the association.

There needs to be a periodic node-level heartbeat exchange between CP and UP to detect if the peer is reachable and active. If peer is determined to be down based on heartbeat messages, then all the data-plane session state associated with the peer should be deleted.



#### Node Association Setup and Maintenance

##### 3.2.4. Node level event notifications

There needs to be support for asynchronous node level event notifications from UP to CP. Example includes switchover

notification in case ports or UP failures when UP node level warm-standby redundancy is enabled. Based on this notification, the CP can create session state for all the sessions associated with the failure domain on the new primary UP.

### 3.3. Management Interface

The CP MUST provide a single point for local management of "CUPS BNG" system to the operator. This requires a management interface between CP and each of its associated UPs for pushing configuration to the UP and retrieving operational state from the UP. The interface MUST minimally include BNG specific configuration and state.

The Management interface SHOULD support transactional configuration from CP to UPs and SHOULD support state retrieval, both based on a well-defined data schema. The management interface SHOULD support unsolicited signaling of state changes (events) from UP to CP i.e. MUST provide telemetry for events. Either gNMI or NETCONF can be considered as acceptable candidates for model driven management interface.

### 4. Resiliency

"CUPS BNG" system MUST be protected against failure of CP VNF and MUST be able to recover the session state without operator intervention and reliance on CPEs. This can be achieved by providing redundancy for processing resources within CP VNF and maintaining redundant instance of session state.

Protection against UP failures based on 1:1 UP (hot-redundancy) and N:M (warm-redundancy) SHOULD be supported. For 1:1 hot-redundancy the CP needs to create data-plane state for sessions on both UPs that form a redundant pair, using the "state control interface". The CP needs to ensure the data-plane state for a session stays synchronized between the two nodes. A given session's data-plane should only be active on one UP in the pair, which serves as active UP for the session. However, sessions that share the redundant UP pair can be distributed between the two UPs for active forwarding.

N:M warm-redundancy ( $N > M$ ) can be supported via creation of data-plane state on the designated backup chassis after the failure has been detected. This would result in longer failover times than 1:1 hot-redundancy.

Redundant network connectivity between CP and UPs MUST be supported. In the "CUPS BNG" architecture, it is important to configure redundant connectivity that doesn't share fate.

## 5. Protocol Selection for CUPS Interfaces

It is important that the selected protocol for "state control interface" between CP and UP works not just for fixed access but also works for converged access on BNG. 3GPP has defined PFCP (Packet Forwarding Control Protocol) in [TS29244] as the interface between CP and UP for LTE gateways. This protocol is suited for large scale state management between CP and UP. Following are some of the key attributes of this protocol:

- o It supports management of forwarding and QOS enforcement state on the UP from CP. It also supports usage reporting from UP to CP.
- o It is over UDP transport and doesn't suffer from any HOL blocking.
- o It provides reliable operation based on request/response with message sequencing and retransmissions.
- o It provides an overload control procedure where overload on UP can be handled gracefully.
- o The protocol is extensible and allows addition of new IEs.

For fixed access BNG, the protocol requires simple extensions in form of additional IEs. The required extensions are mainly due to fact that typically a fixed access BNG requires tighter control over L2 behavior and manages access and subscriber using L2 identifiers (such as VLANs and MAC addresses), whereas mobile access works in terms of L3, either routed or tunneled.

The details of the protocol as applicable to the BNG and the required extensions will be defined in a separate draft.

[TS29244] also describes an in-band signaling channel based on GTP-u tunnel between CP and UP. GTP-u (GPRS Tunneling protocol - User Plane) is defined in 3GPP [TS29281] and defines a tunneling protocol which carries IP payloads. The protocol runs over a UDP/IP stack and uses UDP port number 2152. Data within a tunnel can be multiplexed based on Tunnel Endpoint Identifiers (TEIDs). The protocol supports optional sequence numbers. The protocol supports extension headers to allow development of new features. GTP-u tunnels are signaled between CP and UP, and it is possible

to associate filters to block certain control packets from being forwarded from UP to CP. The payload type carried by GTP-u can be extended to Ethernet (via payload type in extension header). The tunnel encapsulation can also be extended similarly to carry any required meta-data.

## 6. Address Pool Management

The CP MUST support management of IPv4 and IPv6 address pools, where each pool can contain one or more subnets. The pool management MUST support pool selection based on one or more of the following criteria:

- o UP
- o Access port on the UP.
- o Redundancy domain on the UP (e.g. set of access ports that share fate with respect to switchovers due to failures, when UP node level redundancy is enabled).
- o Service (e.g. HSI, VoIP, IPTV etc.).
- o Location (e.g. based on circuit-id/remote-id or part of circuit-id/remote-id in DHCP and PPPoE).

Pool management on CP SHOULD NOT statically link subnets to UPs but SHOULD dynamically allocate subnets to UP based on load i.e. on-demand, and signal allocated subnets using the "state control interface" as described in section 3.2.1. This allows for better IP resource utilization and less subnet fragmentation.

## 7. Security Considerations

For security between CP and UP, Network Domain Security (NDS) as defined in [TS33210] can be considered. As per NDS, the network can be split into security domains. Communication within a single security domain is considered secure, and protocols can operate without any additional security. When communication has to cross security domains, then IPSEC can be used.

## 8. IANA Considerations

None.

## 9. References

### 9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [TS29244] 3GPP, "Interface between the Control Plane and the User Plane Nodes", TS 29.244 15.2.0, June 2018, <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3111>.
- [TS29281] 3GPP, "General Packet Radio System (GPRS) Tunneling Protocol User Plane (GTPv1-U)", TS 29.281 15.3.0, June 2018, <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1699>.
- [TS33210] 3GPP, "Network Domain Security (NDS); IP network layer security", TS 33.210 15.0.0, June 2018, <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2279>.
- [WT378] BBF, "Nodal Requirements for Hybrid Access Broadband Networks", WT-378, 2018.

### 9.2. Informative References

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



Authors' Addresses

Sanjay Wadhwa  
Nokia  
777 East Middlefield Road  
Mountain View  
USA  
  
Email: [Sanjay.wadhwa@nokia.com](mailto:Sanjay.wadhwa@nokia.com)

Killian De Smedt  
Nokia  
Copernicuslaan 50  
Antwerp  
Belgium

Email: Killian.de\_smedt@nokia.com

Rajesh Shinde  
Reliance Jio Infocomm Ltd.  
Reliance Corporate Park  
Thane Belapur Road, Ghansoli  
Navi Mumbai 400710  
India

Email: Rajesh.A.Shinde@ril.com

Jonathan Newton  
Vodafone  
Waterside House  
Bracknell  
United Kingdom

Email: jonathan.newton@vodafone.com

Ryan Hoffman  
TELUS  
1525 10th Ave SW  
Calgary, Alberta  
Canada

Email: ryan.hoffman@telus.com

Praveen Muley  
Nokia  
805. E. Middle Field Rd.  
Mountain View, CA, 94043  
USA

Email: praveen.muley@nokia.com

Subrat Pani  
Juniper Networks

10 Technology Park Dr.  
Westford, MA  
USA

Email: [spani@juniper.net](mailto:spani@juniper.net)

INTERNET-DRAFT  
Intended Status: Informational  
Expires: January 3, 2019

Shen Yan  
Zhe Chen  
Huawei Technologies  
July 2, 2018

The analysis of SRv6 and potential improvement  
draft-yan-rtgwg-srv6-constrain-analysis-00

Abstract

This document analyzes the constraints of Segment Routing (SR), especially in the aspect of the header consumption and multicast of SRv6. Some potential methods have been proposed to improve the performance and enable more functions.

Status of this Memo

This Internet-Draft is submitted to IETF 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/lid-abstracts.html>

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

Copyright and License Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents  
(<http://trustee.ietf.org/license-info>) in effect on the date of

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

## Table of Contents

1	Introduction . . . . .	3
2	Terminology . . . . .	3
3.	Constrain Analysis . . . . .	3
3.1	Segment Consumption . . . . .	3
3.2	Multicast . . . . .	4
4	Potential Improvement . . . . .	5
4.1	Decrease the Consumption . . . . .	5
4.2	Globalize Semantics . . . . .	6
5	Security Considerations . . . . .	7
6	IANA Considerations . . . . .	7
7	References . . . . .	7
7.1	Normative References . . . . .	7
7.2	Informative References . . . . .	8
	Authors' Addresses . . . . .	8

## 1 Introduction

Segment Routing (SR) leverages the source routing paradigm. It allows a headend node to steer a packet flow along any path for specific objectives like Traffic Engineering (TE). A node steers a packet through an SR Policy instantiated as an ordered list of instructions. These instructions are stack-structural and Segment Routing can be directly instantiated on the IPv6 data plane through the use of the Segment Routing Header defined in [I-D.ietf-6man-segment-routing-header]. SRv6 refers to this SR instantiation on the IPv6 data plane. The current SRv6's design is good however there are still some constraints in SID consumption and functional defects.

In this document, we analyze the constraints of SRv6's design and try to propose the potential improvement methods.

## 2 Terminology

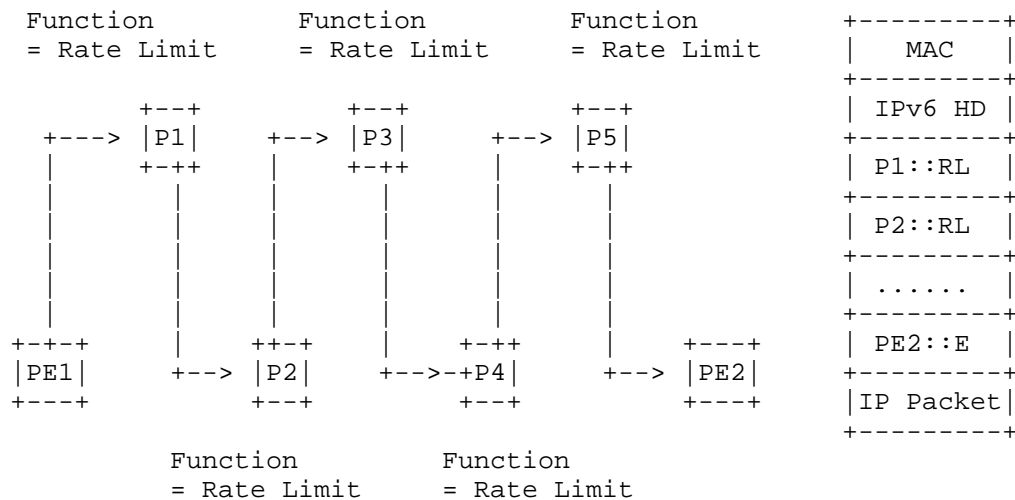
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

## 3. Constrain Analysis

### 3.1 Segment Consumption

The SID of SRv6 is a LOC:FUNCT tuple where FUNCT is a locally defined label. If a router is required to forward the packet to a specific neighbor, or perform a specific function, a corresponding label/SID MUST be put into the header. It means that N IPv6 addresses are needed in the header if the packet is required to pass through N routers.

As shown in Figure 1, PE1 wants to transmit a flow to PE2. Each router in this path is required to execute the function of Rate Limit (RL). In this example, the total cost before the inner IP Packet will be 158 Bytes. This causes two problems. The first one is the low payload proportion. The average packet size on the Internet is around 500 bytes and this design will occupy near 30%. The second one is the low processing efficiency. The current network processor reads normally less than 100 bytes at one time. If the header is too long, it needs more time to process. Conversely, if the header can be compressed shorter, the processing efficiency will be improved a lot.



### Figure 1, Segment Consumption

In this example, theoretically there are four redundant RL instructions. The potential improvements may come from two aspects. Firstly, the coupling of locator and function makes the segments cannot be reused. If the locator and function can be de-coupled by some methods, it potentially decreases the size of whole packet header especially when the number of routers is large and most of them execute the same function when forwarding the packet. Secondly, a simple path label may be employed instead of stacking multiple locators. A shorter path label can decrease the length of instruction list.

### 3.2 Multicast

SR-MPLS supports multicast but SRv6 can hardly do the same. As shown in Figure 2, CE1, CE2, CE3, CE4 and CE5, construct a Blue VPN. CE1 wants to send a broadcast frame to all the other CEs. Because of the localized semantics, different routers use different SIDs for the same instruction / metadata. Therefore, the multicast packet MUST be replicated at PE1 instead of the P-routers (P).

In other words, in current SRv6 scheme, the multicast packet MUST be replicated at the ingress PE because egress PE uses different SIDs for the same VPN, since the locator is contained in SIDs. This is not an unsolvable problems. If the locator and function can be de-coupled meanwhile all the P-routers perform same operation according to a uniform instruction suite, all the multicast packets will be same in

the egress router so that the packet could be replicated at any P-routers.

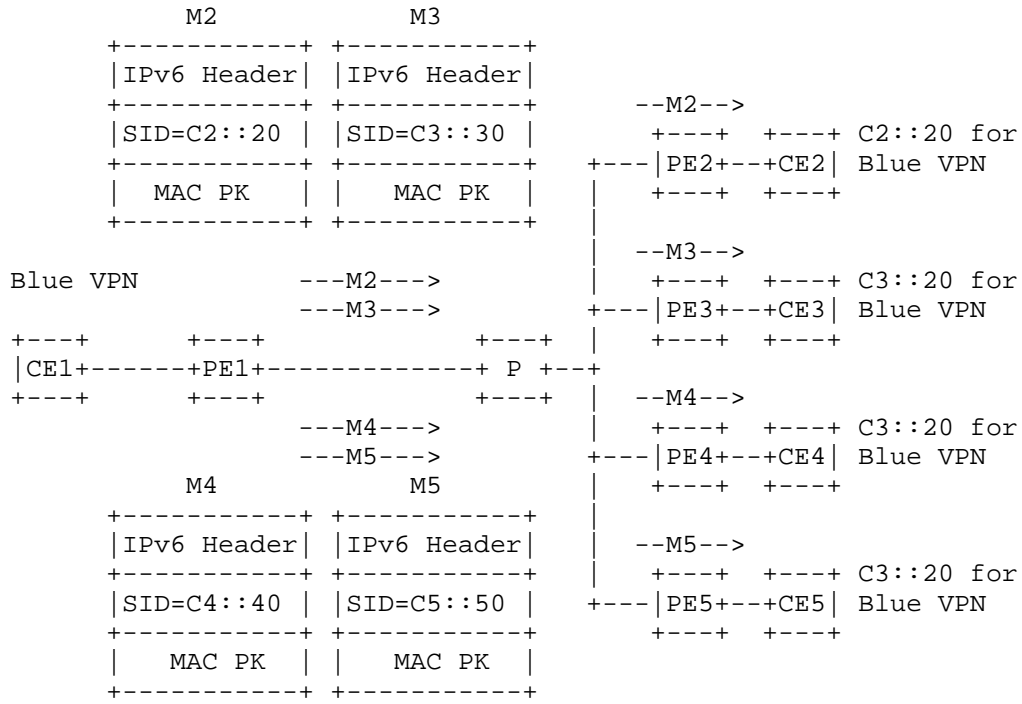


Figure 2, Multicast with SRv6

#### 4 Potential Improvement

This section presents potential solutions to address the constrains discussed above.

The core idea of the solutions is to de-couple instruction and locator carried in the data packet, by adopting globalized instructions. Globalized instruction is an instruction code that could be recognized by every node in a domain. Moreover, the packet processing logic associated with the instruction code SHOULD be same for all nodes in the domain. In this way, the packet header overhead could be significantly compressed, and multicast could be well supported.

##### 4.1 Decrease the Consumption

In particular, the example topology in Figure 1 could be used to



illustrate this idea. To limit packet rate for a specific flow, the packets sent from PE1 to PE2 SHOULD carry two globalized instructions. One has the semantic of "shortest-path forwarding the packet according to PE2's address", and the other has the semantic of "limiting packet rate based on flow identifier and the maximal packet rate". Each node along the forwarding path performs these two globalized instructions accordingly thus achieving the targeted network function (i.e., packet rate limitation) with minimal overhead in the packet header.

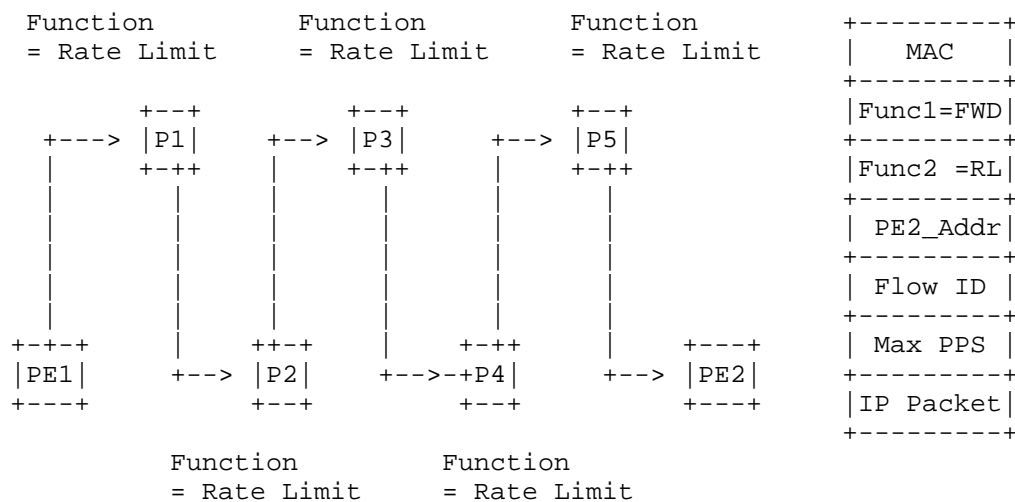


Figure 3, Potential Solution for Speed Limitation

#### 4.2 Globalize Semantics

The example topology in Figure 2 could be used to illustrate how this idea benefits multicast scenarios. As shown in Figure 4, after receiving a broadcast Ethernet frame from CE1, the ingress node (i.e., PE1) only need to encapsulate the frame into a single packet, and the packet SHOULD carry two globalized instructions. One has the semantic of "forwarding and replicating (if needed) the packet according to the multicast address of group PE2-PE5", and the other has the semantic of "if there is no next-hop, striping the encapsulated instructions, looking up the VRF of blue VPN and forwarding the packet accordingly". Each transit node in the network forwards and replicates (if needed) the packet based on the multicast address, and the egress nodes perform corresponding VPN actions. Therefore, globalized instructions enable packet replication in transit nodes, thus eliminating bandwidth wasting.

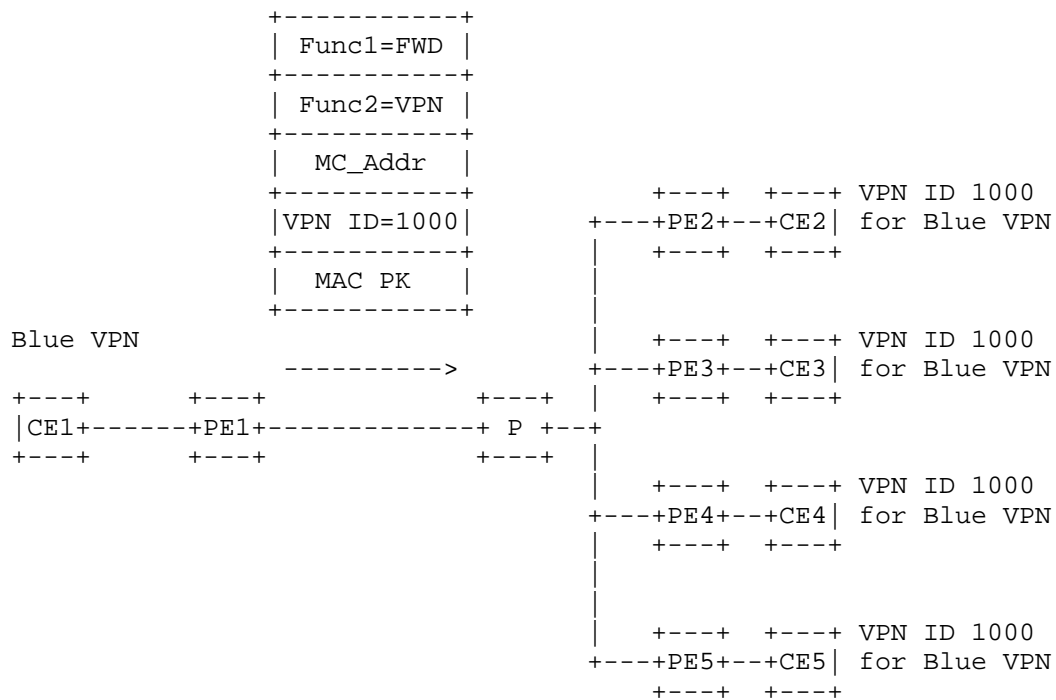


Figure 4, Potential Solution for Multicast VPN

Note that the metadata (e.g., unicast and multicast IP addresses, flow identifier, maximal packet rate, and VPN identifier) associated with the globalized instructions should also be carried in the packets, and there should be an approach to efficiently organize those instructions and metadata into a reasonable packet format. Such an approach will be specified in separated documents.

## 5 Security Considerations

## 6 IANA Considerations

## 7 References

### 7.1 Normative References

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 7.2 Informative References

[SR-MPLS-MCAST] Dave Allan, etc., "A Framework for Computed Multicast Applied to SR-MPLS", Work in Progress, draft-allan-pim-sr-mpls-multicast-framework-00, June 1, 2018

[SR-HEADER] C. Filsfils, Ed., etc., "IPv6 Segment Routing Header (SRH)", Work in Progress, draft-ietf-6man-segment-routing-header-14, June 28, 2018

## Authors' Addresses

Shen Yan  
Huawei Technologies Co. Ltd  
No. 156, Beiqing Rd,  
Haidian Dist, Beijing,  
China, 100095

EMail: yanshen@huawei.com

Zhe Chen  
Huawei Technologies Co. Ltd  
No. 156, Beiqing Rd,  
Haidian Dist, Beijing,  
China, 100095

EMail: chenzhel7@huawei.com