

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: September 6, 2018

J. Arkko  
Ericsson  
J. Tantsura  
Nuagenetworks  
J. Halpern  
B. Varga  
Ericsson  
March 5, 2018

Considerations on Network Virtualization and Slicing  
draft-arkko-arch-virtualization-01

Abstract

This document makes some observations on the effects of virtualization on Internet architecture, as well as provides some guidelines for further work at the IETF relating to virtualization.

This document also provides a summary of IETF technologies that relate to network virtualization. An understanding of what current technologies there exist and what they can or cannot do is the first step in developing plans for possible extensions.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 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

(<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. Definitions . . . . .	3
3. General Observations . . . . .	4
4. Virtualization in 5G Networks . . . . .	6
5. Overview of IETF Virtualization Technologies . . . . .	6
5.1. Selection of Virtual Instances . . . . .	7
5.2. Traffic Separation in VPNs . . . . .	7
5.3. Traffic Engineering and QoS . . . . .	9
5.4. Service Chaining . . . . .	10
5.5. Management Frameworks and Data Models . . . . .	10
6. Architectural Observations . . . . .	12
7. Further Work . . . . .	14
8. Acknowledgements . . . . .	17
9. Informative References . . . . .	17
Authors' Addresses . . . . .	19

## 1. Introduction

Network virtualization is network management pertaining to treating different traffic categories in separate virtual networks, with independent lifecycle management and resource, technology, and topology choices.

This document makes some observations on the effects of virtualization on Internet architecture, as well as provides some guidelines for further work at the IETF relating to virtualization.

This document also provides a summary of IETF technologies that relate to network virtualization. An understanding of what current technologies there exist and what they can or cannot do is the first step in developing plans for possible extensions.

In particular, many IETF discussions earlier in the summer of 2017 started from a top-down view of new virtualization technologies, but were often unable to explain the necessary delta to the wealth of existing IETF technology in this space. This document takes a different, bottom-up approach to the topic and attempts to document existing technology, and then identify areas of needed development.

In particular, whether one calls a particular piece of technology "virtualization", "slicing", "separation", or "network selection" does not matter at the level of a system. Any modern system will use several underlying technology components that may use different terms but provide some separation or management. So, for instance, in a given system you may use VLAN tags in an ethernet segment, MPLS or VPNs across the domain, NAIs to select the right AAA instance, and run all this top of virtualized operating system and software-based switches. As new needs are being recognised in the developing virtualization technology, what should drive the work is the need for specific capabilities rather than the need to distinguish a particular term from another term.

## 2. Definitions

Network function virtualization is defined in Wikipedia as follows:

"Network function virtualization or NFV is a network architecture concept that uses the technologies of IT virtualization to virtualize entire classes of network node functions into building blocks that may connect, or chain together, to create communication services.

NFV relies upon, but differs from, traditional server-virtualization techniques, such as those used in enterprise IT. A virtualized network function, or VNF, may consist of one or more virtual machines running different software and processes, on top of standard high-volume servers, switches and storage devices, or even cloud computing infrastructure, instead of having custom hardware appliances for each network function."

We should not confuse NFV and network virtualization, the former, as the name suggests is about functions virtualization, and not the network.

The idea of network virtualization is almost as old as the networking technology itself. Network virtualization is hierarchical and multilayer in its nature, from layer 1 up to services on top. When talking about virtualization we usually define overlay to underlay relationship between different layers, bottom up. A VPN (Virtual Private Network) [RFC4026] is the most common form of network virtualization. The general benefits and desirability of VPNs have been described many times and in many places ([RFC4110] and [RFC4664]).

The only immutable infrastructure is the "physical" medium, that could be dedicated or "sliced" to provide services (VPNs) in a multi-tenant environment.

The term slicing has been used to describe a virtualization concept in planned 5G networks. The 3GPP architecture specification [TS-3GPP.23.501] defines network slices as having potentially different "supported features and network functions optimisations", and spanning functions from core network to radio access networks.

[I-D.king-teas-applicability-actn-slicing] defined slicing as "an approach to network operations that builds on the concept of network abstraction to provide programmability, flexibility, and modularity. It may use techniques such as Software Defined Networking (SDN) and Network Function Virtualization (NFV) to create multiple logical (virtual) networks, each tailored for a set of services that are sharing the same set of requirements, on top of a common network.

And, [I-D.geng-coms-problem-statement] defines slicing as a management mechanism that an service provider can use to allocate dedicated network resources from shared network infrastructures to a tenant.

### 3. General Observations

#### Software vs. Protocols

Many of the necessary tools for using virtualization are software, e.g., tools that enable running processes or entire machines in a virtual environment decoupled from physical machines and isolated from each other, virtual switches that connect systems together, management tools to set up virtual environments, and so on. From a communications perspective these tools operate largely in the same fashion as their real-world counterparts do, except that there may not be wires or other physical communication channels, and that connections can be made in the desired fashion.

In general, there is no reason for protocols to change just because a function or a connection exists on a virtual platform. However, sometimes there are useful underlying technologies that facilitate connection to virtualized systems, or optimised or additional tools that are needed in the the virtualized environment.

For instance, many underlying technologies enable virtualization at hardware or physical networking level. For instance, Ethernet networks have Virtual LAN (VLAN) tags and mobile networks have a choice of Access Point Names (APNs). These techniques allow users and traffic to be put on specific networks, which in turn may comprise of virtual components.

Other examples of protocols providing helpful techniques include virtual private networking mechanisms or management mechanisms and data models that can assist in setting up and administering virtualized systems.

There may also be situations where scaling demands changes in protocols. An ability to replicate many instances may push the limits of protocol mechanisms that were designed primarily or originally for physical networks.

#### Selection vs. Creation and Orchestration

Two primary tasks in virtualization should be differentiated: selection of a particular virtual instance, and the tasks related to how that virtual instance was created and continues to be managed.

Selection involves choosing a particular virtual instance, or an endpoint to a virtual network. In its simplest form, a customer could be hardwired by configuration to a particular virtual instance. In more complex cases, the connecting devices may have some settings that affect the choice. In the general case, both the connecting devices and the network they are connecting to it have a say in the choice.

The selection choice may even be dynamic in some cases. For instance, traffic pattern analysis may affect the selection.

Typically, however, connecting devices do not have a say in what the virtual instance does. This is directed by the network operator and its customers. An instance is specified, created, and needs to be continuously managed and orchestrated. The creation can be manual and occur rarely, or be more dynamic, e.g., an instance can actually be instantiated automatically, and only when the first connecting device connects to it.

#### Protocols vs. Representations of Virtual Networks

Some of virtualization technology benefits from protocol support either in the data or control plane. But there are also management constructs, such as data models representing virtual services or networks and data models useful in the construction of such services.

There are also conceptual definitions that may be needed when constructing either protocols or data models or when discussing service agreements between providers and consumers.

#### 4. Virtualization in 5G Networks

Goals for the support of virtualization in 5G relate to both the use of virtualized network functions to build the 5G network, and to enabling the separation of different user or traffic classes into separate network constructs called slices.

Slices enable a separation of concerns, allow the creation of dedicated services for special traffic types, allow faster evolution of the network mechanisms by easing gradual migration to new functionality, and enable faster time to market for new new functionality.

In 5G, slice selection happens as a combination of settings in the User Equipment (UE) and the network. Settings in the UE include, for instance, the Access Point Name (APN), Dedicated Core Network Indicator (DCN-ID) [TS-3GPP.23.401], and, with 5G, a slice indicator (Network Slice Selection Assistance Information or NSSAI) [TS-3GPP.23.501]. This information is combined with the information configured in the network for a given subscriber and the policies of the networks involved. Ultimately, a slice is selected.

A 5G access network carries a user's connection attempt to the 5G core network and the Access Management Function (AMF) network function. This function collects information provided by the UE and the subscriber database from home network, and consults the Network Slice Selection Function (NSSF) to make a decision of the slice selected for the user. When the selection has been made, this may also mean that the connection is moved to a different AMF; enabling separate networks to have entirely different network-level service.

The creation and orchestration of slices does not happen at this signalling plane, but rather the slices are separately specified, created, and managed, typically with the help of an orchestrator function.

The exact mechanisms for doing this continue to evolve, but in any case involve multiple layers of technology, ranging from underlying virtualization software to network component configuration mechanisms and models (often in YANG) to higher abstraction level descriptions (often in TOSCA), to orchestrator software.

#### 5. Overview of IETF Virtualization Technologies

General networking protocols are largely agnostic to virtualization. TCP/IP does not care whether it runs on a physical wire or on a computer-created connection between virtual devices.

As a result, virtualization generally does not affect TCP/IP itself or applications running on top. There are some exceptions, though, such as when the need to virtualize has caused previously held assumptions to break, and the Internet community has had to provide new solutions. For instance, early versions of the HTTP protocol assumed a single host served a single website. The advent of virtual hosting and pressure to not use large numbers of IPv4 addresses lead to HTTP 1.1 adopting virtual hosting, where the identified web host is indicated inside the HTTP protocol rather than inferred from the reception of a request at particular IP address [VirtualHosting] [RFC2616].

But where virtualization affects the Internet architecture and implementations is at lower layers, the physical and MAC layers, the systems that deal with the delivery of IP packets to the right destination, management frameworks controlling these systems, and data models designed to help the creation, monitoring, or management of virtualized services.

What follows is an overview of existing technologies and technologies currently under development that support virtualization in its various forms.

#### 5.1. Selection of Virtual Instances

Some L2 technology allows the identification of traffic belonging to a particular virtual network or connection. For instance, Ethernet VLAN tags.

There are some IETF technologies that also allow similar identification of connections setup with the help of IETF protocols. For instance, Network Access Identifiers may identify a particular customer or virtual service within AAA, EAP or IKEv2 VPN connections.

#### 5.2. Traffic Separation in VPNs

Technologies that assist separation and engineering of networks include both end-point and provider-based VPNs. End-point VPN technologies include, for instance, IPsec-based VPNs [RFC4301].

For providing virtualized services, however, provider-based solutions are often the most relevant ones. L1VPN facilitates virtualization of the underlying L0 "physical" medium. L2[IEEE802.1Q] facilitates virtualization of the underlying Ethernet network Tunneling over IP (MPLS, GRE, VxLAN, IPinIP, L2TP, etc) facilitates virtualization of the underlying IP network - MPLS LSP's - either traffic engineered or not belong here L2VPN facilitates virtualization of a L2 network L3VPN facilitates virtualization of a L3 network.

The IETF has defined a multiplicity of technologies that can be used for provider-based VPNs. The technologies choices available can be described along two axes, control mechanisms and dataplane encapsulation mechanisms. The two are not completely orthogonal.

In the data plane, for provider based VPNs, the first important observation is that the most obvious encapsulation is NOT used. While IPSec could be used for provider-based VPNs, it does not appear to be used in practice, and is not the focus for any of the available control mechanisms. Often, when end2end encryption is required it is used as an overlay over MPLS based L3VPN

The common encapsulation for provider-based VPNs is to use MPLS. This is particularly common for VPNs within one operator, and is sometimes supported across operators.

Keyed GRE can be used, particularly for cross-operator cases. However, it seems to be rare in practice.

The usage of MPLS for provider-based VPNs generally follows a pattern of using two (or more) MPLS labels, top (transport) label to represent the remote end point/egress provider-edge device, and bottom (service) label to signal the different VPNs on the remote end point. Using TE might result in a deeper label stack.

L2 VPNs could be signaled thru LDP[RFC4762] or MP-BGP[RFC4761], L3 VPN is signaled thru MP-BGP[RFC4364]

The LDP usage to control VPN establishment falls within the PALS working group, and is used to establish pseudo-wires to carry Ethernet (or lower layer) traffic. The Ethernet cases tend to be called VPLS (Virtual Private LAN Service) for multi-point connectivity and VPWS (Virtual Private Wire Service) for point-to-point connectivity. These mechanism do augment the data plane capabilities with control words that support additional features. In operation, LDP is used to signal the communicating end-points that are interested in communicating with each other in support of specific VPNs. Information about the MAC addresses used behind the provider edges is exchanged using classic Ethernet flooding technology. It has been proposed to use BGP to bootstrap the exchange of information as to who the communicating endpoints are.

BGP can be used to establish Layer 2 or Layer 3 VPNs. Originally, the BGP based MPLS VPN technology was developed to support layer 3 VPNs. the BGP exchanges uses several different features in MP-BGP (specifically route distinguishers and route targets) to control the distribution of information about VPN end-points. The BGP information carries the VPN IP address prefixes, and the MPLS labels



to be used to represent the VPN. This technology combination is generally known as L3VPN.

This usage of BGP for VPNs has been extended to support Layer 2 VPNs. This is known as EVPN. The BGP exchanges are used to carry the MAC address reachability behind each provider edge router, providing an Ethernet multipoint service without a need to flood unknown-destination Ethernet packets.

In theory, the BGP mechanisms can also be used to support other tunnels such as keyed GRE. That is not widely practiced.

There are also hybrid variations, such as adding an ARP / ND proxy service so that an L3VPN can be used with an L2 Access, when the only desired service is IP.

### 5.3. Traffic Engineering and QoS

Traffic Engineering (TE) is the term used to refer to techniques that enable operators to control how specific traffic flows are treated within their networks.

The TEAS working group works on enhancements to traffic-engineering capabilities for MPLS and GMPLS networks:

TE is applied to packet networks via MPLS TE tunnels and LSPs. The MPLS-TE control plane was generalized to additionally support non-packet technologies via GMPLS. RSVP-TE is the signaling protocol used for both MPLS-TE and GMPLS.

The TEAS WG is responsible for:

- \* Traffic-engineering architectures for generic applicability across packet and non-packet networks.
- \* Definition of protocol-independent metrics and parameters.
- \* Functional specification of extensions for routing (OSPF, ISIS), for path computation (PCE), and RSVP-TE to provide general enablers of traffic-engineering systems.
- \* Definition of control plane mechanisms and extensions to allow the setup and maintenance of TE paths and TE tunnels that span multiple domains and/or switching technologies.

A good example of work that is currently considered in the TEAS WG is the set of models that detail earlier IETF-developed topology models with both traffic engineering information and connection to what

services are running on top of the network  
[I-D.bryskin-teas-use-cases-sf-aware-topo-model]  
[I-D.bryskin-teas-sf-aware-topo-model]. These models enable reasoning about the state of the network with respect to those services, and to set up services with optimal network connectivity.

Traffic engineering is a common requirement for many routing systems, and also discussed, e.g., in the context of LISP.

#### 5.4. Service Chaining

The SFC working group has defined the concept of Service Chaining:

Today, common deployment models have service functions inserted on the data-forwarding path between communicating peers. Going forward, however, there is a need to move to a different model, where service functions, whether physical or virtualized, are not required to reside on the direct data path and traffic is instead steered through required service functions, wherever they are deployed.

For a given service, the abstracted view of the required service functions and the order in which they are to be applied is called a Service Function Chain (SFC). An SFC is instantiated through selection of specific service function instances on specific network nodes to form a service graph: this is called a Service Function Path (SFP). The service functions may be applied at any layer within the network protocol stack (network layer, transport layer, application layer, etc.).

#### 5.5. Management Frameworks and Data Models

There have been two working groups at the IETF, focusing on data models describing VPNs. The IETF and the industry in general is currently specifying a set of YANG models for network element and protocol configuration [RFC6020].

YANG is a powerful and versatile data modeling language that was designed from the requirements of network operators for an easy to use and robust mechanism for provisioning devices and services across networks. It was originally designed at the Internet Engineering Task Force (IETF) and has been so successful that it has been adopted as the standard for modeling design in many other standards bodies such as the Metro Ethernet Forum, OpenDaylight, OpenConfig, and others. The number of YANG modules being implemented for interfaces, devices, and service is growing rapidly.

(It should be noted that there are also other description formats, e.g., Topology and Orchestration Specification for Cloud Applications (TOSCA) [TOSCA-1.0] [TOSCA-Profile-1.1], common in many higher abstract level network service descriptions. The ONAP open source project plans to employ it for abstract mobile network slicing models, for instance.)

A service model is an abstract model, at a higher level than network element or protocol configuration. A service model for VPN service describes a VPN in a manner that a customer of the VPN service would see it.

It needs to be clearly understood that such a service model is not a configuration model. That is, it does not provide details for configuring network elements or protocols: that work is expected to be carried out in other protocol-specific working groups. Instead, service models contain the characteristics of the service as discussed between the operators and their customers. A separate process is responsible for mapping this customer service model onto the protocols and network elements depending on how the network operator chooses to realise the service.

The L2SM WG specifies a service model for L2-based VPNs:

The Layer Two Virtual Private Network Service Model (L2SM) working group is a short-lived WG. It is tasked to create a YANG data model that describes a L2VPN service (a L2VPN customer service model). The model can be used for communication between customers and network operators, and to provide input to automated control and configuration applications.

It is recognized that it would be beneficial to have a common base model that addresses multiple popular L2VPN service types. The working group derives a single data model that includes support for the following:

- \* point-to-point Virtual Private Wire Services (VPWS),
- \* multipoint Virtual Private LAN services (VPLS) that use LDP-signaled Pseudowires,
- \* multipoint Virtual Private LAN services (VPLS) that use a Border Gateway Protocol (BGP) control plane as described in [RFC4761] and [RFC6624],
- \* Ethernet VPNs specified in [RFC7432].

Other L2VPN service types may be included if there is consensus in the working group.

Similarly, the L3SM WG specified a service model for L3-based VPNs.

The Layer Three Virtual Private Network Service Model (L3SM) working group is a short-lived WG tasked to create a YANG data model that describes a L3VPN service (a L3VPN service model) that can be used for communication between customers and network operators, and to provide input to automated control and configuration applications.

It needs to be clearly understood that this L3VPN service model is not an L3VPN configuration model. That is, it does not provide details for configuring network elements or protocols. Instead it contains the characteristics of the service.

## 6. Architectural Observations

This section makes some observations about architectural trends and issues.

### Role of Software

An obvious trend is that bigger and bigger parts of the functionality in a network is driven by software, e.g., orchestration or management tools that figure out how to control relatively simple network element functionality. The software components are where the intelligence is, and a smaller fraction of the intelligence resides in network elements, nor is the intelligence encoded in the behaviour rules of the protocols that the network elements use to communicate with each other.

### Centralization of Functions

An interesting architectural trend is that virtualization and data /software driven networking technologies are driving network architectures where functionality moves towards central entities such as various controllers, path computation servers, and orchestration systems.

A natural consequence of this is the simplification (and perhaps commoditization) of network elements, while the "intelligent" or higher value functions migrate to the center.

The benefits are largely in the manageability, control, and speed of change. There are, however, potential pitfalls to be aware of as well. First off, networks need to continue to be operate even

under partial connectivity situations and breakage, and it is key that designs can handle those situations as well.

And it is important that network users and peers continue to be able to operate and connect in the distributed, voluntary manner that we have today. Today's virtualization technology is primarily used to manage single administrative domains and to offer specific service to others. One could imagine centralised models being taken too far as well, limiting the ability of other network owners to manage their own networks.

#### Tailored vs. general-purpose networking

The interest in building tailored solutions, tailored Quality-of-Service offerings vs. building general-purpose "low touch" networks seems to fluctuate over time.

It is important to find the right balance here. From an economics perspective, it may not be feasible to provide specialised service -- at least if it requires human effort -- for large fraction of use cases. Even if those are very useful in critical applications.

#### Need for descriptions

As networks deal more and more with virtual services, there arises a need to have generally understood, portable descriptions of these service. Hence the creation of YANG data models representing abstract VPN services, for instance.

We can also identify some potential architectural principles, such as:

#### Data model layering

Given the heterogeneity of networking technologies and the differing users that data models are being designed for, it seems difficult to provide a single-level model. It seems preferable to construct a layered set of models, for instance abstract, user-facing models that specify services that can then be mapped to concrete configuration model for networks. And these can in turn be mapped to individual network element configuration models.

Getting this layered design right is crucial for our ability to evolve a useful set of data models.

#### Ability to evolve modelling tools and mapping systems

The networks and their models are complex, and mapping from high abstraction level specifications to concrete network configurations is a hard problem.

It is important that each of the components can evolve on its own. It should be possible to plug in a new language that represents network models better. Or replace a software component that performs mapping between layers to one that works better.

While this should normally be possible, there's room to avoid too tight binding between the different aspects of a system. For instance, abstraction layers within software can shield the software from being too closely tied with a particular representation language.

Similarly, it would be an advantage to develop algorithms and mapping approaches separately from the software that actually does that, so that another piece of software could easily follow the same guidelines and provide an alternate implementation. Perhaps there's an opportunity for specification work to focus more on processing rules than protocol behaviours, for instance.

#### General over specific

In the quick pace of important developments, it is tempting to focus on specific concepts and service offerings such as 5G slicing.

But a preferable approach seems to provide general-purpose tools that can be used by 5G and other networks, and whose longevity exceeds that of a version of a specific offering. The quick development pace is likely driving the evolution of concepts in any case, and building IETF tools that provide the ability to deal with different technologies is most useful.

#### 7. Further Work

There may be needs for further work in this area at the IETF. Before discussing the specific needs, it may be useful to classify the types of useful work that might come to question. And perhaps also outline some types of work that is not appropriate for the IETF.

The IETF works primarily on protocols, but in many cases also with data models that help manage systems, as well as operational guidance documents. But the IETF does not work on software, such as abstractions that only need to exist inside computers or ones that do not have an effect on protocols either on real or simulated "wires".

The IETF also does not generally work on system-level design. IETF is best at designing components, not putting those components together to achieve a particular purpose or build a specific application.

As a result, IETF's work on new systems employing virtualization techniques (such as 5G slicing concept) is more at the component improvement level than at the level of the concept. There needs to be a mapping between a vision of a system and how it utilizes various software, hardware, and protocol tools to achieve the particular virtualization capabilities it needs to. Developing a new concept does not necessarily mean that entirely new solutions are needed throughout the stack. Indeed, systems and concepts are usually built on top of solid, well defined components such as the ones produced by the IETF.

That mapping work is necessarily something that those who want to achieve some new functionality need to do; it is difficult for others to take a position on what the new functionality is. But at the same time, IETF working groups and participants typically have a perspective on how their technology should develop and be extended. Those two viewpoints must meet.

The kinds of potential new work in this space falls generally in the following classes:

#### Virtualization selectors

Sometimes protocols need mechanisms that make it possible to use them as multiple instances. E.g., VLAN tags were added to Ethernet frames, NAIs were added to PPP and EAP, and so on. These cases are rare today, because most protocols and mechanisms have some kind of selector that can be used to run multiple instances or connect to multiple different networks.

#### Traffic engineering

A big reason for building specific networks for specific purposes is to provide an engineered service level on delay and other factors to the given customer. There are a number of different tools in the IETF to help manage and engineer networks, but it is also an area that continues to develop and will likely see new functionality.

#### Virtual service data models

Data models -- such as those described by L2SM or L3SM working groups can represent a "service" offered by a network, a setup built for a specific customer or purpose.

Some specific areas where work is likely needed include:

- o The ability to manage heterogenous technologies, e.g., across SDN and traditionally built networks, or manage both general-purpose and very technology-specific parameters such as those associated with 5G radio.
- o The ability to specify "statistical" rather than hard performance parameters. In some networks -- notably with wireless technology -- recent advances have made very high peak rates possible, but with increased bursty-ness of traffic and with potential bottlenecks on the aggregation parts of the networks. The ability to specify statistical performance in data models and in VPN configuration would be important, over different timescales and probabilities.
- o Mapping from high abstraction level specifications to concrete network configurations.

There is a lot of work on data models and templates at various levels and in different representations. There are also many systems built to manage these models and orchestrate network configuration. But the mapping of the abstract models to concrete network configurations remains a hard problem, and it certainly will need more work.

There are even some questions about how to go about this. Is it enough that we specify models, and leave the mapping to "magic" of the software? Are the connections something that different vendors compete in producing good products in? Or are the mapping algorithms something that needs to be specified together, and their ability to work with different types of network equipment verified in some manner?

- o Cross-domain: A big problem is that we have little tools for cross-domain management of virtualized networks and resources.

Finally, there is a question of where all this work should reside. There's an argument that IETF-based virtualization technologies deserve proper management tools, including data models.

And there's another argument that with the extensive use of virtualization technology, solutions that can manage many different networks should be general, and as such, potential IETF work



material. Yet, the IETF is not and should not be in the space of replacing various tools and open source toolkits that have been created for managing virtualization. It seems though that work on commonly usable data models at several layers of abstraction would be good work at the IETF.

Nevertheless, the IETF should understand where the broader community is and what tools they use for what purpose, and try to help by building on those components. Virtualization and slicing are sometimes represented as issues needing a single solution. In reality, they are an interworking of a number of different tools.

## 8. Acknowledgements

The authors would like to thank Gonzalo Camarillo, Gabriel Montenegro, Alex Galis, Adrian Farrell, Liang Geng, Yi Zhao, Hannu Flinck, Yi Zhao, Barry Leiba, Georg Mayer, Benoit Claise, Daniele Ceccarelli, Warren Kumari, Ted Hardie, and many others for interesting discussions in this problem space.

## 9. Informative References

- [CC2015] claffy, kc. and D. Clark, "Adding Enhanced Services to the Internet: Lessons from History", September 2015 ([https://www.caida.org/publications/papers/2015/adding\\_enhanced\\_services\\_internet/adding\\_enhanced\\_services\\_internet.pdf](https://www.caida.org/publications/papers/2015/adding_enhanced_services_internet/adding_enhanced_services_internet.pdf)).
- [I-D.bryskin-teas-sf-aware-topo-model]  
Bryskin, I. and X. Liu, "SF Aware TE Topology YANG Model", draft-bryskin-teas-sf-aware-topo-model-01 (work in progress), March 2018.
- [I-D.bryskin-teas-use-cases-sf-aware-topo-model]  
Bryskin, I., Liu, X., Guichard, J., Lee, Y., Contreras, L., and D. Ceccarelli, "Use Cases for SF Aware Topology Models", draft-bryskin-teas-use-cases-sf-aware-topo-model-02 (work in progress), March 2018.
- [I-D.geng-coms-problem-statement]  
67, 4., Slawomir, S., Qiang, L., Matsushima, S., Galis, A., and L. Contreras, "Problem Statement of Supervised Heterogeneous Network Slicing", draft-geng-coms-problem-statement-00 (work in progress), September 2017.
- [I-D.ietf-sfc-nsh]

Quinn, P., Elzur, U., and C. Pignataro, "Network Service Header (NSH)", draft-ietf-sfc-nsh-28 (work in progress), November 2017.

- [I-D.king-teas-applicability-actn-slicing]  
King, D. and Y. Lee, "Applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Network Slicing", draft-king-teas-applicability-actn-slicing-01 (work in progress), July 2017.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <<https://www.rfc-editor.org/info/rfc2616>>.
- [RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, DOI 10.17487/RFC4026, March 2005, <<https://www.rfc-editor.org/info/rfc4026>>.
- [RFC4110] Callon, R. and M. Suzuki, "A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC 4110, DOI 10.17487/RFC4110, July 2005, <<https://www.rfc-editor.org/info/rfc4110>>.
- [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>>.
- [RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, DOI 10.17487/RFC4664, September 2006, <<https://www.rfc-editor.org/info/rfc4664>>.
- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<https://www.rfc-editor.org/info/rfc4761>>.
- [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>>.

- [RFC6624] Kompella, K., Kothari, B., and R. Cherukuri, "Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling", RFC 6624, DOI 10.17487/RFC6624, May 2012, <<https://www.rfc-editor.org/info/rfc6624>>.
- [RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <<https://www.rfc-editor.org/info/rfc7432>>.
- [RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <<https://www.rfc-editor.org/info/rfc8049>>.
- [TOSCA-1.0]  
OASIS, "Topology and Orchestration Specification for Cloud Applications Version 1.0", OASIS OASIS Standard, <http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0-os.html>, November 2013.
- [TOSCA-Profile-1.1]  
OASIS, "TOSCA Simple Profile in YAML Version 1.1", OASIS OASIS Standard, <http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.1/TOSCA-Simple-Profile-YAML-v1.1.html>, January 2018.
- [TS-3GPP.23.401]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access; (Release 15)", 3GPP Technical Specification 23.401, December 2017.
- [TS-3GPP.23.501]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture and procedures for 5G System; (Release 15)", 3GPP Technical Specification 23.501, December 2017.
- [VirtualHosting]  
Wikipedia, "Virtual Hosting", Wikipedia article [https://en.wikipedia.org/wiki/Virtual\\_hosting](https://en.wikipedia.org/wiki/Virtual_hosting), August 2017.

Authors' Addresses

Jari Arkko  
Ericsson  
Kauniainen 02700  
Finland

Email: jari.arkko@piuha.net

Jeff Tantsura  
Nuagenetworks

Email: jefftant.ietf@gmail.com

Joel Halpern  
Ericsson

Email: joel.halpern@ericsson.com

Balazs Varga  
Ericsson  
Budapest 1097  
Hungary

Email: balazs.a.varga@ericsson.com

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: June 18, 2018

A. Choudhary  
M. Jethanandani  
Cisco Systems  
N. Strahle  
E. Aries  
Juniper Networks  
I. Chen  
Jabil  
December 15, 2017

YANG Model for QoS  
draft-asechoud-rtgwg-qos-model-03

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 June 18, 2018.

Copyright Notice

Copyright (c) 2017 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 . . . . .	3
3. QoS Model Design . . . . .	3
4. DiffServ Model Design . . . . .	3
5. Modules Tree Structure . . . . .	4
6. Modules . . . . .	8
6.1. IETF-QOS-CLASSIFIER . . . . .	8
6.2. IETF-QOS-POLICY . . . . .	14
6.3. IETF-QOS-ACTION . . . . .	17
6.4. IETF-QOS-TARGET . . . . .	29
6.5. IETF-DIFFSERV . . . . .	31
7. Security Considerations . . . . .	34
8. Acknowledgement . . . . .	34
9. References . . . . .	34
9.1. Normative References . . . . .	34
9.2. Informative References . . . . .	35
Appendix A. Company A, Company B and Company C examples . . . . .	35
A.1. Example of Company A Diffserv Model . . . . .	35
A.2. Example of Company B Diffserv Model . . . . .	45
A.3. Example of Company C Diffserv Model . . . . .	58
Authors' Addresses . . . . .	65

## 1. Introduction

This document defines a base YANG [RFC6020] 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 currently not part of this document and will be added later if necessary. 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. It is left up to individual vendors to stitch some of the actions like queues, random-detect (RED) and vendor specific parameters of the DiffServ policy definitions. Designing the module in this manner allows for a very flexible and extensible module that should fit in with most of the vendor requirements. 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.

## 2. Terminology

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

## 3. 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 five YANG modules - four QoS base modules 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 meter-rate?      uint64
            +--rw meter-burst?     uint64
            +--rw conform-action
              +--rw meter-action-params* [meter-action-type]
                +--rw meter-action-type      identityref
                +--rw (meter-action-val)?
            +--rw exceed-action
              +--rw meter-action-params* [meter-action-type]
                +--rw meter-action-type      identityref
                +--rw (meter-action-val)?
        +--:(one-rate-tri-color-meter-type)
          +--rw one-rate-tri-color-meter
            +--rw committed-rate?      uint64
            +--rw committed-burst?     uint64
            +--rw excess-burst?        uint64
            +--rw conform-action

```

```

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

```

ietf-qos-target module contains reference of qos-policy and augments ietf-interfaces [RFC7223] module. A single policy of a particular policy-type can be applied on an interface in each direction of traffic. Policy-type is of type identity and is populated in a vendor specific manner. This way it provides greater flexibility for each vendor to define different policy types each with its own capabilities and restrictions.

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

```

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

```

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

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

```

module: ietf-diffserv
augment "/classifier:classifiers/classifier:classifier-entry" +
  "/classifier:filter-entry":
  +--rw (filter-param)?
    +--:(dscp)
      |   +--rw dscp-cfg* [dscp-min dscp-max]
      |   |   +--rw dscp-min      inet:dscp
      |   |   +--rw dscp-max      inet:dscp
      +--:(source-ip-address)
      |   +--rw source-ip-address-cfg* [source-ip-addr]
      |   |   +--rw source-ip-addr  inet:ip-prefix
      +--:(destination-ip-address)
      |   +--rw destination-ip-address-cfg* [destination-ip-addr]
      |   |   +--rw destination-ip-addr  inet:ip-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
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-ip-address)
      |   +--rw source-ip-address-cfg* [source-ip-addr]
      |   |   +--rw source-ip-addr  inet:ip-prefix
      +--:(destination-ip-address)
      |   +--rw destination-ip-address-cfg* [destination-ip-addr]

```

```

    |      +---rw destination-ip-addr      inet:ip-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
augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg":
+---rw (action-cfg-params)?
    +---:(dscp-marking)
        +---rw dscp-cfg
        +---rw dscp?      inet:dscp

```

## 6. Modules

### 6.1. IETF-QOS-CLASSIFIER

```

<CODE BEGINS>file "ietf-qos-classifier@2016-03-03.yang"
module iETF-qos-classifier {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-classifier";
  prefix classifier;
  import iETF-inet-types {
    prefix inet;
  }
  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) 2014 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 2016-03-03 {
  description
    "Latest revision of qos base classifier module";
  reference "RFC XXXX";
}
feature policy-inline-classifier-config {
  description
    " This feature allows classifier configuration
    directly under policy.";
}
identity filter-type {
  description
    "This is identity of base filter-type";
}
identity dscp {
  base filter-type;
  description
    "Differentiated services code point filter-type";
}
identity source-ip-address {
  base filter-type;
  description
    "source ipv4 and ipv6 address filter-type";
}
identity destination-ip-address {
  base filter-type;
  description
    "destination ipv4 and ipv6 address filter-type";
}
identity source-port {
  base filter-type;
  description
    "source port filter-type";
}
identity destination-port {
```

```
    base filter-type;
    description
      "destination port filter-type";
  }
  identity protocol {
    base filter-type;
    description
      "protocol type filter-type";
  }
  identity classifier-entry-filter-operation-type {
    description
      "Classifier entry filter logical operation";
  }
  identity match-any-filter {
    base classifier-entry-filter-operation-type;
    description
      "Classifier entry filter logical OR operation";
  }
  identity match-all-filter {
    base classifier-entry-filter-operation-type;
    description
      "Classifier entry filter logical AND operation";
  }
  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-ip-address-cfg {
    list source-ip-address-cfg {
      key "source-ip-addr";
      description
        "list of source ipv4 or ipv6 address";
      leaf source-ip-addr {
```

```
        type inet:ip-prefix;
        description
            "source ipv4 or ipv6 prefix";
    }
}
description
    "Filter grouping containing list of source ip addresses";
}
grouping destination-ip-address-cfg {
    list destination-ip-address-cfg {
        key "destination-ip-addr";
        description
            "list of destination ipv4 or ipv6 address";
        leaf destination-ip-addr {
            type inet:ip-prefix;
            description
                "destination ipv4 or ipv6 prefix";
        }
    }
}
description
    "Filter grouping containing list of destination ip 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 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-any-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 attribute" +
          "must be true";
      }
      key "filter-type filter-logical-not";
      uses filters;
      description
        "Filters configured inline in a policy";
    }
  }
}
container classifiers {
  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@2016-03-03.yang"
module ietf-qos-policy {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-policy";
  prefix policy;
  import 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) 2014 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 2016-03-03 {
  description
    "Latest revision of qos policy";
  reference "RFC XXXX";
}
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@2016-06-15.yang"
module iETF-qos-action {
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-action";
  prefix action;
  import iETF-inet-types {
    prefix inet;
  }
  import iETF-qos-policy {
    prefix policy;
  }
  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) 2014 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 2016-06-15 {
    description

```

```
        "Latest revision for qos actions";
        reference "RFC XXXX";
    }
    feature meter-template-support {
        description
            " This feature allows support of meter-template.";
    }

    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 dscp-marking {
    base policy:action-type;
    description
        "dscp marking action type";
}
identity meter-inline {
    base policy:action-type;
    description
        "meter-inline action type";
}
```

```
identity meter-reference {
  base policy:action-type;
  description
    "meter reference action type";
}
identity min-rate {
  base policy:action-type;
  description
    "min-rate action type";
}
identity max-rate {
  base policy:action-type;
  description
    "max-rate action type";
}
identity queue {
  base policy:action-type;
  description
    "queue action type";
}
identity scheduler {
  base policy:action-type;
  description
    "scheduler action type";
}
identity discard {
  base policy:action-type;
  description
    "discard action type";
}
identity child-policy {
  base policy:action-type;
  description
    "child-policy action type";
}
identity count {
  base policy:action-type;
  description
    "discard 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";
    }
    identity two-rate-tri-color-meter-type {
        base meter-type;
        description
            "two rate three color meter action type";
    }

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

    identity meter-action-type {
        description
            "action type in a meter";
    }
    identity meter-action-drop {
        base meter-action-type;
        description
            "drop action type in a meter";
    }
    identity meter-action-mark-dscp {
        base meter-action-type;
        description
            "dscp mark 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 size or interval configuration";
    choice burst-type {
      case size {
        leaf burst-size {
          type uint64;
          units "bytes";
          description
            "burst size";
        }
      }
      case interval {
        leaf burst-interval {
          type uint64;
          units "microsecond";
          description
            "burst interval";
        }
      }
    }
    description
      "Choice of burst type";
  }
}

grouping threshold {
  description
    "Threshold Parameters";
  container threshold {
    description
      "threshold";
    choice threshold-type {
      case size {
        leaf threshold-size {
          type uint64;
          units "bytes";
          description
```

```
        "Threshold size";
    }
}
case interval {
    leaf threshold-interval {
        type uint64;
        units "microsecond";
        description
            "Threshold interval";
    }
}
description
    "Choice of threshold type";
}
}
}

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

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

grouping meter-action-params {
    description
        "meter action parameters";
    list meter-action-params {
        key "meter-action-type";
        ordered-by user;
        description
```

```
    "Configuration of basic-meter & associated actions";
  leaf meter-action-type {
    type identityref {
      base meter-action-type;
    }
    description
      "meter action type";
  }
  choice 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 meter-rate {
      type uint64;
      units "bits-per-second";
      description
        "meter rate";
    }
    leaf meter-burst {
      type uint64;
      units "bytes";
      description
        "burst size";
    }
  }
  container conform-action {
    uses meter-action-params;
    description
      "conform action";
  }
  container exceed-action {
    uses 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";
  leaf committed-rate {
    type uint64;
    units "bits-per-second";
    description
      "meter rate";
  }
  leaf committed-burst {
    type uint64;
    units "bytes";
    description
      "committed burst size";
  }
  leaf excess-burst {
    type uint64;
    units "bytes";
    description
      "excess burst size";
  }
  container conform-action {
    uses meter-action-params;
    description
      "conform, or green action";
  }
  container exceed-action {
    uses meter-action-params;
    description
      "exceed, or yellow action";
  }
  container violate-action {
    uses 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";
  }
  leaf committed-rate {
    type uint64;
    units "bits-per-second";
    description
```

```
        "meter rate";
    }
    leaf committed-burst {
        type uint64;
        units "bytes";
        description
            "committed burst size";
    }
    leaf peak-rate {
        type uint64;
        units "bits-per-second";
        description
            "meter rate";
    }
    leaf peak-burst {
        type uint64;
        units "bytes";
        description
            "committed burst size";
    }
    container conform-action {
        uses meter-action-params;
        description
            "conform, or green action";
    }
    container exceed-action {
        uses meter-action-params;
        description
            "exceed, or yellow action";
    }
    container violate-action {
        uses 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-type {
            type identityref {
                base meter-type;
            }
            description
                "This leaf defines type of the filter";
        }
        description
            "meter reference";
    }
    description
        "meter reference";
}
```

```
grouping count {
  container count-cfg {
    leaf count-action {
      type empty;
      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";
  }
  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 max-rate {
  container max-rate-cfg {
    uses rate-value-unit;
    uses burst;
    description
      "maximum rate attributes container";
  }
  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@2017-12-12.yang"
module iETF-qos-target {
    yang-version 1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-qos-target";
    prefix target;
    import iETF-interfaces {
        prefix if;
    }
    import iETF-qos-policy {
        prefix policy;
    }
    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>";
    description
        "This module contains a collection of YANG definitions for

```

```
    configuring qos specification implementations.
    Copyright (c) 2014 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 2017-12-12 {
  description
    "Latest revision qos based policy applied to a target";
  reference "RFC XXXX";
}
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@2017-12-12.yang"
module iETF-diffserv {
    yang-version 1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv";
    prefix diffserv;

    import iETF-qos-classifier {
        prefix classifier;
    }
    import iETF-qos-policy {
        prefix policy;
    }
    import iETF-qos-action {
        prefix action;
    }

    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>";
    description
        "This module contains a collection of YANG definitions for
        configuring diffserv specification implementations.
        Copyright (c) 2014 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 2017-12-12 {
  description
    "Latest revision of diffserv based classifier";
  reference "RFC XXXX";
}

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

```
        description
            "Filter Type Protocol";
    }
}
description
    "augments diffserv filters to qos classifier";
}
augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry/policy:filter-entry" {
    choice filter-params {
        description
            "Choice of action types";
        case dscp {
            uses classifier:dscp-cfg;
            description
                "Filter containing list of dscp ranges";
        }
        case source-ip-address {
            uses classifier:source-ip-address-cfg;
            description
                "Filter containing list of source ip addresses";
        }
        case destination-ip-address {
            uses classifier:destination-ip-address-cfg;
            description
                "Filter containing list of destination ip address";
        }
        case source-port {
            uses classifier:source-port-cfg;
            description
                "Filter containing list of source-port ranges";
        }
        case destination-port {
            uses classifier:destination-port-cfg;
            description
                "Filter containing list of destination-port ranges";
        }
        case protocol {
            uses classifier:protocol-cfg;
            description
                "Filter Type Protocol";
        }
    }
}
description
    "Augments Diffserv Classifier with common filter types";
}
augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry/" +
```

```
        "policy:classifier-action-entry-cfg" {
    choice action-cfg-params {
        description
            "Choice of action types";
        case dscp-marking {
            uses action:dscp-marking;
        }
    }
    description
        "augments dscp-marking and meter to qos policy";
    }
}
<CODE ENDS>
```

## 7. Security Considerations

## 8. Acknowledgement

## 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>>.
- [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>>.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <<https://www.rfc-editor.org/info/rfc7223>>.

## 9.2. 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>>.

## 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
- support of hierarchial policy.
- use of inline actions in a policy
- flexibility in marking dscp or metadata at ingress and/or egress.

```
module example-compa-diffserv {  
  namespace "urn:ietf:params:xml:ns:yang:example-compa-diffserv";  
  prefix example;  
  
  import ietf-interfaces {  
    prefix if;  
  }  
  import ietf-qos-classifier {  
    prefix classifier;  
  }  
  import ietf-qos-policy {  
    prefix policy;  
  }
```

```
}
import ietf-qos-action {
  prefix action;
}
import ietf-qos-target {
  prefix target;
}
import ietf-diffserv {
  prefix diffserv;
}

organization "Company A";
contact
  "Editor:   XYZ
   <mailto:xyz@compa.com>";
description
  "This module contains a collection of YANG definitions of
   companyA diffserv specification extension.";
revision 2016-06-15 {
  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";
}

/*****
* Augmentation to Classifier Module
*****/

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";
}

/*****
* Augmentation to Policy Module
*****/

augment "/policy:policies/policy:policy-entry/" +
    "policy:classifier-entry/" +
    "policy:classifier-action-entry-cfg/" +
    "policy:action-cfg-params" {
    case priority {
        uses action:priority;
    }
    case min-rate {
        uses action:min-rate;
    }
    case max-rate {
        uses action:max-rate;
    }
    case random-detect {
        uses randomdetect;
    }
    case meter-inline {
        uses action:meter;
    }
    case child-policy {
        leaf child-policy {
            type leafref {
                path "/policy:policies/policy:policy-entry/" +
                "policy:policy-name";
            }
            description
                "Child Policy in the hierarchial configuration";
        }
    }
    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" +
        "/example:meter-inline" +
        "/example:meter-type" +
        "/example:one-rate-two-color-meter-type" +
        "/example:one-rate-two-color-meter" +
        "/example:conform-action" +
        "/example:meter-action-params" +
        "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:one-rate-two-color-meter-type" +
    "/example:one-rate-two-color-meter" +
    "/example:exceed-action" +
    "/example:meter-action-params" +
    "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:one-rate-tri-color-meter-type" +
    "/example:one-rate-tri-color-meter" +
    "/example:conform-action" +
    "/example:meter-action-params" +
    "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:one-rate-tri-color-meter-type" +
    "/example:one-rate-tri-color-meter" +
    "/example:exceed-action" +
    "/example:meter-action-params" +
    "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:one-rate-tri-color-meter-type" +
    "/example:one-rate-tri-color-meter" +
    "/example:violate-action" +
    "/example:meter-action-params" +
    "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
```

```
        "/example:two-rate-tri-color-meter-type" +
        "/example:two-rate-tri-color-meter" +
        "/example:conform-action" +
        "/example:meter-action-params" +
        "/example: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" +
        "/example:meter-inline" +
        "/example:meter-type" +
        "/example:two-rate-tri-color-meter-type" +
        "/example:two-rate-tri-color-meter" +
        "/example:exceed-action" +
        "/example:meter-action-params" +
        "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:two-rate-tri-color-meter-type" +
    "/example:two-rate-tri-color-meter" +
    "/example:violate-action" +
    "/example:meter-action-params" +
    "/example: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" +
    "/example:meter-inline" +
    "/example:meter-type" +
    "/example:one-rate-two-color-meter-type" +
    "/example: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" +
  "/example:meter-inline" +
  "/example:meter-type" +
  "/example:one-rate-tri-color-meter-type" +
  "/example: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" +
  "/example:meter-inline" +
  "/example:meter-type" +
  "/example:two-rate-tri-color-meter-type" +
  "/example: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";
    }
}

/*****
* Augmentation to Target Module
*****/

augment "/if:interfaces/if:interface/" +
    "target:qos-target-entry/" +
    "target:qos-target-classifier-statistics/" +
    "diffserv:diffserv-action-statistics" {
    uses target:queuing-stats;
    description
        "Augment the statistics to policy entry";
}
augment "/if:interfaces/if:interface/" +
    "target:qos-target-entry/" +
    "target:qos-target-classifier-statistics" {
    leaf relative-path {
        type string;
        description
            "Relative Path of the classifier entry in the
            hierarchial policy";
    }
    description
        "Augment the statistics to policy entry";
}
}

```

#### 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;
  namespace "urn:ietf:params:xml:ns:yang:" +
    "example-compb-diffserv-filter-policy";
  prefix compb-filter-policy;

  import ietf-qos-classifier {
    prefix classifier;
  }
  import ietf-qos-policy {
    prefix policy;
  }
  import ietf-qos-action {
    prefix action;
  }
  import ietf-diffserv {
    prefix diffserv;
  }

  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) 2014 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 2015-04-07 {
    description
      "Latest revision of diffserv policy";
```

```
    reference "RFC XXXX";
  }

/*
 * The policy must be of either type v4 or v6. Corresponding
 * address types must be used. Enforce with "must" statement?
 */
identity v4-diffserv-policy-type {
  base policy:policy-type;
  description
    "This defines default policy-type";
}

identity v6-diffserv-policy-type {
  base policy:policy-type;
  description
    "This defines default policy-type";
}

/*****
 * 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";
}

augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:classifier-action-entry-cfg" +
  "/diffserv: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 meter-reference {
    uses action:meter-reference;
    description
      "Assign a meter as an action";
  }
  case discard {
    uses action:discard;
    description
      "Discard action";
  }
  case count {
    uses action:count;
    description
      "Count action - explicit count configuration";
  }
  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:meter-action-params" +
        "/action:meter-action-val" {
case meter-action-drop {
    description
        "meter drop";
    uses action:drop;
}

description
    "Augment the actions to basic meter";
}
}

module example-compb-queue-policy {
    yang-version 1;
    namespace "urn:ietf:params:xml:ns:yang:example-compb-queue-policy";
    prefix queue-plcy;
```

```
import ietf-qos-classifier {
  prefix classifier;
}
import ietf-qos-policy {
  prefix policy;
}

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) 2014 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 2015-04-07 {
  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;
  namespace "urn:ietf:params:xml:ns:yang:ietf-compb-queue";
  prefix compb-queue;

  import ietf-qos-action {
    prefix action;
  }

  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 2015-04-07 {
    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 guaranteed 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;
    namespace "urn:ietf:params:xml:ns:yang:" +
        "example-compb-scheduler-policy";
    prefix scheduler-plcy;

    import ietf-qos-action {
        prefix action;
    }

    import ietf-qos-policy {
        prefix policy;
    }

    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 2015-04-07 {
```

```

        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-compc-qos-policy {
  yang-version 1.1;
  namespace "urn:example-compc-qos-policy";
  prefix "compcqos";

  import ietf-qos-policy {
    prefix "pol";
  }

  import ietf-qos-action {
    prefix "action";
  }

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

  revision 2016-09-26 {
    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 drived 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 = 'compc-qos-policy'" {
    description
        "Configurations for a classifier-policy-type policy";
}
case metering-or-policing-policy {
    when "../../../sub-type = 'policing-policy' or "
        + "../../../sub-type = 'metering-policy'" {
    }
    container dscp-marking {
        uses action:dscp-marking;
    }
    container precedence-marking {
        uses action:dscp-marking;
    }
    container priority-marking {
        uses action:priority;
    }
    container rate-limiting {
        uses action:one-rate-two-color-meter;
    }
}
case mdrd-queueing-policy {
    when "../../../sub-type = 'mdrr-queueing-policy'" {
        description
            "MDRR queue handling properties for the traffic " +
            "classified into current queue";
    }
    leaf mdrd-queue-weight {
        type uint8 {
            range "20..100";
        }
        units percentage;
    }
}
case pwfq-queueing-policy {
    when "../../../sub-type = 'pwfq-queueing-policy'" {
        description
            "PWFQ queue handling properties for traffic " +
            "classified into current queue";
    }
    leaf pwfq-queue-weight {
        type uint8 {
            range "20..100";
        }
        units percentage;
    }
    leaf pwfq-queue-priority {
```

```
        type uint8;
    }
    leaf pwfq-queue-rate {
        type uint8;
    }
}
case forwarding-policy {
    when "../../../sub-type = 'forwarding-policy'" {
        description
            "Forward policy handling properties for traffic " +
            "in this classifier";
    }
    uses redirect-action-grp;
}
description
    "Add the classify action configuration";
}
```

#### Authors' Addresses

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

Email: asechoud@cisco.com

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

Email: mjethanandani@gmail.com

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

Email: nstrahle@juniper.net

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

Email: [exa@juniper.net](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: May 1, 2018

A. Choudhary  
Cisco Systems  
I. Chen  
Kuatro Technologies  
October 28, 2017

YANG Model for QoS Operational Parameters  
draft-asechoud-rtgwg-qos-oper-model-00

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 May 1, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	2
3. QoS Operational Model Design . . . . .	2
4. Modules Tree Structure . . . . .	3
5. Modules . . . . .	5
5.1. IETF-QOS-OPER . . . . .	5
6. Security Considerations . . . . .	11
7. Revision Tracking . . . . .	11
8. Acknowledgement . . . . .	11
9. Normative References . . . . .	11
Authors' Addresses . . . . .	12

## 1. Introduction

This document defines a base YANG [RFC6020] 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 draft-asechoud-rtgwg-qos-model.

This document doesn't include operational parameters for random-detect (RED, which is left to individual vendor to augment it.

## 2. Terminology

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

## 3. 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.

draft-asechoud-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.



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.

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.

```

module: ietf-qos-oper
augment /if:interfaces/if:interface:
  +--ro qos-classifier-statistics*
    |   +--ro policy-name?          string
    |   +--ro direction?           identityref
    |   +--ro classifier-entry-name? string
    |   +--ro classifier-entry-statistics
    |     +--ro classified-pkts?    uint64
    |     +--ro classified-bytes?   uint64
    |     +--ro classified-rate?    uint64
  +--ro qos-named-statistics*
    |   +--ro stats-name?    string
    |   +--ro pkts?          uint64
    |   +--ro bytes?         uint64
    |   +--ro rate?          uint64
  +--ro metering-statistics*
    |   +--ro policy-name?    string
    |   +--ro direction?     identityref
    |   +--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 policy-name?    string
    |   +--ro direction?     identityref
    |   +--ro queue-id?       string
    |   +--ro queueing-statistics
    |     +--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

```

## 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;
  }

  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>";
  description
    "This module contains a collection of YANG definitions for
    configuring qos specification implementations.
    Copyright (c) 2014 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 2017-10-28 {
    description
      "Latest revision qos based policy applied to a target";
    reference "RFC XXXX";
  }

  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
            "Classifier Counters";
        container classifier-entry-statistics {
            config false;
            description
                "
                This group defines the classifier filter statistics 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 qos-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 queuing-stats {
  description
    "Queuing Counters";
  container queuing-statistics {
    description
      "queue related statistics ";
    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 ";
    }
}

grouping meter-stats {
    description
        "Metering Statistics";
    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";
    }
}

augment "/if:interfaces/if:interface" {
    description
        "Augments Qos Target Entry to Interface module";
    list qos-classifier-statistics {

```

```

    config false;
    description
        "Statistics for each Classifier Entry in a Policy applied
        in a particular direction";
    leaf policy-name {
        type string;
        description
            "Policy entry name";
    }
    leaf direction {
        type identityref {
            base direction;
        }
        description
            "Direction fo the traffic flow either inbound or outbound";
    }
    leaf classifier-entry-name {
        type string;
        description
            "Classifier Entry Name";
    }
    uses classifier-entry-stats;
}
list qos-named-statistics {
    config false;
    description
        "Matched Statistics for a statistics-name";
    leaf stats-name {
        type string;
        description
            "Statistics name";
    }
    uses qos-named-stats;
}
list metering-statistics {
    config false;
    description
        "Statistics for each Meter associated with the Policy";
    leaf policy-name {
        type string;
        description
            "Policy entry name";
    }
    leaf direction {
        type identityref {
            base direction;
        }
        description

```



```
        "Direction fo the traffic flow either inbound or outbound";
    }
    leaf meter-id {
        type string;
        description
            "Meter Identifier";
    }
    uses meter-stats;
}
list queueing-statistics {
    config false;
    description
        "Statistics for each Queue associated with the Policy";
    leaf policy-name {
        type string;
        description
            "Policy entry name";
    }
    leaf direction {
        type identityref {
            base direction;
        }
        description
            "Direction fo the traffic flow either inbound or outbound";
    }
    leaf queue-id {
        type string;
        description
            "Queue Identifier";
    }
    uses queueing-stats;
}
}
```

<CODE ENDS>

6. Security Considerations
7. Revision Tracking
8. Acknowledgement
9. 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>>.
- [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>>.

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  
Kuatro Technologies

Email: [ichen@kuatrotech.com](mailto:ichen@kuatrotech.com)

Routing Area Working Group  
Internet-Draft  
Intended status: Informational  
Expires: September 6, 2018

S. Bryant  
J. Dong  
Huawei  
Z. Li  
China Mobile  
T. Miyasaka  
KDDI Corporation  
March 05, 2018

Enhanced Virtual Private Networks (VPN+)  
draft-bryant-rtgwg-enhanced-vpn-02

Abstract

This draft describes a number of enhancements that need to be made to virtual private networks (VPNs) to support the needs of new applications, particularly applications that are associated with 5G services. A network enhanced with these properties may form the underpin of network slicing, but will also be of use in its own right.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 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
2. Requirements Language . . . . .	4
3. Overview of the Requirements . . . . .	4
3.1. Isolation between Virtual Networks . . . . .	4
3.2. Diverse Performance Guarantees . . . . .	6
3.3. A Pragmatic Approach to Isolation . . . . .	7
3.4. Integration . . . . .	8
3.5. Dynamic Configuration . . . . .	8
3.6. Customized Control Plane . . . . .	9
4. Architecture and Components of VPN+ . . . . .	9
4.1. Communications Layering . . . . .	9
4.2. Multi-Point to Multi-point . . . . .	10
4.3. Candidate Underlay Technologies . . . . .	10
4.3.1. FlexE . . . . .	11
4.3.2. Dedicated Queues . . . . .	12
4.3.3. Time Sensitive Networking . . . . .	12
4.3.4. Deterministic Networking . . . . .	12
4.3.5. MPLS Traffic Engineering (MPLS-TE) . . . . .	13
4.3.6. Segment Routing . . . . .	13
4.4. Control Plane Considerations . . . . .	16
4.5. Application Specific Network Types . . . . .	17
4.6. Integration with Service Functions . . . . .	17
5. Scalability Considerations . . . . .	17
5.1. Maximum Stack Depth . . . . .	18
5.2. RSVP scalability . . . . .	18
6. OAM and Instrumentation . . . . .	19
7. Enhanced Resiliency . . . . .	19
8. Security Considerations . . . . .	20
9. IANA Considerations . . . . .	20
10. References . . . . .	21
10.1. Normative References . . . . .	21
10.2. Informative References . . . . .	21
Authors' Addresses . . . . .	22

## 1. Introduction

Virtual networks, often referred to as virtual private networks (VPNs) have served the industry well as a means of providing different groups of users with logically isolated access to a common network. The common or base network that is used to provide the VPNs

is often referred to as the underlay, and the VPN is often called an overlay.

Driven largely by needs surfacing from 5G, the concept of network slicing has gained traction. There is a need to create a VPN with enhanced characteristics. Specifically there is a need for a transport network supporting a set of virtual networks each of which provides the client with dedicated (private) networking, computing and storage resources drawn from a shared pool. The tenant of such a network can require a degree of isolation and performance that previously could only be satisfied by dedicated networks. Additionally the tenant may ask for some level of control of their virtual network e.g. to customize the service paths in the network slice.

These properties cannot be met with pure overlay networks, as they require tighter coordination and integration between the underlay and the overlay network. This document introduces a new network service called enhanced VPN (VPN+). VPN+ refers to a virtual network which has dedicated network resources allocated from the underlay network. Unlike traditional VPN, an enhanced VPN can achieve greater isolation and guaranteed performance.

These new network layer properties, which have general applicability, may also be of interest as part of a network slicing solution.

This document specifies a framework for using the existing, modified and potential new networking technologies as components to provide an enhanced VPN (VPN+) service. Specifically we are concerned with:

- o The design of the enhanced VPN data-plane
- o The necessary protocols in both, underlay and the overlay of enhanced VPN, and
- o The mechanisms to achieve integration between overlay and underlay
- o The necessary method of monitoring an enhanced VPN
- o The methods of instrumenting an enhanced VPN to ensure that the required tenant Service Level Agreement (SLA) is maintained

The required layer structure necessary to achieve this is shown in Section 4.1.

One use for enhanced VPNs is to create network slices with different isolation requirements. Such slices may be used to provide different tenants of vertical industrial markets with their own virtual network

with the explicit characteristics required. These slices may be "hard" slices providing a high degree of confidence that the VPN+ characteristics will be maintained over the slice life cycle, or they may be "soft" slices in which case some degree of interaction may be experienced.

## 2. Requirements Language

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

## 3. Overview of the Requirements

In this section we provide an overview of the requirements of an enhanced VPN.

### 3.1. Isolation between Virtual Networks

The requirement is to provide both hard and soft isolation between the tenants/applications using one enhanced VPN and the tenants/applications using another enhanced VPN. Hard isolation is needed so that applications with exacting requirements can function correctly despite a flash demand being created on another VPN competing for the underlying resources. An example might be a network supporting both emergency services and public broadband multi-media services.

During a major incident the VPNs supporting these services would both be expected to experience high data volumes, and it is important that both make progress in the transmission of their data. In these circumstances the VPNs would require an appropriate degree of isolation to be able to continue to operate acceptably.

We introduce the terms hard (static) and soft (dynamic) isolation to cover cases such as the above. A VPN has soft isolation if the traffic of one VPN cannot be inspected by the traffic of another. Both IP and MPLS VPNs are examples of soft isolated VPNs because the network delivers the traffic only to the required VPN endpoints. However the traffic from one or more VPNs and regular network traffic may congest the network resulting in delays for other VPNs operating normally. The ability for a VPN to be sheltered from this effect is called hard isolation, and this property is required by some critical applications. Although these isolation requirements are triggered by the needs of 5G networks, they have general utility. In the remainder of this section we explore how isolation may be achieved in packet networks.

It is of course possible to achieve high degrees of isolation in the optical layer. However this is done at the cost of allocating resources on a long term basis and end-to-end basis. Such an arrangement means that the full cost of the resources must be borne by the service that is allocated the resources. On the other hand, isolation at the packet layer allows the resources to be shared amongst many services and only dedicated to a service on a temporary basis. This allows greater statistical multiplexing of network resources and amortizes the cost over many services, leading to better economy. However, the degree of isolation required by network slicing cannot easily be met with MPLS-TE packet LSPs as they guarantee long-term bandwidth, but not latency.

Thus some trade-off between the two approaches needs to be considered to provide the required isolation between virtual networks while still allows reasonable sharing inside each VPN.

The work of the IEEE project on Time Sensitive Networking is introducing the concept of packet scheduling where a high priority packet stream may be given a scheduled time slot thereby guaranteeing that it experiences no queuing delay and hence a reduced latency. However where no scheduled packet arrives its reserved time-slot is handed over to best effort traffic, thereby improving the economics of the network. Such a scheduling mechanism may be usable directly, or with extension to achieve isolation between multiple VPNs.

One of the key areas in which isolation needs to be provided is at the interfaces. If nothing is done the system falls back to the router queuing system in which the ingress places it on a selected output queue. Modern routers have quite sophisticated output queuing systems, traditionally these have not provided the type of scheduling system needed to support the levels of isolation needed for the applications that are the target of VPN+ networks. However some of the more modern approaches to queuing allow the construction of logical virtual channelized sub-interfaces (VCSI). With VCSIs there is only one physical interface, and routing sees a single adjacency, but the queuing system is used to provide virtual interfaces at various priorities. Sophisticated queuing systems of this type may be used to provide end-to-end virtual isolation between tenant's traffic in an otherwise homogeneous network.

[FLEXE] provides the ability to multiplex multiple channels over an Ethernet link in a way that provides hard isolation. However it is a only a link technology. When packets are received by the downstream node they need to be processed in a way that preserves that isolation. This in turn requires a queuing and forwarding implementation that preserves the isolation, such as a sliced hardware system, or an LVI system of the type described above.

### 3.2. Diverse Performance Guarantees

There are several aspects to guaranteed performance, guaranteed maximum packet loss, guaranteed maximum delay and guaranteed delay variation.

Guaranteed maximum packet loss is a common parameter, and is usually addressed by setting the packet priorities, queue size and discard policy. However this becomes more difficult when the requirement is combine with the latency requirement. The limiting case is zero congestion loss, and than is the goal of the Deterministic Networking work that the IETF and IEEE are pursuing. In modern optical networks loss due to transmission errors is already asymptotic to zero due, but there is always the possibility of failure of the interface and the fiber itself. This can only be addressed by some form of packet duplication and transmission over diverse paths.

Guaranteed maximum latency is required in a number of applications particularly real-time control applications and some types of virtual reality applications. The work of the IETF Deterministic Networking (DetNet) Working Group is relevant, however the scope needs to be extended to methods of enhancing the underlay to better support the delay guarantee, and to integrate these enhancements with the overall service provision.

Guaranteed maximum delay variation is a service that may also be needed. Time transfer is one example of a service that needs this, although the fungible nature of time means that it might be delivered by the underlay as a shared service and not provided through different virtual networks. Alternatively a dedicated virtual network may be used to provide this as a shared service. The need for guaranteed maximum delay variation as a general requirement is for further study.

This leads to the concept that there is a spectrum of grades of service guarantee that need to be considered when deploying and enhanced VPN. As a guide to understanding the design requirements we can consider four types:

- o Guaranteed latency,
- o Enhanced delivery
- o Assured bandwidth,
- o Best effort



In Section 3.1 we considered the work of the IEEE Time Sensitive Networking (TSN) project and the work of the IETF DetNet Working group in the context of isolation. However this work is of greater relevance in assuring end-to-end packet latency. It is also of importance in considering enhanced delivery.

A service that is guaranteed latency has a latency upper bound provided by the network. It is important to note that assuring the upper bound is more important than achieving the minimum latency.

A service that is offered enhanced delivery is one in which the network (at layer 3) attempts to deliver the packet through multiple paths in the hope of avoiding transient congestion [I-D.ietf-detnet-dp-sol].

A useful mechanism to provide these guarantees is to use Flex Ethernet [FLEXE] as the underlay. This is a method of bonding Ethernets together and of providing time-slot based channelization over an Ethernet bearer. Such channels are fully isolated from other channels running over the same Ethernet bearer. As noted elsewhere this produces hard isolation but at the cost of making the reclamation of unused bandwidth harder.

These approaches can usefully be used in tandem. It is possible to use FlexE to provide tenant isolation, and then to use the TSN approach over FlexE to provide service performance guarantee inside the a slice/tenant VPN.

### 3.3. A Pragmatic Approach to Isolation

A key question to consider is whether whether it is possible to achieve hard isolation in packet networks? Packet networks were never designed to support hard isolation, just the opposite, they were designed to provide a high degree of statistical multiplexing and hence a significant economic advantage when compared to a dedicated, or a Time Division Multiplexing (TDM) network. However the key thing to bear in mind is that the concept of hard isolation needs to be viewed from the perspective of the application, and there is no need to provide any harder isolation than is required by the application. From a historical perspective it is good to think about pseudowires [RFC3985] which emulate services that in many would have had hard isolation in their native form. However experience has shown that in most cases an approximation to this requirement is sufficient for most uses.

Thus, for example, using FlexE or channelized sub-interface, together with packet scheduling as interface slicing, and optionally, also together with the slicing of node resources (Network Processor Unit

(NPU), etc.), it may be possible to provide a type of hard isolation that is adequate for many applications. Other applications may be satisfied with a classical VPN and reserved bandwidth, but yet others may require dedicated point to point fiber. The requirement is thus to qualify the needs of each application and provide an economic solution that satisfies those needs without over-engineering.

### 3.4. Integration

A solution to the enhanced VPN problem will need to provide seamless integration of both Overlay VPN and the underlay network resources. This needs to be done in a flexible and scalable way so that it can be widely deployed in operator networks. Given the targeting of both this technology and service function chaining at mobile networks and in particular 5G the co-integration of service functions is a likely requirement.

### 3.5. Dynamic Configuration

It is necessary that new enhanced VPNs can be introduced to the network, modified, and removed from the network according to service demand. In doing so due regard must be given to the impact of other enhanced VPNs that are operational. An enhanced VPN that requires hard isolation must not be disrupted by the installation or modification of another enhanced VPN.

Whether modification of an enhanced VPN can be disruptive to that VPN, and in particular the traffic in flight is to be determined, but is likely to be a difficult problem to address.

The data-plane aspect of this are discussed further in Section 4.3.

The control-plane and management-plane aspects of this, particularly the garbage collection are likely to be challenging and are for further study.

As well as managing dynamic changes to the VPN in a seamless way, dynamic changes to the underlay and its transport network need to be managed in order to avoid disruption to sensitive services.

In addition to non-disruptively managing the network as a result of gross change such as the inclusion of a new VPN endpoint or a change to a link, consideration has to be given to the need to move VPN traffic as a result of traffic volume changes.

### 3.6. Customized Control Plane

In some cases it is desirable that an enhanced VPN has a custom control-plane, so that the tenant of the enhanced VPN can have some control to the resources and functions partitioned for this VPN. Each enhanced VPN may have its own dedicated controller, it may be provided with an interface to a control-plane that is shared with a set of other tenants, or it may be provided with an interface to the control-plane of the underlay provided by the underlay network operator.

Further detail on this requirement will be provided in a future version of the draft.

## 4. Architecture and Components of VPN+

Normally a number of enhanced VPN services will be provided by a common network infrastructure. Each enhanced VPN consists of both the overlay and a specific set of dedicated network resources and functions allocated in the underlay to satisfy the needs of the VPN tenant. The integration between overlay and underlay ensures the isolation and between different enhanced VPNs, and facilitates the guaranteed performance for different services.

An enhanced VPN needs to be designed with consideration given to:

- o Isolation of enhanced VPN data plane.
- o A scalable control plane to match the data plane isolation.
- o The amount of state in the packet vs the amount of state in the control plane.
- o Mechanism for diverse performance guarantee within an enhanced VPN
- o Support of the required integration between network functions and service functions.

### 4.1. Communications Layering

The communications layering model use to build an enhanced VPN is shown in Figure 1.

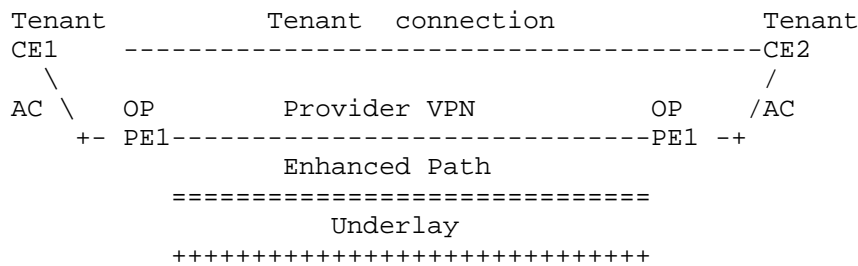


Figure 1: Communication Layering

The network operator is required to provide a tenant connection between the tenant's Customer Equipment (CE) (CE1 and CE2). These CEs attach to the Operator's Provider Edge Equipments (PE) (PE1 and PE2 respectively). The attachment circuits (AC) are outside the scope of this document other than to note that they obviously need to provide a connection of sufficient quality in terms of isolation, latency etc so as to satisfy the needs of the user. The subtlety to be aware of is that the ACs are often provided by a network rather than a fixed point to point connection and thus the considerations in this document may apply to the network that provides the AC.

A provider VPN is constructed between PE1 and PE2 to carry tenant traffic. This is a normal VPN, and provides one stage of isolation between tenants.

An enhanced path is constructed to carry the provider VPN using dedicated resources drawn from the underlay.

## 4.2. Multi-Point to Multi-point

At a VPN level connections are frequently multi-point-to-multi-point (MP2MP). As far as such services are concerned the underlay is also an abstract MP2MP medium. However when service guarantees are provided, such as with an enhanced VPN, each point to point path through the underlay needs to be specifically engineered to meet the required performance guarantees.

### 4.3. Candidate Underlay Technologies

A VPN is a network created by applying a multiplexing technique to the underlying network (the underlay) in order to distinguish the traffic of one VPN from that of another. A VPN path that travels by other than the shortest path through the underlay normally requires state in the underlay to specify that path. State is normally applied to the underlay through the use of the RSVP Signaling protocol, or directly through the use of an SDN controller, although

other techniques may emerge as this problem is studied. This state gets harder to manage as the number of VPN paths increases. Furthermore, as we increase the coupling between the underlay and the overlay to support the VPN which requires enhanced VPN service, this state will increase further.

In an enhanced VPN different subsets of the underlay resources are dedicated to different VPNs. Any enhanced VPN solution thus needs tighter coupling with underlay than is the case with classical VPNs. We cannot for example share the tunnel between enhanced VPNs which require hard isolation.

In the following sections we consider a number of candidate underlay solutions for proving the required VPN separation.

- o FlexE
- o Time Sensitive Networking
- o Deterministic Networking
- o Dedicated Queues

We then consider the problem of slice differentiation and resource representation. Candidate technologies are:

- o MPLS
- o MPLS-SR
- o Segment Routing over IPv6 (SRv6)

#### 4.3.1. FlexE

FlexE [FLEXE] is a method of creating a point-to-point Ethernet with a specific fixed bandwidth. FlexE supports the bonding of multiple links, which supports creating larger links out of multiple slower links in a more efficient way than traditional link aggregation. FlexE also supports the sub-rating of links, which allows an operator to only use a portion of a link. FlexE also supports the channelization of links, which allows one link to carry several lower-speed or sub-rated links from different sources.

If different FlexE channels are used for different services, then no sharing is possible between the services. This in turn means that it is not possible to dynamically re-distribute unused bandwidth to lower priority services increasing the cost of operation of the network. FlexE can on the other hand be used to provide hard

isolation between different tenants by providing hard isolation on an interface. The tenant can then use other methods to manage the relative priority of their own traffic.

Methods of dynamically re-sizing FlexE channels and the implication for enhanced VPN are under study.

#### 4.3.2. Dedicated Queues

In an enhanced VPN providing multiple isolated virtual networks the conventional Diff-Serv based queuing system is insufficient for our purposes due to the limited number of queues which cannot differentiate between traffic of different VPNs and the range of service classes that each need to provide their tenants. This problem is particularly acute with an MPLS underlay due to the small number of traffic class services available. In order to address this problem and thus reduce the interference between VPNs, it is likely to be necessary to steer traffic of VPNs to dedicated input and output queues.

#### 4.3.3. Time Sensitive Networking

Time Sensitive Networking (TSN) is an IEEE project that is designing a method of carrying time sensitive information over Ethernet. As Ethernet this can obviously be tunneled over a Layer 3 network in a pseudowire. However the TSN payload would be opaque to the underlay and thus not treated specifically as time sensitive data. The preferred method of carrying TSN over a layer 3 network is through the use of deterministic networking as explained in the following section of this document.

The mechanisms defined in TSN can be used to meet the requirements of time sensitive services of an enhanced VPN.

#### 4.3.4. Deterministic Networking

Deterministic Networking (DetNet) [I-D.ietf-detnet-architecture] is a technique being developed in the IETF to enhance the ability of layer 3 networks to deliver packets more reliably and with greater control over the delay. The design cannot use classical re-transmission techniques such as TCP since can add delay that is above the maximum tolerated by the applications. Even the delay improvements that are achieved with SCTP-PR are outside the bounds set by application demands. The approach is to pre-emptively send copies of the packet over various paths in the expectation that this minimizes the chance of all packets being lost, but to trim duplicate packets to prevent excessive flooding of the network and to prevent multiple packets being delivered to the destination. It also seeks to set an upper

bound on latency. Note that it is not the goal to minimize latency, and the optimum upper bound paths may not be the minimum latency paths.

DetNet is based on flows. It currently makes no comment on the underlay, and so at this stage must be assumed to use the base topology. To be of use in this application DetNet there needs to be a description of how to deal with the concept of flows within an enhanced VPN.

How we use DetNet in a multi-tenant (VPN) network, and how to improve the scalability of DetNet in a multi-tenant (VPN) network is for further study.

#### 4.3.5. MPLS Traffic Engineering (MPLS-TE)

Normal MPLS runs on the base topology and has the concepts of reserving end to end bandwidth for an LSP, and of creating VPNs. VPN traffic can be run over RSVP-TE tunnels to provide reserved bandwidth for a specific VPN connection. This is rarely deployed in practice due to scaling and management overhead concerns.

#### 4.3.6. Segment Routing

Segment Routing [I-D.ietf-spring-segment-routing] is a method that prepends instructions to packets at entry and sometimes at various points as it passes through the network. These instructions allow packets to be routed on paths other than the shortest path for various traffic engineering reasons. These paths can be strict or loose paths, depending on the compactness required of the instruction list and the degree of autonomy granted to the network (for example to support ECMP).

With SR, a path needs to be dynamically created through a set of resources by simply specifying the Segment IDs (SIDs), i.e. instructions rooted at a particular point in the network. Thus if a path is to be provisioned from some ingress point A to some egress point B in the underlay, A is provided with the A..B SID list and instructions on how to identify the packets to which the SID list is to be prepended.

By encoding the state in the packet, as is done in Segment Routing, state is transitioned out of the network.

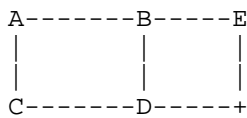


Figure 2: An SR Network Fragment

Consider the network fragment shown in Figure 2. To send a packet from A to E via B, D & E: Node A prepends the ordered list of SIDs: D, E to the packet and pushes the packet to B. SID list {B, D, E} can be used as a VPN path. Thus, to create a VPN, a set of SID Lists is created and provided to each ingress node of the VPN together with packet selection criteria. In this way it is possible to create a VPN with no state in the core. However this is at the expense of creating a larger packet with possible MTU and hardware restriction limits that need to be overcome.

Note in the above if A and E support multiple VPN an additional VPN identifier will need to be added to the packet, but this is omitted from this text for simplicity.

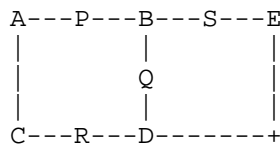


Figure 3: Another SR Network Fragment

Consider a further network fragment shown in Figure 3, and further consider VPN A+D+E.

A has lists: {P, B, Q, D}, {P, B, S, E}  
 D has lists: {Q, B, P, A}, {E}  
 E has lists: {S, B, P, A}, {D}

To create a new VPN C+D+B the following list are introduced:

C lists: {R, D}, {A, P, B}  
 D lists: {R, C}, {Q, B}  
 B lists: {Q, D}, {P, A, C}

Thus VPN C+D+B was created without touching the settings of the core routers, indeed it is possible to add endpoints to the VPNs, and move the paths around simply by providing new lists to the affected endpoints.



There are a number of limitations in SR as it is currently defined that limit its applicability to enhanced VPNs:

- o Segments are shared between different VPNs,
- o There is no reservation of bandwidth,
- o There is limited differentiation in the data plane.

Thus some extensions to SR are needed to provide isolation between different enhanced VPNs. This can be achieved by including a finer granularity of state in the core in anticipation of its future use by authorized services. We therefore need to evaluate the balance between this additional state and the performance delivered by the network.

Both MPLS Segment Routing and SRv6 Segment Routing are candidate technologies for enhanced VPN.

With current segment routing, the instructions are used to specify the nodes and links to be traversed. However, in order to achieve the required isolation between different services, new instructions can be created which can be prepended to a packet to steer it through specific dedicated network resources and functions, e.g. links, queues, processors, services etc.

Clearly we can use traditional constructs to create a VPN, but there are advantages to the use of other constructs such as Segment Routing (SR) in the creation of virtual networks with enhanced properties.

Traditionally a traffic engineered path operates with a granularity of a link with hints about priority provided through the use of the traffic class field in the header. However to achieve the latency and isolation characteristics that are sought by VPN+ users, steering packets through specific queues resources will likely be required. The extent to which these needs can be satisfied through existing QoS mechanisms is to be determined. What is clear is that a fine control of which services wait for which, with a fine granularity of queue management policy is needed. Note that the concept of a queue is a useful abstraction for many types of underlay mechanism that may be used to provide enhanced latency support. From the perspective of the control plane and from the perspective of the segment routing the method of steering a packet to a queue that provides the required properties is a universal construct. How the queue satisfies the requirement is outside the scope of these aspect of the enhanced VPN system. Thus for example a FlexE channel, or time sensitive networking packet scheduling slot are abstracted to the same concept and bound to the data plane in a common manner.

We can introduce the specification of finer, deterministic, granularity to path selection through extensions to traditional path construction techniques such as RSVP-TE and MPLS-TP.

We can also introduce it by specifying the queue through an SR instruction list. Thus new SR instructions may be created to specify not only which resources are traversed, but in some cases how they are traversed. For example, it may be possible to specify not only the queue to be used but the policy to be applied when enqueueing and dequeuing.

This concept can be further generalized, since as well as queuing to the output port of a router, it is possible to queue to any resource, for example:

- o A network processor unit (NPU)
- o A Central Processing Unit (CPU) Core
- o A Look-up engine such as TCAMs

#### 4.4. Control Plane Considerations

It is expected that VPN+ would be based on a hybrid control mechanism, which takes advantage of the logically centralized controller for on-demand provisioning and global optimization, whilst still relies on distributed control plane to provide scalability, high reliability, fast reaction, automatic failure recovery etc. Extension and optimization to the distributed control plane is needed to support the enhanced properties of VPN+.

Where SR is used as a the data-plane construct it needs to be noted that it does not have the capability of reserving resources along the path nor do its currently specified distributed control plane (the link state routing protocols). An SDN controller can clearly do this, from the controllers point of view, and no resource reservation is done on the device. Thus if a distributed control plane is needed either in place of an SDN controller or as an assistant to it, the design of the control system needs to ensure that resources are uniquely allocated to the correct service, and no allocated to multiple services causing unintended resource conflict. This needs further study.

On the other hand an advantage of using an SR approach is that it provides a way of efficiently binding the network underlay and the enhanced VPN overlay. With a technology such as RSVP-TE LSPs, each virtual path in the VPN is bound to the underlay with a dedicated TE-LSP.

RSVP-TE could be enhanced to bind the VPN to specific resources within the underlay, but as noted elsewhere in this document there are concerns as to the scalability of this approach. With an SR-based approach to resource reservation (per-slice reservation), it is straightforward to create dedicated SR network slices, and the VPN can be bound to a particular SR network slice.

#### 4.5. Application Specific Network Types

Although a lot of the traffic that will be carried over the enhanced VPN will likely be IPv4 or IPv6, the design has to be capable of carrying other traffic types. In particular the design SHOULD be capable of carrying Ethernet traffic. This is easily accomplished through the various pseudowire (PW) techniques [RFC3985]. Where the underlay is MPLS Ethernet can be carried over the enhanced VPN encapsulated according to the method specified in [RFC4448]. Where the underlay is IP Layer Two Tunneling Protocol - Version 3 (L2TPv3) [RFC3931] can be used with Ethernet traffic carried according to [RFC4719]. Encapsulations have been defined for most of the common layer two type for both PW over MPLS and for L2TPv3.

#### 4.6. Integration with Service Functions

There is a significant overlap between the problem of routing a packet through a set of network resources and the problem of routing a packet through a set of compute resources. Service Function Chain technology is designed to forward a packet through a set of compute resources.

A future version of this document will discuss this further.

### 5. Scalability Considerations

For a packet to transit a network, other than on a best effort, shortest path basis, it is necessary to introduce additional state, either in the packet, or in the network or some combination of both.

There are at least three ways of doing this:

- o Introduce the complete state into the packet. That is how SR does this, and this allows the controller to specify the precise series of forwarding and processing instructions that will happen to the packet as it transits the network. The cost of this is an increase in the packet header size. The cost is also that systems will have capabilities enabled in case they are called upon by a service. This is a type of latent state, and increases as we more precisely specify the path and resources that need to be exclusively available to a VPN.

- o Introduce the state to the network. This is normally done by creating a path using RSVP-TE, which can be extended to introduce any element that needs to be specified along the path, for example explicitly specifying queuing policy. It is of course possible to use other methods to introduce path state, such as via a Software Defined Network (SDN) controller, or possibly by modifying a routing protocol. With this approach there is state per path per path characteristic that needs to be maintained over its life-cycle. This is more state than is needed using SR, but the packet are shorter.
- o Provide a hybrid approach based on using binding SIDs to create path fragments, and bind them together with SR.

Dynamic creation of a VPN path using SR requires less state maintenance in the network core at the expense of larger VPN headers on the packet. The scaling properties will reduce roughly from a function of  $(N/2)^2$  to a function of  $N$ , where  $N$  is the VPN path length in intervention points (hops plus network functions). Reducing the state in the network is important to VPN+, as VPN+ requires the overlay to be more closely integrated with the underlay than with traditional VPNs. This tighter coupling would normally mean that significant state needed to be created and maintained in the core. However, a segment routed approach allows much of this state to be spread amongst the network ingress nodes, and transiently carried in the packets as SIDs.

These approaches are for further study.

#### 5.1. Maximum Stack Depth

One of the challenges with SR is the stack depth that nodes are able to impose on packets. This leads to a difficult balance between adding state to the network and minimizing stack depth, or minimizing state and increasing the stack depth.

#### 5.2. RSVP scalability

The traditional method of creating a resource allocated path through an MPLS network is to use the RSVP protocol. However there have been concerns that this requires significant continuous state maintenance in the network. There are ongoing works to improve the scalability of RSVP-TE LSPs in the control plane [I-D.ietf-teas-rsvp-te-scaling-rec]. This will be considered further in a future version of this document.

There is also concern at the scalability of the forwarder footprint of RSVP as the number of paths through an LSR grows

[I-D.sitaraman-mpls-rsvp-shared-labels] proposes to address this by employing SR within a tunnel established by RSVP-TE. This work will be considered in a future version of this document.

## 6. OAM and Instrumentation

A study of OAM in SR networks has been documented in [I-D.ietf-spring-oam-usecase].

The enhanced VPN OAM design needs to consider the following requirements:

- o Instrumentation of the underlay so that the network operator can be sure that the resources committed to a tenant are operating correctly and delivering the required performance.
- o Instrumentation of the overlay by the tenant. This is likely to be transparent to the network operator and to use existing methods. Particular consideration needs to be given to the need to verify the isolation and the various committed performance characteristics.
- o Instrumentation of the overlay by the network provider to proactively demonstrate that the committed performance is being delivered. This needs to be done in a non-intrusive manner, particularly when the tenant is deploying a performance sensitive application
- o Verification of the conformity of the path to the service requirement. This may need to be done as part of a commissioning test.

These issues will be discussed in a future version of this document.

## 7. Enhanced Resiliency

Each enhanced VPN, of necessity, has a life-cycle, and needs modification during deployment as the needs of its user change. Additionally as the network as a whole evolves there will need to be garbage collection performed to consolidate resources into usable quanta.

Systems in which the path is imposed such as SR, or some form of explicit routing tend to do well in these applications because it is possible to perform an atomic transition from one path to another. However implementations and the monitoring protocols need to make sure that the new path is up before traffic is transitioned to it.

There are however two manifestations of the latency problem that are for further study in any of these approaches:

- o The problem of packets overtaking one and other if a path latency reduces during a transition.
- o The problem of the latency transient in either direction as a path migrates.

There is also the matter of what happens during failure in the underlay infrastructure. Fast reroute is one approach, but that still produces a transient loss with a normal goal of rectifying this within 50ms. An alternative is some form of N+1 delivery such as has been used for many years to support protection from service disruption. This may be taken to a different level using the techniques proposed by the IETF deterministic network work with multiple in-network replication and the culling of later packets.

In addition to the approach used to protect high priority packets, consideration has to be given to the impact of best effort traffic on the high priority packets during a transient. Specifically if a conventional re-convergence process is used there will inevitably be micro-loops and whilst some form of explicit routing will protect the high priority traffic, lower priority traffic on best effort shortest paths will micro-loop without the use of a loop prevention technology. To provide the highest quality of service to high priority traffic, either this traffic must be shielded from the micro-loops, or micro-loops must be prevented.

## 8. Security Considerations

All types of virtual network require special consideration to be given to the isolation between the tenants. However in an enhanced virtual network service hard isolation needs to be considered. If a service requires a specific latency then it can be damaged by simply delaying the packet through the activities of another tenant. In a network with virtual functions, depriving a function used by another tenant of compute resources can be just as damaging as delaying transmission of a packet in the network.

## 9. IANA Considerations

There are no requested IANA actions.

## 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>>.

### 10.2. Informative References

- [FLEXE] "Flex Ethernet Implementation Agreement", March 2016, <<http://www.oiforum.com/wp-content/uploads/OIF-FLEXE-01.0.pdf>>.
- [I-D.ietf-detnet-architecture] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-04 (work in progress), October 2017.
- [I-D.ietf-detnet-dp-sol] Korhonen, J., Andersson, L., Jiang, Y., Finn, N., Varga, B., Farkas, J., Bernardos, C., Mizrahi, T., and L. Berger, "DetNet Data Plane Encapsulation", draft-ietf-detnet-dp-sol-01 (work in progress), January 2018.
- [I-D.ietf-spring-oam-usecase] Geib, R., Filsfils, C., Pignataro, C., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", draft-ietf-spring-oam-usecase-10 (work in progress), December 2017.
- [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-teas-rsvp-te-scaling-rec] Beeram, V., Minei, I., Shakir, R., Pacella, D., and T. Saad, "Techniques to Improve the Scalability of RSVP Traffic Engineering Deployments", draft-ietf-teas-rsvp-te-scaling-rec-09 (work in progress), February 2018.

- [I-D.sitaraman-mpls-rsvp-shared-labels]  
Sitaraman, H., Beeram, V., Parikh, T., and T. Saad,  
"Signaling RSVP-TE tunnels on a shared MPLS forwarding  
plane", draft-sitaraman-mpls-rsvp-shared-labels-03 (work  
in progress), December 2017.
- [NETCALC] "Applicability of Network Calculus to DetNet", November  
2017, <[https://datatracker.ietf.org/meeting/100/materials/  
slides-100-detnet-applicability-of-network-calculus-to-  
detnet](https://datatracker.ietf.org/meeting/100/materials/slides-100-detnet-applicability-of-network-calculus-to-detnet)>.
- [RFC3931] Lau, J., Ed., Townsley, M., Ed., and I. Goyret, Ed.,  
"Layer Two Tunneling Protocol - Version 3 (L2TPv3)",  
RFC 3931, DOI 10.17487/RFC3931, March 2005,  
<<https://www.rfc-editor.org/info/rfc3931>>.
- [RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation  
Edge-to-Edge (PWE3) Architecture", RFC 3985,  
DOI 10.17487/RFC3985, March 2005,  
<<https://www.rfc-editor.org/info/rfc3985>>.
- [RFC4448] Martini, L., Ed., Rosen, E., El-Aawar, N., and G. Heron,  
"Encapsulation Methods for Transport of Ethernet over MPLS  
Networks", RFC 4448, DOI 10.17487/RFC4448, April 2006,  
<<https://www.rfc-editor.org/info/rfc4448>>.
- [RFC4719] Aggarwal, R., Ed., Townsley, M., Ed., and M. Dos Santos,  
Ed., "Transport of Ethernet Frames over Layer 2 Tunneling  
Protocol Version 3 (L2TPv3)", RFC 4719,  
DOI 10.17487/RFC4719, November 2006,  
<<https://www.rfc-editor.org/info/rfc4719>>.

## Authors' Addresses

Stewart Bryant  
Huawei

Email: [stewart.bryant@gmail.com](mailto:stewart.bryant@gmail.com)

Jie Dong  
Huawei

Email: [jie.dong@huawei.com](mailto:jie.dong@huawei.com)



Zhenqiang Li  
China Mobile

Email: lizhenqiang@chinamobile.com

Takuya Miyasaka  
KDDI Corporation

Email: ta-miyasaka@kddi.com

INTERNET-DRAFT  
Intended status: Informational

R. Gu  
S. Hu  
China Mobile  
M. Wang  
D. Eastlake  
Huawei  
F. Hu  
ZTE  
December 12, 2018

Expires: June 11, 2019

Control Plane and User Plane Separated BNG Deployment Model  
draft-cuspdrt-rtgwg-cu-separation-bng-deployment-02

Abstract

This document describes the deployment model for a Broadband Network Gateway (BNG) device with Control Plane (CP) and User Plane(UP) separation. It is intended to give guidance for the deployment of CP and UP separated BNG devices in an operators' network.

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 BESS working group mailing list: [bess@ietf.org](mailto:bess@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>.

Gu, et al.  
\*

[Page 1]

INTERNET-DRAFT

Separated BNG Deployment Model

Table of Contents

1. Introduction and Overview.....	3
2. Concept and Terminology.....	5

2.1 Terminology.....	5
3. BNG with CP and UP Separation Deployment Model.....	6
3.1 CP and UP of BNG Deployment Within One District.....	6
3.2. CP and UP of BNG Deployment in Multiple Districts.....	7
4. The Process of BNG with CUPS in Home Service.....	10
5. High Availability Considerations.....	11
6. Security Considerations.....	12
7. IANA Considerations.....	12
Normative References.....	13
Informative References.....	13
Authors' Addresses.....	14

<

## 1. Introduction and Overview

A Broadband Network Gateway (BNG) is an Ethernet-centric IP edge router and acts as the aggregation point for the user traffic with some additional functions such as address management and cooperating with AAA (Radius/Diameter) systems and subscriber management. Because of the rapid development of new services, such as 4K, IoT, etc. and the increasing numbers of distributed home broadband service users, high resource utilization, high-efficiency management, and fast service provisioning are required. This calls for a new BNG architecture with CP and UP separation, which is also called Cloud BNG, as proposed in [BBF-CloudCO] [TR-384].

The CP and UP separation architecture of the BNG is composed of a Control Plane and a User Plane, with the concentrated CP responsible for control and management of the UP's resources and subscribers' information, and with the distributed UP taking charge of policy implementation and traffic forwarding. The obvious advantages of this new architecture are listed below.

**Resource Utilization Improvement:** A centralized Control Plane provides unified management capability for network resources and users information. The CP has an overview of all the resources and can distribute resources as specific users require, thus resources can be totally controlled and balanced.

**Management with High Efficiency:** A centralized CP provides a unified management interface to the outside systems such as EMS, DHCP Server, AAA Server, etc. In this situation, management can be easier for the centralized CP as it's the only device interfacing with the outside systems.

**Dynamic and Flexible:** The CP can be virtualized as a VNF with MANO management in an NFVI, while the UP can be a virtual machine or physical device as needed. A software-oriented CP can be designed with flexibility. The CP can handle all the situations dynamically over a wide range from few users accessing to large numbers of users accessing.

**Fast TTM:** The CP and UP can be deployed separately with the CP deployed centrally and the UP deployed in distribution closer to users. Thus, according to different situations such as session overload or extremely high throughput, the CP and UP can be extended separately. This can help shorten the time to market (TTM).

As noted, the new BNG architecture has CP and UP separation. The CP and UP are deployed with separation due to practical requirements. This document gives the CU separation BNG deployment model for actual

deployments.

## 2. Concept and 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.

### 2.1 Terminology

**BNG:** Broadband Network Gateway. A broadband remote access server (BRAS, 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).

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

**CUPS:** Control/User Plane Separation

**UP:** User Plane. The 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.

**TTM:** Time to Market. It is the length of time it takes from a product or a service being conceived until it is available for sale.

**MANO:** Management and Orchestration. Functions are collectively provided by NFVO, VNFM and VIM.

**VNF:** Virtual Network Function. Implementation of a Network Function that can be deployed on a Network Function Virtualization Infrastructure (NFVI).

**PNF:** Physical Network Function

**DHCP:** Dynamic Host Configuration Protocol

**PPPoE:** Point-to-Point Protocol over Ethernet

**IPoE:** Internet Protocol over Ethernet

### 3. BNG with CP and UP Separation Deployment Model

#### 3.1 CP and UP of BNG Deployment Within One District

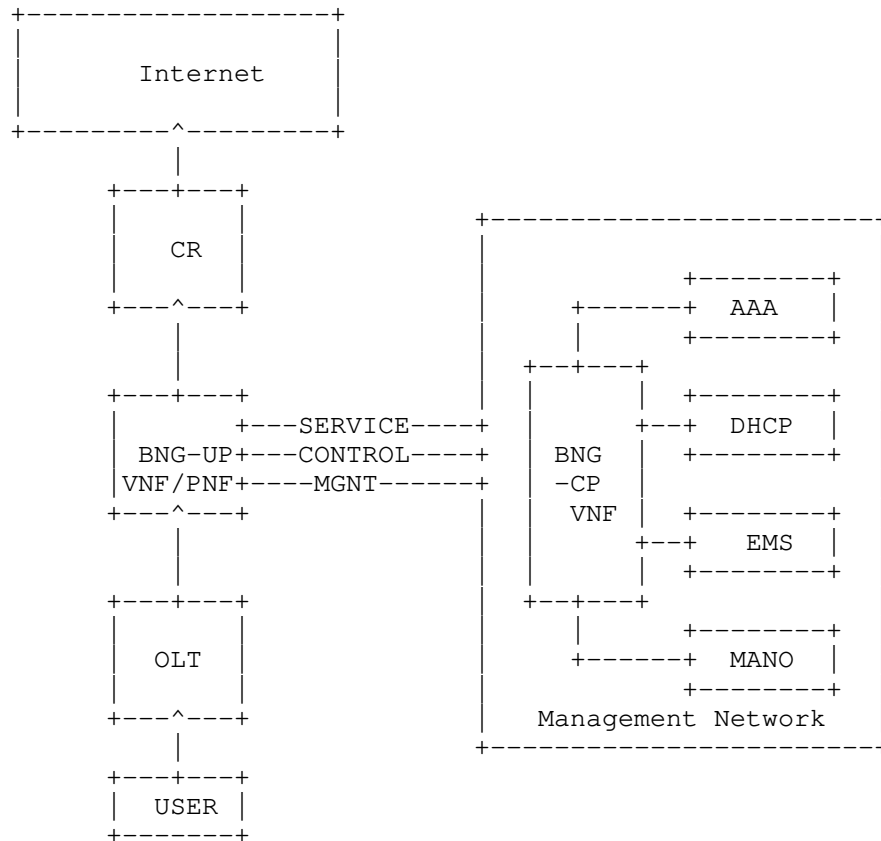


Figure 1: Cloud BNG Deployed in One District

Take a one district example as in Figure 1. Here BNG-CP and BNG-UP are separated as deployed. Since the CP is computationally intensive, a virtualized CP acting as a VNF can meet the requirements of flexibility and fast calculation. The UP is traffic intensive, which can be virtualized or stay physical depending on traffic. The virtualized UP with low expense and high flexibility can be suitable for light traffic. In high traffic, special hardware is needed with high traffic forwarding performance.

In order to fulfill the function of a BNG, the BNG-CP needs to communicate with outside systems such as a AAA (Radius/Diameter) server and many others in the management network. In addition, the

BNG-CP has three interfaces with the BNG-UP separated by their traffic categories: Service Interface, Control Interface, and Management Interface.

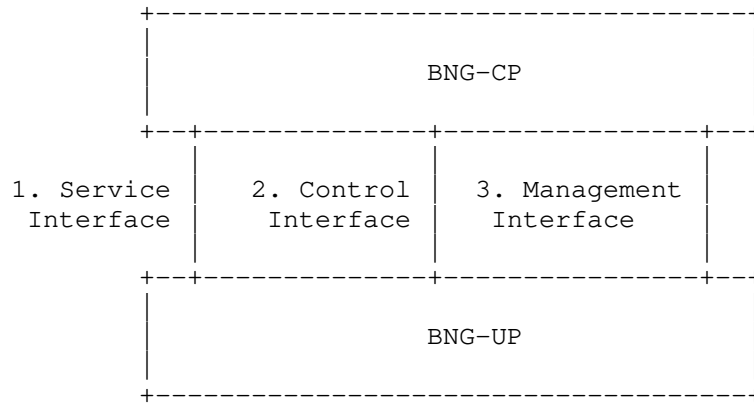


Figure 2. Internal Interfaces Between the BNG CP and UP

The functions of the three interfaces are as follows:

**Service Interface:** The CP and UP use this interface to establish VXLAN tunnels with each other and transmit PPPoE and IPoE packets over the VXLAN tunnels for authentication.

**Control Interface:** The CP uses this interface to deliver service entries to the UP, and the UP uses this interface to report service events to the CP.

**Management Interface:** The CP uses this interface to deliver basic configurations to the UP. This interface uses NETCONF.

Several related drafts exist describing these interfaces in detail. The VXLAN-GPE extension draft for C/U separated BNG is related to the Service Interface [huang-nov3-vxlan-gpe-extension-for-vbng]. The draft YANG data model for CU separated BNG focuses on Management Interface, seeing in [cuspdrtgwg-cu-separation-yang-model]. Another two drafts [cuspdrtgwg-cusp-requirements] and [cuspdrtgwg-cu-separation-infor-model] are related to the control interface giving an information model abstraction and suitable protocol.

### 3.2. CP and UP of BNG Deployment in Multiple Districts



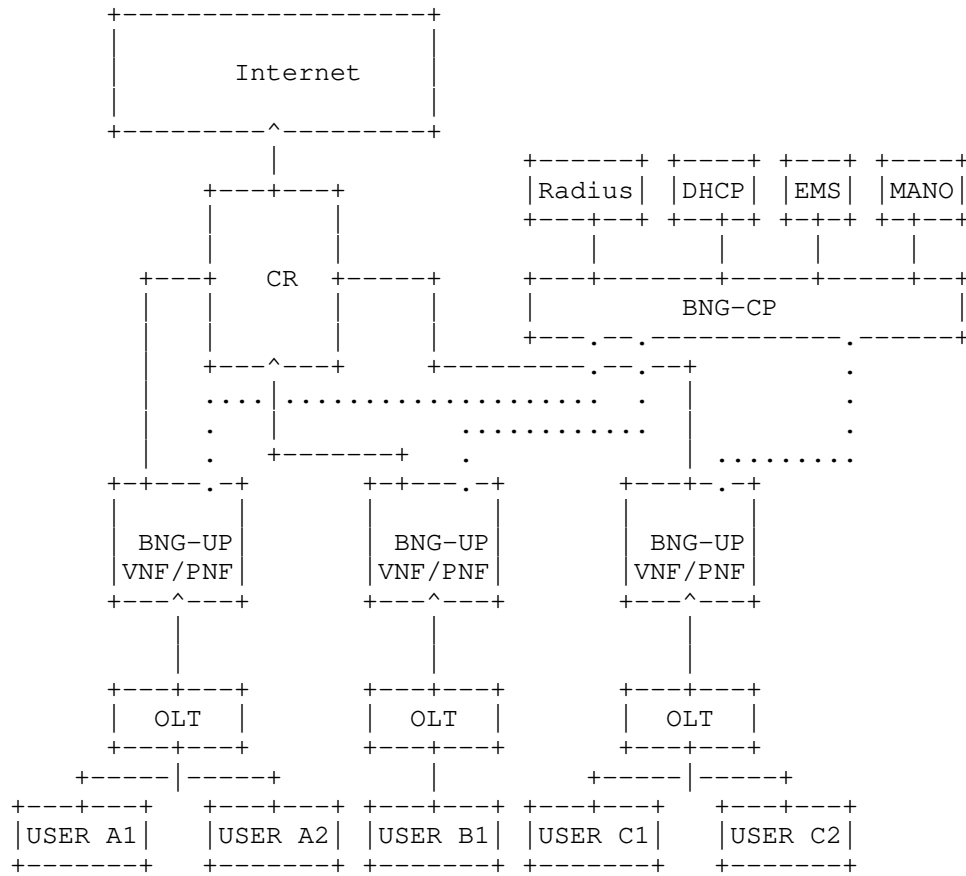


Figure 3: Cloud BNG Deployed in Several Districts

If subscribers are distributed in several districts, the CP can be deployed centrally with the UP deployed in different districts close to subscribers as shown in Figure 3. Thus the deployment model can be a bit complex.

Take three districts A, B. and C for example. Here three UPs are placed with one shared CP. The CP is usually deployed in a Core Data Center such as in a provincial datacenter with UPs in edge Data Centers such as city datacenters. In this Data Centers design, we have core data centers and edge data centers according to their location and responsibility. Core data centers are often planned in provinces for control and management, while edge data centers are in cities or towns for easy service access.

In this scenario, a centralized CP interfaces to the subsystems outside and communicate with all these UPs for control and management.

Under the CP's control, the corresponding traffic is forwarded by UP to the Internet.

#### 4. The Process of BNG with CUPS in Home Service

Take a user Bob accessing to the Internet using Home Broadband Service as an example. The process includes the service traffic from user to the internet and signaling traffic between BNG-UP and BNG-CP. Below is the whole process.

- (1) User Bob dials up with packets of PPPoE or IPoE from BNG-UP which will be sent to the BNG-CP with the user's information. This is signaling traffic.
- (2) The BNG-CP processes the dialup packets. Confirming with the outside neighboring systems in the management network, the BNG-CP makes the decision to permit or deny of the dial access through certification. In this step, the BNG-CP manages resources and generates tables with information such as User Info, IP Info, QoS Info, etc. This is signaling traffic.
- (3) The BNG-CP sends tables to the corresponding UP or to one UP it chooses from the corresponding UPs. This is signaling traffic.
- (4) The BNG-UP receives the tables, matches rules and performs corresponding actions.
- (5) If Bob is certificated and permitted, the UP forwards their traffic into the Internet with related policies such as limited bandwidth, etc. Otherwise, Bob is denied to access the Internet. This is service traffic.

From Step 2 to Step 4, the information model defined in [cuspd-t-rtgw-g-cu-separation-infor-model] can be used.

## 5. High Availability Considerations

As the BNG-CP takes responsibility for control and management, such as communicating with outside systems, generating flow tables, and managing the UP's resources, high availability of this key component should be considered. Some redundancy should be adopted for reliability, such as N+N or N+K active standby BNG-CPs. N+N standby means 1:1 backup for each BNG-CP, which enables easy rapid switch of any number of BNG-CP to their backup but is expensive because it requires a large number of backup CPs. N+K means a smaller number of backup CPs, for example N2:1 backup where  $N2 < N$  which is less expensive but does not handle more than 1 failure in the N2 subset of N BNG-CPs.

## 6. Security Considerations

TBD.

## 7. 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>> .in -10

## Informative References

- [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.
- [cuspdrt-rtgwg-cu-separation-yang-model] Hu, F., "YANG Data Model for Configuration Interface of Control-Plane and User-Plane separation BNG", draft-cuspdrt-rtgwg-cu-separation-yang-model, work in progress, 2018.
- [cuspdrt-rtgwg-cusp-requirements] Hu, S., "Requirements for Control Plane and User Plane Separated BNG Protocol", draft-cuspdrt-rtgwg-cusp-requirements, work in progress, 2018.
- [cuspdrt-rtgwg-cu-separation-infor-model] Wang, Z., "Information Model of Control-Plane and User- Plane separation BNG", draft-cuspdrt-rtgwg-cu-separation-infor-model, work in progress, 2018.
- [TR-384] BroadBand Forum, "Cloud Central Office Reference Architectural Framework", January 2018.

Authors' Addresses

Rong Gu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: gurong\_cmcc@outlook.com

Sujun Hu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: shujun\_hu@outlook.com

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

Email: wangzitao@huawei.com

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

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

Fangwei Hu  
ZTE Corporation  
No.889 Bibo Rd  
Shanghai 201203  
China

Phone: +86 21 68896273  
Email: hu.fangwei@zte.com.cn

Copyright, Disclaimer, and Additional IPR Provisions

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.





rtgwg  
Internet-Draft  
Intended status: Informational  
Expires: April 25, 2019

S. Hu  
China Mobile  
Donald. Eastlake  
M. Wang, Ed.  
Huawei  
V. Lopez  
Telefonica  
F. Qin  
Z. Li  
China Mobile  
T. Chua  
Singapore Telecommunications Limited  
October 22, 2018

Information Model of Control-Plane and User-Plane Separation BNG  
draft-cuspd-rtgwg-cu-separation-infor-model-03

Abstract

To improve network resource utilization and reduce operational expense, the Control-Plane and User-Plane separation concept is defined in Broadband Forum TR-384. This document describes the information model for the interface between the Control-Plane (CP) and the User-Plane (UP) in the CP/UP separation BNG. This information model may involve both the control channel interface and the configuration channel interface. The interface for the control channel allows the Control-Plane to send flow tables to the User-Plane, such as user's information table, user's interface table, and user's QoS table. And it also allows the User-Plane to report resource and statistics information to the Control-Plane. The interface for the configuration channel is in charge of the protocol version negotiation between the Control-Plane and User-Plane, the configuration for devices of the Control-Plane and User-Plane, and the reports of User-Plane's capabilities, etc. The information model defined in this document supports defining a standardized data model. Such a data model can be used to specify an interface to the CU separation BNG.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <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 April 25, 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 . . . . .	3
2. Concept and Terminology . . . . .	3
2.1. Terminology . . . . .	3
3. Control Plane and User Plane Separation BNG Information Model Overview . . . . .	4
3.1. Service Data Model Usage . . . . .	6
4. Information Model . . . . .	8
4.1. Information Model for Control-Plane . . . . .	9
4.1.1. User-Related Information . . . . .	11
4.1.1.1. User Basic Information Model . . . . .	11
4.1.1.2. IPv4 Information Model . . . . .	12
4.1.1.3. IPv6 Information Model . . . . .	13
4.1.1.4. QoS Information Model . . . . .	14
4.1.2. Interface Related Information . . . . .	15
4.1.2.1. Interface Information Model . . . . .	15
4.1.3. Device Related Information . . . . .	16
4.1.3.1. Addressfield distribute Table . . . . .	17
4.2. Information Model for User Plane . . . . .	17
4.2.1. Port Resources of UP . . . . .	18
4.2.2. Traffic Statistics Infor . . . . .	19
5. Security Considerations . . . . .	20
6. IANA Considerations . . . . .	20
7. References . . . . .	20
7.1. Normative References . . . . .	20

7.2. Informative References . . . . .	21
Authors' Addresses . . . . .	21

## 1. Introduction

To improve network resource utilization and reduce operational expense, the Control-Plane and User-Plane separation concept is defined in Broadband Forum [TR-384]. The motivation for and architecture of the Control-Plane and User-Plane separation BNG is discussed in [I.D.cuspdrt-gwgc-cu-separation-bng-architecture].

This document describes an information model for the interface between the Control-Plane (CP) and the User-Plane (UP) separation in the CP / UP Separated BNG. This information model may involve both the control channel interface and the configuration channel interface. The interface for control channel allows the Control-Plane to send several flow tables to the User-Plane, such as user's information table, user's interface table, and user's QoS table, etc. And it also allows the User-Plane to report the resources and statistics information to the Control-Plane. The interface for configuration channel is in charge of the protocol version negotiation of protocols between the Control-Plane and User-Plane, the configuration for the devices of Control-Plane and User-Plane, and the report of User-Plane's capabilities, etc. The information model defined in this document enables supports defining a standardized data model. Such a data model can be used to define an interface to the CU separation BNG.

## 2. Concept and Terminology

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

### 2.1. Terminology

**BNG:** Broadband Network Gateway. A broadband remote access server (BRAS, 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).

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

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. Control Plane and User Plane Separation BNG Information Model Overview

Briefly, a CU separation BNG is made up of a centralized CP and a set of UPs. The CP is a user control management component that manages UP's resources such as the user entry and forwarding policy, for example, the access bandwidth and priority management. The UP is a network edge and user policy implementation component. It can support the forwarding plane functions on traditional BNG devices, such as traffic forwarding, QoS, and traffic statistics collection, and it can also support the control plane functions on traditional BNG devices, such as routing, multicast, etc.

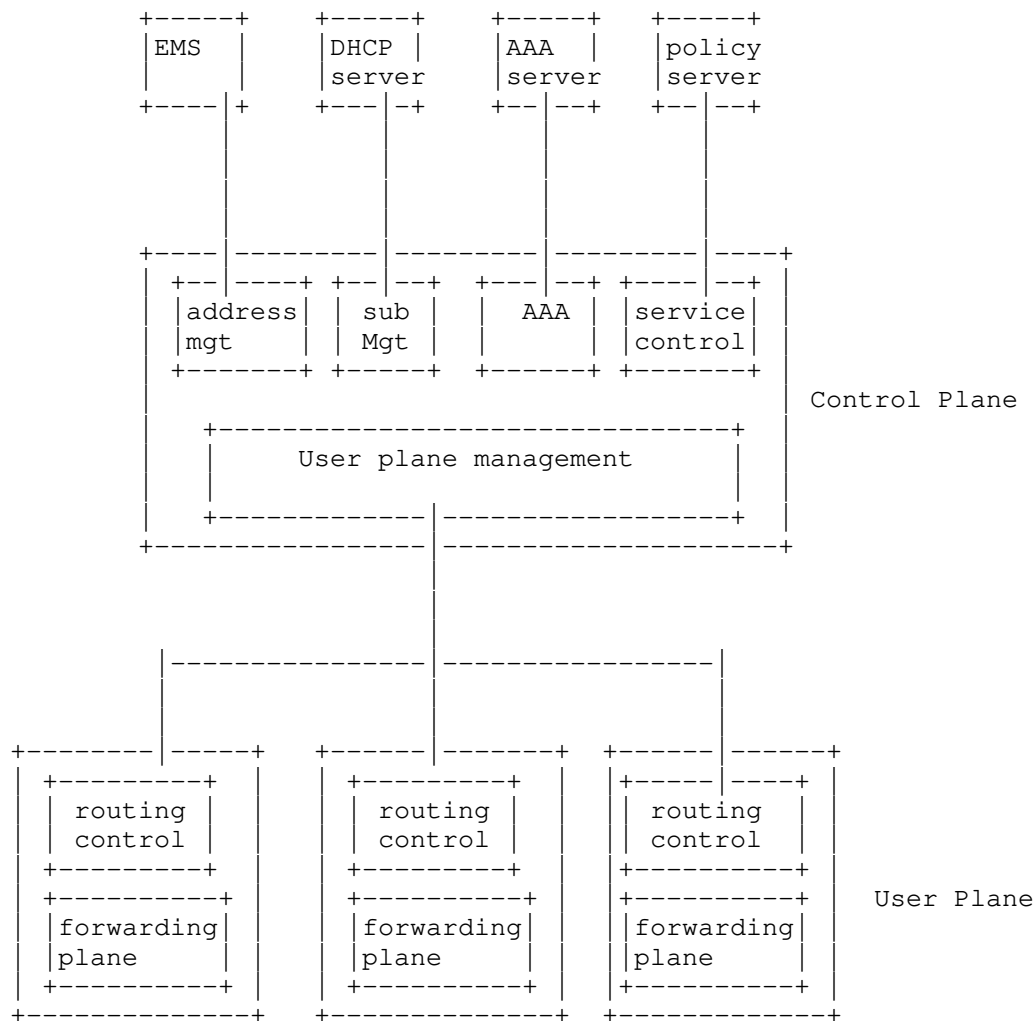


Figure 1. CU Separated BNG

The CU separated BNG is shown 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, maintenance of channels between Control Planes and User Planes. Other modules in the BNG control plane, such as address management, AAA, and etc., are responsible for the connection with outside subsystems in order to provide the service. The routing control and forwarding Plane in the BNG User Plane (local) could be distributed across the infrastructure.

### 3.1. Service Data Model Usage

The idea of this information model is to propose a set of generic and abstract information models to be used in both Control Plane and User Planes. A typical scenario would be that this model can be used as a compendium to realize the communication between the Control Plane and User Planes of the CU separation BNG.

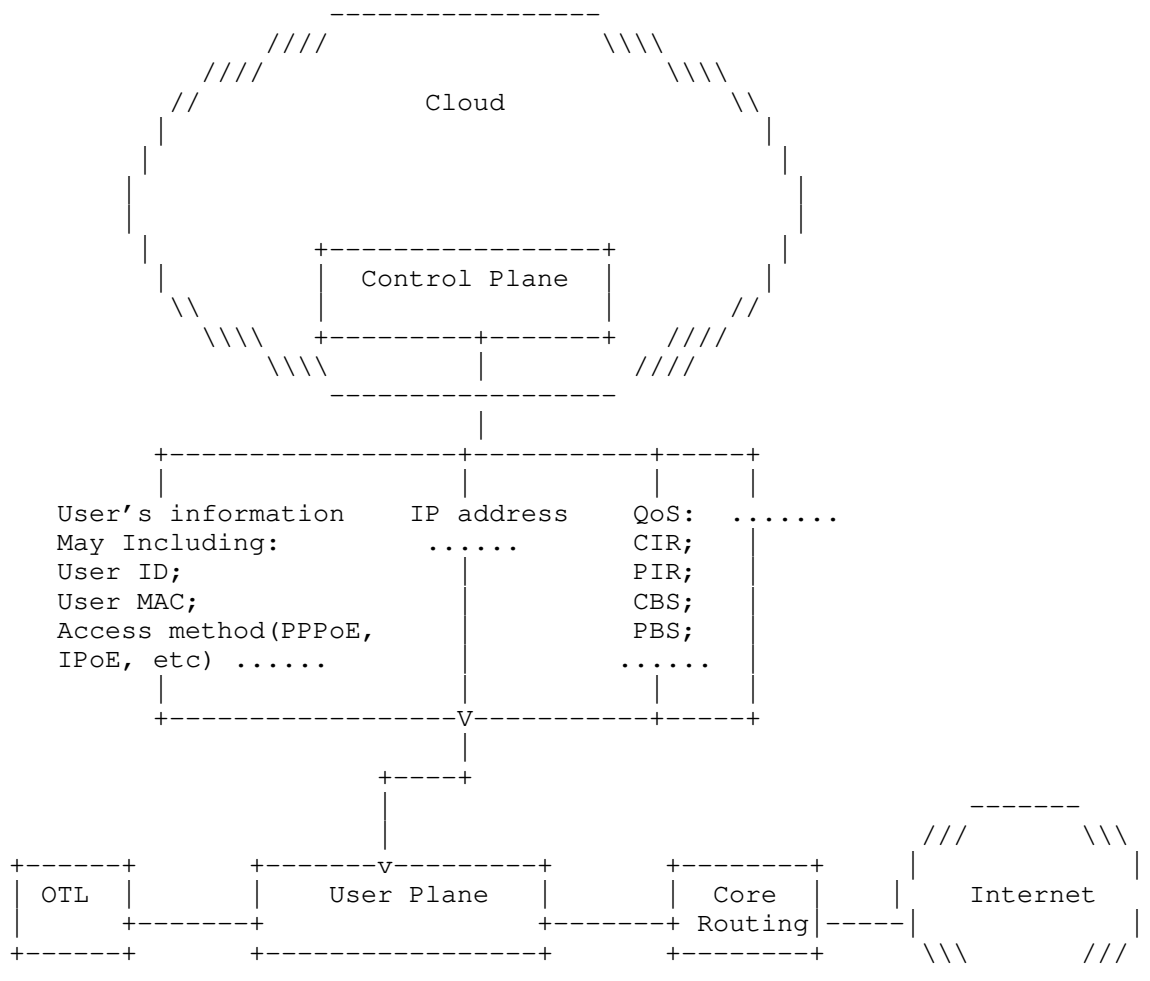


Figure 2. CU Separation BNG

As shown in Figure 2, when users access the BNG network, the control plane solicits these users' information (such as user's ID, user's MAC, user's access methods, for example via PPPoE/IPoE), associates them with available bandwidth which is reported by User planes, and based on the service's requirements generates a set of tables, which may include user's information, user's IP address, and QoS. Then the control plane can transmit these tables to the User planes. User planes receive these tables, parse them, matches these rules, and then performs corresponding actions.



#### 4. Information Model

This section specifies the information model in Routing Backus-Naur Form [RFC5511]. This grammar intends to help readers better understand the English text description in order to derive a data model. However it may not provide all the details provided by the English text. When there is a lack of clarity, the English text will take precedence.

This section describes the information model that represents the interface of the CU separation BNG that is language and protocol neutral.

The following Routing BNF grammar describes the Overview of Information Model for CU separation BNG.

```

<cu-separation-bng-infor-model>::=<control-plane-information-model>
                                <user-plane-information-model>

<control-plane-information-model>::=<user-related-infor-model>
                                <interface-related-infor-model>
                                <device-related-infor-model>

<user-related-infor-model>::= <user-basic-information>
                                [<ipv4-informatiom>] | [<ipv6-information>]
                                [<qos-information>]

<user-basic-information> :: = <USER_ID> <MAC_ADDRESS>
                                [<ACCESS_TYPE>] [<SESSION_ID>]
                                [<INNER_VLAN-ID>] [<OUTER_VLAN_ID>]
                                <USER_INTERFACE>

<ipv4-informatiom>::=<USER_ID><USER_IPV4>
                                <MASK_LENGTH><GATEWAY>
                                <VRF>

<ipv6-information>::=<USER_ID>(<USER_IPV6>
                                <PREFIX_LEN>) | (<PD_ADDRESS><PD_PREFIX_LEN>)
                                <VRF>

<qos-information>::=<USER_ID>
                                (<CIR><PIR><CBS><PBS>)
                                [<QOS_PROFILE>]

<interface-related-infor-model>::=<interface-information>

<interface-information>::=<IFINDEX><BAS_ENABLE>
                                <service-type>

```

```

<service-type>::=<PPP_Only><IPV4_TRIG>
               <IPV6_TRIG><ND-TRIG>
               <ARP_PROXY>

<device-related-infor-model>::=<address-field-distribute>

<address-field-distribute>::=<ADDRESS_SEGMENT><ADDRESS_SEGMENT_MASK>
                             <ADDRESS_SEGMENT_VRF><NEXT_HOP>
                             <IF_INDEX><MASK_LENGTH>

<user-plane-information-model>::=<port-resources-infor-model>
                                <traffic-statistics>

<port-resource-information>::=<IF_INDEX><IF_NAME>
                             <IF_TYPE><LINK_TYPE>
                             <MAC_ADDRESS><IF_PHY_STATE>
                             <MTU>

<traffic-statistics-information>::=<USER_ID><STATISTICS_TYPE>
                                   <INGRESS_STATISTICS_PACKETS>
                                   <INGRESS_STATISTICS_BYTES>
                                   <EGRESS_STATISTICS_PACKETS>
                                   <EGRESS_STATISTICS_BYTES>

```

#### 4.1. Information Model for Control-Plane

This section describes information model for the Control-Plane (CP). As mentioned in Section 3, the Control Plane is a user control management component which manages the user's information, User-Plane's resources and forwarding policy, etc. The control plane can generate several tables which contain a set of rules based on the resources and specific requirements of user's service. After that, the control plane sends the tables to User Planes, and User planes receive the tables, parse them, match the rules, and then perform corresponding actions.

The Routing Backus-Naur Form grammar below specifies the Information model for Control-Plane:

```

<control-plane-information-model>::=<user-related-infor-model>
    <interface-related-infor-model>
    <device-related-infor-model>

<user-related-infor-model>::= <user-basic-information>
    [<ipv4-information>] | [<ipv6-information>]
    [<qos-information>]

<user-basic-information> ::= <USER_ID> <MAC_ADDRESS>
    [<ACCESS_TYPE>] [<SESSION_ID>]
    [<INNER_VLAN-ID>] [<OUTER_VLAN_ID>]
    <USER_INTERFACE>

<ipv4-information>::=<USER_ID><USER_IPV4>
    <MASK_LENGTH><GATEWAY>
    <VRF>

<ipv6-information>::=<USER_ID>(<USER_IPV6>
    <PREFIX_LEN>) | (<PD_ADDRESS><PD_PREFIX_LEN>)
    <VRF>

<qos-information>::=<USER_ID>
    (<CIR><PIR><CBS><PBS>)
    [<QOS_PROFILE>]

<interface-related-infor-model>::=<interface-information>

<interface-information>::=<IFINDEX><BAS_ENABLE>
    <service-type>

<service-type>::=<PPP_Only><IPV4_TRIG>
    <IPV6_TRIG><ND-TRIG>
    <ARP_PROXY>

<device-related-infor-model>::=<address-field-distribute>

<address-field-distribute>::=<ADDRESS_SEGMENT><ADDRESS_SEGMENT_MASK>
    <ADDRESS_SEGMENT_VRF><NEXT_HOP>
    <IF_INDEX><MASK_LENGTH>

```

user-related-infor-model: presents the attributes that can describe the user's profile, such as user's basic information, qos, and IP address.

interface-related-infor-model: presents the attributes that relate to some physical/virtual interface. This model can be used to indicate which kinds of service can be supported by interfaces.

device-related-infor-model: presents the attributes which relate to a specific device. For example the control plane can manage and distribute the users, which belong to same subnet, to some specific devices. And the user plane's devices provide corresponding service for these users.

#### 4.1.1.1. User-Related Information

The user related information is a collection of attributes bound to specific users. For example, the control plane can use a unified ID to distinguish different users and distribute the IP address and QoS rules to a specific user. In this section, the user related information models are presented. The user related information models include the user information model, IPv4/IPv6 information model, QoS information model, etc.

The Routing Backus-Naur Form grammar below specifies the user related information model:

```
<user-related-infor-model>::= <user-basic-information>
                               [<ipv4-information>] | [<ipv6-information>]
                               [<qos-information>]

<user-basic-information> ::= <USER_ID> <MAC_ADDRESS>
                             [<ACCESS_TYPE>] [<SESSION_ID>]
                             [<INNER_VLAN_ID>] [<OUTER_VLAN_ID>]
                             <USER_INTERFACE>

<ipv4-information>::=<USER_ID><USER_IPV4>
                    <MASK_LENGTH><GATEWAY>
                    <VRF>

<ipv6-information>::=<USER_ID>(<USER_IPV6>
                             <PREFIX_LEN>) | (<PD_ADDRESS><PD_PREFIX_LEN>)
                             <VRF>

<qos-information>::=<USER_ID>
                  (<CIR><PIR><CBS><PBS>)
                  [<QOS_PROFILE>]
```

##### 4.1.1.1.1. User Basic Information Model

The User Basic Information model contains a set of attributes to describe the basic information of a specific user, such as the user's MAC address, access type (via PPPoE, IPoE, etc), inner VLAN ID, outer VLAN ID, etc.

The Routing Backus-Naur Form grammar below specifies the user basic information model:

```
<user-basic-information> ::= <USER_ID> <MAC_ADDRESS>  
                             [<ACCESS_TYPE>] [<SESSION_ID>]  
                             [<INNER_VLAN-ID>] [<OUTER_VLAN_ID>]  
                             <USER_INTERFACE>
```

USER\_ID (4 bytes): is the identifier for a user. This parameter is unique and mandatory. It can be used to distinguish different users.

MAC\_ADDRESS (6 bytes): is the MAC address of the user.

ACCESS\_TYPE (2 bytes): This attribute is an optional parameter. It can be used to indicate the protocol being used for the user's access, such as PPPoE, IPoE, etc.

SESSION\_ID (4 bytes): This attribute is an optional parameter. It can be used as the identifier of PPPoE session.

INNER\_VLAN-ID (2 bytes): The 12-bit identifier of user's inner VLAN in network byte order. The unused high-order 4 bits MUST be sent as zero and ignored on receipt.

OUTER\_VLAN\_ID (2 bytes): The 12-bit identifier of user's outer VLAN in network byte order. The unused high-order 4 bits MUST be sent as zero and ignored on receipt.

USER\_INTERFACE (4 bytes): This attribute specifies the binding interface of a specific user. The IfIndex of the interface MAY be included. This is the 32-bit IfIndex assigned to the interface by the device as specified by the Interfaces Group MIB [RFC2863]. The IfIndex can be utilized within a management domain to map to an actual interface, but it is also valuable in public applications [RFC5837]. The IfIndex can be used as an opaque token to discern which interface of the User-Plane is providing corresponding service for specific user.

#### 4.1.1.2. IPv4 Information Model

The IPv4 information model presents the user's IPv4 parameters. It is an optional constructs. The Routing Backus-Naur Form grammar below specifies the user's IPv4 information model:

```
<ipv4-information> ::= <USER_ID> <USER_IPV4>  
                       <MASK_LENGTH> <GATEWAY>  
                       <VRF>
```

USER\_ID (4 bytes): is the identifier of user. This parameter is unique and mandatory. This attribute is used to distinguish different users. In conjunction with other IPv4 parameters it links the user to the user's IPv4 information.

USER\_IPV4 (4 bytes): This attribute specifies the user's IPv4 address, It is usually used in user plane discovery and ARP reply message.

MASK\_LENGTH (4 bytes): This attribute specifies the user's subnet mask length which can identify a range of IP addresses that are on the same network.

GATEWAY (4 bytes): This attribute specifies the user's gateway, and is usually used in User Plane discovery and ARP reply message.

VRF (4 bytes): is the identifier of VRF instance.

#### 4.1.1.3. IPv6 Information Model

The IPv6 information model presents the user's IPv6 parameters. It is an optional constructs. The Routing Backus-Naur Form grammar below specifies the user's IPv6 information model:

```
<ipv6-information>::=<USER_ID>(<USER_IPV6>  
                                <PREFIX_LEN> | (<PD_ADDRESS><PD_PREFIX_LEN>  
                                <VRF>
```

USER\_ID (4 bytes): is the identifier of user. This parameter is unique and mandatory. This attribute is used to distinguish different users. in conjunction with other IPv6 parameters, I tlink the user to the user's IPv6 information.

USER\_IPV6 (2 bytes): This attribute specifies the user's IPv6 address. It is usually used in neighbor discovery (ND discovery).

PREFIX\_LEN (4 bytes): This attribute specifies the user's subnet prefix lengths which can identify a range of IP addresses that are on the same network.

PD\_ADDRESS (4 bytes): In IPv6 networking, DHCPv6 prefix delegation is used to assign a network address prefix and automate configuration and provisioning of the public routable addresses for the network. This attribute specifies the user's DHCPv6 prefix delegation address, and is usually used in neighbor discovery (ND discovery).

PD\_PREFIX\_LEN (4 bytes): This attribute specifies the user's DHCPv6 delegation prefix length, and it's usually used in neighbor discovery (ND discovery).

VRF (4 bytes): is the identifier of a VRF instance

#### 4.1.1.4. QoS Information Model

In the CU separation BNG, the Control-Plane (CP) generates the QoS table base based on the UP's bandwidth resources and the specific QoS requirements of the user's services. This table contains a set of QoS matching rules such as user's committed information rate, peak information rate, committed burst size, etc. And it is an optional construct. The Routing Backus-Naur Form grammar below illustrates the user's qos information model:

```
<qos-information>::=<USER_ID>
                    (<CIR><PIR><CBS><PBS>)
                    [<QOS_PROFILE>]
```

USER\_ID (4 bytes): is the identifier of user. This parameter is unique and mandatory. This attribute is used to distinguish different users. within conjunction with other qos parameters it links the user to the user's qos information.

CIR (4 bytes): In a BNG network, the Committed Information Rate (CIR) is the bandwidth available for a user guaranteed by an internet service provider under normal conditions. This attribute is used to indicate the user's committed information rate, and it usually appears with other qos attributes (such as PIR, CBS, PBS, etc) to give the user's QoS profile.

PIR (4 bytes): Peak Information Rate (PIR) is a burstable rate set on routers and/or switches that allow throughput bursts. This attribute is used to indicate the user's peak information rate. In conjunction with other QoS attributes (such as CIR, CBS, PBS, etc) it is used to give the user's QoS profile.

CBS (4 bytes): The Committed Burst Size (CBS) specifies the relative amount of reserved buffers for a specific ingress network's forwarding class queue or egress network's forwarding class queue. This attribute is used to indicate the user's committed burst size. In conjunction with other qos attributes (such as CIR, PIR, PBS, etc) it is used to give the user's QoS profile.

PBS (4 bytes): The Peak Burst Size (PBS) specifies the maximum size of the first token bucket. This attribute is used to indicate the

user's peak burst size. In conjunction with other qos attributes (such as CIR, PIR, CBS, etc) it is used to give the user's QoS profile.

**QOS\_PROFILE (4 bytes):** This attribute specifies the standard profile provided by the operator. It can be used as a QoS template that is defined as a list of classes of services and associated properties. The properties may include:

- o **Rate-limit:** used to rate-limit the class of service. The value is expressed as a percentage of the global service bandwidth.
- o **latency:** used to define the latency constraint of the class. The latency constraint can be expressed as the lowest possible latency or a latency boundary expressed in milliseconds.
- o **jitter:** used to define the jitter constraint of the class. The jitter constraint can be expressed as the lowest possible jitter or a jitter boundary expressed in microseconds.
- o **bandwidth:** used to define a guaranteed amount of bandwidth for the class of service. It is expressed as a percentage.

#### 4.1.2. Interface Related Information

This model contains the necessary information for an interface. It is used to indicate which kind of service can be supported by this interface. The Routing Backus-Naur Form grammar below specifies the interface related information model:

```
<interface-related-infor-model>::=<interface-information>

<interface-information>::=<IFINDEX><BAS_ENABLE>
                        <service-type>

<service-type>::=<PPP_Only><IPV4_TRIG>
                <IPV6_TRIG><ND-TRIG>
                <ARP_PROXY>
```

##### 4.1.2.1. Interface Information Model

The interface model mentioned here is a logical construct that identifies a specific process or a type of network service. In CU separation BNG network, the Control-Plane (CP) generates the Interface-Infor table based on the available resources, which are received from the User-Plane (UP), and the specific requirements of user's services.



The Routing Backus-Naur Form grammar below specifies the interface information model:

```
<interface-information>::=<IFINDEX><BAS_ENABLE>  
                           <service-type>  
  
<service-type>::=<PPP_Only><IPV4_TRIG>  
                  <IPV6_TRIG><ND-TRIG>  
                  <ARP_PROXY>
```

IFINDEX (4 bytes): The IfIndex is the 32-bit index assigned to the interface by the device as specified by the Interfaces Group MIB [RFC2863]. The IfIndex can be utilized within a management domain to map to an actual interface, but it is also valuable in public applications. The IfIndex can be used as an opaque token to discern which interface of the User-Plane is providing the corresponding service for specific user.

BAS\_ENABLE (2 bytes): This is a flag, and if it is TRUE, the BRAS is enabled on this interface.

PPP\_Only (2 bytes): This is a flag, and if it is TRUE, the interface only supports PPP user.

IPV4\_TRIG (2 bytes): This is a flag, and if it is TRUE, the interface supports the user being triggered to connect to the internet by using an IPv4 message.

IPV6\_TRIG (2 bytes): This is a flag, and if it is TRUE, the interface supports that the user being triggered to connect to the internet by using an IPv6 message.

ND-TRIG (2 bytes): This is a flag, and if it is TRUE, the interface supports the user being triggered to connect to the internet by using a neighbor discovery message.

ARP\_PROXY (2 bytes): This is a flag, and if it is TRUE, ARP PROXY is enabled on this interface.

#### 4.1.3. Device Related Information

The device related information model presents the attributes which relate to a specific device. For example the control plane can manage and distribute the users, who belong to the same subnet, to some specific devices. And then the user plane's devices can provide the corresponding service for these users. The Routing Backus-Naur Form grammar below specifies the device related information model:

```
<device-related-infor-model>::=<address-field-distribute>

<address-field-distribute>::=<ADDRESS_SEGMENT><ADDRESS_SEGMENT_MASK>
                               <ADDRESS_SEGMENT_VRF><NEXT_HOP>
                               <IF_INDEX><MASK_LENGTH>
```

#### 4.1.3.1. Address field distribute Table

In the CU separation BNG information model, the Control-Plane (CP) generates and sends this Address field distribute table to UP. Based on this table, the user-plane's devices can be divided into several blocks, and each block is in charge of working for users with the same subnet. The Routing Backus-Naur Form grammar below illustrates the address field sepcifies information model:

```
<address-field-distribute>::=<ADDRESS_SEGMENT><ADDRESS_SEGMENT_MASK>
                               <ADDRESS_SEGMENT_VRF><NEXT_HOP>
                               <IF_INDEX><MASK_LENGTH>
```

#### 4.2. Information Model for User Plane

This section describes the information model for the interface of to the User Plane (UP). As mentioned in section Section 3, the UP is a network edge and user policy implementation component. It supports the following: Forwarding plane functions on traditional BNG devices, including traffic forwarding, QoS, and traffic statistics collection and Control plane functions on traditional BNG devices, including routing, multicast, and MPLS.

In CU separation BNG information model, the CP generates tables and provides the entries. The UP plays two roles:

1. It receives these tables, parses them, then performs corresponding actions.
2. It also generates several tables to report the available resources (such as usable interfaces, etc.) and statistical information to CP.

The Routing Backus-Naur Form grammar below specifies the User Plane information model:

```

<user-plane-information-model>::=<port-resources-infor-model>
                                <traffic-statistics>

port-resource-information>::=<IF_INDEX><IF_NAME>
                             <IF_TYPE><LINK_TYPE>
                             <MAC_ADDRESS><IF_PHY_STATE>
                             <MTU>

<traffic-statistics-information>::=<USER_ID><STATISTICS_TYPE>
                                     <INGRESS_STATISTICS_PACKETS>
                                     <INGRESS_STATISTICS_BYTES>
                                     <EGRESS_STATISTICS_PACKETS>
                                     <EGRESS_STATISTICS_BYTES>

```

#### 4.2.1. Port Resources of UP

The User Plane can generate the network resource table, which contains a bunch of attributes to present the available network resources, for example the usable interfaces.

The Figure Routing BNF grammar below illustrates specifies the Port Resources Information Table of User-Plane:

```

<port-resource-information>::<IF_INDEX><IF_NAME>
                             <IF_TYPE><LINK_TYPE>
                             <MAC_ADDRESS><IF_PHY_STATE>
                             <MTU>

```

**IFINDEX (4 bytes):** IfIndex is the 32-bit index assigned to the interface by the device as specified by the Interfaces Group MIB [RFC2863]. The IfIndex can be utilized within a management domain to map to an actual interface, but it is also valuable in public applications. The IfIndex can be used as an opaque token to discern which interface of the User-Plane is available.

**IF\_NAME (64 bytes):** the textual name of the interface. The value of this object should be the name of the interface as assigned by the local device and should be suitable for use in commands entered at the device's "console". This might be a text name, such as "le0" or a simple port number, such as "1", depending on the interface naming syntax of the device. If several entries in the ifTable together represent a single interface as named by the device, then each will have the same value of ifName.

**IF\_TYPE (4 bytes):** the type of interface, such as Ethernet, GE, Eth-Trunk, etc.

LINK\_TYPE (4 bytes): This attribute specifies the type of link, such as point-to-point, broadcast, multipoint, point-to-multipoint, private and public (accessibility and ownership), etc.

MAC\_ADDRESS (6 bytes): This attribute specifies the available interface's MAC address.

IF\_PHY\_STATE (1 byte): The current operational state of the interface. This is an enumeration type node:

- 1- Up: ready to pass packets;
- 2- Down
- 3- Testing: in some test mode;
- 4- Unknow: status cannot be determined for some reason;
- 5- Dormant;
- 6- Not present: some component is missing.

MTU: This attribute specifies the available interface's MTU (Maximum Transmission Unit).

#### 4.2.2. Traffic Statistics Infor

The user-plane also generates the traffic statistics table to report the current traffic statistics.

The Figure below specifies the Traffic Statistics Infor model of User-Plane:

```
<traffic-statistics-information>::=<USER_ID><STATISTICS_TYPE>  
                                <INGRESS_STATISTICS_PACKETS>  
                                <INGRESS_STATISTICS_BYTES>  
                                <EGRESS_STATISTICS_PACKETS>  
                                <EGRESS_STATISTICS_BYTES>
```

USER\_ID (4 bytes): is the identifier of user. This parameter is unique and mandatory. This attribute is used to distinguish different users. In conjunction with other statistics parameters such as ingress packets, egress packets, etc, it is used to report the user's status profile.

STATISTICS\_TYPE (4 bytes): This attribute specifies the traffic type such as IPv4, IPv6, etc.

INGRESS\_STATISTICS\_PACKETS (8 bytes): This attribute specifies the Ingress Statistics Packets of specific user.

INGRESS\_STATISTICS\_BYTES (8 bytes): This attribute specifies the Ingress Statistics Bytes of specific user.

EGRESS\_STATISTICS\_PACKETS (8 bytes): This attribute specifies the Egress Statistics Packets of specific user.

EGRESS\_STATISTICS\_BYTES (8 bytes): This attribute specifies the Egress Statistics Bytes of specific user.

## 5. Security Considerations

None.

## 6. IANA Considerations

None.

## 7. References

### 7.1. Normative References

- [I-D.cuspdtdt-rtgwg-cu-separation-bng-architecture]  
Hu, S., Qin, F., Li, Z., Chua, T., Wang, Z., and J. Song,  
"Architecture for Control Plane and User Plane Separated  
BNG", draft-cuspdtdt-rtgwg-cu-separation-bng-architecture-01  
(work in progress), July 2018.
- [I-D.cuspdtdt-rtgwg-cu-separation-bng-deployment]  
Gu, R., Hu, S., and Z. Wang, "Deployment Model of Control  
Plane and User Plane Separation BNG", draft-cuspdtdt-rtgwg-  
cu-separation-bng-deployment-00 (work in progress),  
October 2017.
- [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>>.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group  
MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000,  
<<https://www.rfc-editor.org/info/rfc2863>>.

- [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>>.
- [RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <<https://www.rfc-editor.org/info/rfc5837>>.
- [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>>.

## 7.2. Informative References

- [TR-384] Broadband Forum, "Cloud Central Office Reference Architectural Framework", BBF TR-384, January. 2018.

### Authors' Addresses

Shujun Hu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: [hushujun@chinamobile.com](mailto:hushujun@chinamobile.com)

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

Email: [d3e3e3@gmail.com](mailto:d3e3e3@gmail.com)

Michael Wang (editor)  
Huawei  
101 Software Avenue, Yuhua District  
Nanjing, Jiangsu 210012  
China

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

Victor Lopez  
Telefonica  
Sur 3 building, 3rd floor, Ronda de la Comunicacion s/n  
Madrid 28050  
Spain

Email: victor.lopezalvarez@telefonica.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

rtgwg  
Internet-Draft  
Intended status: Informational  
Expires: April 25, 2019

S. Hu  
China Mobile  
V. Lopez  
Telefonica  
F. Qin  
Z. Li  
China Mobile  
T. Chua  
Singapore Telecommunications Limited  
Donald. Eastlake  
M. Wang  
J. Song  
Huawei  
October 22, 2018

Requirements for Control Plane and User Plane Separated BNG Protocol  
draft-cuspd-rtgwg-cusp-requirements-03

Abstract

This document introduces the Control Plane and User Plane separated BNG (Broadband Network Gateway) architecture and defines a set of associated terminology. It also specifies a set of protocol requirements for communication between the BNG-CP and the BNG-UPs in the Control Plane and User Plane Separated BNG.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <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 April 25, 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. Concept and Terminology . . . . .	3
2.1. Terminology . . . . .	3
3. CU Separated BNG Model . . . . .	3
3.1. Internal interfaces between the CP and UP . . . . .	5
4. The usage of CU separation BNG protocol . . . . .	6
5. Control Plane and User Plane Separation Protocol Requirements	7
5.1. Transmit information tables . . . . .	7
5.2. Message Priority . . . . .	7
5.3. Reliability . . . . .	7
5.4. Support for Secure Communication . . . . .	8
5.5. Version negotiation . . . . .	8
5.6. Capability Exchange . . . . .	9
5.7. CP primary/backup capability . . . . .	9
5.8. Event Notification . . . . .	9
5.9. Query Statistics . . . . .	10
6. Security Considerations . . . . .	10
7. IANA Considerations . . . . .	10
8. References . . . . .	10
8.1. Normative References . . . . .	10
8.2. Informative References . . . . .	11
Authors' Addresses . . . . .	11

## 1. Introduction

A 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 CU separated BNG is introduced [TR-384]. The CU separated Service Control Plane could be virtualized and centralized;

it is responsible for user access authentication and sending forwarding entries to user planes. The routing control and forwarding plane, i.e. BNG user plane (local), could be distributed across the infrastructure.

This document introduces the Control Plane and User Plane separated BNG architecture and modeling. This document also defines the protocol requirements for Control Plane and User Plane Separated BNG (CUSP).

## 2. Concept and 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.

### 2.1. Terminology

**BNG:** Broadband Network Gateway. A broadband remote access server (BRAS, 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 UP's resources such as the user entry and user's QoS policy.

**CUSP:** Control Plane and User Plane Separated BNG Protocol.

**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 Model

Figure 1 shows the architecture of CU separated BNG

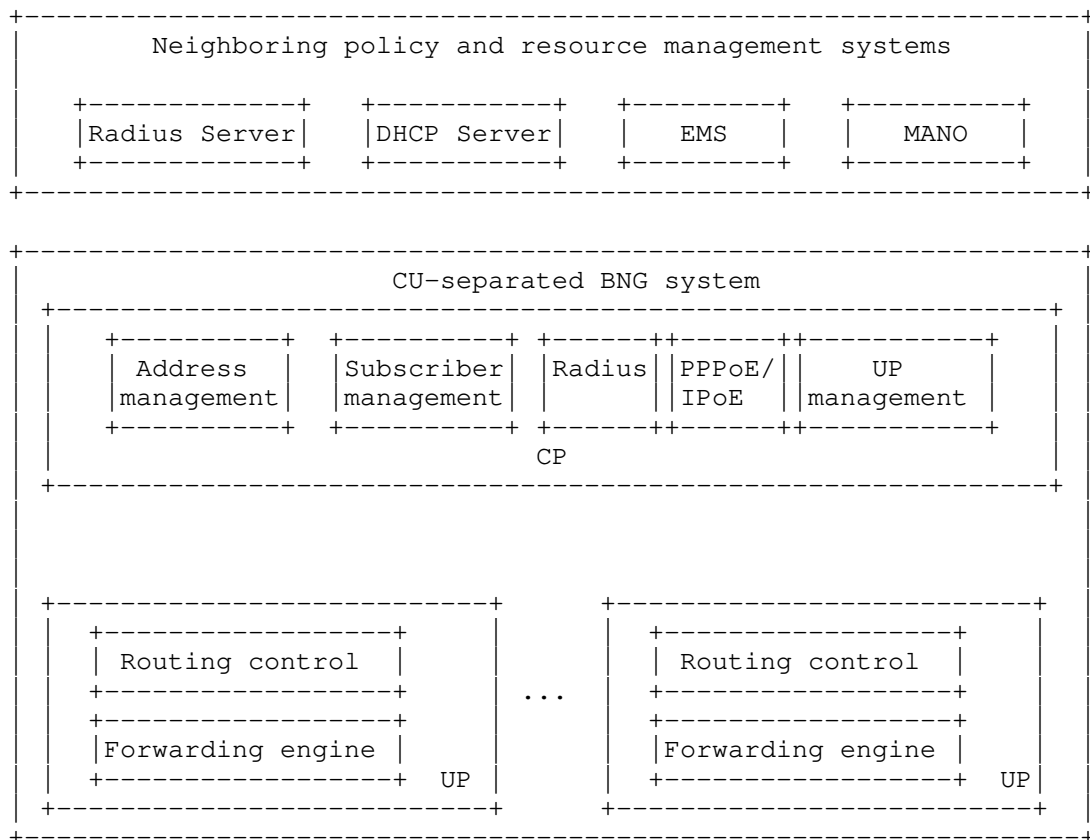


Figure 1. Architecture of CU Separated BNG

Briefly, a CU separated BNG is made up of a Control Plane (CP) and a set of User Planes (UPs) [TR-384], [I-D.cuspdrt-gwg-cu-separation-bng-deployment]. The Control Plane is a user control management component which manages UP's resources such as the user entry and user's Quality of Service (QoS) policy, for example, the access bandwidth and priority management. This Control Plane could be virtualized and centralized. The functional modules 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, update, and maintenance of channels between control planes and user planes [TR-384], [I-D.cuspdrt-gwg-cu-separation-bng-deployment]. The User Plane (UP) is a network edge and user policy implementation component. It can support the forwarding plane functions on traditional BNG devices,

such as traffic forwarding, QoS, and traffic statistics collection, and it can also support the control plane functions on traditional BNG devices, such as routing, multicast, etc [TR-384], [I-D.cuspdrt-gwg-cu-separation-bng-deployment].

### 3.1. Internal interfaces between the CP and UP

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

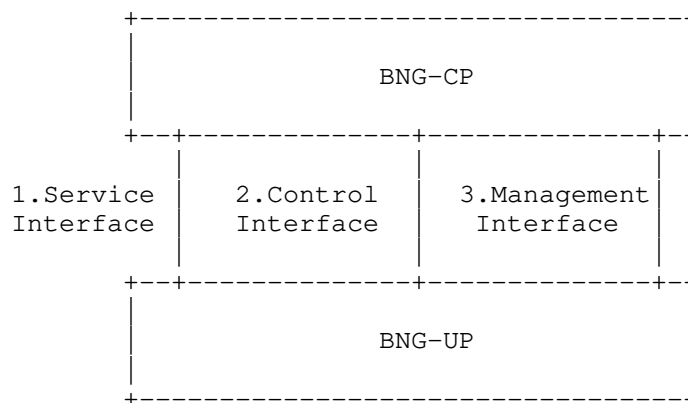


Figure 2. Interfaces between the BNG-CP and the BNG-UP

**Service interface:** The CP and UP use this interface to establish VXLAN tunnels with each other and transmit PPPoE and IPoE packets over the VXLAN tunnels.

**Control interface:** The CP uses this interface to deliver service entries, and the UP uses this interface to report service events to the CP.

**Management interface:** The CP uses this interface to deliver configurations to the UP. This interface uses NETCONF.

The CUSP (Control plane and User plane Separated BNG protocol) defines the control interface, and specifies the communication between the centralized control plane and user planes. This protocol should be designed to support establishing and maintaining a conversation between CP and UPs, and transporting the tables that are specified in [draft-cuspdrt-gwg-cu-separation-infor-model].

#### 4. The usage of CU separation BNG protocol

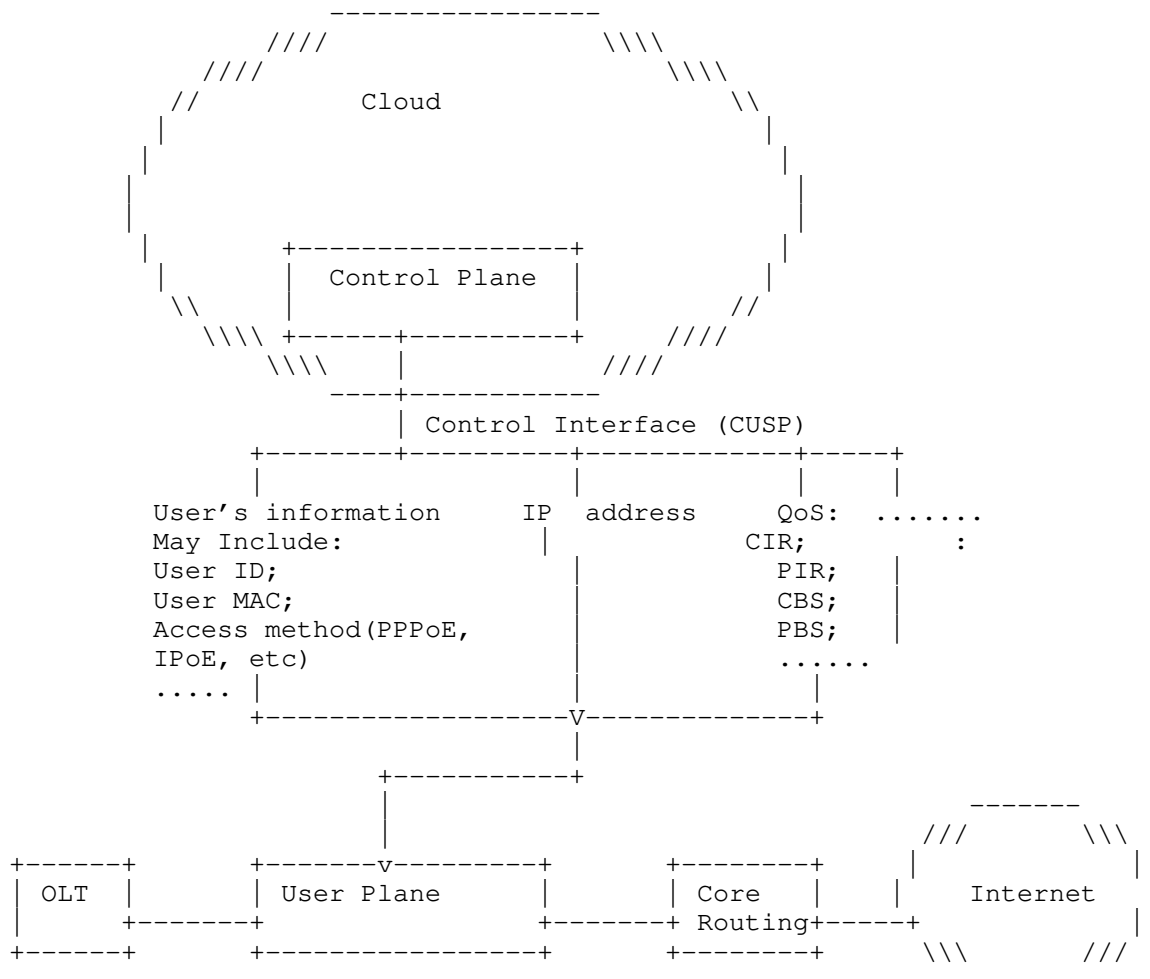


Figure 3. CU Separation BNG protocol usage

As shown in Figure 3, when users access the BNG network, the control plane solicits user information (such as user's ID, user's MAC, user's access methods, for example via PPPoE/IPoE), associates users with available bandwidth which is reported by User planes, and, based on the service's requirement, generates a set of tables, which may include user's information, UP's IP segment, and QoS, etc. Then the control plane can transmit these tables to the User planes. User

planes receive these tables, parse them, and then perform corresponding actions.

## 5. Control Plane and User Plane Separation Protocol Requirements

This section specifies the requirements for the CU separation protocol.

### 5.1. Transmit information tables

The Control Plane and User Plane Separation Protocol **MUST** allow the CP to send tables to each User Plane device.

a) The current BNG service requires that the UP should support at least 2000 users being accessed every second. And every user requires at least 2000 bytes. To achieve high performance, the CU Separation protocol **SHOULD** be lightweight.

b) CU separation protocol should support data encoded as either XML or binary. It allows user information data to be read, saved, and manipulated with tools specific to XML or binary.

c) In order to provide centralized session management, high scalability for subscriber management capacity, and cost-efficient redundancy, batching ability should be provided. The CU Separation protocol should be able to group an ordered set of commands to a UP device. Each such group of commands **SHOULD** be sent to the UP in as few messages as practical. Furthermore, the protocol **MUST** support the ability to specify if a command group **MUST** have all-or-nothing semantics.

d) The CU Separation protocol **SHOULD** be able to support at least hundreds of UP devices and tens of thousands of ports. For example, the protocol field sizes corresponding to UP or port numbers **SHALL** be large enough to support the minimum required numbers. This requirement does not relate to the performance of the system as the number of UPs or ports in the system grows.

### 5.2. Message Priority

The CU Separation protocol **MUST** provide a means to express the protocol message priorities.

### 5.3. Reliability

Heartbeat is a periodic signal generated by hardware or software to test for some aspects of normal operation or to synchronize other parts of network system.

In the CU separation BNG, a heartbeat is sent between CP and UPs at a regular interval on the order of seconds. If the CP/UP does not receive a heartbeat for a time--usually a few heartbeat intervals--the CP/UP that should have sent the heartbeat is assumed to have failed.

The CU separation protocol should support some kind of heartbeat monitoring mechanism. And this mechanism should have ability to distinguish whether the interruption is an actual failure. For example, in some scenarios (i.e. CP/UP update, etc), the connection between the UP and CP need to be interrupted. In this case, the interruption should not be reported.

#### 5.4. Support for Secure Communication

As mentioned above, CP may send some information tables to the UP which may be critical to the network function (e.g, User Information, IPv4/IPv6 information) and may reflect the business information (e.g, QoS, service level agreements, etc). Therefore, supporting the integrity of all CU Separation protocol messages and protecting against man-in-the-middle attacks MUST be supported.

The CP Separation protocol should support security in a variety of scenarios. For example, the connections between the CP and UPs could be dedicated lines, VPNs within one domain, or could cross several domains, that is, cross third party networks. Thus it is likely that more than one security mechanism SHOULD be supported. TLS and IPsec are good candidates for such mechanisms.

#### 5.5. Version negotiation

The CU separated BNG may consist of different vendors' devices implementing different versions of protocol. Therefore, the CU separation protocol MUST provide some mechanisms to perform the version negotiation.

Version negotiation is the process that the CU separated BNG's Control-Plane uses to evaluate the protocol versions supported by both the control-plane and the user-plane devices. Then a suitable protocol version is selected for communication in CUSP. The process is a "negotiation" because it requires identifying the most recent protocol version that is supported by both the control-plane and the user-plane devices or determining that they have no version in common.

## 5.6. Capability Exchange

The UP Capability Report displays the device's profile, service capability, and other assigned capabilities within the CU separated BNG. The CU separation protocol should MUST provide some mechanism to exchange the UP device's capabilities.

## 5.7. CP primary/backup capability

A backup CP for failure recovery is required for the CU separated BNG network. And the CUSP should provide some mechanism to implement the backup CP:

- a) In some scenarios, there may be two CP devices both declaring the primary CP. Thus the CUSP should support or associate with some mechanisms to determine which CP is the primary device.
- b) In the scenario of the primary CP down, the CUSP should support switching between primary and backup CP.

## 5.8. Event Notification

The CUSP protocol SHOULD be able to asynchronously notify the CP of events on the UP such as failures and changes in available resources and capabilities. Some scenarios that may initiate event notifications are listed below.

- a) Sending response message: As mentioned above, the control plane solicits users' information, associates them with available bandwidth, and generates a set of tables based on the service's requirement. Then the control plane transmits these tables to the corresponding User plane. The UP should respond with an event notification to inform the CP that the tables are received.
- b) User trace: The user trace mechanism can support the Control Plane tracing and monitoring the network status for users (for example the real-time bandwidth, etc), to help debug the user's application. Therefore, the UPs SHOULD be able to notify the CP with the User trace message.
- c) Sending statistics parameters: In CU separation BNG, the User-plane will report the traffic statistics parameters to the Control-plane, such as the ingress packets, ingress bytes, egress packets, egress bytes, etc. These parameters can help measure the BNG network performance. Available network resources can be allocated basing on the statistics parameters by the BNG-CP. Therefore, the UPs SHOULD be able to notify the CP with statistics parameters.



d) Report the result of User Detect: "User Detect" message will be send periodically to detect user dial-up and disconnect. The UPs SHOULD be able to notify the CP with the result of User Detect.

#### 5.9. Query Statistics

The CUSP protocol MUST provide a means for the CP to be able to query statistics (performance monitoring) from the UP.

#### 6. Security Considerations

As this is an Informational requirements document, detailed technical Security Considerations are not included. However, Section 5.4 covers general security requirements and Section 5.7 covers backup requirements relevant to some denial of service scenarios.

#### 7. IANA Considerations

This document requires no IANA actions.

#### 8. References

##### 8.1. Normative References

- [I-D.cuspdtdt-rtgwg-cu-separation-bng-deployment]  
Gu, R., Hu, S., and Z. Wang, "Deployment Model of Control Plane and User Plane Separation BNG", draft-cuspdtdt-rtgwg-cu-separation-bng-deployment-00 (work in progress), October 2017.
- [I-D.cuspdtdt-rtgwg-cu-separation-infor-model]  
Wang, Z., Gu, R., Lopezalvarez, V., and S. Hu, "Information Model of Control-Plane and User-Plane separation BNG", draft-cuspdtdt-rtgwg-cu-separation-infor-model-00 (work in progress), February 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [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>>.

## 8.2. Informative References

- [TR-384] Broadband Forum, "Cloud Central Office Reference Architectural Framework", BBF TR-384, January. 2018.

## Authors' Addresses

Shujun Hu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: shujun\_hu@outlook.com

Victor Lopez  
Telefonica  
Sur 3 building, 3rd floor, Ronda de la Comunicacion s/n  
Madrid 28050  
Spain

Email: victor.lopezalvarez@telefonica.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

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

Email: d3e3e3@gmail.com

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

Email: wangzitao@huawei.com

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

Email: song.jun@huawei.com

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: July 2019

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

Seamless Interconnect Underlay to Cloud Overlay Problem Statement  
draft-dm-net2cloud-problem-statement-07

Abstract

This document describes the problems that enterprises face today when connecting their branch offices to dynamic workloads in third party data centers (a.k.a. Cloud DCs).

It examines some of the approaches interconnecting cloud DCs with enterprises' on-premises DCs & branch offices. This document also describes some of the (network) problems that many enterprises face when they have workloads & applications & data split among hybrid data centers, especially for those enterprises with multiple sites that are already interconnected by VPNs (e.g., MPLS L2VPN/L3VPN).

Current operational problems are examined to determine whether there is a need to improve existing protocols or whether a new protocol is necessary to solve them.

Status of this Memo

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

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

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

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

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

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

This Internet-Draft will expire on August 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.....	3
2. Definition of terms.....	4
3. Current Practices in Interconnecting Enterprise Sites with Cloud DCs.....	5
3.1. Interconnect to Cloud DCs.....	5
3.2. Interconnect to Hybrid Cloud DCs.....	7
3.3. Connecting workloads among hybrid Cloud DCs.....	7
4. Desired Properties for Networks that interconnect Hybrid Clouds	8
5. Problems with MPLS-based VPNs extending to Hybrid Cloud DCs....	9
6. Problem with using IPsec tunnels to Cloud DCs.....	10
6.1. Complexity of multi-point any-to-any interconnection....	10
6.2. Poor performance over long distance.....	11
6.3. Scaling Issues with IPsec Tunnels.....	11
7. Problems of Using SD-WAN to connect to Cloud DCs.....	12
7.1. SD-WAN among branch offices vs. interconnect to Cloud DCs	12

8. End-to-End Security Concerns for Data Flows.....	15
9. Requirements for Dynamic Cloud Data Center VPNs.....	15
10. Security Considerations.....	16
Solution drafts resulting from this work will address security concerns inherent to the solution(s), including both protocol aspects and the importance (for example) of securing workloads in cloud DCs and the use of secure interconnection mechanisms.....	
IANA Considerations.....	16
11. References.....	16
11.1. Normative References.....	16
11.2. Informative References.....	16
12. Acknowledgments.....	17

## 1. Introduction

The ever-increasing use of cloud applications for communication services change the way corporate business works and shares information. Such cloud applications use resources hosted in third party DCs that also host services for other customers.

With the advent of widely available third party cloud DCs in diverse geographic locations and the advancement of tools for monitoring and predicting application behaviors, it is technically feasible for enterprises to instantiate applications and workloads in locations that are geographically closest to their end-users. Such proximity improves end-to-end latency and overall user experience. Conversely, an enterprise can easily shutdown applications and workloads whenever end-users are in motion (thereby modifying the networking connection of subsequently relocated applications and workloads). In addition, an enterprise may wish to take advantage of more and more business applications offered by third party private cloud DCs.

Most of those enterprise branch offices & on-premises data centers are already connected via VPNs, such as MPLS-based L2VPNs and L3VPNs. Then connecting to the cloud-hosted resources may not be straightforward if the provider of the VPN service does not have direct connections to the corresponding cloud DCs. Under those circumstances, the enterprise can upgrade the CPEs deployed in its various premises to utilize SD-WAN techniques to reach cloud resources (without any assistance from the VPN service provider), or wait for their VPN service provider to make new agreements with data

center providers to connect to the cloud resources. Either way has additional infrastructure and operational costs.

In addition, it is an uptrend with more enterprises instantiating their apps & workloads in different cloud DCs to maximize the benefits of geographical proximity, elasticity and special features offered by different cloud DCs.

## 2. Definition of terms

**Cloud DC:** Third party Data Centers that usually host applications and workload owned by different organizations or tenants.

**Controller:** Used interchangeably with SD-WAN controller to manage SD-WAN overlay path creation/deletion and monitoring the path conditions between two or more sites.

**DSVPN:** Dynamic Smart Virtual Private Network. DSVPN is a secure network that exchanges data between sites without needing to pass traffic through an organization's headquarter virtual private network (VPN) server or router.

**Heterogeneous Cloud:** applications & workloads split among Cloud DCs owned & managed by different operators.

**Hybrid Clouds:** Hybrid Clouds (usually plural) refer to enterprises using their own premises DCs in addition to Cloud services provided by multiple cloud operators. For example, an enterprise not only have applications running in their own DCs, but also have applications hosted in multiple third party cloud DCs ((AWS, Azure, Google, Salesforces, SAP, etc). . ONUG also has a notion of heterogeneous cloud, refers to enterprises does not have its own DC, only uses services by 3rd party cloud operators.

**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).

VPC: Virtual Private Cloud. A service offered by Cloud DC operators to allocate logically-isolated cloud resources, including compute, networking and storage.

### 3. Current Practices in Interconnecting Enterprise Sites with Cloud DCs

#### 3.1. Interconnect to Cloud DCs

Most Cloud operators offer some type of network gateway through which an enterprise can reach their workloads hosted in the Cloud DCs. For example, AWS (Amazon Web Services) offers the following options to reach workloads in AWS Cloud DCs:

- Internet gateway for any external entities to reach the workloads hosted in AWS Cloud DC via the Internet.
- Virtual gateway (vGW) where IPsec tunnels [RFC6071] are established between an enterprise's own gateway and AWS vGW, so that the communications between those gateways can be secured from the underlay (which might be the public Internet).
- Direct Connect, which allows enterprises to purchase direct connect from network service providers to get a private leased line interconnecting the enterprises gateway(s) and the AWS Direct Connect routers. Via Direct Connect, an AWS Transit Gateway can be used to interconnect multiple VPCs in different Availability Zones.

CPEs at one Enterprise branch office are connected to the Internet to reach AWS's vGW via IPsec tunnels. Other ports of such CPEs are connected to AWS DirectConnect via a private network (without any encryption).



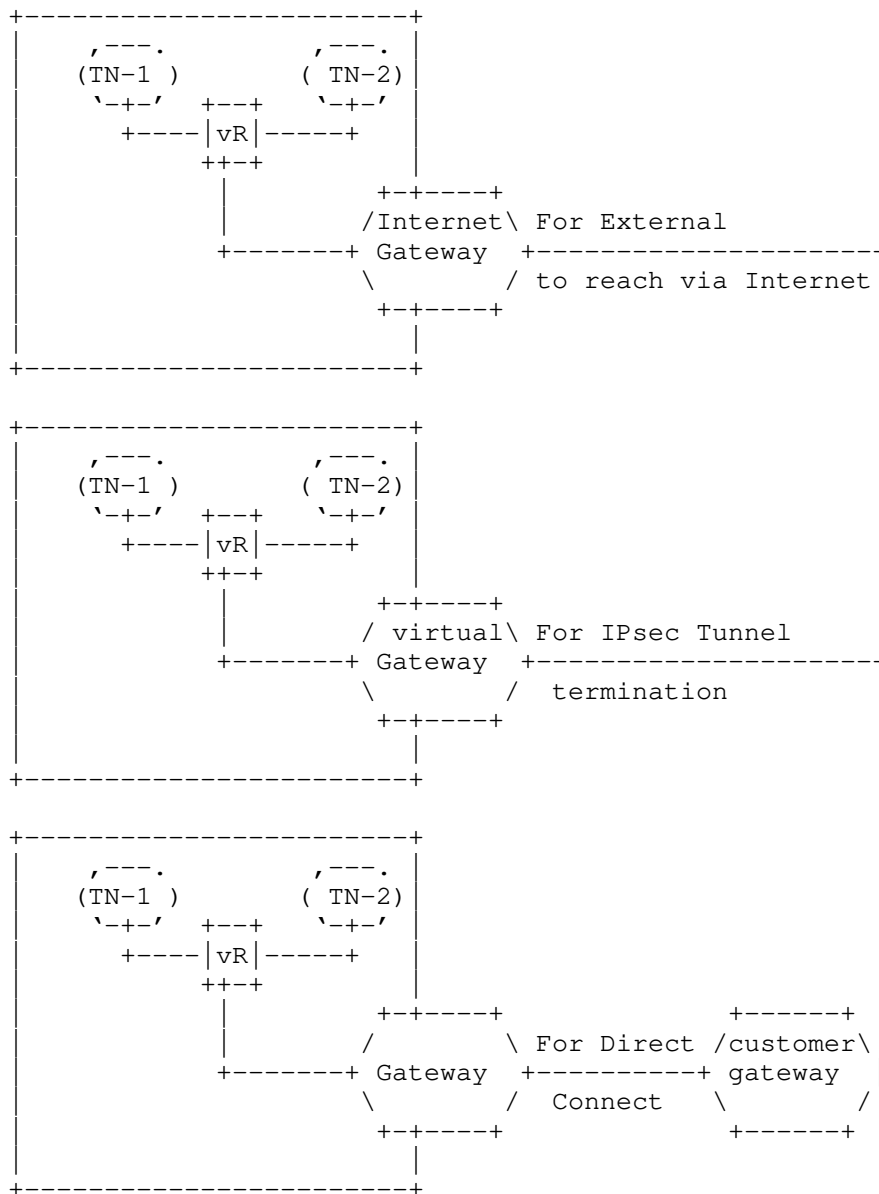


Figure 1: Examples of Cloud DC connections.

### 3.2. Interconnect to Hybrid Cloud DCs

According to Gartner, by 2020 "hybrid will be the most common usage of the cloud" as more enterprises see the benefits of integrating public and private cloud infrastructures. However, enabling the growth of hybrid cloud deployments in the enterprise requires fast and safe interconnection between public and private cloud services. For an enterprise to connect to applications & workloads hosted in multiple Cloud DCs, the enterprise can use IPsec tunnels established over the Internet or a (virtualized) leased line service to connect its on-premises gateways to each of the Cloud DC's gateways, virtual routers instantiated in the Cloud DCs, or any other suitable design (including a combination thereof).

Some enterprises prefer to instantiate their own virtual CPEs/routers inside the Cloud DC to connect the workloads within the Cloud DC. Then an overlay path is established between customer gateways to the virtual CPEs/routers for reaching the workloads inside the cloud DC.

### 3.3. Connecting workloads among hybrid Cloud DCs

There are multiple approaches to interconnect workloads among different Cloud DCs:

- Utilize Cloud DC provided transit gateways, which usually does not work if Cloud DCs are owned and managed by different Cloud providers.
- Hairpin all the traffic through the customer gateway, which creates additional transmission delay & incurs cost when exiting Cloud DCs, or
- Establish direct tunnels among different VPCs (Virtual Private Clouds) via client's own virtual routers instantiated within Cloud DCs. DMVPN (Dynamic Multipoint Virtual Private Network) or DSVPN (Dynamic Smart VPN) techniques can be used to establish direct Multi-point-to-Point or multi-point-to multi-point tunnels among those client's own virtual routers.

DMVPN & DSVPN use NHRP (Next Hop Resolution Protocol) [RFC2735] so that spoke nodes can register their IP addresses & WAN ports with the hub node. The IETF ION (Internetworking over NBMA (non-broadcast

multiple access) WG standardized NHRP for connection-oriented NBMA network (such as ATM) network address resolution more than two decades ago.

There are many differences between virtual routers in Public Cloud DCs and the nodes in an NBMA network. NHRP & DSVPN are not cannot be used for registering virtual routers in Cloud DCs unless an extension of such protocols is developed for that purpose. Other protocols such as BGP can be used, as described in [BGP-SDWAN].

#### 4. Desired Properties for Networks that interconnect Hybrid Clouds

The networks that interconnect hybrid cloud DCs must address the following requirements:

- High availability at any time, whatever the duration of the connection to the cloud DC.  
Many enterprises include cloud infrastructures in their disaster recovery strategy, e.g., by enforcing periodic backup policies within the cloud, or by running backup applications in the Cloud, etc. Therefore, the connection to the cloud DCs may not be permanent, but rather needs to be on-demand.
- Global reachability from different geographical zones, thereby facilitating the proximity of applications as a function of the end users' location, to improve latency.
- Elasticity and mobility, to instantiate additional applications at Cloud DCs when end-users' usages increase and shut down applications at locations when there are fewer end-users. Some enterprises have front-end web portals running in cloud DCs and database servers in their on-premises DCs. Those Front-end web portals need to be reachable from the public Internet. The backend connection to the sensitive data in database servers hosted in the on-premises DCs might need secure connections.
- Scalable security management. IPsec is commonly used to interconnect cloud gateways with CPEs deployed in the enterprise premises. For enterprises with a large number or branch offices, managing the IPsec's Security Associations among many nodes can be very difficult.

## 5. Problems with MPLS-based VPNs extending to Hybrid Cloud DCs

Traditional MPLS-based VPNs have been widely deployed as an effective way to support businesses and organizations that require network performance and reliability. MPLS shifted the burden of managing a VPN service from enterprises to service providers. The CPEs attached to MPLS VPNs are also simpler and less expensive, since they do not need to manage routes to remote sites; they simply pass all outbound traffic to the MPLS VPN PEs to which the CPEs are attached (albeit multi-homing scenarios require more processing logic on CPEs). MPLS has addressed the problems of scale, availability, and fast recovery from network faults, and incorporated traffic-engineering capabilities.

However, traditional MPLS-based VPN solutions are sub-optimized for connecting end-users to dynamic workloads/applications in cloud DCs because:

- The Provider Edge (PE) nodes of the enterprise's VPNs might not have direct connections to third party cloud DCs that are used for hosting workloads with the goal of providing an easy access to enterprises' end-users.
- It usually takes some time to deploy provider edge (PE) routers at new locations. When enterprise's workloads are changed from one cloud DC to another (i.e., removed from one DC and re-instantiated to another location when demand changes), the enterprise branch offices need to be connected to the new cloud DC, but the network service provider might not have PEs located at the new location.

One of the main drivers for moving workloads into the cloud is the widely available cloud DCs at geographically diverse locations, where apps can be instantiated so that they can be as close to their end-users as possible. When the user base changes, the applications may be migrated to a new cloud DC location closest to the new user base.

- Most of the cloud DCs do not expose their internal networks, so the MPLS-based VPNs can only reach Cloud DC's Gateways, not to the workloads hosted inside.
- Many cloud DCs use an overlay to connect their gateways to the workloads located inside the DC. There has not been any standard to address the interworking between the Cloud Overlay and the enterprise' existing underlay networks.

Another roadblock is the lack of a standard way to express and enforce consistent security policies for workloads that not only use virtual addresses, but in which are also very likely hosted in different locations within the Cloud DC [RFC8192]. The current VPN path computation and bandwidth allocation schemes may not be flexible enough to address the need for enterprises to rapidly connect to dynamically instantiated (or removed) workloads and applications regardless of their location/nature (i.e., third party cloud DCs).

#### 6. Problem with using IPsec tunnels to Cloud DCs

As described in the previous section, many Cloud operators expose their gateways for external entities (which can be enterprises themselves) to directly establish IPsec tunnels. Enterprises can also instantiate virtual routers within Cloud DCs to connect to their on-premises devices via IPsec tunnels. If there is only one enterprise location that needs to reach the Cloud DC, an IPsec tunnel is a very convenient solution.

However, many medium-to-large enterprises usually have multiple sites and multiple data centers. For workloads and apps hosted in cloud DCs, multiple sites need to communicate securely with those cloud workloads and apps. This section documents some of the issues associated with using IPsec tunnels to connect enterprise premises with cloud gateways.

##### 6.1. Complexity of multi-point any-to-any interconnection

The dynamic workload instantiated in cloud DC needs to communicate with multiple branch offices and on-premises data centers. Most enterprises need multi-point interconnection among multiple locations, which can be provided by means of MPLS L2/L3 VPNs.

Using IPsec overlay paths to connect all branches & on-premises data centers to cloud DCs requires CPEs to manage routing among Cloud DCs gateways and the CPEs located at other branch locations, which can dramatically increase the complexity of the design, possibly at the cost of jeopardizing the CPE performance.

The complexity of requiring CPEs to maintain routing among other CPEs is one of the reasons why enterprises migrated from Frame Relay based services to MPLS-based VPN services.

MPLS-based VPNs have their PE directly connected to the CPEs. Therefore, CPEs only need to forward all traffic to the directly attached PEs, which are therefore responsible for enforcing the routing policy within the corresponding VPNs. Even for multi-homed CPEs, the CPEs only need to forward traffic among the directly connected PEs. However, when using IPsec tunnels between CPEs and Cloud DCs, the CPEs need to compute, select, establish and maintain routes for traffic to be forwarded to Cloud DCs, to remote CPEs via VPN, or directly.

#### 6.2. Poor performance over long distance

When enterprise CPEs or gateways are far away from cloud DC gateways or across country/continent boundaries, performance of IPsec tunnels over the public Internet can be problematic and unpredictable. Even though there are many monitoring tools available to measure delay and various performance characteristics of the network, the measurement for paths over the Internet is passive and past measurements may not represent future performance.

Many cloud providers can replicate workloads in different available zones. An App instantiated in a cloud DC closest to clients may have to cooperate with another App (or its mirror image) in another region or database server(s) in the on-premises DC. This kind of coordination requires predictable networking behavior/performance among those locations.

#### 6.3. Scaling Issues with IPsec Tunnels

IPsec can achieve secure overlay connections between two locations over any underlay network, e.g., between CPEs and Cloud DC Gateways.

If there is only one enterprise location connected to the cloud gateway, a small number of IPsec tunnels can be configured on-demand

between the on-premises DC and the Cloud DC, which is an easy and flexible solution.

However, for multiple enterprise locations to reach workloads hosted in cloud DCs, the cloud DC gateway needs to maintain multiple IPsec tunnels to all those locations (e.g., as a hub & spoke topology). For a company with hundreds or thousands of locations, there could be hundreds (or even thousands) of IPsec tunnels terminating at the cloud DC gateway, which is not only very expensive (because Cloud Operators usually charge their customers based on connections), but can be very processing intensive for the gateway. Many cloud operators only allow a limited number of (IPsec) tunnels & bandwidth to each customer. Alternatively, you could use a solution like group encryption where a single IPsec SA is necessary at the GW but the drawback here is key distribution and maintenance of a key server, etc.

#### 7. Problems of Using SD-WAN to connect to Cloud DCs

SD-WAN can establish parallel paths over multiple underlay networks between two locations on-demand, for example, to support the connections established between two CPEs interconnected by a traditional MPLS VPN ([RFC4364] or [RFC4664]) or by IPsec [RFC6071] tunnels.

SD-WAN lets enterprises augment their current VPN network with cost-effective, readily available Broadband Internet connectivity, enabling some traffic offloading to paths over the Internet according to differentiated, possibly application-based traffic forwarding policies, or when the MPLS VPN connection between the two locations is congested, or otherwise undesirable or unavailable.

##### 7.1. SD-WAN among branch offices vs. interconnect to Cloud DCs

SD-WAN interconnection of branch offices is not as simple as it appears. For an enterprise with multiple sites, using SD-WAN overlay paths among sites requires each CPE to manage all the addresses that local hosts have the potential to reach, i.e., map internal VPN addresses to appropriate SD-WAN paths. This is similar to the complexity of Frame Relay based VPNs, where each CPE needed to maintain mesh routing for all destinations if they were to avoid an extra hop through a hub router. Even though SD-WAN CPEs can get assistance from a central controller (instead of running a routing protocol) to resolve the mapping between destinations and SD-WAN paths, SD-WAN CPEs are still responsible for routing table

maintenance as remote destinations change their attachments, e.g., the dynamic workload in other DCs are de-commissioned or added.

Even though originally envisioned for interconnecting branch offices, SD-WAN offers a very attractive way for enterprises to connect to Cloud DCs.

The SD-WAN for interconnecting branch offices and the SD-WAN for interconnecting to Cloud DCs have some differences:

- SD-WAN for interconnecting branch offices usually have two end-points (e.g., CPEs) controlled by one entity (e.g., a controller or management system operated by the enterprise).
- SD-WAN for Cloud DC interconnects may consider CPEs owned or managed by the enterprise, while remote end-points are being managed or controlled by Cloud DCs (For the ease of description, let's call such CPEs asymmetrically-managed CPEs).



- Cloud DCs may have different entry points (or devices) with one entry point that terminates a private direct connection (based upon a leased line for example) and other entry points being devices terminating the IPsec tunnels, as shown in Figure 2.

Therefore, the SD-WAN design becomes asymmetric.

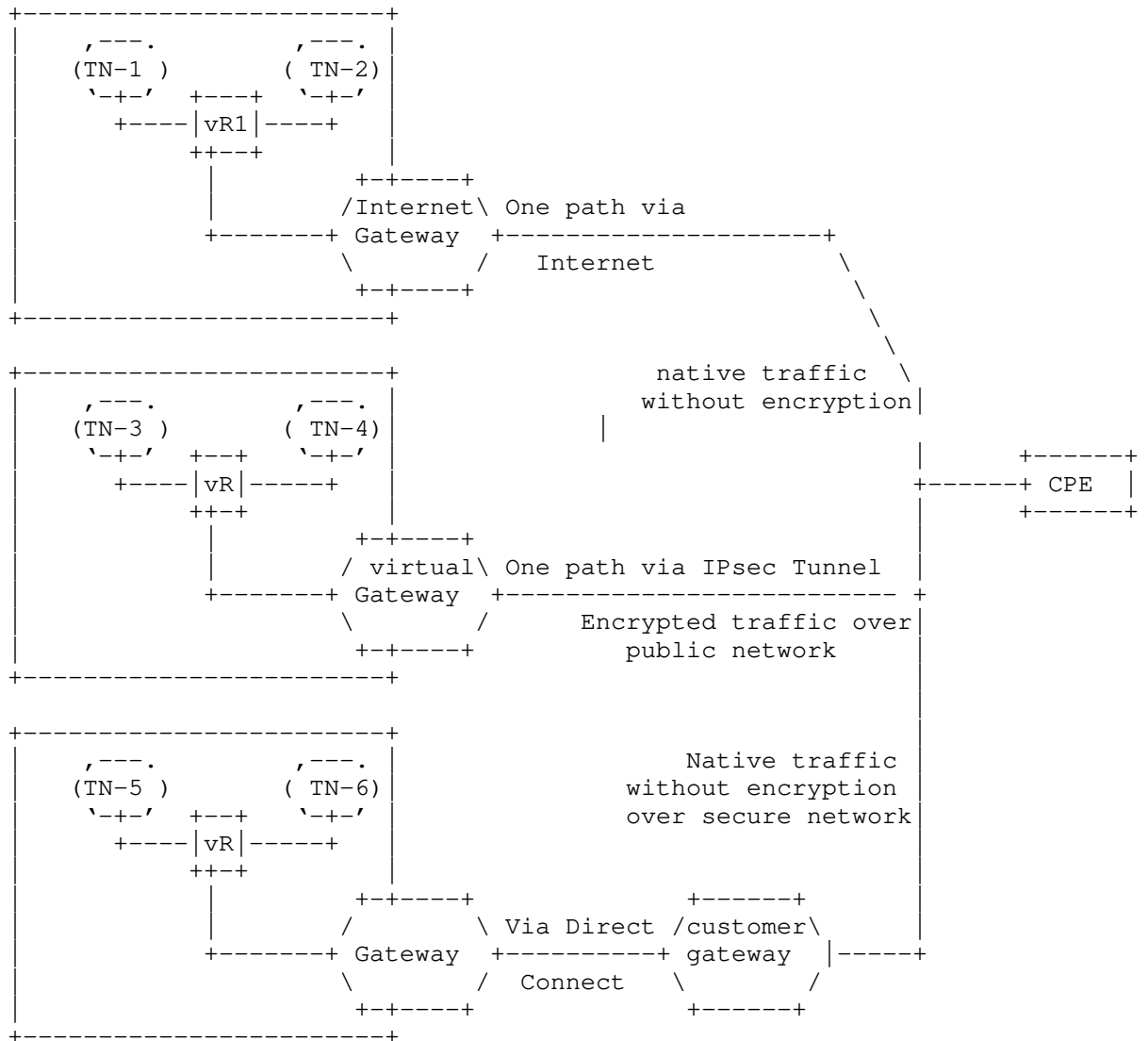


Figure 2: Different Underlays to Reach Cloud DC

## 8. End-to-End Security Concerns for Data Flows

When IPsec tunnels established from enterprise on-premises CPEs are terminated at the Cloud DC gateway where the workloads or applications are hosted, some enterprises have concerns regarding traffic to/from their workload being exposed to others behind the data center gateway (e.g., exposed to other organizations that have workloads in the same data center).

To ensure that traffic to/from workloads is not exposed to unwanted entities, IPsec tunnels may go all the way to the workload (servers, or VMs) within the DC.

## 9. Requirements for Dynamic Cloud Data Center VPNs

In order to address the aforementioned issues, any solution for enterprise VPNs that includes connectivity to dynamic workloads or applications in cloud data centers should satisfy a set of requirements:

- The solution should allow enterprises to take advantage of the current state-of-the-art in VPN technology, in both traditional MPLS-based VPNs and IPsec-based VPNs (or any combination thereof) that run over the public Internet.
- The solution should not require an enterprise to upgrade all their existing CPEs.
- The solution should support scalable IPsec key management among all nodes involved in DC interconnect schemes.
- The solution needs to support easy and fast, on-the-fly, VPN connections to dynamic workloads and applications in third party data centers, and easily allow these workloads to migrate both within a data center and between data centers.
- Allow VPNs to provide bandwidth and other performance guarantees.
- Be a cost-effective solution for enterprises to incorporate dynamic cloud-based applications and workloads into their existing VPN environment.

## 10. Security Considerations

The draft discusses security requirements as a part of the problem space, particularly in sections 4, 5, and 8.

Solution drafts resulting from this work will address security concerns inherent to the solution(s), including both protocol aspects and the importance (for example) of securing workloads in cloud DCs and the use of secure interconnection mechanisms.

### IANA Considerations

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

## 11. References

### 11.1. Normative References

### 11.2. Informative References

[RFC2735] B. Fox, et al "NHRP Support for Virtual Private networks". Dec. 1999.

[RFC8192] S. Hares, et al "Interface to Network Security Functions (I2NSF) Problem Statement and Use Cases", July 2017

[ITU-T-X1036] ITU-T Recommendation X.1036, "Framework for creation, storage, distribution and enforcement of policies for network security", Nov 2007.

[RFC6071] S. Frankel and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", Feb 2011.

[RFC4364] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", Feb 2006

[RFC4664] L. Andersson and E. Rosen, "Framework for Layer 2 Virtual Private Networks (L2VPNs)", Sept 2006.

[BGP-SDWAN] L. Dunbar, et al. "BGP Extension for SDWAN Overlay Networks", draft-dunbar-idr-bgp-sdwan-overlay-ext-03, work-in-progress, Nov 2018.

## 12. Acknowledgments

Many thanks to Ignas Bagdonas, Michael Huang, Liu Yuan Jiao, Katherine Zhao, and Jim Guichard for the discussion and contributions.

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

Mehmet Toy  
Verizon  
One Verizon Way  
Basking Ridge, NJ 07920  
Email: mehmet.toy@verizon.com



IETF RTGWG  
Internet-Draft  
Intended status: Standards Track  
Expires: September 20, 2018

Fangwei Hu  
RongRong Hua  
ZTE Corporation  
Shujun Hu  
Rong Gu  
China Mobile  
Mar 19, 2018

YANG Data Model for Configuration Interface of Control-Plane and User-  
Plane separation BNG  
draft-hu-rtgwg-cu-separation-yang-model-03.txt

Abstract

This document defines the YANG data model for operation management of Control-Plane and User-Plane separation BNG (Broadband Network Gateway).

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 20, 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
2. Conventions used in this document . . . . .	4
2.1. Terminology . . . . .	4
2.2. Requirements Language . . . . .	4
3. Design Tree . . . . .	4
3.1. Global Configuration . . . . .	4
3.2. BNG-UP Interface Configuration . . . . .	5
3.3. Control Channel Configuration . . . . .	5
3.4. Service Channel Configuration . . . . .	6
3.5. Multicast Service . . . . .	6
3.6. PPPOX Configuration . . . . .	7
3.7. Acl Configuration . . . . .	8
3.8. QoS Configuration . . . . .	9
4. vBNG YANG Data Model . . . . .	9
5. Security Considerations . . . . .	29
6. Acknowledgements . . . . .	30
7. IANA Considerations . . . . .	30
8. References . . . . .	30
8.1. Normative References . . . . .	30
8.2. Informative References . . . . .	31
Authors' Addresses . . . . .	32

## 1. Introduction

The main idea of BNG Control-Plane and User-Plane separation is to extract and centralize the user management functions of multiple BNG devices, forming an unified and centralized control plane (CP), while the traditional router's Control Plane and forwarding plane are both preserved on BNG devices in the form of a user plane (UP). We name the control-Plane and User-plane separation BNG as vBNG.

The architecture of Control-plane and user-plane separated BNG is shown as the following figure.



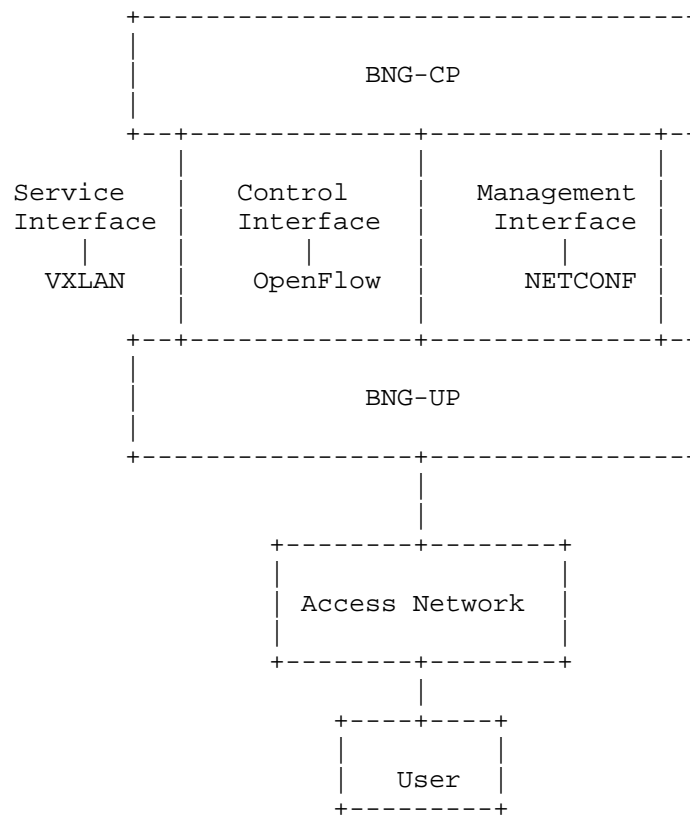


Figure 1: Architecture of C/U separated BNG

There are three interfaces between BNG-CP and BNG-UP: Service interface, control interface and management interface. The service interface is used to carry PPPoE/IPoE dialup packets between user plane and control plane. The requirement and possible solution is defined in the [I-D.huang-nvo3-vxlan-extension-for-vbras]. Control interface is used for setting forwarding entries of user plane through OpenFlow or other protocols. Management interface is used by BNG-CP to carry out related configurations of BNG-UP through NETCONF protocol [RFC6241].

This document defines the YANG data model for vBNG(BNG-CP and BNG-UP). There are three types of YANG data model for vBNG in this document: The YANG data models for BNG-CP, the YANG data models for BNG-UP by network management directly, and the YANG data models for BNG-UP through the management interfaces among the BNG-UP and BNG-CP.

The YANG data models through the management interfaces include: The BNG-UP interfaces configuration(Section 3.2), the controller channel configuration(Section 3.3), the ACL configuration for BNG-UP(Section 3.6) and QoS configuration for BNG-UP (section 3.7), etc.

## 2. Conventions used in this document

### 2.1. Terminology

**BNG:** Broadband Network Gateway. A broadband remote access server 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.

**BNG-CP:** BNG Control Plane. The BNG-CP is a user control management component which support to manage UP's resources such as the user entry and forwarding policy.

**BNG-UP:** BNG User Plane. BNG-UP is a network edge and user policy implementation component.

**vBNG:** Virtualization Broadband Network Gateway. An vBNG is to extract and centralize the user management functions of multiple BNG devices, and to form an unified and centralized control plane (CP). The vBNG devices include BNG-UP and BNG-CP.

### 2.2. Requirements Language

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

## 3. Design Tree

### 3.1. Global Configuration

The BNG-UP or BNG-CP part can be a physical or logical network element. The LNE model [I-D.ietf-rtgwg-lne-model] is augmented to define the YANG data models for BNG-UP and BNG-CP in this document.

```

module: ietf-vbng
  augment /lne:logical-network-elements/lne:logical-network-element:
    +--rw ietf-vbng
      +--rw bng-cp
        | +--rw bng-cp-name?   string
        | +--rw enable?       boolean
      +--rw bng-up!
        | +--rw bng-up* [shelf-no]
        | | +--rw shelf-no      uint8
        | | +--rw bng-up-name?  string
        | | +--rw netconf-server!
        | | | +--rw ip          inet:ipv4-address
        | | | +--rw user-name?  string
        | | | +--rw password?   string
        | | | +--rw port?       uint32
        | | +--rw keepalive-sink? enumeration

```

### 3.2. BNG-UP Interface Configuration

The BNG-UP interface configuration is to configure the basic interface informations of BNG-UP element, such as interface name, the VLAN parameters for the sub-interface. The BNG-UP interface data models are configured through the management interfaces between BGN-UP and BNG-CP by netconf protocol.

The tree structure for BNG-UP interface configuration is as following:

```

+--rw interfaces
  | +--rw interface* [name]
  | | +--rw name      if:interface-ref
  | | +--rw ethernet
  | | | +--rw lacp?   boolean
  | | +--rw mac-offset? uint32
  | | +--rw vlans
  | | | +--rw tag* [index]
  | | | | +--rw index    uint8
  | | | | +--rw tag
  | | | | | +--rw tag-type? string
  | | | | +--rw vlan-id?   vlan-id

```

### 3.3. Control Channel Configuration

The control channel is to configure the control channel parameters. The control channel data models are configured through the management interfaces between BGN-UP and BNG-CP by netconf protocol

The control channel parameters include: name, id, port, disconnect. The tree structure for control channel configuration parameters are as following:

```

+--rw control-channel
|   +--rw address-family* [af]
|   |   +--rw af          address-family-type
|   |   +--rw control-ip?  inet:ip-address
|   +--rw name?           string
|   +--rw id?             uint32
|   +--rw port?           uint32
|   +--rw disconnect
|   |   +--rw (response-delay)?
|   |   |   +--:(nolimitflag)
|   |   |   |   +--rw forever?      enumeration
|   |   |   +--:(range)
|   |   |   +--rw delay-time?      uint32

```

### 3.4. Service Channel Configuration

The VXLAN tunnel is the suggestion service interface protocol between BNG-CP and BNG-UP. The VXLAN tunnel parameters include: tunnel-source-ip, tunnel-destination-ip, vxlan-id, vxlan-tunnel-id, vxlan-tunnel-name, etc.

```

+--rw vxlan-channel* [vxlan-tunnel-id]
|   +--rw vxlan-tunnel-id      uint32
|   +--rw vxlan-tunnel-name?   string
|   +--rw address-family* [af]
|   |   +--rw af              address-family-type
|   |   +--rw tunnel-source-ip?  inet:ip-address
|   |   +--rw tunnel-destination-ip?  inet:ip-address
|   +--rw bind-vxlan-id* [vxlan-id]
|   |   +--rw vxlan-id        vxlan-id

```

### 3.5. Multicast Service

The multicast service parameters are configured through management interfaces. Both IGMP and MLD multicast services are supported by bng. The multicast service YANG data model are only configured to BNG-CP.

```

+--rw multicast-service
|   +--rw multicast-global
|   |   +--rw keepalive-timer?    enumeration
|   |   +--rw query-interval?     uint16
|   +--rw igmp-service-profile
|   |   +--rw igmp-service-profile* [service-profile-num]
|   |   |   +--rw service-profile-num    uint8
|   |   |   +--rw access-group?          string
|   |   |   +--rw description?           string
|   |   |   +--rw max-groups?            uint8
|   |   |   +--rw max-prw-groups?        uint8
|   |   |   +--rw prw-group* [ipv4-address ipv4-address-mask]
|   |   |   |   +--rw ipv4-address        inet:ipv4-address
|   |   |   |   +--rw ipv4-address-mask   inet:ipv4-address
|   |   |   |   +--rw max-prw-count?      uint32
|   |   |   |   +--rw prw-resume-interval? uint32
|   |   |   +--rw static-group* [ipv4-address]
|   |   |   |   +--rw ipv4-address        inet:ipv4-address
|   +--rw mld-service-profile
|   |   +--rw mld-service-profile* [service-profile-num]
|   |   |   +--rw service-profile-num    uint8
|   |   |   +--rw access-group?          string
|   |   |   +--rw description?           string
|   |   |   +--rw max-groups?            uint8
|   |   |   +--rw max-prw-groups?        uint8
|   |   |   +--rw prw-group* [ipv6-address ipv6-prefixlen]
|   |   |   |   +--rw ipv6-address        inet:ipv6-address
|   |   |   |   +--rw ipv6-prefixlen      uint8
|   |   |   |   +--rw max-prw-count?      uint32
|   |   |   |   +--rw prw-resume-interval? uint32
|   |   |   +--rw static-group* [ipv6-address]
|   |   |   |   +--rw ipv6-address        inet:ipv6-address

```

### 3.6. PPPOX Configuration

The pppox parameters are only configured to BNG-CP. The parameters are as following:

```

+--rw bras-pppox
|   +--rw pppox-ipv6cp-cfg
|   |   +--rw ipv6cp-extension?    enumeration
|   |   +--rw ipv6cp-aftr?         uint8
|   |   +--rw ipv6cp-ipv6-address? uint8
|   |   +--rw ipv6cp-ipv6-prefix?  uint8
|   |   +--rw ipv6-dns!
|   |   |   +--rw ipv6cp-ipv6-dns-secondary    uint8
|   |   |   +--rw ipv6cp-ipv6-dns-primary     uint8

```

```

|   +--rw ipv4-dns!
|   |   +--rw ipv6cp-ipv4-dns-primary      uint8
|   |   +--rw ipv6cp-ipv4-dns-secondary   uint8
+--rw pppox-ipcp-cfg
|   +--rw ipcp-flag?                      enumeration
|   +--rw option-type?                    uint8
|   +--rw br-address?                     inet:ipv4-address
|   +--rw ipv6-rapid-deployment!
|   |   +--rw v4-mask-len      uint8
|   |   +--rw v6-pref         inet:ipv6-address
|   |   +--rw v6-mask-len     uint8
+--rw pppoe-switch
|   +--rw delay-time?              uint16
|   +--rw keepalive-timer?         enumeration
|   +--rw ppp-max-payload?         enumeration
|   +--rw service?                 enumeration
|   +--rw ppp-mru-verify?          enumeration
|   +--rw keepalive-fast-reply?    enumeration
+--rw pppoe-cfg* [template]
|   +--rw template                  uint32
|   +--rw ppp-authentication?       enumeration
|   +--rw ppp-check-magic-num?      enumeration
|   +--rw ppp-mru?                  uint32
|   +--rw pppoe-ac-name?            string
|   +--rw pppoe-service-name-omit?  enumeration
|   +--rw pppoe-ac-cookie-check?    enumeration
|   +--rw pppoe-password-string?    string
|   +--rw pppoe-username-string?    string
|   +--rw (ppp-quick-redial)?
|   |   +--:(quick-redial-disable)
|   |   |   +--rw ppp-quick-redial-disable?  enumeration
|   |   +--:(fast-response)
|   |   |   +--rw ppp-fast-response?          enumeration
|   |   |   +--rw ppp-quick-redial-enable?    enumeration
+--rw ppp-keepalive
|   +--rw ppp-keepalive-timer?  uint32
|   +--rw ppp-keepalive-count?  uint16
+--rw ppp-timeout
|   +--rw ppp-timeout-negtimeoutsec?  uint8
|   +--rw ppp-timeout-authentication? uint8

```

### 3.7. Acl Configuration

The acl information for BNG-UP is configured through netconf protocol from BNG-CP. The ACL information includes ipv4-acl, ipv6-acl, link-acl, etc. The YANG data model for ACL refers to [I-D.ietf-netmod-acl-model]

### 3.8. QoS Configuration

The QoS information for BNG-UP is also configured through netconf protocol from BNG-CP. The support QoS information includes IP-DSCP, MPLS, VPLS, VPWS etc. The YANG data model for QoS refers to [I-D.asechoud-rtgwg-qos-model]

## 4. vBNG YANG Data Model

```
<CODE BEGINS> file "ietf-vbng@2018-03-18.yang"
module ietf-vbng{
  namespace "urn:ietf:params:xml:ns:yang:ietf-vbng";
  prefix "vbng";

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-interfaces {
    prefix if;
  }

  import ietf-logical-network-element {
    prefix lne;
  }

  organization
    "IETF NETCONF Working Group";

  contact
    "
      WG List:  <mailto:netconf@ietf.org>

      Editor:   Fangwei Hu
                <mailto:hu.fangwei@zte.com.cn>
    ";

  description
    "The YANG module defines a generic configuration
    model for vbng";

  revision 2018-03-20{
    description "Change the control channel protocol name";
    reference
      "draft-hu-rtgwg-cu-separation-yang-model-03";
  }
}
```

```
    revision 2018-02-27{
    description "Correct some type of nodes.";
    reference
        "draft-hu-rtgwg-cu-separation-yang-model-02";
    }

    revision 2018-01-18{
    description "add multicast service configuration and pppox configuration,
        and update the OpenFlow channel parameters.";
    reference
        "draft-hu-rtgwg-cu-separation-yang-model-01";
    }

revision 2017-07-16{
    description "Initial revision";
    reference
        "draft-hu-rtgwg-cu-separation-yang-model-00";
    }

/* Typedefs */

    typedef vlan-id {
    type uint16 {
        range "0..4094";
    }
    description
        "Typedef for VLAN ID.";
    }

    typedef vxlan-id {
    type uint32;
    description
        "Typedef for VxLAN ID.";
    }

    typedef address-family-type {
    type enumeration {
        enum ipv4 {
            description
                "IPv4";
        }
        enum ipv6 {
            description
                "IPv6";
        }
    }
    description
        "Typedef for address family type.";
```



```
    }

    /* Configuration Data */

    augment /lne:logical-network-elements/lne:logical-network-element {
    container ietf-vbng{

        container bng-cp {
        leaf bng-cp-name {
            type string;
            description "configure bng-cp name";
        }

        leaf enable {
            type boolean;
            description "'true' to support bng control plane and user plane
separation";
        }
        description "configure bng-cp";
    }

    container bng-up {
        presence bng-up;
        list bng-up {
            key shelf-no ;
            leaf shelf-no {
                type uint8 {
                    range 1..127 ;
                }
                description 'Configure shelf-no of forwarder,1-127.';
            }
        }

        leaf bng-up-name {
            type string {
                length 1..31 ;
            }
            description 'Configure bng up name.' ;
        }
    }

    container netconf-server {
        presence netconf-server ;
        leaf ip {
            type inet:ipv4-address;
            mandatory true ;
            description 'Configure ip address of netconf server.';
        }

        leaf user-name {
```

```
    type string {
      length 1..65 ;
    }
    description 'configure user name, default: "who".';
  }

  leaf password {
    type string {
      length 3..32 ;
    }
    description 'configure password, default: "who".';
  }

  leaf port {
    type uint32;
    description 'Configure port.';
  }
  description 'Configure netconf server.';
}

leaf keepalive-sink {
  type enumeration {
    enum enable {
      value 1 ;
      description 'enable the keepalive-sink function';
    }
    enum disable {
      value 0 ;
      description 'disable keepalive-sink function';
    }
  }
  description "configure keepalive-sink";
}
description "configure bng up list";
}
description "configure bng up container";
}

container interfaces {
  list interface {
    key name;
    leaf name {
      type if:interface-ref;
      description "interface name";
    }
    container ethernet {
      leaf lacp {
        type boolean;
      }
    }
  }
}
```

```

        description "enable lacp function";
    }
        description "configure ethernet interface";
    }
    leaf mac-offset {
        type uint32;
        description "configure mac offset";
    }

    container vlans {
        list tag {
            key index;
            max-elements 2;

            leaf index {
                type uint8 {
                    range "0..1";
                }
                must ". = 0 or
count(.../.../tag[index = 0]/index) > 0" {
                    error-message "An inner tag can only be specified if an
                        outer tag has also been specified";
                    description "Ensure that an inner tag cannot be
                        specified without an outer tag'";
                }
            }

            description "The index into the tag stack, outermost tag
                assigned index 0";
        }

        container tag{
            leaf tag-type {
                type string;
                description "tag type";
            }
            leaf vlan-id {
                type vlan-id;
                description "vlan id value";
            }
        }

        description "tag";
    }
    description "tag list";
}
description "vlans";
}
description "interfaces list";
}
description "interface container";

```

```
    }

    container control-channel {
list address-family {
    key "af";
    leaf af {
        type address-family-type;
        description "Address family type value.";
    }
    leaf control-ip {
        type inet:ip-address;
        description
            "Set the IP address of for control channel protocol session";
    }
        description "Per-af params.";
    }

    leaf name {
        type string;
        description "control channel protocol logical name";
    }

    leaf id {
        type uint32;
        description "id value";
    }

    leaf of-port {
        type uint32;
        description "control channel udp port number";
    }

container disconnect {
    choice response-delay {
        default range ;
        case nolimitflag {
            leaf forever {
                type enumeration {
                    enum forever {
                        value 1 ;
                                description "Delay forever";
                    }
                }
            }
        }
        description 'Delay forever';
    }
    description 'The flag of no time limite';
}
    case range {
```

```
        leaf delay-time {
            type uint32 {
                range 0..2880 ;
            }
            description 'Delay time ,default 240 minutes';
        }
        description 'Set delay time range';
    }
    description 'Set delay time';
}
description 'Set delay time after control channel protocol discon
nect';
}
    description "configure control channel value";
}

list vxlan-channel{
    key vxlan-tunnel-id;
    leaf vxlan-tunnel-id {
        type uint32;
        description
            "Static VxLAN tunnel ID.";
    }

    leaf vxlan-tunnel-name {
        type string;
        description
            "Name of the static VxLAN tunnel.";
    }

    list address-family {
        key "af";
        leaf af {
            type address-family-type;
            description
                "Address family type value.";
        }

        leaf tunnel-source-ip {
            type inet:ip-address;
            description
                "Source IP address for the static VxLAN tunnel";
        }

        leaf tunnel-destination-ip {
            type inet:ip-address;
            description
                "Destination IP address for the static VxLAN tunnel";
        }
    }
}
```

```
        list bind-vxlan-id {
            key vxlan-id;
            leaf vxlan-id {
                type vxlan-id;
                description
                    "VxLAN ID.";
            }
            description
                "VxLAN ID list for the VTEP.";
        }

        description
            "Per-af params.";
    }
    description
        "Configure VxLAN channel";
}

container multicast-service{
    container multicast-global {
    leaf keepalive-timer {
        type enumeration {
            enum start {
                value 1 ;
                description 'open switch';
            }
            enum stop {
                value 2 ;
                description 'close switch';
            }
            enum always {
                value 3 ;
                description 'always keepalive';
            }
        }
        default start ;
        description 'the switch of sending keepalive packet';
    }
    leaf query-interval {
        type uint16 {
            range 1..65535 ;
        }
        default 125 ;
        description 'multicast query interval';
    }
    description 'multicast global configuration';
}
```

```
container igmp-service-profile {
  list igmp-service-profile {
    key service-profile-num ;
    leaf service-profile-num {
      type uint8 {
        range 1..100 ;
      }
      description 'service profile number';
    }
    leaf access-group {
      type string {
        length 1..31 ;
      }
      description 'acl name';
    }

    leaf description {
      type string {
        length 1..31 ;
      }
      description 'description of service profile';
    }

    leaf max-groups {
      type uint8 {
        range 1..128 ;
      }
      default 10 ;
      description 'max groups of the service profile';
    }

    leaf max-prw-groups {
      type uint8 {
        range 1..128 ;
      }
      default 10 ;
      description 'max preview groups of the service profile';
    }

    list prw-group {
      key 'ipv4-address ipv4-address-mask';
      leaf ipv4-address {
        type inet:ipv4-address ;
        description 'ipv4 address of the preview group';
      }

      leaf ipv4-address-mask {
        type inet:ipv4-address;
      }
    }
  }
}
```

```
        description 'ipv4 mask of the preview group';
    }

    leaf max-prw-count {
        type uint32 {
            range 1..1800 ;
        }
        default 10 ;
        description 'max preview times';
    }

    leaf prw-resume-interval {
        type uint32 {
            range 1..86400 ;
        }
        default 60 ;
        description 'preview interval';
    }
    description 'configure preview group';
}

list static-group {
    key ipv4-address ;
    leaf ipv4-address {
        type inet:ipv4-address ;
        description 'ipv4 address of the static group';
    }
    description 'configure static group';
}
description 'configuration of igmp service profile';
}
description 'configuration of igmp service profile';
}

container mld-service-profile {
    list mld-service-profile {
        key service-profile-num ;
        leaf service-profile-num {
            type uint8 {
                range 1..100 ;
            }
            description 'service profile number';
        }
        leaf access-group {
            type string {
                length 1..31 ;
            }
            description 'acl name';
        }
    }
}
```



```
}
leaf description {
    type string {
        length 1..31 ;
    }
    description 'description of service profile';
}
leaf max-groups {
    type uint8 {
        range 1..128 ;
    }
    default 10 ;
    description 'max groups of the service profile';
}
leaf max-prw-groups {
    type uint8 {
        range 1..128 ;
    }
    default 10 ;
    description 'max preview groups of the service profile';
}
list prw-group {
    key 'ipv6-address ipv6-prefixlen'
    ;
    leaf ipv6-address {
        type inet:ipv6-address ;
        description 'ipv6 address of the preview group';
    }
    leaf ipv6-prefixlen {
        type uint8 {
            range 1..128 ;
        }
        description 'ipv6 prefix length';
    }
    leaf max-prw-count {
        type uint32 {
            range 1..1800 ;
        }
        default 10 ;
        description 'max preview times';
    }
    leaf prw-resume-interval {
        type uint32 {
            range 1..86400 ;
        }
        default 60 ;
    }
}
```

```
        description 'preview interval';
    }
    description 'configure preview group';
}
list static-group {
    key ipv6-address ;
    leaf ipv6-address {
        type inet:ipv6-address;
        description 'ipv6 address of the static group';
    }
    description 'configure static group';
}
description 'configuration of mld service profile';
}
description 'configuration of mld service profile';
}
    description 'multicast service configuration';
}

container bras-pppox {
    container pppox-ipv6cp-cfg {
        leaf ipv6cp-extension {
            type enumeration {
                enum enable {
                    value 1 ;
                    description "enable the IPv6cp extension!";
                }
                enum disable {
                    value 0 ;
                    description "disable the IPv6cp extension!";
                }
            }
            default disable ;
            description 'Ipv6cp extension flag';
        }
        leaf ipv6cp-aftr {
            type uint8 {
                range 1..255 ;
            }
            description 'AFTR option type value';
        }
        leaf ipv6cp-ipv6-address {
            type uint8 {
                range 1..255 ;
            }
        }
    }
}
```

```
        description 'Ipv6 address option type value';
    }
    leaf ipv6cp-ipv6-prefix {
        type uint8 {
            range 1..255 ;
        }
        description 'Ipv6 prefix option type value';
    }
    container ipv6-dns {
        presence ipv6-dns ;
        leaf ipv6cp-ipv6-dns-secondary {
            type uint8 {
                range 1..255 ;
            }
            mandatory true ;
            description 'IPV6 primary DNS option type value';
        }
        leaf ipv6cp-ipv6-dns-primary {
            type uint8 {
                range 1..255 ;
            }
            mandatory true ;
            description 'IPV6 secondary DNS option type value';
        }
        description 'Ipv6 DNS option type value';
    }
    container ipv4-dns {
        presence ipv4-dns ;
        leaf ipv6cp-ipv4-dns-primary {

            type uint8 {
                range 1..255 ;
            }
            mandatory true ;
            description 'IPV4 primary DNS option type value';
        }
        leaf ipv6cp-ipv4-dns-secondary {

            type uint8 {
                range 1..255 ;
            }
            mandatory true ;
            description 'IPV4 secondary DNS option type value';
        }
        description 'Ipv4 DNS option type value';
    }
    description 'Configuration about IPV6CP extension.';
}
```

[illegible]

```
    }
    description '6RD BR IPv4 address';
  }
  container ipv6-rapid-deployment {
    presence ipv6-rapid-deployment ;
    leaf v4-mask-len {
      type uint8 {
        range 0..32 ;
      }
      mandatory true ;
      description 'IPv4 address mask length';
    }
    leaf v6-pref {
      type inet:ipv6-address ;
      mandatory true ;
      description 'IPv6 prefix';
    }
    leaf v6-mask-len {
      type uint8 {
        range 1..128 ;
      }
      mandatory true ;
      description 'IPv6 prefix length';
    }
    description 'Ipv6 rapid deployment';
  }
  description 'Configuration about IPCP extension.';
}
container pppoe-switch {
  leaf delay-time {
    type uint16 {
      range 1..300 ;
    }
    description 'Trigger user offline when VCC phys-interface down';
  }
  leaf keepalive-timer {
    type enumeration {
      enum start {
        value 1 ;
        description "start keepalive timer";
      }
      enum stop {
        value 0 ;
        description "stop keepalive timer";
      }
    }
    default start ;
    description 'Start or stop send keepalive packet';
  }
}
```

```
}
leaf ppp-max-payload {
  type enumeration {
    enum disable {
      value 0 ;
      description "disable ppp max payload";
    }
    enum enable {
      value 1 ;
      description "enable ppp max payload";
    }
  }
  default disable ;
  description 'Enable or disable pppoe ppp-max-payload';
}
leaf service {
  type enumeration {
    enum advertise {
      value 1 ;
      description "enable ppp service!";
    }
    enum disable {
      value 0 ;
      description "disable ppp service!";
    }
  }
  default advertise ;
  description 'Open or close pppoe service';
}
leaf ppp-mru-verify {
  type enumeration {
    enum open {
      value 1 ;
      description "enable ppp mru verify!";
    }
    enum close {
      value 0 ;
      description "disable ppp mru!";
    }
  }
  default close ;
  description 'set ppp lcp mru verify when mru over 1492';
}

leaf keepalive-fast-reply {
  type enumeration {
    enum enable {
      value 1 ;
```

```
        description 'Enable keepalive fast reply!';
    }
    enum disable {
        value 0 ;
        description 'Disable keepalive fast reply!';
    }
}
description 'Set keepalive fast reply flag.';
}
description 'Configuration about pppoe switch.';
}
list pppoe-cfg {
    key template ;
    leaf template {
        type uint32 {
            range 1..1000 ;
        }
        description 'PPPoX template number';
    }
}
leaf ppp-authentication {
    type enumeration {
        enum pap {
            value 1 ;
            description "configure pap authentication!";
        }
        enum chap {
            value 2 ;
            description "configure chap authentication!";
        }
        enum mschapv1 {
            value 6 ;
            description "configure mschapv1 authentication!";
        }
        enum mschapv2 {
            value 7 ;
            description "configure mschapv2 authentication!";
        }
        enum pap-chap {
            value 21 ;
            description "configure pap-chap authentication!";
        }
    }
}
default pap-chap ;
description 'Set ppp authentication';
}
leaf ppp-check-magic-num {
    type enumeration {
        enum disable {
```

```
        value 0 ;
        description 'disable ppp magic check';
    }
    enum enable {
        value 1 ;
        description 'enable ppp magic check';
    }
}
default enable ;
description 'Check magic number or not';
}
leaf ppp-mru {
    type uint32 {
        range 320..9000 ;
    }
    default 1492 ;
    description 'Set mru value';
}
leaf pppoe-ac-name {
    type string ;
    description 'Set ac-name';
}
leaf pppoe-service-name-omit {
    type enumeration {
        enum disable {
            value 0 ;
            description "disable pppoe service name omit";
        }
        enum enable {
            value 1 ;
            description "enable pppoe service name omit";
        }
    }
    default disable ;
    description 'Check service-name value';
}
leaf pppoe-ac-cookie-check {
    type enumeration {
        enum disable {
            value 0 ;
            description "disable pppoe ac cookie check";
        }
        enum enable {
            value 1 ;
            description "enable pppoe ac cookie check";
        }
    }
}
```



```
    default enable ;
        description 'Check options';
    }
    leaf pppoe-password-string {
        type string ;
        description 'Set authen fail password error string';
    }
    leaf pppoe-username-string {
        type string ;
        description 'Set authen fail username error string';
    }
}

choice ppp-quick-redial {
    case quick-redial-disable {
        leaf ppp-quick-redial-disable {
            type enumeration {
                enum disable {
                    value 0 ;
                    description "disable ppp quick redial";
                }
            }
        }
        default disable ;
        description 'disable quick-redial';
    }
    description 'disable quick-redial';
}

case fast-response {
    leaf ppp-fast-response {
        type enumeration {
            enum diable {
                value 0 ;
                description "disable ppp fast response";
            }
            enum enable {
                value 1 ;
                description "enable ppp fast response";
            }
        }
        description 'set Response the access request immediately';
    }
    leaf ppp-quick-redial-enable {
        type enumeration {
            enum enable {
                value 1 ;
                description "enable ppp quick redial";
            }
        }
        default enable ;
    }
}
```

```

        description 'Enable quick-redial';
    }
    description 'set quick-redial or Response the access request immediately';
}
    default quick-redial-disable ;
    description 'Enable or disable quick-redial';
}
container ppp-keepalive {
    leaf ppp-keepalive-timer {
        type uint32 {
            range 10..14400 ;
        }
        default 60 ;
        description 'Set keepalive time(unit:seconds)';
    }
    leaf ppp-keepalive-count {
        type uint16 {
            range 1..10 ;
        }
        default 3 ;
        description 'Set keepalive counter';
    }
    description 'Set keepalive time and counter';
}
container ppp-timeout {
    leaf ppp-timeout-negtimeoutsec {
        type uint8 {
            range 1..10 ;
        }
        default 3 ;
        description 'Set ppp negtimeoutsec timeout(unit:seconds)';
    }
    leaf ppp-timeout-authentication {
        type uint8 {
            range 1..10 ;
        }
        default 3 ;
        description 'Set ppp authentication timeout(unit:seconds)';
    }
    description 'Set ppp negtimeoutsec and authentication timeout';
}
    description 'Configuration pppoe template';
}
    description 'Configuration vBRAS PPPoX.';
}
    description "ietf-bng configuration!";
}
description "augment lne model";

```

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

## 5. 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 [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

vBNG represents device and network configuration information based on the LNE. As such, the security of this information is important, but it is fundamentally no different than any other interface or device configuration information that has already been covered in other documents such as [I-D.ietf-rtgwg-lne-model].

The vulnerable "config true" parameters and subtree are the following:

/lne:logical-network-elements/lne:logical-network-element/ietf-vbng/bng-cp: this subtree specifies the global configuration of bng control plane. Modify the configuration can cause the bng control plane instance disabled.

/lne:logical-network-elements/lne:logical-network-element/ietf-vbng/bng-up: this subtree specifies the global configuration of BNG user plane. Modify the configuration can cause the BNG user plane instance disabled.

/lne:logical-network-elements/lne:logical-network-element/ietf-vbng/control-channel: this subtree specifies the configuration control channel parameters among bng user planes and control plane. Modify the configuration can cause the control channel and control channel protocol interrupted.

/lne:logical-network-elements/lne:logical-network-element/ietf-vbng/vxlan-channel: this subtree specifies the configuration VXLAN channel parameters among BNG user planes and control plane. Modify the configuration can cause the VXLAN channel interrupted.

Unauthorized access to any of these lists can adversely affect the security of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

## 6. Acknowledgements

## 7. 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-vbng.

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-vbng
namespace:	urn:ietf:params:xml:ns:yang:ietf-vbng
prefix:	vbng
reference:	RFC XXXX

## 8. References

### 8.1. Normative References

[I-D.asechoud-rtgwg-qos-model]

Choudhary, A., Jethanandani, M., Strahle, N., Aries, E., and I. Chen, "YANG Model for QoS", draft-asechoud-rtgwg-qos-model-05 (work in progress), March 2018.

[I-D.ietf-netmod-acl-model]

Jethanandani, M., Huang, L., Agarwal, S., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-18 (work in progress), March 2018.

[I-D.ietf-rtgwg-lne-model]

Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Model for Logical Network Elements", draft-ietf-rtgwg-lne-model-09 (work in progress), March 2018.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [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>>.
- [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>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [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>>.

## 8.2. Informative References

- [I-D.huang-nvo3-vxlan-extension-for-vbras]  
Huang, L. and S. Hu, "VxLAN Extension Requirement for Signaling Exchange Between Control and User Plane of vBras", draft-huang-nvo3-vxlan-extension-for-vbras-00 (work in progress), March 2017.

Authors' Addresses

Fangwei Hu  
ZTE Corporation  
No.889 Bibo Rd  
Shanghai 201203  
China

Phone: +86 21 68896273  
Email: hu.fangwei@zte.com.cn

RongRong Hua  
ZTE Corporation  
No.50 Software Avenue,Yuhuatai District  
Nanjing, Jiangsu Province 210012  
China

Email: hua.rongrong@zte.com.cn

Shujun Hu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing 100053  
China

Email: shujun\_hu@outlook.com

Rong Gu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: gurong\_cmcc@outlook.com

Routing Working Group  
Internet-Draft  
Intended status: Informational  
Expires: February 1, 2020

F. Baker  
  
C. Bowers  
Juniper Networks  
J. Linkova  
Google  
July 31, 2019

Enterprise Multihoming using Provider-Assigned IPv6 Addresses without  
Network Prefix Translation: Requirements and Solutions  
draft-ietf-rtgwg-enterprise-pa-multihoming-12

Abstract

Connecting an enterprise site to multiple ISPs over IPv6 using provider-assigned addresses is difficult without the use of some form of Network Address Translation (NAT). Much has been written on this topic over the last 10 to 15 years, but it still remains a problem without a clearly defined or widely implemented solution. Any multihoming solution without NAT requires hosts at the site to have addresses from each ISP and to select the egress ISP by selecting a source address for outgoing packets. It also requires routers at the site to take into account those source addresses when forwarding packets out towards the ISPs.

This document examines currently available mechanisms for providing a solution to this problem for a broad range of enterprise topologies. It covers the behavior of routers to forward traffic taking into account source address, and it covers the behavior of hosts to select appropriate default source addresses. It also covers any possible role that routers might play in providing information to hosts to help them select appropriate source addresses. In the process of exploring potential solutions, this document also makes explicit requirements for how the solution would be expected to behave from the perspective of an enterprise site network administrator.

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 1, 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 . . . . .	4
2. Requirements Language . . . . .	6
3. Terminology . . . . .	6
4. Enterprise Multihoming Use Cases . . . . .	8
4.1. Simple ISP Connectivity with Connected SERs . . . . .	8
4.2. Simple ISP Connectivity Where SERs Are Not Directly Connected . . . . .	10
4.3. Enterprise Network Operator Expectations . . . . .	12
4.4. More complex ISP connectivity . . . . .	14
4.5. ISPs and Provider-Assigned Prefixes . . . . .	16
4.6. Simplified Topologies . . . . .	17
5. Generating Source-Prefix-Scoped Forwarding Tables . . . . .	17
6. Mechanisms For Hosts To Choose Good Default Source Addresses In A Multihomed Site . . . . .	24
6.1. Default Source Address Selection Algorithm on Hosts . . . .	26
6.2. Selecting Default Source Address When Both Uplinks Are Working . . . . .	29
6.2.1. Distributing Default Address Selection Policy Table with DHCPv6 . . . . .	29
6.2.2. Controlling Default Source Address Selection With Router Advertisements . . . . .	30
6.2.3. Controlling Default Source Address Selection With ICMPv6 . . . . .	32
6.2.4. Summary of Methods For Controlling Default Source	



Address Selection To Implement Routing Policy . . . .	33
6.3. Selecting Default Source Address When One Uplink Has Failed . . . . .	34
6.3.1. Controlling Default Source Address Selection With DHCPv6 . . . . .	35
6.3.2. Controlling Default Source Address Selection With Router Advertisements . . . . .	36
6.3.3. Controlling Default Source Address Selection With ICMPv6 . . . . .	37
6.3.4. Summary Of Methods For Controlling Default Source Address Selection On The Failure Of An Uplink . . . .	38
6.4. Selecting Default Source Address Upon Failed Uplink Recovery . . . . .	38
6.4.1. Controlling Default Source Address Selection With DHCPv6 . . . . .	38
6.4.2. Controlling Default Source Address Selection With Router Advertisements . . . . .	39
6.4.3. Controlling Default Source Address Selection With ICMP . . . . .	39
6.4.4. Summary Of Methods For Controlling Default Source Address Selection Upon Failed Uplink Recovery . . . .	40
6.5. Selecting Default Source Address When All Uplinks Failed	40
6.5.1. Controlling Default Source Address Selection With DHCPv6 . . . . .	40
6.5.2. Controlling Default Source Address Selection With Router Advertisements . . . . .	40
6.5.3. Controlling Default Source Address Selection With ICMPv6 . . . . .	41
6.5.4. Summary Of Methods For Controlling Default Source Address Selection When All Uplinks Failed . . . . .	41
6.6. Summary Of Methods For Controlling Default Source Address Selection . . . . .	41
6.7. Solution Limitations . . . . .	43
6.7.1. Connections Preservation . . . . .	43
6.8. Other Configuration Parameters . . . . .	44
6.8.1. DNS Configuration . . . . .	44
7. Deployment Considerations . . . . .	45
7.1. Deploying SADR Domain . . . . .	46
7.2. Hosts-Related Considerations . . . . .	46
8. Other Solutions . . . . .	47
8.1. Shim6 . . . . .	47
8.2. IPv6-to-IPv6 Network Prefix Translation . . . . .	47
8.3. Multipath Transport . . . . .	47
9. IANA Considerations . . . . .	48
10. Security Considerations . . . . .	48
11. Acknowledgements . . . . .	49
12. References . . . . .	49
12.1. Normative References . . . . .	49

12.2. Informative References . . . . .	51
Authors' Addresses . . . . .	52

## 1. Introduction

Site multihoming, the connection of a subscriber network to multiple upstream networks using redundant uplinks, is a common enterprise architecture for improving the reliability of its Internet connectivity. If the site uses provider-independent (PI) addresses, all traffic originating from the enterprise can use source addresses from the PI address space. Site multihoming with PI addresses is commonly used with both IPv4 and IPv6, and does not present any new technical challenges.

It may be desirable for an enterprise site to connect to multiple ISPs using provider-assigned (PA) addresses, instead of PI addresses. Multihoming with provider-assigned addresses is typically less expensive for the enterprise relative to using provider-independent addresses as it does not require obtaining and maintaining PI address space as well as running BGP between the enterprise and the ISPs (for small/medium networks running BGP might be not just undesirable but impossible, especially if residential-type ISP connections are used). PA multihoming is also a practice that should be facilitated and encouraged because it does not add to the size of the Internet routing table, whereas PI multihoming does. Note that PA is also used to mean "provider-aggregatable". In this document we assume that provider-assigned addresses are always provider-aggregatable.

With PA multihoming, for each ISP connection, the site is assigned a prefix from within an address block allocated to that ISP by its National or Regional Internet Registry. In the simple case of two ISPs (ISP-A and ISP-B), the site will have two different prefixes assigned to it (prefix-A and prefix-B). This arrangement is problematic. First, packets with the "wrong" source address may be dropped by one of the ISPs. In order to limit denial of service attacks using spoofed source addresses, BCP38 [RFC2827] recommends that ISPs filter traffic from customer sites to only allow traffic with a source address that has been assigned by that ISP. So a packet sent from a multihomed site on the uplink to ISP-B with a source address in prefix-A may be dropped by ISP-B.

However, even if ISP-B does not implement BCP38 or ISP-B adds prefix-A to its list of allowed source addresses on the uplink from the multihomed site, two-way communication may still fail. If the packet with source address in prefix-A was sent to ISP-B because the uplink to ISP-A failed, then if ISP-B does not drop the packet and the packet reaches its destination somewhere on the Internet, the return packet will be sent back with a destination address in prefix-

A. The return packet will be routed over the Internet to ISP-A, but it will not be delivered to the multihomed site because the site uplink with ISP-A has failed. Two-way communication would require some arrangement for ISP-B to advertise prefix-A when the uplink to ISP-A fails.

Note that the same may be true with a provider that does not implement BCP 38, if his upstream provider does, or has no corresponding route to deliver the ingress traffic to the multihomed site. The issue is not that the immediate provider implements ingress filtering; it is that someone upstream does (so egress traffic is blocked), or lacks a route (causing blackholing of the ingress traffic).

Another issue with asymmetric traffic flow (when the egress traffic leaves the site via one ISP but the return traffic enters the site via another uplink) is related to stateful firewalls/middleboxes. Keeping state in that case might be problematic, even impossible.

With IPv4, this problem is commonly solved by using [RFC1918] private address space within the multi-homed site and Network Address Translation (NAT) or Network Address/Port Translation (NAPT) on the uplinks to the ISPs. However, one of the goals of IPv6 is to eliminate the need for and the use of NAT or NAPT. Therefore, requiring the use of NAT or NAPT for an enterprise site to multihome with provider-assigned addresses is not an attractive solution.

[RFC6296] describes a translation solution specifically tailored to meet the requirements of multi-homing with provider-assigned IPv6 addresses. With the IPv6-to-IPv6 Network Prefix Translation (NPTv6) solution, within the site an enterprise can use Unique Local Addresses [RFC4193] or the prefix assigned by one of the ISPs. As traffic leaves the site on an uplink to an ISP, the source address gets translated to an address within the prefix assigned by the ISP on that uplink in a predictable and reversible manner. [RFC6296] is currently classified as Experimental, and it has been implemented by several vendors. See Section 8.2, for more discussion of NPTv6.

This document defines routing requirements for enterprise multihoming. This document focuses on the following general class of solutions.

Each host at the enterprise has multiple addresses, at least one from each ISP-assigned prefix. Each host, as discussed in Section 6.1 and [RFC6724], is responsible for choosing the source address applied to each packet it sends. A host is expected to be able respond dynamically to the failure of an uplink to a given ISP by no longer sending packets with the source address corresponding to that ISP. Potential mechanisms for the communication of changes in the network

to the host are Neighbor Discovery Router Advertisements ([RFC4861]), DHCPv6 ([RFC8415]), and ICMPv6 ([RFC4443]).

The routers in the enterprise network are responsible for ensuring that packets are delivered to the "correct" ISP uplink based on source address. This requires that at least some routers in the site network are able to take into account the source address of a packet when deciding how to route it. That is, some routers must be capable of some form of Source Address Dependent Routing (SADR), if only as described in the section 4.3 of [RFC3704]. At a minimum, the routers connected to the ISP uplinks (the site exit routers or SERs) must be capable of Source Address Dependent Routing. Expanding the connected domain of routers capable of SADR from the site exit routers deeper into the site network will generally result in more efficient routing of traffic with external destinations.

This document is organized as follows. Section 4 looks in more detail at the enterprise networking environments in which this solution is expected to operate. The discussion of Section 4 uses the concepts of source-prefix-scoped routing advertisements and forwarding tables and provides a description of how source-prefix-scoped routing advertisements are used to generate source-prefix-scoped forwarding tables. Instead, this detailed description is provided in Section 5. Section 6 discusses existing and proposed mechanisms for hosts to select the default source address to be used by applications. It also discusses the requirements for routing that are needed to support these enterprise network scenarios and the mechanisms by which hosts are expected to update default source addresses based on network state. Section 7 discusses deployment considerations, while Section 8 discusses other solutions.

## 2. Requirements Language

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

## 3. Terminology

PA (provider-assigned or provider-aggregatable) address space: a block of IP addresses assigned by an Regional Internet Registry (RIR) to a Local Internet Registry (LIR), used to create allocations to end sites. Can be aggregated and present in the routing table as one route.

PI (provider-independent) address space: a block of IP addresses assigned by an Regional Internet Registry (RIR) directly to end site/end customer.

ISP: Internet Service Provider.

LIR (Local Internet Registry): an organisation (usually an ISP or an enterprise/academic) which receives IP addresses allocation from its Regional Internet Registry, then assign parts of that allocation to its customers.

RIR (Regional Internet Registry): an organization which manages the Internet number resources (such as IP addresses and AS numbers) within a geographical region of the world.

SADR (Source Address Dependent Routing): Routing which takes into account the source address of a packet in addition to the packet destination address.

SADR domain: a routing domain where some (or all) routers exchange source-dependent routing information.

Source-Prefix-Scoped Routing/Forwarding Table: a routing (or forwarding) table which contains routing (or forwarding) information which is applicable to packets with source addresses from the specific prefix only.

Unscoped Routing/Forwarding Table: a routing (or forwarding) table which can be used to route/forward packets with any source addresses.

SER (Site Edge Router): a router which connects the site to an ISP (terminates an ISP uplink)..

LLA (Link-Local Address): IPv6 Unicast Address from fe80::/10 prefix ([RFC4291]).

ULA (Unique Local IPv6 Unicast Address): IPv6 unicast addresses from FC00::/7 prefix. They are globally unique and intended for local communications ([RFC4193]).

GUA (Global Unicast Address): globally routable IPv6 addresses of the global scope ([RFC4291]).

SLAAC (IPv6 Stateless Address Autoconfiguration): a stateless process of configuring network stack on IPv6 hosts ([RFC4862]).

RA (Router Advertisement): a message sent by an IPv6 router to advertise its presence to hosts together with various network-related parameters required for hosts to perform SLAAC ([RFC4861]).

PIO (Prefix Information Option): a part of RA message containing information about IPv6 prefixes which could be used by hosts to generate global IPv6 addresses ([RFC4862]).

RIO (Route Information Option): a part of RA message containing information about more specific IPv6 prefixes reachable via the advertising router ([RFC4191]).

#### 4. Enterprise Multihoming Use Cases

##### 4.1. Simple ISP Connectivity with Connected SERs

We start by looking at a scenario in which a site has connections to two ISPs, as shown in Figure 1. The site is assigned the prefix 2001:db8:0:a000::/52 by ISP-A and prefix 2001:db8:0:b000::/52 by ISP-B. We consider three hosts in the site. H31 and H32 are on a LAN that has been assigned subnets 2001:db8:0:a010::/64 and 2001:db8:0:b010::/64. H31 has been assigned the addresses 2001:db8:0:a010::31 and 2001:db8:0:b010::31. H32 has been assigned 2001:db8:0:a010::32 and 2001:db8:0:b010::32. H41 is on a different subnet that has been assigned 2001:db8:0:a020::/64 and 2001:db8:0:b020::/64.

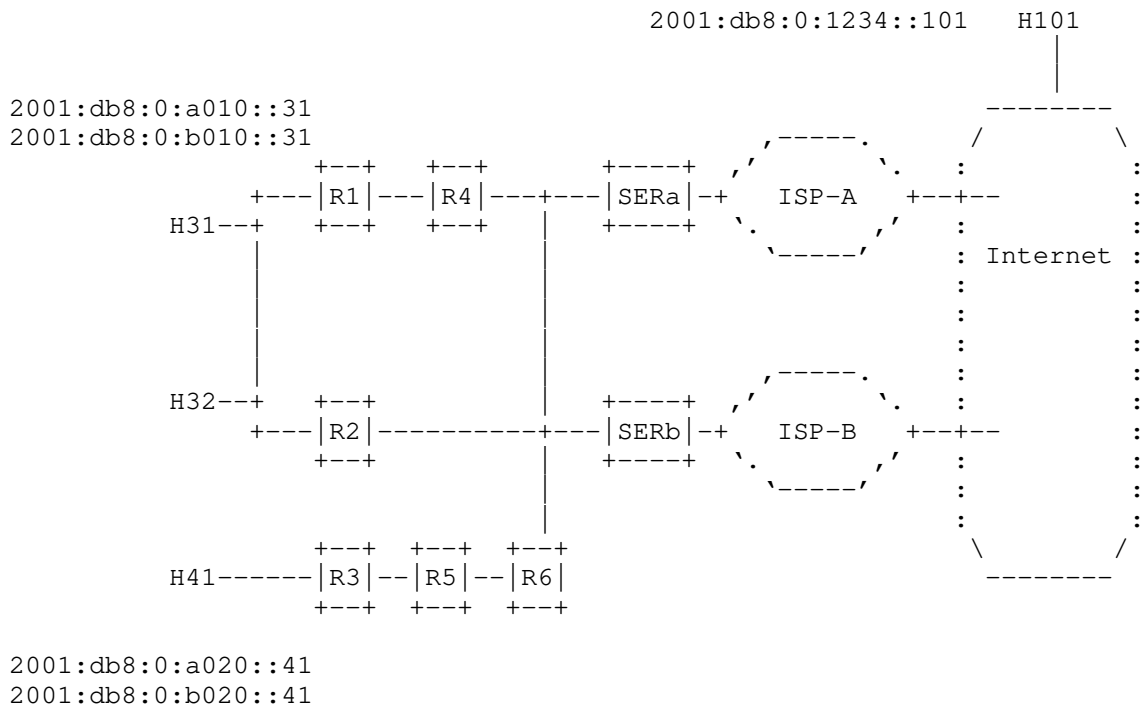


Figure 1: Simple ISP Connectivity With Connected SERs

We refer to a router that connects the site to an ISP as a site edge router (SER). Several other routers provide connectivity among the internal hosts (H31, H32, and H41), as well as connecting the internal hosts to the Internet through SERa and SERb. In this example SERa and SERb share a direct connection to each other. In Section 4.2, we consider a scenario where this is not the case.

For the moment, we assume that the hosts are able to make good choices about which source addresses through some mechanism that doesn't involve the routers in the site network. Here, we focus on primary task of the routed site network, which is to get packets efficiently to their destinations, while sending a packet to the ISP that assigned the prefix that matches the source address of the packet. In Section 6, we examine what role the routed network may play in helping hosts make good choices about source addresses for packets.

With this solution, routers will need some form of Source Address Dependent Routing, which will be new functionality. It would be useful if an enterprise site does not need to upgrade all routers to

support the new SADR functionality in order to support PA multi-homing. We consider if this is possible and what are the tradeoffs of not having all routers in the site support SADR functionality.

In the topology in Figure 1, it is possible to support PA multihoming with only SERa and SERb being capable of SADR. The other routers can continue to forward based only on destination address, and exchange routes that only consider destination address. In this scenario, SERa and SERb communicate source-scoped routing information across their shared connection. When SERa receives a packet with a source address matching prefix 2001:db8:0:b000::/52, it forwards the packet to SERb, which forwards it on the uplink to ISP-B. The analogous behaviour holds for traffic that SERb receives with a source address matching prefix 2001:db8:0:a000::/52.

In Figure 1, when only SERa and SERb are capable of source address dependent routing, PA multi-homing will work. However, the paths over which the packets are sent will generally not be the shortest paths. The forwarding paths will generally be more efficient as more routers are capable of SADR. For example, if R4, R2, and R6 are upgraded to support SADR, then can exchange source-scoped routes with SERa and SERb. They will then know to send traffic with a source address matching prefix 2001:db8:0:b000::/52 directly to SERb, without sending it to SERa first.

#### 4.2. Simple ISP Connectivity Where SERs Are Not Directly Connected

In Figure 2, we modify the topology slightly by inserting R7, so that SERa and SERb are no longer directly connected. With this topology, it is not enough to just enable SADR routing on SERa and SERb to support PA multi-homing. There are two solutions to enable PA multihoming in this topology.



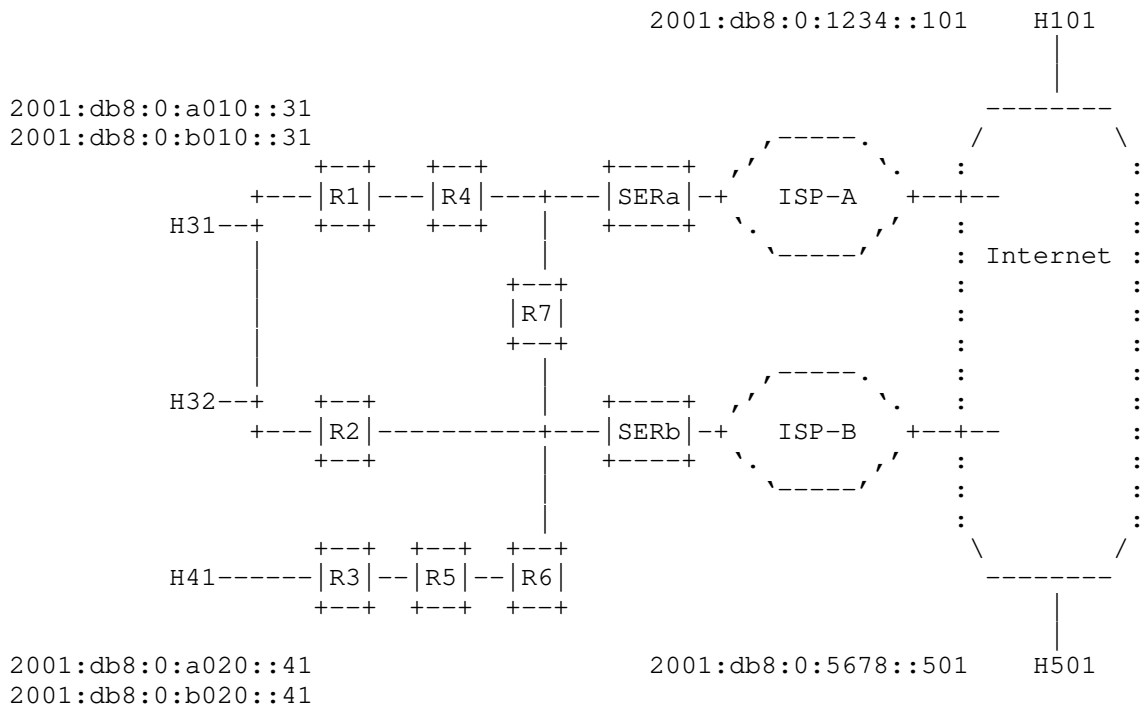


Figure 2: Simple ISP Connectivity Where SERs Are Not Directly Connected

One option is to effectively modify the topology by creating a logical tunnel between SERa and SERb, using GRE ([RFC7676]) for example. Although SERa and SERb are not directly connected physically in this topology, they can be directly connected logically by a tunnel.

The other option is to enable SADR functionality on R7. In this way, R7 will exchange source-scoped routes with SERa and SERb, making the three routers act as a single SADR domain. This illustrates the basic principle that the minimum requirement for the routed site network to support PA multi-homing is having all of the site exit routers be part of a connected SADR domain. Extending the connected SADR domain beyond that point can produce more efficient forwarding paths.

#### 4.3. Enterprise Network Operator Expectations

Before considering a more complex scenario, let's look in more detail at the reasonably simple multihoming scenario in Figure 2 to understand what can reasonably be expected from this solution. As a general guiding principle, we assume an enterprise network operator will expect a multihomed network to behave as close as to a single-homed network as possible. So a solution that meets those expectations where possible is a good thing.

For traffic between internal hosts and traffic from outside the site to internal hosts, an enterprise network operator would expect there be no visible change in the path taken by this traffic, since this traffic does not need to be routed in a way that depends on source address. It is also reasonable to expect that internal hosts should be able to communicate with each other using either of their source addresses without restriction. For example, H31 should be able to communicate with H41 using a packet with S=2001:db8:0:a010::31, D=2001:db8:0:b020::41, regardless of the state of uplink to ISP-B.

These goals can be accomplished by having all of the routers in the network continue to originate normal unscoped destination routes for their connected networks. If we can arrange so that these unscoped destination routes get used for forwarding this traffic, then we will have accomplished the goal of keeping forwarding of traffic destined for internal hosts, unaffected by the multihoming solution.

For traffic destined for external hosts, it is reasonable to expect that traffic with a source address from the prefix assigned by ISP-A to follow the path to that the traffic would follow if there is no connection to ISP-B. This can be accomplished by having SERa originate a source-scoped route of the form (S=2001:db8:0:a000::/52, D=::/0) . If all of the routers in the site support SADR, then the path of traffic exiting via ISP-A can match that expectation. If some routers don't support SADR, then it is reasonable to expect that the path for traffic exiting via ISP-A may be different within the site. This is a tradeoff that the enterprise network operator may decide to make.

It is important to understand how this multihoming solution behaves when an uplink to one of the ISPs fails. To simplify this discussion, we assume that all routers in the site support SADR. We first start by looking at how the network operates when the uplinks to both ISP-A and ISP-B are functioning properly. SERa originates a source-scoped route of the form (S=2001:db8:0:a000::/52, D=::/0), and SERb is originates a source-scoped route of the form (S=2001:db8:0:b000::/52, D=::/0). These routes are distributed through the routers in the site, and they establish within the

routers two set of forwarding paths for traffic leaving the site. One set of forwarding paths is for packets with source address in 2001:db8:0:a000::/52. The other set of forwarding paths is for packets with source address in 2001:db8:0:b000::/52. The normal destination routes which are not scoped to these two source prefixes play no role in the forwarding. Whether a packet exits the site via SERa or via SERb is completely determined by the source address applied to the packet by the host. So for example, when host H31 sends a packet to host H101 with (S=2001:db8:0:a010::31, D=2001:db8:0:1234::101), the packet will only be sent out the link from SERa to ISP-A.

Now consider what happens when the uplink from SERa to ISP-A fails. The only way for the packets from H31 to reach H101 is for H31 to start using the source address for ISP-B. H31 needs to send the following packet: (S=2001:db8:0:b010::31, D=2001:db8:0:1234::101).

This behavior is very different from the behavior that occurs with site multihoming using PI addresses or with PA addresses using NAT. In these other multi-homing solutions, hosts do not need to react to network failures several hops away in order to regain Internet access. Instead, a host can be largely unaware of the failure of an uplink to an ISP. When multihoming with PA addresses and NAT, existing sessions generally need to be re-established after a failure since the external host will receive packets from the internal host with a new source address. However, new sessions can be established without any action on the part of the hosts. Multihoming with PA addresses and NAT has created the expectation of a fairly quick and simple recovery from network failures. Alternatives should to be evaluated in terms of the speed and complexity of the recovery mechanism.

Another example where the behavior of this multihoming solution differs significantly from that of multihoming with PI address or with PA addresses using NAT is in the ability of the enterprise network operator to route traffic over different ISPs based on destination address. We still consider the fairly simple network of Figure 2 and assume that uplinks to both ISPs are functioning. Assume that the site is multihomed using PA addresses and NAT, and that SERa and SERb each originate a normal destination route for D=::/0, with the route origination dependent on the state of the uplink to the respective ISP.

Now suppose it is observed that an important application running between internal hosts and external host H101 experience much better performance when the traffic passes through ISP-A (perhaps because ISP-A provides lower latency to H101.) When multihoming this site with PI addresses or with PA addresses and NAT, the enterprise

network operator can configure SERa to originate into the site network a normal destination route for D=2001:db8:0:1234::/64 (the destination prefix to reach H101) that depends on the state of the uplink to ISP-A. When the link to ISP-A is functioning, the destination route D=2001:db8:0:1234::/64 will be originated by SERa, so traffic from all hosts will use ISP-A to reach H101 based on the longest destination prefix match in the route lookup.

Implementing the same routing policy is more difficult with the PA multihoming solution described in this document since it doesn't use NAT. By design, the only way to control where a packet exits this network is by setting the source address of the packet. Since the network cannot modify the source address without NAT, the host must set it. To implement this routing policy, each host needs to use the source address from the prefix assigned by ISP-A to send traffic destined for H101. Mechanisms have been proposed to allow hosts to choose the source address for packets in a fine grained manner. We will discuss these proposals in Section 6. However, interacting with host operating systems in some manner to ensure a particular source address is chosen for a particular destination prefix is not what an enterprise network administrator would expect to have to do to implement this routing policy.

#### 4.4. More complex ISP connectivity

The previous sections considered two variations of a simple multihoming scenario where the site is connected to two ISPs offering only Internet connectivity. It is likely that many actual enterprise multihoming scenarios will be similar to this simple example. However, there are more complex multihoming scenarios that we would like this solution to address as well.

It is fairly common for an ISP to offer a service in addition to Internet access over the same uplink. Two variations of this are reflected in Figure 3. In addition to Internet access, ISP-A offers a service which requires the site to access host H51 at 2001:db8:0:5555::51. The site has a single physical and logical connection with ISP-A, and ISP-A only allows access to H51 over that connection. So when H32 needs to access the service at H51 it needs to send packets with (S=2001:db8:0:a010::32, D=2001:db8:0:5555::51) and those packets need to be forward out the link from SERa to ISP-A.

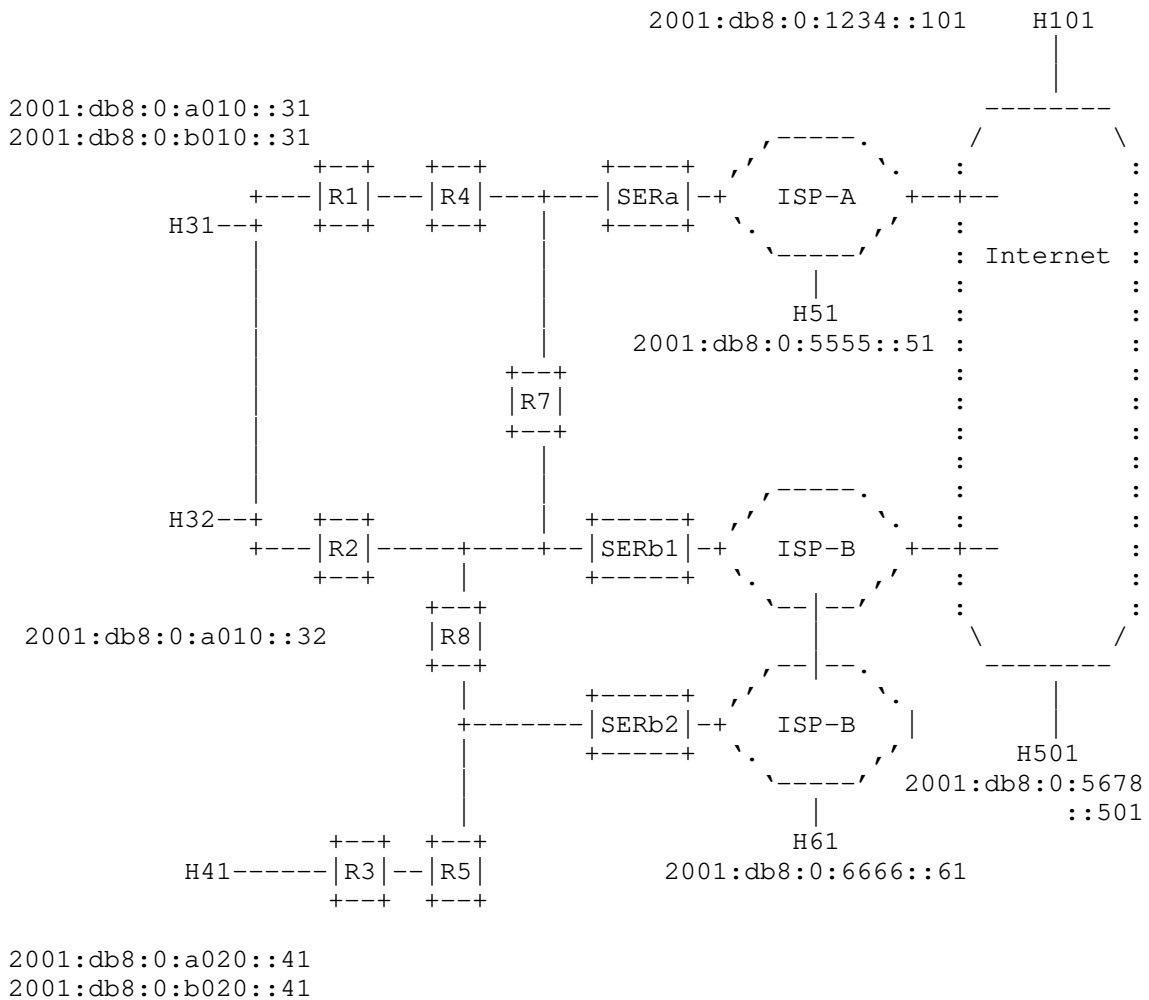


Figure 3: Internet access and services offered by ISP-A and ISP-B

ISP-B illustrates a variation on this scenario. In addition to Internet access, ISP-B also offers a service which requires the site to access host H61. The site has two connections to two different parts of ISP-B (shown as SERb1 and SERb2 in Figure 3). ISP-B expects Internet traffic to use the uplink from SERb1, while it expects traffic destined for the service at H61 to use the uplink from SERb2. For either uplink, ISP-B expects the ingress traffic to have a source address matching the prefix it assigned to the site, 2001:db8:0:b000::/52.

As discussed before, we rely completely on the internal host to set the source address of the packet properly. In the case of a packet sent by H31 to access the service in ISP-B at H61, we expect the packet to have the following addresses: (S=2001:db8:0:b010::31, D=2001:db8:0:6666::61). The routed network has two potential ways of distributing routes so that this packet exits the site on the uplink at SERb2.

We could just rely on normal destination routes, without using source-prefix scoped routes. If we have SERb2 originate a normal unscoped destination route for D=2001:db8:0:6666::/64, the packets from H31 to H61 will exit the site at SERb2 as desired. We should not have to worry about SERa needing to originate the same route, because ISP-B should choose a globally unique prefix for the service at H61.

The alternative is to have SERb2 originate a source-prefix-scoped destination route of the form (S=2001:db8:0:b000::/52, D=2001:db8:0:6666::/64). From a forwarding point of view, the use of the source-prefix-scoped destination route would result in traffic with source addresses corresponding only to ISP-B being sent to SERb2. Instead, the use of the unscoped destination route would result in traffic with source addresses corresponding to ISP-A and ISP-B being sent to SERb2, as long as the destination address matches the destination prefix. It seems like either forwarding behavior would be acceptable.

However, from the point of view of the enterprise network administrator trying to configure, maintain, and trouble-shoot this multihoming solution, it seems much clearer to have SERb2 originate the source-prefix-scoped destination route correspond to the service offered by ISP-B. In this way, all of the traffic leaving the site is determined by the source-prefix-scoped routes, and all of the traffic within the site or arriving from external hosts is determined by the unscoped destination routes. Therefore, for this multihoming solution we choose to originate source-prefix-scoped routes for all traffic leaving the site.

#### 4.5. ISPs and Provider-Assigned Prefixes

While we expect that most site multihoming involves connecting to only two ISPs, this solution allows for connections to an arbitrary number of ISPs to be supported. However, when evaluating scalable implementations of the solution, it would be reasonable to assume that the maximum number of ISPs that a site would connect to is five (topologies with two redundant routers each having two uplinks to different ISPs plus a tunnel to a headoffice acting as fifth one are not unheard of).

It is also useful to note that the prefixes assigned to the site by different ISPs will not overlap. This must be the case, since the provider-assigned addresses have to be globally unique.

#### 4.6. Simplified Topologies

The topologies of many enterprise sites using this multihoming solution may in practice be simpler than the examples that we have used. The topology in Figure 1 could be further simplified by having all hosts directly connected to the LAN connecting the two site exit routers, SERa and SERb. The topology could also be simplified by having the uplinks to ISP-A and ISP-B both connected to the same site exit router. However, it is the aim of this document to provide a solution that applies to a broad a range of enterprise site network topologies, so this document focuses on providing a solution to the more general case. The simplified cases will also be supported by this solution, and there may even be optimizations that can be made for simplified cases. This solution however needs to support more complex topologies.

We are starting with the basic assumption that enterprise site networks can be quite complex from a routing perspective. However, even a complex site network can be multihomed to different ISPs with PA addresses using IPv4 and NAT. It is not reasonable to expect an enterprise network operator to change the routing topology of the site in order to deploy IPv6.

#### 5. Generating Source-Prefix-Scoped Forwarding Tables

So far we have described in general terms how the routers in this solution that are capable of Source Address Dependent Routing will forward traffic using both normal unscoped destination routes and source-prefix-scoped destination routes. Here we give a precise method for generating a source-prefix-scoped forwarding table on a router that supports SADR.

1. Compute the next-hops for the source-prefix-scoped destination prefixes using only routers in the connected SADR domain. These are the initial source-prefix-scoped forwarding table entries.
2. Compute the next-hops for the unscoped destination prefixes using all routers in the IGP. This is the unscoped forwarding table.
3. For a given source-prefix-scoped forwarding table  $T$  (scoped to source prefix  $P$ ), consider a source-prefix-scoped forwarding table  $T'$ , whose source prefix  $P'$  contains  $P$ . We call  $T$  the more specific source-prefix-scoped forwarding table, and  $T'$  the less specific source-prefix-scoped forwarding table. We select

entries in the less specific source-prefix-scoped forwarding table to augment the more specific source-prefix-scoped forwarding table based on the following rules. If a destination prefix of an entry in the less specific source-prefix-scoped forwarding table exactly matches the destination prefix of an existing entry in the more specific source-prefix-scoped forwarding table (including destination prefix length), then do not add the entry to the more specific source-prefix-scoped forwarding table. If the destination prefix does NOT match an existing entry, then add the entry to the more specific source-prefix-scoped forwarding table. As the unscoped forwarding table is considered to be scoped to `::/0`, this process will propagate routes from the unscoped forwarding table to the more specific source-prefix-scoped forwarding table. If there exist multiple source-prefix-scoped forwarding tables whose source prefixes contain `P`, these source-prefix-scoped forwarding tables should be processed in order from most specific to least specific.

The forwarding tables produced by this process are used in the following way to forward packets.

1. Select the most specific (longest prefix match) source-prefix-scoped forwarding table that matches the source address of the packet (again, the unscoped forwarding table is considered to be scoped to `::/0`).
2. Look up the destination address of the packet in the selected forwarding table to determine the next-hop for the packet.

The following example illustrates how this process is used to create a forwarding table for each provider-assigned source prefix. We consider the multihomed site network in Figure 3. Initially we assume that all of the routers in the site network support SADR. Figure 4 shows the routes that are originated by the routers in the site network.



Routes originated by SERa:  
(S=2001:db8:0:a000::/52, D=2001:db8:0:5555/64)  
(S=2001:db8:0:a000::/52, D=::/0)  
(D=2001:db8:0:5555::/64)  
(D=::/0)

Routes originated by SERb1:  
(S=2001:db8:0:b000::/52, D=::/0)  
(D=::/0)

Routes originated by SERb2:  
(S=2001:db8:0:b000::/52, D=2001:db8:0:6666::/64)  
(D=2001:db8:0:6666::/64)

Routes originated by R1:  
(D=2001:db8:0:a010::/64)  
(D=2001:db8:0:b010::/64)

Routes originated by R2:  
(D=2001:db8:0:a010::/64)  
(D=2001:db8:0:b010::/64)

Routes originated by R3:  
(D=2001:db8:0:a020::/64)  
(D=2001:db8:0:b020::/64)

Figure 4: Routes Originated by Routers in the Site Network

Each SER originates destination routes which are scoped to the source prefix assigned by the ISP that the SER connects to. Note that the SERs also originate the corresponding unscoped destination route. This is not needed when all of the routers in the site support SADR. However, it is required when some routers do not support SADR. This will be discussed in more detail later.

We focus on how R8 constructs its source-prefix-scoped forwarding tables from these route advertisements. R8 computes the next hops for destination routes which are scoped to the source prefix 2001:db8:0:a000::/52. The results are shown in the first table in Figure 5. (In this example, the next hops are computed assuming that all links have the same metric.) Then, R8 computes the next hops for destination routes which are scoped to the source prefix 2001:db8:0:b000::/52. The results are shown in the second table in Figure 5. Finally, R8 computes the next hops for the unscoped destination prefixes. The results are shown in the third table in Figure 5.

```

forwarding entries scoped to
source prefix = 2001:db8:0:a000::/52
=====
D=2001:db8:0:5555/64      NH=R7
D=::/0                    NH=R7

forwarding entries scoped to
source prefix = 2001:db8:0:b000::/52
=====
D=2001:db8:0:6666/64      NH=SERb2
D=::/0                    NH=SERb1

unscoped forwarding entries
=====
D=2001:db8:0:a010::/64    NH=R2
D=2001:db8:0:b010::/64    NH=R2
D=2001:db8:0:a020::/64    NH=R5
D=2001:db8:0:b020::/64    NH=R5
D=2001:db8:0:5555::/64    NH=R7
D=2001:db8:0:6666::/64    NH=SERb2
D=::/0                    NH=SERb1

```

Figure 5: Forwarding Entries Computed at R8

The final step is for R8 to augment the more specific source-prefix-scoped forwarding tables with entries from less specific source-prefix-scoped forwarding tables. The unscoped forwarding table is considered as being scoped to `::/0`, so both `2001:db8:0:a000::/52` and `2001:db8:0:b000::/52` are more specific prefixes of `::/0`. Therefore, entries in the unscoped forwarding table will be evaluated to be added to these two more specific source-prefix-scoped forwarding tables. If a forwarding entry from the less specific source-prefix-scoped forwarding table has the exact same destination prefix (including destination prefix length) as the forwarding entry from the more specific source-prefix-scoped forwarding table, then the existing forwarding entry in the more specific source-prefix-scoped forwarding table wins.

As an example of how the source scoped forwarding entries are augmented, we consider how the two entries in the first table in Figure 5 (the table for source prefix = `2001:db8:0:a000::/52`) are augmented with entries from the third table in Figure 5 (the table of unscoped or scoped to `::/0` forwarding entries). The first four unscoped forwarding entries (`D=2001:db8:0:a010::/64`, `D=2001:db8:0:b010::/64`, `D=2001:db8:0:a020::/64`, and `D=2001:db8:0:b020::/64`) are not an exact match for any of the existing entries in the forwarding table for source prefix `2001:db8:0:a000::/52`. Therefore, these four entries are added to the

final forwarding table for source prefix 2001:db8:0:a000::/52. The result of adding these entries is reflected in the first four entries the first table in Figure 6.

The next less specific scoped (scope is ::/0) forwarding table entry is for D=2001:db8:0:5555::/64. This entry is an exact match for the existing entry in the forwarding table for the more specific source prefix 2001:db8:0:a000::/52. Therefore, we do not replace the existing entry with the entry from the unscoped forwarding table. This is reflected in the fifth entry in the first table in Figure 6. (Note that since both scoped and unscoped entries have R7 as the next hop, the result of applying this rule is not visible.)

The next less specific prefix scoped (scope is ::/0) forwarding table entry is for D=2001:db8:0:6666::/64. This entry is not an exact match for any existing entries in the forwarding table for source prefix 2001:db8:0:a000::/52. Therefore, we add this entry. This is reflected in the sixth entry in the first table in Figure 6.

The next less specific prefix scoped (scope is ::/0) forwarding table entry is for D=::/0. This entry is an exact match for the existing entry in the forwarding table for more specific source prefix 2001:db8:0:a000::/52. Therefore, we do not overwrite the existing source-prefix-scoped entry, as can be seen in the last entry in the first table in Figure 6.

if source address matches 2001:db8:0:a000::/52  
 then use this forwarding table

```
=====
D=2001:db8:0:a010::/64    NH=R2
D=2001:db8:0:b010::/64    NH=R2
D=2001:db8:0:a020::/64    NH=R5
D=2001:db8:0:b020::/64    NH=R5
D=2001:db8:0:5555::/64    NH=R7
D=2001:db8:0:6666::/64    NH=SERb2
D=::/0                    NH=R7
```

else if source address matches 2001:db8:0:b000::/52  
 then use this forwarding table

```
=====
D=2001:db8:0:a010::/64    NH=R2
D=2001:db8:0:b010::/64    NH=R2
D=2001:db8:0:a020::/64    NH=R5
D=2001:db8:0:b020::/64    NH=R5
D=2001:db8:0:5555::/64    NH=R7
D=2001:db8:0:6666::/64    NH=SERb2
D=::/0                    NH=SERb1
```

else if source address matches ::/0 use this forwarding table

```
=====
D=2001:db8:0:a010::/64    NH=R2
D=2001:db8:0:b010::/64    NH=R2
D=2001:db8:0:a020::/64    NH=R5
D=2001:db8:0:b020::/64    NH=R5
D=2001:db8:0:5555::/64    NH=R7
D=2001:db8:0:6666::/64    NH=SERb2
D=::/0                    NH=SERb1
```

Figure 6: Complete Forwarding Tables Computed at R8

The forwarding tables produced by this process at R8 have the desired properties. A packet with a source address in 2001:db8:0:a000::/52 will be forwarded based on the first table in Figure 6. If the packet is destined for the Internet at large or the service at D=2001:db8:0:5555/64, it will be sent to R7 in the direction of SERa. If the packet is destined for an internal host, then the first four entries will send it to R2 or R5 as expected. Note that if this packet has a destination address corresponding to the service offered by ISP-B (D=2001:db8:0:5555::/64), then it will get forwarded to SERb2. It will be dropped by SERb2 or by ISP-B, since the packet has a source address that was not assigned by ISP-B. However, this is expected behavior. In order to use the service offered by ISP-B, the host needs to originate the packet with a source address assigned by ISP-B.

In this example, a packet with a source address that doesn't match 2001:db8:0:a000::/52 or 2001:db8:0:b000::/52 must have originated from an external host. Such a packet will use the unscoped forwarding table (the last table in Figure 6). These packets will flow exactly as they would in absence of multihoming.

We can also modify this example to illustrate how it supports deployments where not all routers in the site support SADR. Continuing with the topology shown in Figure 3, suppose that R3 and R5 do not support SADR. Instead they are only capable of understanding unscoped route advertisements. The SADR routers in the network will still originate the routes shown in Figure 4. However, R3 and R5 will only understand the unscoped routes as shown in Figure 7.

Routes originated by SERa:  
(D=2001:db8:0:5555::/64)  
(D=::/0)

Routes originated by SERb1:  
(D=::/0)

Routes originated by SERb2:  
(D=2001:db8:0:6666::/64)

Routes originated by R1:  
(D=2001:db8:0:a010::/64)  
(D=2001:db8:0:b010::/64)

Routes originated by R2:  
(D=2001:db8:0:a010::/64)  
(D=2001:db8:0:b010::/64)

Routes originated by R3:  
(D=2001:db8:0:a020::/64)  
(D=2001:db8:0:b020::/64)

Figure 7: Routes Advertisements Understood by Routers that do not Support SADR

With these unscoped route advertisements, R5 will produce the forwarding table shown in Figure 8.

forwarding table

```
=====
D=2001:db8:0:a010::/64      NH=R8
D=2001:db8:0:b010::/64      NH=R8
D=2001:db8:0:a020::/64      NH=R3
D=2001:db8:0:b020::/64      NH=R3
D=2001:db8:0:5555::/64      NH=R8
D=2001:db8:0:6666::/64      NH=SERb2
D=::/0                      NH=R8
```

Figure 8: Forwarding Table For R5, Which Doesn't Understand Source-Prefix-Scoped Routes

As all SERs belong to the SADR domain any traffic that needs to exit the site will eventually hit a SADR-capable router. To prevent routing loops involving SADR-capable and non-SADR-capable routers, traffic that enters the SADR-capable domain does not leave the domain until it exits the site. Therefore all SADR-capable routers within the domain MUST be logically connected.

Note that the mechanism described here for converting source-prefix-scoped destination prefix routing advertisements into forwarding state is somewhat different from that proposed in [I-D.ietf-rtgwg-dst-src-routing]. The method described in the current document is functionally equivalent, but it is based on application of existing mechanisms for the described scenarios.

## 6. Mechanisms For Hosts To Choose Good Default Source Addresses In A Multihomed Site

Until this point, we have made the assumption that hosts are able to choose the correct source address using some unspecified mechanism. This has allowed us to just focus on what the routers in a multihomed site network need to do in order to forward packets to the correct ISP based on source address. Now we look at possible mechanisms for hosts to choose the correct source address. We also look at what role, if any, the routers may play in providing information that helps hosts to choose source addresses.

It should be noted that this section discussed how hosts could select the default source address for new connections. Any connection which already exists on a host is bound to the specific source address which can not be changed. Section 6.7 discusses the connections preservation issue in more details.

Any host that needs to be able to send traffic using the uplinks to a given ISP is expected to be configured with an address from the prefix assigned by that ISP. The host will control which ISP is used

for its traffic by selecting one of the addresses configured on the host as the source address for outgoing traffic. It is the responsibility of the site network to ensure that a packet with the source address from an ISP is now sent on an uplink to that ISP.

If all of the ISP uplinks are working, the choice of source address by the host may be driven by the desire to load share across ISP uplinks, or it may be driven by the desire to take advantage of certain properties of a particular uplink or ISP (if some information about various path properties has been made available to the host somehow - see [I-D.ietf-intarea-provisioning-domains] as an example). If any of the ISP uplinks is not working, then the choice of source address by the host can cause packets to get dropped.

How a host should make good decisions about source address selection in a multihomed site is not a solved problem. We do not attempt to solve this problem in this document. Instead we discuss the current state of affairs with respect to standardized solutions and implementation of those solutions. We also look at proposed solutions for this problem.

An external host initiating communication with a host internal to a PA multihomed site will need to know multiple addresses for that host in order to communicate with it using different ISPs to the multihomed site (knowing just one address would undermine all benefits of redundant connectivity provided by multihoming). These addresses are typically learned through DNS. (For simplicity, we assume that the external host is single-homed.) The external host chooses the ISP that will be used at the remote multihomed site by setting the destination address on the packets it transmits. For a session originated from an external host to an internal host, the choice of source address used by the internal host is simple. The internal host has no choice but to use the destination address in the received packet as the source address of the transmitted packet.

For a session originated by a host inside the multi-homed site, the decision of what source address to select is more complicated. We consider three main methods for hosts to get information about the network. The two proactive methods are Neighbor Discovery Router Advertisements and DHCPv6. The one reactive method we consider is ICMPv6. Note that we are explicitly excluding the possibility of having hosts participate in or even listen directly to routing protocol advertisements.

First we look at how a host is currently expected to select the default source and destination addresses to be used for a new connection.

### 6.1. Default Source Address Selection Algorithm on Hosts

[RFC6724] defines the algorithms that hosts are expected to use to select source and destination addresses for packets. It defines an algorithm for selecting a source address and a separate algorithm for selecting a destination address. Both of these algorithms depend on a policy table. [RFC6724] defines a default policy which produces certain behavior.

The rules in the two algorithms in [RFC6724] depend on many different properties of addresses. While these are needed for understanding how a host should choose addresses in an arbitrary environment, most of the rules are not relevant for understanding how a host should choose among multiple source addresses in multihomed environment when sending a packet to a remote host. Returning to the example in Figure 3, we look at what the default algorithms in [RFC6724] say about the source address that internal host H31 should use to send traffic to external host H101, somewhere on the Internet.

There is no choice to be made with respect to destination address. H31 needs to send a packet with D=2001:db8:0:1234::101 in order to reach H101. So H31 have to choose between using S=2001:db8:0:a010::31 or S=2001:db8:0:b010::31 as the source address for this packet. We go through the rules for source address selection in Section 5 of [RFC6724].

Rule 1 (Prefer same address) is not useful to break the tie between source addresses, because neither the candidate source addresses equals the destination address.

Rule 2 (Prefer appropriate scope) is also not used in this scenario, because both source addresses and the destination address have global scope.

Rule 3 (Avoid deprecated addresses) applies to an address that has been autoconfigured by a host using stateless address autoconfiguration as defined in [RFC4862]. An address autoconfigured by a host has a preferred lifetime and a valid lifetime. The address is preferred until the preferred lifetime expires, after which it becomes deprecated. A deprecated address is not used if there is a preferred address of the appropriate scope available. When the valid lifetime expires, the address cannot be used at all. The preferred and valid lifetimes for an autoconfigured address are set based on the corresponding lifetimes in the Prefix Information Option in Neighbor Discovery Router Advertisements. So a possible tool to control source address selection in this scenario would be for a host to make an address deprecated by having routers on that link, R1 and R2 in Figure 3, send a Router Advertisement message containing a



Prefix Information Option for the source prefix to be discouraged (or prohibited) with the preferred lifetime set to zero. This is a rather blunt tool, because it discourages or prohibits the use of that source prefix for all destinations. However, it may be useful in some scenarios. For example, if all uplinks to a particular ISP fail, it is desirable to prevent hosts from using source addresses from that ISP address space.

Rule 4 (Avoid home addresses) does not apply here because we are not considering Mobile IP.

Rule 5 (Prefer outgoing interface) is not useful in this scenario, because both source addresses are assigned to the same interface.

Rule 5.5 (Prefer addresses in a prefix advertised by the next-hop) is not useful in the scenario when both R1 and R2 will advertise both source prefixes. However potentially this rule may allow a host to select the correct source prefix by selecting a next-hop. The most obvious way would be to make R1 to advertise itself as a default router and send PIO for 2001:db8:0:a010::/64, while R2 is advertising itself as a default router and sending PIO for 2001:db8:0:b010::/64. We'll discuss later how Rule 5.5 can be used to influence a source address selection in single-router topologies (e.g. when H41 is sending traffic using R3 as a default gateway).

Rule 6 (Prefer matching label) refers to the Label value determined for each source and destination prefix as a result of applying the policy table to the prefix. With the default policy table defined in Section 2.1 of [RFC6724],  $\text{Label}(2001:\text{db8}:0:\text{a010}::31) = 5$ ,  $\text{Label}(2001:\text{db8}:0:\text{b010}::31) = 5$ , and  $\text{Label}(2001:\text{db8}:0:1234::101) = 5$ . So with the default policy, Rule 6 does not break the tie. However, the algorithms in [RFC6724] are defined in such a way that non-default address selection policy tables can be used. [RFC7078] defines a way to distribute a non-default address selection policy table to hosts using DHCPv6. So even though the application of rule 6 to this scenario using the default policy table is not useful, rule 6 may still be a useful tool.

Rule 7 (Prefer temporary addresses) has to do with the technique described in [RFC4941] to periodically randomize the interface portion of an IPv6 address that has been generated using stateless address autoconfiguration. In general, if H31 were using this technique, it would use it for both source addresses, for example creating temporary addresses 2001:db8:0:a010:2839:9938:ab58:830f and 2001:db8:0:b010:4838:f483:8384:3208, in addition to 2001:db8:0:a010::31 and 2001:db8:0:b010::31. So this rule would prefer the two temporary addresses, but it would not break the tie between the two source prefixes from ISP-A and ISP-B.

Rule 8 (Use longest matching prefix) dictates that between two candidate source addresses the one which has longest common prefix length with the destination address. For example, if H31 were selecting the source address for sending packets to H101, this rule would not be a tie breaker as for both candidate source addresses 2001:db8:0:a101::31 and 2001:db8:0:b101::31 the common prefix length with the destination is 48. However if H31 were selecting the source address for sending packets H41 address 2001:db8:0:a020::41, then this rule would result in using 2001:db8:0:a101::31 as a source (2001:db8:0:a101::31 and 2001:db8:0:a020::41 share the common prefix 2001:db8:0:a000::/58, while for 2001:db8:0:b101::31 and 2001:db8:0:a020::41 the common prefix is 2001:db8:0:a000::/51). Therefore rule 8 might be useful for selecting the correct source address in some but not all scenarios (for example if ISP-B services belong to 2001:db8:0:b000::/59 then H31 would always use 2001:db8:0:b010::31 to access those destinations).

So we can see that of the 8 source selection address rules from [RFC6724], four actually apply to our basic site multihoming scenario. The rules that are relevant to this scenario are summarized below.

- o Rule 3: Avoid deprecated addresses.
- o Rule 5.5: Prefer addresses in a prefix advertised by the next-hop.
- o Rule 6: Prefer matching label.
- o Rule 8: Prefer longest matching prefix.

The two methods that we discuss for controlling the source address selection through the four relevant rules above are SLAAC Router Advertisement messages and DHCPv6.

We also consider a possible role for ICMPv6 for getting traffic-driven feedback from the network. With the source address selection algorithm discussed above, the goal is to choose the correct source address on the first try, before any traffic is sent. However, another strategy is to choose a source address, send the packet, get feedback from the network about whether or not the source address is correct, and try another source address if it is not.

We consider four scenarios where a host needs to select the correct source address. The first is when both uplinks are working. The second is when one uplink has failed. The third one is a situation when one failed uplink has recovered. The last one is failure of both (all) uplinks.

It should be noted that [RFC6724] only defines the behavior of IPv6 hosts to select default addresses that applications and upper-layer protocols can use. Applications and upper-layer protocols can make their own choices on selecting source addresses. The mechanism proposed in this document attempts to ensure that the subset of source addresses available for applications and upper-layer protocols is selected with the up-to-date network state in mind. The rest of the document discusses various aspects of the default source address selection defined in [RFC6724], calling it for the sake of brevity "the source address selection".

## 6.2. Selecting Default Source Address When Both Uplinks Are Working

Again we return to the topology in Figure 3. Suppose that the site administrator wants to implement a policy by which all hosts need to use ISP-A to reach H101 at D=2001:db8:0:1234::101. So for example, H31 needs to select S=2001:db8:0:a010::31.

### 6.2.1. Distributing Default Address Selection Policy Table with DHCPv6

This policy can be implemented by using DHCPv6 to distribute an address selection policy table that assigns the same label to destination address that match 2001:db8:0:1234::/64 as it does to source addresses that match 2001:db8:0:a000::/52. The following two entries accomplish this.

Prefix	Precedence	Label
2001:db8:0:1234::/64	50	33
2001:db8:0:a000::/52	50	33

Figure 9: Policy table entries to implement a routing policy

This requires that the hosts implement [RFC6724], the basic source and destination address framework, along with [RFC7078], the DHCPv6 extension for distributing a non-default policy table. Note that it does NOT require that the hosts use DHCPv6 for address assignment. The hosts could still use stateless address autoconfiguration for address configuration, while using DHCPv6 only for policy table distribution (see [RFC8415]). However this method has a number of disadvantages:

- o DHCPv6 support is not a mandatory requirement for IPv6 hosts ([RFC6434]), so this method might not work for all devices.
- o Network administrators are required to explicitly configure the desired network access policies on DHCPv6 servers. While it might be feasible in the scenario of a single multihomed network, such approach might have some scalability issues, especially if the

centralized DHCPv6 solution is deployed to serve a large number of multiomed sites.

#### 6.2.2. Controlling Default Source Address Selection With Router Advertisements

Neighbor Discovery currently has two mechanisms to communicate prefix information to hosts. The base specification for Neighbor Discovery (see [RFC4861]) defines the Prefix Information Option (PIO) in the Router Advertisement (RA) message. When a host receives a PIO with the A-flag set, it can use the prefix in the PIO as source prefix from which it assigns itself an IP address using stateless address autoconfiguration (SLAAC) procedures described in [RFC4862]. In the example of Figure 3, if the site network is using SLAAC, we would expect both R1 and R2 to send RA messages with PIOs for both source prefixes 2001:db8:0:a010::/64 and 2001:db8:0:b010::/64 with the A-flag set. H31 would then use the SLAAC procedure to configure itself with the 2001:db8:0:a010::31 and 2001:db8:0:b010::31.

Whereas a host learns about source prefixes from PIO messages, hosts can learn about a destination prefix from a Router Advertisement containing Route Information Option (RIO), as specified in [RFC4191]. The destination prefixes in RIOs are intended to allow a host to choose the router that it uses as its first hop to reach a particular destination prefix.

As currently standardized, neither PIO nor RIO options contained in Neighbor Discovery Router Advertisements can communicate the information needed to implement the desired routing policy. PIO's communicate source prefixes, and RIO communicate destination prefixes. However, there is currently no standardized way to directly associate a particular destination prefix with a particular source prefix.

[I-D.pfister-6man-sadr-ra] proposes a Source Address Dependent Route Information option for Neighbor Discovery Router Advertisements which would associate a source prefix and with a destination prefix. The details of [I-D.pfister-6man-sadr-ra] might need tweaking to address this use case. However, in order to be able to use Neighbor Discovery Router Advertisements to implement this routing policy, an extension that allows R1 and R2 to explicitly communicate to H31 an association between S=2001:db8:0:a000::/52 D=2001:db8:0:1234::/64 would be needed.

However, Rule 5.5 of the default source address selection algorithm (discussed in Section 6.1 above), together with default router preference (specified in [RFC4191]) and RIO can be used to influence a source address selection on a host as described below. Let's look

at source address selection on the host H41. It receives RAs from R3 with PIOs for 2001:db8:0:a020::/64 and 2001:db8:0:b020::/64. At that point all traffic would use the same next-hop (R3 link-local address) so Rule 5.5 does not apply. Now let's assume that R3 supports SADR and has two scoped forwarding tables, one scoped to S=2001:db8:0:a000::/52 and another scoped to S=2001:db8:0:b000::/52. If R3 generates two different link-local addresses for its interface facing H41 (one for each scoped forwarding table, LLA\_A and LLA\_B) and starts sending two different RAs: one is sent from LLA\_A and includes PIO for 2001:db8:0:a020::/64, another is sent from LLA\_B and includes PIO for 2001:db8:0:b020::/64. Now it is possible to influence H41 source address selection for destinations which follow the default route by setting default router preference in RAs. If it is desired that H41 reaches H101 (or any destinations in the Internet) via ISP-A, then RAs sent from LLA\_A should have default router preference set to 01 (high priority), while RAs sent from LLA\_B should have preference set to 11 (low). Then LLA\_A would be chosen as a next-hop for H101 and therefore (as per rule 5.5) 2001:db8:0:a020::41 would be selected as the source address. If, at the same time, it is desired that H61 is accessible via ISP-B then R3 should include a RIO for 2001:db8:0:6666::/64 to its RA sent from LLA\_B. H41 would chose LLA\_B as a next-hop for all traffic to H61 and then as per Rule 5.5, 2001:db8:0:b020::41 would be selected as a source address.

If in the above mentioned scenario it is desirable that all Internet traffic leaves the network via ISP-A and the link to ISP-B is used for accessing ISP-B services only (not as ISP-A link backup), then RAs sent by R3 from LLA\_B should have Router Lifetime set to 0 and should include RIOs for ISP-B address space. It would instruct H41 to use LLA\_A for all Internet traffic but use LLA\_B as a next-hop while sending traffic to ISP-B addresses.

The description of the mechanism above assumes SADR support by the first-hop routers as well as SERs. However, a first-hop router can still provide a less flexible version of this mechanism even without implementing SADR. This could be done by providing configuration knobs on the first-hop router that allow it to generate different link-local addresses and to send individual RAs for each prefix.

The mechanism described above relies on Rule 5.5 of the default source address selection algorithm defined in [RFC6724]. [RFC8028] states that "A host SHOULD select default routers for each prefix it is assigned an address in". It also recommends that hosts should implement Rule 5.5. of [RFC6724]. Hosts following the recommendations specified in [RFC8028] therefore should be able to benefit from the solution described in this document. No standards need to be updated in regards to host behavior.

### 6.2.3. Controlling Default Source Address Selection With ICMPv6

We now discuss how one might use ICMPv6 to implement the routing policy to send traffic destined for H101 out the uplink to ISP-A, even when uplinks to both ISPs are working. If H31 started sending traffic to H101 with S=2001:db8:0:b010::31 and D=2001:db8:0:1234::101, it would be routed through SER-b1 and out the uplink to ISP-B. SERb1 could recognize that this traffic is not following the desired routing policy and react by sending an ICMPv6 message back to H31.

In this example, we could arrange things so that SERb1 drops the packet with S=2001:db8:0:b010::31 and D=2001:db8:0:1234::101, and then sends to H31 an ICMPv6 Destination Unreachable message with Code 5 (Source address failed ingress/egress policy). When H31 receives this packet, it would then be expected to try another source address to reach the destination. In this example, H31 would then send a packet with S=2001:db8:0:a010::31 and D=2001:db8:0:1234::101, which will reach SERa and be forwarded out the uplink to ISP-A.

However, we would also want it to be the case that SERb1 does not enforce this routing policy when the uplink from SERa to ISP-A has failed. This could be accomplished by having SERa originate a source-prefix-scoped route for (S=2001:db8:0:a000::/52, D=2001:db8:0:1234::/64) and have SERb1 monitor the presence of that route. If that route is not present (because SERa has stopped originating it), then SERb1 will not enforce the routing policy, and it will forward packets with S=2001:db8:0:b010::31 and D=2001:db8:0:1234::101 out its uplink to ISP-B.

We can also use this source-prefix-scoped route originated by SERa to communicate the desired routing policy to SERb1. We can define an EXCLUSIVE flag to be advertised together with the IGP route for (S=2001:db8:0:a000::/52, D=2001:db8:0:1234::/64). This would allow SERa to communicate to SERb that SERb should reject traffic for D=2001:db8:0:1234::/64 and respond with an ICMPv6 Destination Unreachable Code 5 message, as long as the route for (S=2001:db8:0:a000::/52, D=2001:db8:0:1234::/64) is present. The definition of an EXCLUSIVE flag for SADR advertisements in IGPs would require future standardization work.

Finally, if we are willing to extend ICMPv6 to support this solution, then we could create a mechanism for SERb1 to tell the host what source address it should be using to successfully forward packets that meet the policy. In its current form, when SERb1 sends an ICMPv6 Destination Unreachable Code 5 message, it is basically saying, "This source address is wrong. Try another source address." In the absence of a clear indication which address to try next, the

host will iterate over all addresses assigned to the interface (e.g. various privacy addresses) which would lead to significant delays and degraded user experience. It would be better if the ICMPv6 message could say, "This source address is wrong. Instead use a source address in S=2001:db8:0:a000::/52."

However using ICMPv6 for signaling source address information back to hosts introduces new challenges. Most routers currently have software or hardware limits on generating ICMP messages. A site administrator deploying a solution that relies on the SERs generating ICMP messages could try to improve the performance of SERs for generating ICMP messages. However, in a large network, it is still likely that ICMP message generation limits will be reached. As a result hosts would not receive ICMPv6 back which in turn leads to traffic blackholing and poor user experience. To improve the scalability of ICMPv6-based signaling hosts SHOULD cache the preferred source address (or prefix) for the given destination (which in turn might cause issues in case of the corresponding ISP uplinks failure – see Section 6.3). In addition, the same source prefix SHOULD be used for other destinations in the same /64 as the original destination address. The source prefix to the destination mapping SHOULD have a specific lifetime. Expiration of the lifetime SHOULD trigger the source address selection algorithm again.

Using ICMPv6 Destination Unreachable Messages with Code 5 to influence source address selection introduces some security challenges which are discussed in Section 10.

As currently standardized in [RFC4443], the ICMPv6 Destination Unreachable Message with Code 5 would allow for the iterative approach to retransmitting packets using different source addresses. As currently defined, the ICMPv6 message does not provide a mechanism to communicate information about which source prefix should be used for a retransmitted packet. The current document does not define such a mechanism but it might be a useful extension to define in a different document. However this approach has some security implications such as an ability for an attacker to send spoofed ICMPv6 messages to signal invalid/unreachable source prefix causing DoS-type attack.

#### 6.2.4. Summary of Methods For Controlling Default Source Address Selection To Implement Routing Policy

So to summarize this section, we have looked at three methods for implementing a simple routing policy where all traffic for a given destination on the Internet needs to use a particular ISP, even when the uplinks to both ISPs are working.

The default source address selection policy cannot distinguish between the source addresses needed to enforce this policy, so a non-default policy table using associating source and destination prefixes using Label values would need to be installed on each host. A mechanism exists for DHCPv6 to distribute a non-default policy table but such solution would heavily rely on DHCPv6 support by host operating system. Moreover there is no mechanism to translate desired routing/traffic engineering policies into policy tables on DHCPv6 servers. Therefore using DHCPv6 for controlling address selection policy table is not recommended and SHOULD NOT be used.

At the same time Router Advertisements provide a reliable mechanism to influence source address selection process via PIO, RIO and default router preferences. As all those options have been standardized by IETF and are supported by various operating systems no changes are required on hosts. First-hop routers in the enterprise network need to be able of sending different RAs for different SLAAC prefixes (either based on scoped forwarding tables or based on pre-configured policies).

SERs can enforce the routing policy by sending ICMPv6 Destination Unreachable messages with Code 5 (Source address failed ingress/egress policy) for traffic that is being sent with the wrong source address. The policy distribution could be automated by defining an EXCLUSIVE flag for the source-prefix-scoped route which can be set on the SER that originates the route. As ICMPv6 message generation can be rate-limited on routers, it SHOULD NOT be used as the only mechanism to influence source address selection on hosts. While hosts SHOULD select the correct source address for a given destination the network SHOULD signal any source address issues back to hosts using ICMPv6 error messages.

### 6.3. Selecting Default Source Address When One Uplink Has Failed

Now we discuss if DHCPv6, Neighbor Discovery Router Advertisements, and ICMPv6 can help a host choose the right source address when an uplink to one of the ISPs has failed. Again we look at the scenario in Figure 3. This time we look at traffic from H31 destined for external host H501 at D=2001:db8:0:5678::501. We initially assume that the uplink from SERa to ISP-A is working and that the uplink from SERb1 to ISP-B is working.

We assume there is no particular routing policy desired, so H31 is free to send packets with S=2001:db8:0:a010::31 or S=2001:db8:0:b010::31 and have them delivered to H501. For this example, we assume that H31 has chosen S=2001:db8:0:b010::31 so that the packets exit via SERb to ISP-B. Now we see what happens when the link from SERb1 to ISP-B fails. How should H31 learn that it needs



to start sending the packet to H501 with S=2001:db8:0:a010::31 in order to start using the uplink to ISP-A? We need to do this in a way that doesn't prevent H31 from still sending packets with S=2001:db8:0:b010::31 in order to reach H61 at D=2001:db8:0:6666::61.

#### 6.3.1. Controlling Default Source Address Selection With DHCPv6

For this example we assume that the site network in Figure 3 has a centralized DHCP server and all routers act as DHCP relay agents. We assume that both of the addresses assigned to H31 were assigned via DHCP.

We could try to have the DHCP server monitor the state of the uplink from SERb1 to ISP-B in some manner and then tell H31 that it can no longer use S=2001:db8:0:b010::31 by setting its valid lifetime to zero. The DHCP server could initiate this process by sending a Reconfigure Message to H31 as described in Section 18.3 of [RFC8415]. Or the DHCP server can assign addresses with short lifetimes in order to force clients to renew them often.

This approach would prevent H31 from using S=2001:db8:0:b010::31 to reach a host on the Internet. However, it would also prevent H31 from using S=2001:db8:0:b010::31 to reach H61 at D=2001:db8:0:6666::61, which is not desirable.

Another potential approach is to have the DHCP server monitor the uplink from SERb1 to ISP-B and control the choice of source address on H31 by updating its address selection policy table via the mechanism in [RFC7078]. The DHCP server could initiate this process by sending a Reconfigure Message to H31. Note that [RFC8415] requires that Reconfigure Message use DHCP authentication. DHCP authentication could be avoided by using short address lifetimes to force clients to send Renew messages to the server often. If the host is not obtaining its IP addresses from the DHCP server, then it would need to use the Information Refresh Time option defined in [RFC8415].

If the following policy table can be installed on H31 after the failure of the uplink from SERb1, then the desired routing behavior should be achieved based on source and destination prefix being matched with label values.

Prefix	Precedence	Label
::/0	50	44
2001:db8:0:a000::/52	50	44
2001:db8:0:6666::/64	50	55
2001:db8:0:b000::/52	50	55

Figure 10: Policy Table Needed On Failure Of Uplink From SERb1

The described solution has a number of significant drawbacks, some of them already discussed in Section 6.2.1.

- o DHCPv6 support is not required for an IPv6 host and there are operating systems which do not support DHCPv6. Besides that, it does not appear that [RFC7078] has been widely implemented on host operating systems.
- o [RFC7078] does not clearly specify this kind of a dynamic use case where address selection policy needs to be updated quickly in response to the failure of a link. In a large network it would present scalability issues as many hosts need to be reconfigured in very short period of time.
- o Updating DHCPv6 server configuration each time an ISP uplink changes its state introduces some scalability issues, especially for mid/large distributed scale enterprise networks. In addition to that, the policy table needs to be manually configured by administrators which makes that solution prone to human error.
- o No mechanism exists for making DHCPv6 servers aware of network topology/routing changes in the network. In general DHCPv6 servers monitoring network-related events sounds like a bad idea as completely new functionality beyond the scope of DHCPv6 role is required.

#### 6.3.2. Controlling Default Source Address Selection With Router Advertisements

The same mechanism as discussed in Section 6.2.2 can be used to control the source address selection in the case of an uplink failure. If a particular prefix should not be used as a source for any destinations, then the router needs to send RA with Preferred Lifetime field for that prefix set to 0.

Let's consider a scenario when all uplinks are operational and H41 receives two different RAs from R3: one from LLA\_A with PIO for 2001:db8:0:a020::/64, default router preference set to 11 (low) and another one from LLA\_B with PIO for 2001:db8:0:a020::/64, default

router preference set to 01 (high) and RIO for 2001:db8:0:6666::/64. As a result H41 is using 2001:db8:0:b020::41 as a source address for all Internet traffic and those packets are sent by SERs to ISP-B. If SERb1 uplink to ISP-B failed, the desired behavior is that H41 stops using 2001:db8:0:b020::41 as a source address for all destinations but H61. To achieve that R3 should react to SERb1 uplink failure (which could be detected as the scoped route (S=2001:db8:0:b000::/52, D=::/0) disappearance) by withdrawing itself as a default router. R3 sends a new RA from LLA\_B with Router Lifetime value set to 0 (which means that it should not be used as default router). That RA still contains PIO for 2001:db8:0:b020::/64 (for SLAAC purposes) and RIO for 2001:db8:0:6666::/64 so H41 can reach H61 using LLA\_B as a next-hop and 2001:db8:0:b020::41 as a source address. For all traffic following the default route, LLA\_A will be used as a next-hop and 2001:db8:0:a020::41 as a source address.

If all uplinks to ISP-B have failed and therefore source addresses from ISP-B address space should not be used at all, the forwarding table scoped S=2001:db8:0:b000::/52 contains no entries. Hosts can be instructed to stop using source addresses from that block by sending RAs containing PIO with Preferred Lifetime set to 0.

#### 6.3.3. Controlling Default Source Address Selection With ICMPv6

Now we look at how ICMPv6 messages can provide information back to H31. We assume again that at the time of the failure H31 is sending packets to H501 using (S=2001:db8:0:b010::31, D=2001:db8:0:5678::501). When the uplink from SERb1 to ISP-B fails, SERb1 would stop originating its source-prefix-scoped route for the default destination (S=2001:db8:0:b000::/52, D=::/0) as well as its unscoped default destination route. With these routes no longer in the IGP, traffic with (S=2001:db8:0:b010::31, D=2001:db8:0:5678::501) would end up at SERa based on the unscoped default destination route being originated by SERa. Since that traffic has the wrong source address to be forwarded to ISP-A, SERa would drop it and send a Destination Unreachable message with Code 5 (Source address failed ingress/egress policy) back to H31. H31 would then know to use another source address for that destination and would try with (S=2001:db8:0:a010::31, D=2001:db8:0:5678::501). This would be forwarded to SERa based on the source-prefix-scoped default destination route still being originated by SERa, and SERa would forward it to ISP-A. As discussed above, if we are willing to extend ICMPv6, SERa can even tell H31 what source address it should use to reach that destination. The expected host behaviour has been discussed in Section 6.2.3. Using ICMPv6 would have the same scalability/rate limiting issues discussed in Section 6.2.3. ISP-B uplink failure immediately makes source addresses from 2001:db8:0:b000::/52 unsuitable for external communication and might

trigger a large number of ICMPv6 packets being sent to hosts in that subnet.

#### 6.3.4. Summary Of Methods For Controlling Default Source Address Selection On The Failure Of An Uplink

It appears that DHCPv6 is not particularly well suited to quickly changing the source address used by a host in the event of the failure of an uplink, which eliminates DHCPv6 from the list of potential solutions. On the other hand Router Advertisements provides a reliable mechanism to dynamically provide hosts with a list of valid prefixes to use as source addresses as well as prevent particular prefixes to be used. While no additional new features are required to be implemented on hosts, routers need to be able to send RAs based on the state of scoped forwarding tables entries and to react to network topology changes by sending RAs with particular parameters set.

The use of ICMPv6 Destination Unreachable messages generated by the SER (or any SADR-capable) routers seem like they have the potential to provide a support mechanism together with RAs to signal source address selection errors back to hosts, however scalability issues may arise in large networks in case of sudden topology change. Therefore it is highly desirable that hosts are able to select the correct source address in case of uplinks failure with ICMPv6 being an additional mechanism to signal unexpected failures back to hosts.

The current behavior of different host operating system when receiving ICMPv6 Destination Unreachable message with code 5 (Source address failed ingress/egress policy) is not clear to the authors. Information from implementers, users, and testing would be quite helpful in evaluating this approach.

#### 6.4. Selecting Default Source Address Upon Failed Uplink Recovery

The next logical step is to look at the scenario when a failed uplink on SERb1 to ISP-B is coming back up, so hosts can start using source addresses belonging to 2001:db8:0:b000::/52 again.

##### 6.4.1. Controlling Default Source Address Selection With DHCPv6

The mechanism to use DHCPv6 to instruct the hosts (H31 in our example) to start using prefixes from ISP-B space (e.g. S=2001:db8:0:b010::31 for H31) to reach hosts on the Internet is quite similar to one discussed in Section 6.3.1 and shares the same drawbacks.

#### 6.4.2. Controlling Default Source Address Selection With Router Advertisements

Let's look at the scenario discussed in Section 6.3.2. If the uplink(s) failure caused the complete withdrawal of prefixes from 2001:db8:0:b000::/52 address space by setting Preferred Lifetime value to 0, then the recovery of the link should just trigger new RA being sent with non-zero Preferred Lifetime. In another scenario discussed in Section 6.3.2, the SERb1 uplink to ISP-B failure leads to disappearance of the (S=2001:db8:0:b000::/52, D=::/0) entry from the forwarding table scoped to S=2001:db8:0:b000::/52 and, in turn, caused R3 to send RAs from LLA\_B with Router Lifetime set to 0. The recovery of the SERb1 uplink to ISP-B leads to (S=2001:db8:0:b000::/52, D=::/0) scoped forwarding entry re-appearance and instructs R3 that it should advertise itself as a default router for ISP-B address space domain (send RAs from LLA\_B with non-zero Router Lifetime).

#### 6.4.3. Controlling Default Source Address Selection With ICMP

It looks like ICMPv6 provides a rather limited functionality to signal back to hosts that particular source addresses have become valid again. Unless the changes in the uplink state a particular (S,D) pair, hosts can keep using the same source address even after an ISP uplink has come back up. For example, after the uplink from SERb1 to ISP-B had failed, H31 received ICMPv6 Code 5 message (as described in Section 6.3.3) and allegedly started using (S=2001:db8:0:a010::31, D=2001:db8:0:5678::501) to reach H501. Now when the SERb1 uplink comes back up, the packets with that (S,D) pair are still routed to SERa1 and sent to the Internet. Therefore H31 is not informed that it should stop using 2001:db8:0:a010::31 and start using 2001:db8:0:b010::31 again. Unless SERa has a policy configured to drop packets (S=2001:db8:0:a010::31, D=2001:db8:0:5678::501) and send ICMPv6 back if SERb1 uplink to ISP-B is up, H31 will be unaware of the network topology change and keep using S=2001:db8:0:a010::31 for Internet destinations, including H51.

One of the possible option may be using a scoped route with EXCLUSIVE flag as described in Section 6.2.3. SERa1 uplink recovery would cause (S=2001:db8:0:a000::/52, D=2001:db8:0:1234::/64) route to reappear in the routing table. In the absence of that route packets to H101 which were sent to ISP-B (as ISP-A uplink was down) with source addresses from 2001:db8:0:b000::/52. When the route reappears SERb1 would reject those packets and sends ICMPv6 back as discussed in Section 6.2.3. Practically it might lead to scalability issues which have been already discussed in Section 6.2.3 and Section 6.4.3.

#### 6.4.4. Summary Of Methods For Controlling Default Source Address Selection Upon Failed Uplink Recovery

Once again DHCPv6 does not look like reasonable choice to manipulate source address selection process on a host in the case of network topology changes. Using Router Advertisement provides the flexible mechanism to dynamically react to network topology changes (if routers are able to use routing changes as a trigger for sending out RAs with specific parameters). ICMPv6 could be considered as a supporting mechanism to signal incorrect source address back to hosts but should not be considered as the only mechanism to control the address selection in multihomed environments.

#### 6.5. Selecting Default Source Address When All Uplinks Failed

One particular tricky case is a scenario when all uplinks have failed. In that case there is no valid source address to be used for any external destinations while it might be desirable to have intra-site connectivity.

##### 6.5.1. Controlling Default Source Address Selection With DHCPv6

From DHCPv6 perspective uplinks failure should be treated as two independent failures and processed as described in Section 6.3.1. At this stage it is quite obvious that it would result in quite complicated policy table which needs to be explicitly configured by administrators and therefore seems to be impractical.

##### 6.5.2. Controlling Default Source Address Selection With Router Advertisements

As discussed in Section 6.3.2 an uplink failure causes the scoped default entry to disappear from the scoped forwarding table and triggers RAs with zero Router Lifetime. Complete disappearance of all scoped entries for a given source prefix would cause the prefix being withdrawn from hosts by setting Preferred Lifetime value to zero in PIO. If all uplinks (SERa, SERb1 and SERb2) failed, hosts either lost their default routers and/or have no global IPv6 addresses to use as a source. (Note that 'uplink failure' might mean 'IPv6 connectivity failure with IPv4 still being reachable', in which case hosts might fall back to IPv4 if there is IPv4 connectivity to destinations). As a result, intra-site connectivity is broken. One of the possible way to solve it is to use ULAs.

All hosts have ULA addresses assigned in addition to GUAs and used for intra-site communication even if there is no GUA assigned to a host. To avoid accidental leaking of packets with ULA sources SADR-capable routers SHOULD have a scoped forwarding table for ULA source

for internal routes but MUST NOT have an entry for D=::/0 in that table. In the absence of (S=ULA\_Prefix; D=::/0) first-hop routers will send dedicated RAs from a unique link-local source LLA\_ULA with PIO from ULA address space, RIO for the ULA prefix and Router Lifetime set to zero. The behaviour is consistent with the situation when SERb1 lost the uplink to ISP-B (so there is no Internet connectivity from 2001:db8:0:b000::/52 sources) but those sources can be used to reach some specific destinations. In the case of ULA there is no Internet connectivity from ULA sources but they can be used to reach another ULA destinations. Note that ULA usage could be particularly useful if all ISPs assign prefixes via DHCP-PD. In the absence of ULAs, upon the all uplinks failure hosts would lost all their GUAs upon prefix lifetime expiration which again makes intra-site communication impossible.

It should be noted that the Rule 5.5 (prefer a prefix advertised by the selected next-hop) takes precedence over the Rule 6 (prefer matching label, which ensures that GUA source addresses are preferred over ULAs for GUA destinations). Therefore if ULAs are used, the network administrator needs to ensure that while the site has an Internet connectivity, hosts do not select a router which advertises ULA prefixes as their default router.

#### 6.5.3. Controlling Default Source Address Selection With ICMPv6

In case of all uplinks failure all SERs will drop outgoing IPv6 traffic and respond with ICMPv6 error message. In the large network when many hosts are trying to reach Internet destinations it means that SERs need to generate an ICMPv6 error to every packet they receive from hosts which presents the same scalability issues discussed in Section 6.3.3

#### 6.5.4. Summary Of Methods For Controlling Default Source Address Selection When All Uplinks Failed

Again, combining SADR with Router Advertisements seems to be the most flexible and scalable way to control the source address selection on hosts.

#### 6.6. Summary Of Methods For Controlling Default Source Address Selection

To summarize the scenarios and options discussed above:

While DHCPv6 allows administrators to manipulate source address selection policy tables, this method has a number of significant disadvantages which eliminates DHCPv6 from a list of potential solutions:

1. It required hosts to support DHCPv6 and its extension (RFC7078);
2. DHCPv6 server needs to monitor network state and detect routing changes.
3. The use of policy tables requires manual configuration and might be extremely complicated, especially in the case of distributed network when large number of remote sites are being served by centralized DHCPv6 servers.
4. Network topology/routing policy changes could trigger simultaneous re-configuration of large number of hosts which present serious scalability issues.

The use of Router Advertisements to influence the source address selection on hosts seem to be the most reliable, flexible and scalable solution. It has the following benefits:

1. no new (non-standard) functionality needs to be implemented on hosts (except for [RFC4191] RIO support, which remains at the time of this writing not widely implemented);
2. no changes in RA format;
3. routers can react to routing table changes by sending RAs which would minimize the failover time in the case of network topology changes;
4. information required for source address selection is broadcast to all affected hosts in case of topology change event which improves the scalability of the solution (comparing to DHCPv6 reconfiguration or ICMPv6 error messages).

To fully benefit from the RA-based solution, first-hop routers need to implement SADR, belong to the SADR domain and be able to send dedicated RAs per scoped forwarding table as discussed above, reacting to network changes with sending new RAs. It should be noted that the proposed solution would work even if first-hop routers are not SADR-capable but still able to send individual RAs for each ISP prefix and react to topology changes as discussed above (e.g. via configuration knobs).

The RA-based solution relies heavily on hosts correctly implementing default address selection algorithm as defined in [RFC6724]. While the basic (and most common) multihoming scenario (two or more Internet uplinks, no 'walled gardens') would work for any host supporting the minimal implementation of [RFC6724], more complex use cases (such as "walled garden" and other scenarios when some ISP



resources can only be reached from that ISP address space) require that hosts support Rule 5.5 of the default address selection algorithm. There is some evidence that not all host OSes have that rule implemented currently. However it should be noted that [RFC8028] states that Rule 5.5 should be implemented.

ICMPv6 Code 5 error message SHOULD be used to complement RA-based solution to signal incorrect source address selection back to hosts, but it SHOULD NOT be considered as the stand-alone solution. To prevent scenarios when hosts in multihomed environments incorrectly identify onlink/offlink destinations, hosts SHOULD treat ICMPv6 Redirects as discussed in [RFC8028].

## 6.7. Solution Limitations

### 6.7.1. Connections Preservation

The proposed solution is not designed to preserve connection state in case of an uplink failure. When all uplinks to an ISP go down all transport connections established to/from that ISP address space will be interrupted (unless the transport protocol has specific multihoming support). That behaviour is similar to the scenario of IPv4 multihoming with NAT when an uplink failure causes all connections to be NATed to completely different public IPv4 addresses. While it does sound suboptimal, it is determined by the nature of PA address space: if all uplinks to the particular ISP have failed, there is no path for the ingress traffic to reach the site and the egress traffic is supposed to be dropped by the BCP38 [RFC2827] ingress filters. The only potential way to overcome this limitation would be running BGP with all ISPs and advertise all site prefixes to all uplinks – a solution which shares all drawbacks of using PI address space without having its benefits. Networks willing and capable of running BGP and using PI are out of scope of this document.

It should be noted that in case of IPv4 NAT-based multihoming uplink recovery could cause connection interruptions as well (unless packet forwarding is integrated with existing NAT sessions tracking so the egress interface for the existing sessions is not changed). However the proposed solution has a benefit of preserving the existing sessions during/after the failed uplink restoration. Unlike the uplink failure event which causes all addresses from the affected prefix to be deprecated the recovery would just add new preferred addresses to a host without making any addresses unavailable. Therefore connections established to/from those addresses do not have to be interrupted.

While it's desirable for active connections to survive ISP failover events, for sites using PA address space such events affect the reachability of IP addresses assigned to hosts. Unless the transport (or even higher level protocols) are capable of surviving the host renumbering, the active connections will be broken. The proposed solution focuses on minimizing the impact of failover for new connections and for multipath-aware protocols.

Another way to preserve connection state would be using multipath transport as discussed in Section 8.3.

## 6.8. Other Configuration Parameters

### 6.8.1. DNS Configuration

In multihomed environment each ISP might provide their own list of DNS servers. For example, in the topology shown in Figure 3, ISP-A might provide recursive DNS server H51 2001:db8:0:5555::51, while ISP-B might provide H61 2001:db8:0:6666::61 as a recursive DNS server. [RFC8106] defines IPv6 Router Advertisement options to allow IPv6 routers to advertise a list of DNS recursive server addresses and a DNS Search List to IPv6 hosts. Using RDNSS together with 'scoped' RAs as described above would allow a first-hop router (R3 in the Figure 3) to send DNS server addresses and search lists provided by each ISP (or the corporate DNS servers addresses if the enterprise is running its own DNS servers - as discussed below DNS split-horizon problem is too hard to solve without running a local DNS server).

As discussed in Section 6.5.2, failure of all ISP uplinks would cause deprecation of all addresses assigned to a host from the address space of all ISPs. If any intra-site IPv6 connectivity is still desirable (most likely to be the case for any mid/large scale network), then ULAs should be used as discussed in Section 6.5.2. In such a scenario, the enterprise network should run its own recursive DNS server(s) and provide its ULA addresses to hosts via RDNSS in RAs send for ULA-scoped forwarding table as described in Section 6.5.2.

There are some scenarios when the final outcome of the name resolution might be different depending on:

- o which DNS server is used;
- o which source address the client uses to send a DNS query to the server (DNS split horizon).

There is no way currently to instruct a host to use a particular DNS server out of the configured servers list for resolving a particular name. Therefore it does not seem feasible to solve the problem of

DNS server selection on the host (it should be noted that this particular issue is protocol-agnostic and happens for IPv4 as well). In such a scenario it is recommended that the enterprise runs its own local recursive DNS server.

To influence host source address selection for packets sent to a particular DNS server the following requirements must be met:

- o the host supports RIO as defined in [RFC4191];
- o the routers send RIO for routes to DNS server addresses.

For example, if it is desirable that host H31 reaches the ISP-A DNS server H51 2001:db8:0:5555::51 using its source address 2001:db8:0:a010::31, then both R1 and R2 should send the RIO containing the route to 2001:db8:0:5555::51 (or covering route) in their 'scoped' RAs, containing LLA\_A as the default router address and the PO for SLAAC prefix 2001:db8:0:a010::/64. In that case the host H31 (if it supports the Rule 5.5) would select LLA\_A as a next-hop and then chose 2001:db8:0:a010::31 as the source address for packets to the DNS server.

It should be noted that [RFC6106] explicitly prohibits using DNS information if the RA router Lifetime expired: "An RDNSS address or a DNSSL domain name MUST be used only as long as both the RA router Lifetime (advertised by a Router Advertisement message) and the corresponding option Lifetime have not expired.". Therefore hosts might ignore RDNSS information provided in ULA-scoped RAs as those RAs would have router lifetime set to 0. However the updated version of RFC6106 ([RFC8106]) has that requirement removed.

As discussed above the DNS split-horizon problem and selecting the correct DNS server in a multihomed environment is not an easy one to solve. The proper solution would require hosts to support the concept of multiple Provisioning Domains (PvD, a set of configuration information associated with a network, [RFC7556]).

## 7. Deployment Considerations

The solution described in this document requires certain mechanisms to be supported by the network infrastructure and hosts. It requires some routers in the enterprise site to support some form of Source Address Dependent Routing (SADR). It also requires hosts to be able to learn when the uplink to an ISP changes its state so the corresponding source addresses should (or should not) be used. Ongoing work to create mechanisms to accomplish this are discussed in this document, but they are still a work in progress.

### 7.1. Deploying SADR Domain

The proposed solution provides does not prescribe particular details regarding deploying an SADR domain within a multihomed enterprise network. However the following guidelines could be applied:

- o The SADR domain is usually limited by the multihomed site border.
- o The minimal deployable scenario requires enabling SADR on all SERs and including them into a single SADR domain.
- o As discussed in Section 4.2, extending the connected SADR domain beyond that point down to the first-hop routers can produce more efficient forwarding paths and allow the network to fully benefit from SADR. it would also simplify the operation of the SADR domain.
- o During the incremental SADR domain expansion from the SERs down towards first-hop routers it's important to ensure that at any moment of time all SADR-capable routers within the domain are logically connected (see Section 5).

### 7.2. Hosts-Related Considerations

The solution discussed in this document relies on the default address selection algorithm ([RFC6724]) Rule 5.5. While [RFC6724] considers this rule as optional, the recent [RFC8028] states that "A host SHOULD select default routers for each prefix it is assigned an address in". It also recommends that hosts should implement Rule 5.5. of [RFC6724]. Therefore while RFC8028-compliant hosts already have mechanism to learn about ISP uplinks state changes and selecting the source addresses accordingly, many hosts do not have such mechanism supported yet.

It should be noted that multihomed enterprise network utilizing multiple ISP prefixes can be considered as a typical multiple provisioning domain (mPVD) scenario, as described in [RFC7556]. This document defines a way for the network to provide the PVD information to hosts indirectly, using the existing mechanisms. At the same time [I-D.ietf-intarea-provisioning-domains] takes one step further and describes a comprehensive mechanism for hosts to discover the whole set of configuration information associated with different PVD/ISPs. [I-D.ietf-intarea-provisioning-domains] complements this document in terms of making hosts being able to learn about ISP uplink states and selecting the corresponding source addresses.

## 8. Other Solutions

### 8.1. Shim6

The Shim6 working group specified the Shim6 protocol [RFC5533] which allows a host at a multihomed site to communicate with an external host and exchange information about possible source and destination address pairs that they can use to communicate. It also specified the REAP protocol [RFC5534] to detect failures in the path between working address pairs and find new working address pairs. A fundamental requirement for Shim6 is that both internal and external hosts need to support Shim6. That is, both the host internal to the multihomed site and the host external to the multihomed site need to support Shim6 in order for there to be any benefit for the internal host to run Shim6. The Shim6 protocol specification was published in 2009, but it has not been widely implemented. Therefore Shim6 is not considered as a viable solution for enterprise multihoming.

### 8.2. IPv6-to-IPv6 Network Prefix Translation

IPv6-to-IPv6 Network Prefix Translation (NPTv6) [RFC6296] is not the focus of this document. NPTv6 suffers from the same fundamental issue as any other address translation approaches: it breaks end-to-end connectivity. Therefore NPTv6 is not considered as desirable solution and this document intentionally focuses on solving enterprise multihoming problem without any form of address translations.

With increasing interest and ongoing work in bringing path awareness to transport and application layer protocols hosts might be able to determine the properties of the various network paths and choose among paths available to them. As selecting the correct source address is one of the possible mechanisms path-aware hosts may utilize, address translation negatively affects hosts path-awareness which makes NPTv6 even more undesirable solution.

### 8.3. Multipath Transport

Using multipath transport (such as MPTCP, [RFC6824] or multipath capabilities in QUIC) might solve the problems discussed in Section 6 since it would allow hosts to use multiple source addresses for a single connection and switch between source addresses when a particular address becomes unavailable or a new address gets assigned to the host interface. Therefore if all hosts in the enterprise network are only using multipath transport for all connections, the signaling solution described in Section 6 might not be needed (it should be noted that the Source Address Dependent Routing would still be required to deliver packets to the correct uplinks). At the time

this document was written, multipath transport alone could not be considered a solution for the problem of selecting the source address in a multihomed environment. There are significant number of hosts which do not use multipath transport currently and it seems unlikely that the situation is going to change in any foreseeable future (even if new releases of operating systems get multipath protocols support there will be a long tail of legacy hosts). The solution for enterprise multihoming needs to work for the least common denominator: hosts without multipath transport support. In addition, not all protocols are using multipath transport. While multipath transport would complement the solution described in Section 6, it could not be considered as a sole solution to the problem of source address selection in multihomed environments.

On the other hand PA-based multihoming could provide additional benefits for multipath protocol, should those protocols be deployed in the network. Multipath protocols could leverage source address selection to achieve maximum path diversity (and potentially improved performance).

Therefore deploying multipath protocols could not be considered as an alternative to the approach proposed in this document. Instead both solutions complement each other so deploying multipath protocols in PA-based multihomed network proves mutually beneficial.

#### 9. IANA Considerations

This memo asks the IANA for no new parameters.

#### 10. Security Considerations

Section 6.2.3 discusses a mechanism for controlling source address selection on hosts using ICMPv6 messages. Using ICMPv6 to influence source address selection allows an attacker to exhaust the list of candidate source addresses on the host by sending spoofed ICMPv6 Code 5 for all prefixes known on the network (therefore preventing a victim from establishing a communication with the destination host). Another possible attack vector is using ICMPv6 Destination Unreachable Messages with Code 5 to steer the egress traffic towards the particular ISP (for example if the attacker has the ability of doing traffic sniffing or man-in-the-middle attack in that ISP network).

To prevent those attacks hosts SHOULD verify that the original packet header included into ICMPv6 error message was actually sent by the host (to ensure that the ICMPv6 message was triggered by a packet sent by the host).

As ICMPv6 Destination Unreachable Messages with Code 5 could be originated by any SADR-capable router within the domain (or even come from the Internet), GTSM ([RFC5082]) can not be applied. Filtering such ICMOV6 messages at the site border can not be recommended as it would break the legitimate end2end error signalling mechanism ICMPv6 is designed for.

The security considerations of using stateless address autoconfiguration are discussed in [RFC4862].

## 11. Acknowledgements

The original outline was suggested by Ole Troan.

The authors would like to thank the following people (in alphabetical order) for their review and feedback: Olivier Bonaventure, Deborah Brungard, Brian E Carpenter, Lorenzo Colitti, Roman Danyliw, Benjamin Kaduk, Suresh Krishnan, Mirja Kuhlewind, David Lamparter, Nicolai Leymann, Acee Lindem, Philip Matthews, Robert Raszuk, Alvaro Retana, Pete Resnick, Dave Thaler, Michael Tuxen, Martin Vigoureux, Eric Vyncke, Magnus Westerlund.

## 12. References

### 12.1. Normative References

- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<https://www.rfc-editor.org/info/rfc1918>>.
- [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>>.
- [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <<https://www.rfc-editor.org/info/rfc2827>>.
- [RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <<https://www.rfc-editor.org/info/rfc4191>>.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<https://www.rfc-editor.org/info/rfc4193>>.

- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [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>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/info/rfc4861>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC6106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 6106, DOI 10.17487/RFC6106, November 2010, <<https://www.rfc-editor.org/info/rfc6106>>.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, DOI 10.17487/RFC6296, June 2011, <<https://www.rfc-editor.org/info/rfc6296>>.
- [RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <<https://www.rfc-editor.org/info/rfc6724>>.
- [RFC7078] Matsumoto, A., Fujisaki, T., and T. Chown, "Distributing Address Selection Policy Using DHCPv6", RFC 7078, DOI 10.17487/RFC7078, January 2014, <<https://www.rfc-editor.org/info/rfc7078>>.
- [RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <<https://www.rfc-editor.org/info/rfc7556>>.
- [RFC8028] Baker, F. and B. Carpenter, "First-Hop Router Selection by Hosts in a Multi-Prefix Network", RFC 8028, DOI 10.17487/RFC8028, November 2016, <<https://www.rfc-editor.org/info/rfc8028>>.



- [RFC8106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 8106, DOI 10.17487/RFC8106, March 2017, <<https://www.rfc-editor.org/info/rfc8106>>.
- [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>>.
- [RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <<https://www.rfc-editor.org/info/rfc8415>>.

## 12.2. Informative References

- [I-D.ietf-intarea-provisioning-domains] Pfister, P., Vyncke, E., Pauly, T., Schinazi, D., and W. Shao, "Discovering Provisioning Domain Names and Data", draft-ietf-intarea-provisioning-domains-05 (work in progress), June 2019.
- [I-D.ietf-rtgwg-dst-src-routing] Lamparter, D. and A. Smirnov, "Destination/Source Routing", draft-ietf-rtgwg-dst-src-routing-07 (work in progress), March 2019.
- [I-D.pfister-6man-sadr-ra] Pfister, P., "Source Address Dependent Route Information Option for Router Advertisements", draft-pfister-6man-sadr-ra-01 (work in progress), June 2015.
- [RFC3704] Baker, F. and P. Savola, "Ingress Filtering for Multihomed Networks", BCP 84, RFC 3704, DOI 10.17487/RFC3704, March 2004, <<https://www.rfc-editor.org/info/rfc3704>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <<https://www.rfc-editor.org/info/rfc5082>>.

- [RFC5533] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, DOI 10.17487/RFC5533, June 2009, <<https://www.rfc-editor.org/info/rfc5533>>.
- [RFC5534] Arkko, J. and I. van Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming", RFC 5534, DOI 10.17487/RFC5534, June 2009, <<https://www.rfc-editor.org/info/rfc5534>>.
- [RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, DOI 10.17487/RFC6434, December 2011, <<https://www.rfc-editor.org/info/rfc6434>>.
- [RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <<https://www.rfc-editor.org/info/rfc6824>>.
- [RFC7676] Pignataro, C., Bonica, R., and S. Krishnan, "IPv6 Support for Generic Routing Encapsulation (GRE)", RFC 7676, DOI 10.17487/RFC7676, October 2015, <<https://www.rfc-editor.org/info/rfc7676>>.

## Authors' Addresses

Fred Baker  
Santa Barbara, California 93117  
USA

Email: FredBaker.IETF@gmail.com

Chris Bowers  
Juniper Networks  
Sunnyvale, California 94089  
USA

Email: cbowers@juniper.net

Jen Linkova  
Google  
1 Darling Island Rd  
Pyrmont, NSW 2009  
AU

Email: furry@google.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

```



```

    "Route metric type.";
}
description
    "Set the metric type for the route.";
}
container set-route-level {
    leaf route-level {
        type identityref {
            base route-level;
        }
        description
            "Route import level.";
    }
    description
        "Set the level for importation or
        exportation of routes.";
}
leaf set-route-preference {
    type uint16;
    description
        "Set the preference for the route. It is also
        known as 'administrative distance', allows for
        selecting the preferred route among routes with
        the same destination prefix. A smaller value is
        more preferred.";
}
leaf set-tag {
    type tag-type;
    description
        "Set the tag for the route.";
}
leaf set-application-tag {
    type tag-type;
    description
        "Set the application tag for the route.
        The application-specific tag is an additional tag
        that can be used by applications that require
        semantics and/or policy different from that of the
        tag. For example, the tag is usually automatically
        advertised in OSPF AS-External Link State
        Advertisements (LSAs) while this application-specific
        tag is not advertised implicitly.";
}
}
}
}
}
}

```

```
}  
}  
<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)

Internet Engineering Task Force (IETF)  
Internet-Draft  
Intended status: Experimental  
Expires: April 20, 2021

Khaled Omar  
The Road  
L. Camara  
October 20, 2020

Numbering Exchange Protocol (NEP)  
Specification  
draft-omar-nep-09

Abstract

This document specifies the Numbering Exchange Protocol (NEP), an Interior Gateway Protocol (IGP) that combines three metrics: delay, bandwidth and number of hops.

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 April 20, 2021.

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 .....	1
2. Numbering Exchange Protocol (NEP) .....	2
2.1. RIDs and their Advertising .....	4
2.1.1. Advertising Beyond a Router's Neighbours .....	5
2.1.2. Detecting Conflict on RIDs.....	5
2.2. Echo Mechanism .....	5
2.2.1. Detecting Routers that Leaved a NEP Network .....	5
2.3. Topology Advertisement .....	6
2.3.1. The NEP Metric .....	7
3. NEP Loop Prevention .....	7
4. Subnet Advertisement .....	8
4.1. Specification .....	10
4.2. Routing Packets within a NEP AS .....	11
5. IANA Considerations .....	11
6. Security Considerations .....	12
7. References .....	12
7.1. Normative References .....	12
7.2. Informative References .....	12
8. Authors' Addresses .....	13
Appendix A. NEP Advertisement Format .....	13
A.1. NEP Header .....	13
A.2. Topology Advertisement .....	14
A.3. Subnet Advertisements .....	14
A.3.1. Mask-Based Subnet Advertisements .....	15
A.3.2. Prefix-Based Subnet Advertisements .....	15
A.4. Echo Mechanism .....	16
A.4.1. Echo Messages .....	16
A.4.2. Delay Calculated Message .....	16
A.5. Hello Messages .....	16
A.6. Router Left Message .....	17

This contribution has been withdrawn.

## 7. References

## 8. Authors' Addresses

Khaled Omar Ibrahim Omar  
The Road  
6th of October City, Giza  
Egypt

Phone: +2 01003620284  
E-mail: eng.khaled.omar@hotmail.com  
National ID No.: 28611262102992

Luis Camara  
Portugal  
E-mail: luis.camara@live.com.pt

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.

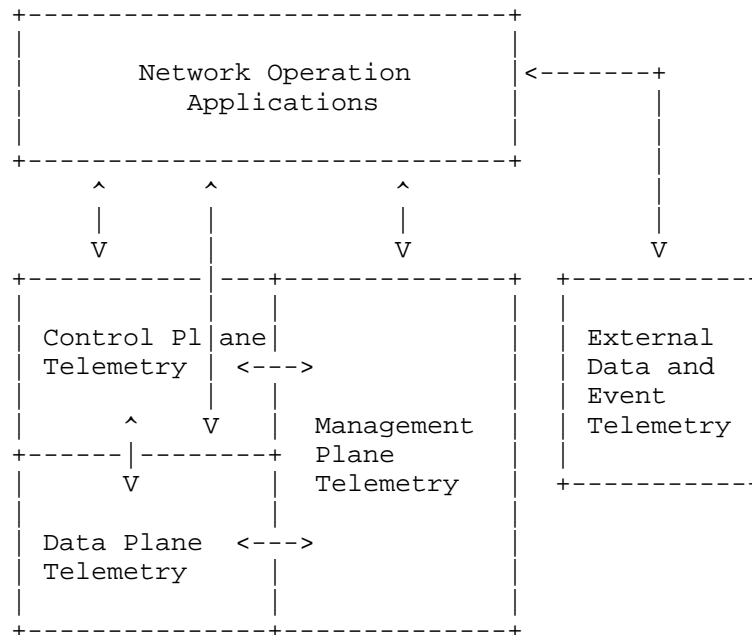


Figure 1: Layer Category of the Network Telemetry Framework

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.

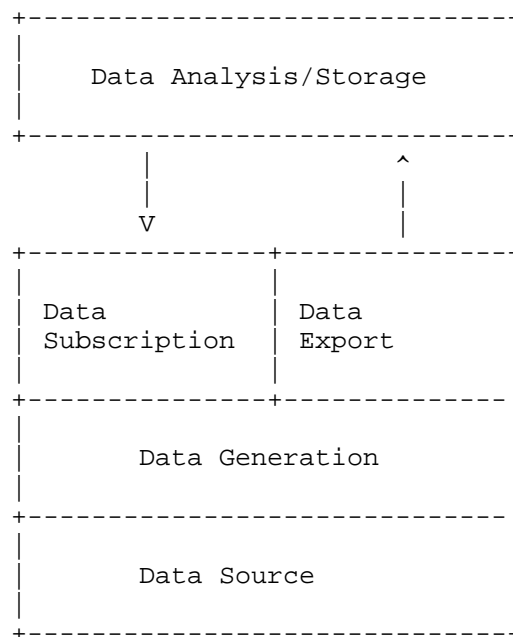


Figure 2: Components in the Network Telemetry Framework

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.

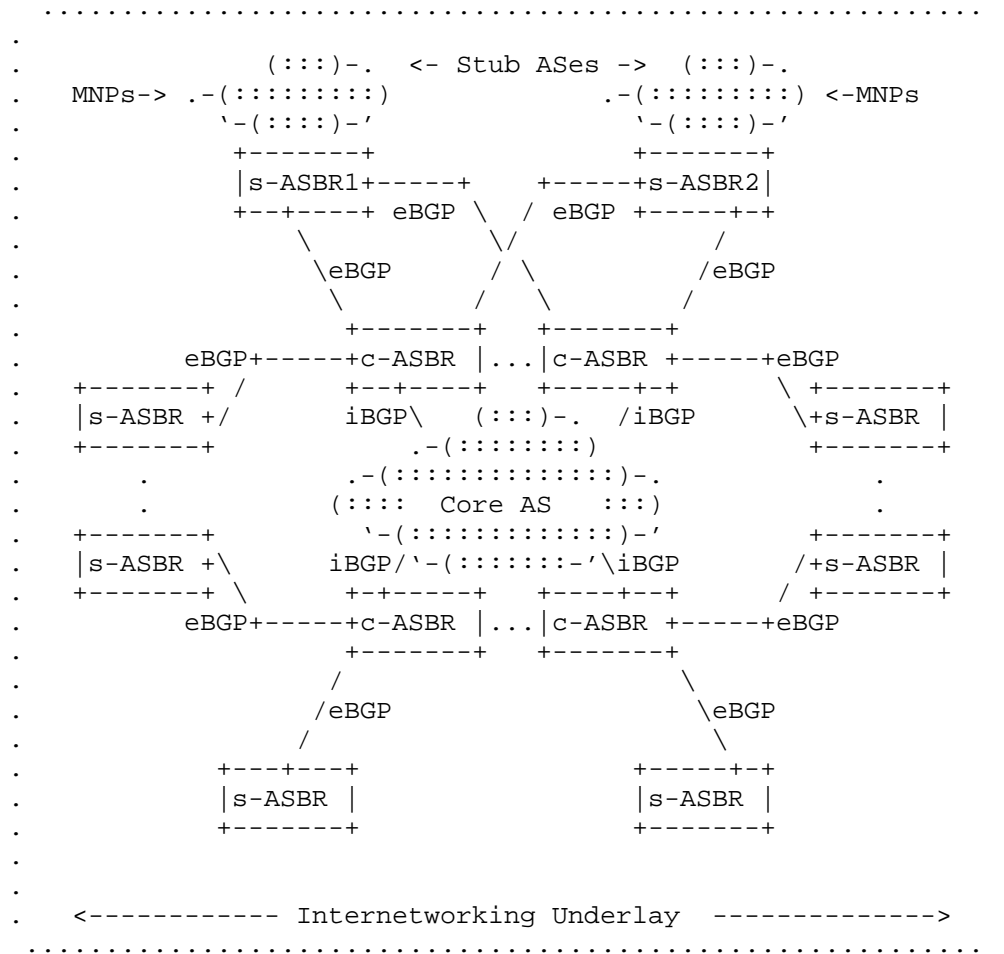


Figure 1: Reference Deployment

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.

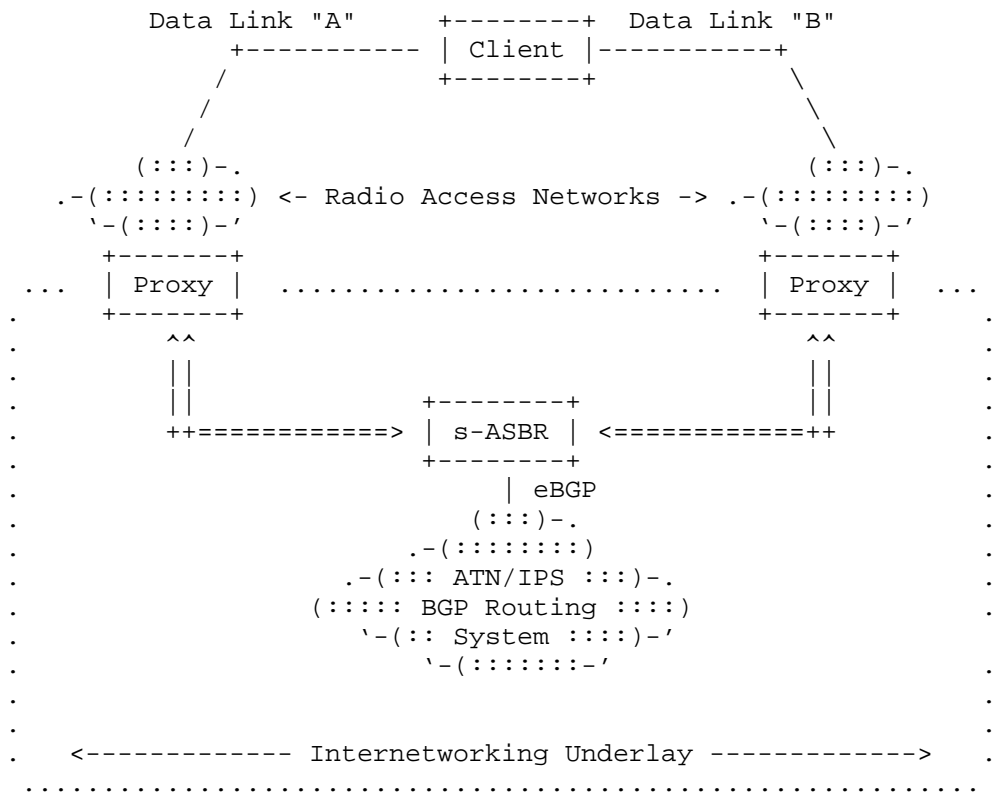


Figure 2: ATN/IPS RAN Architecture

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.

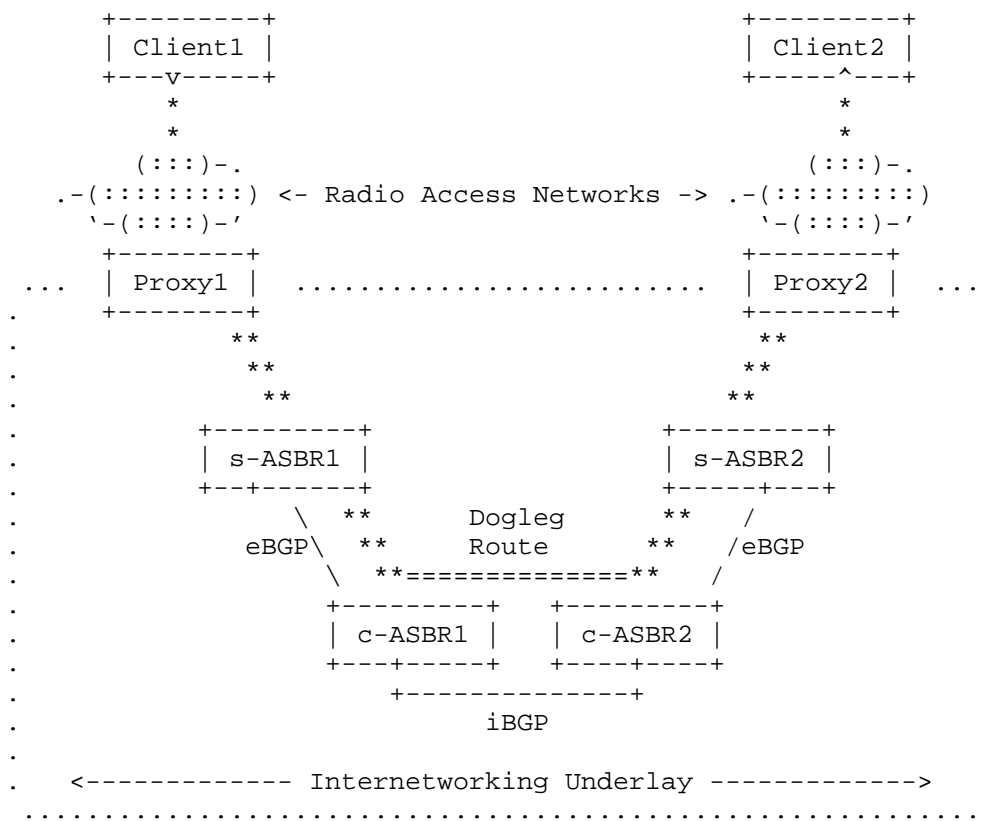


Figure 3: Dogleg Route Before Optimization

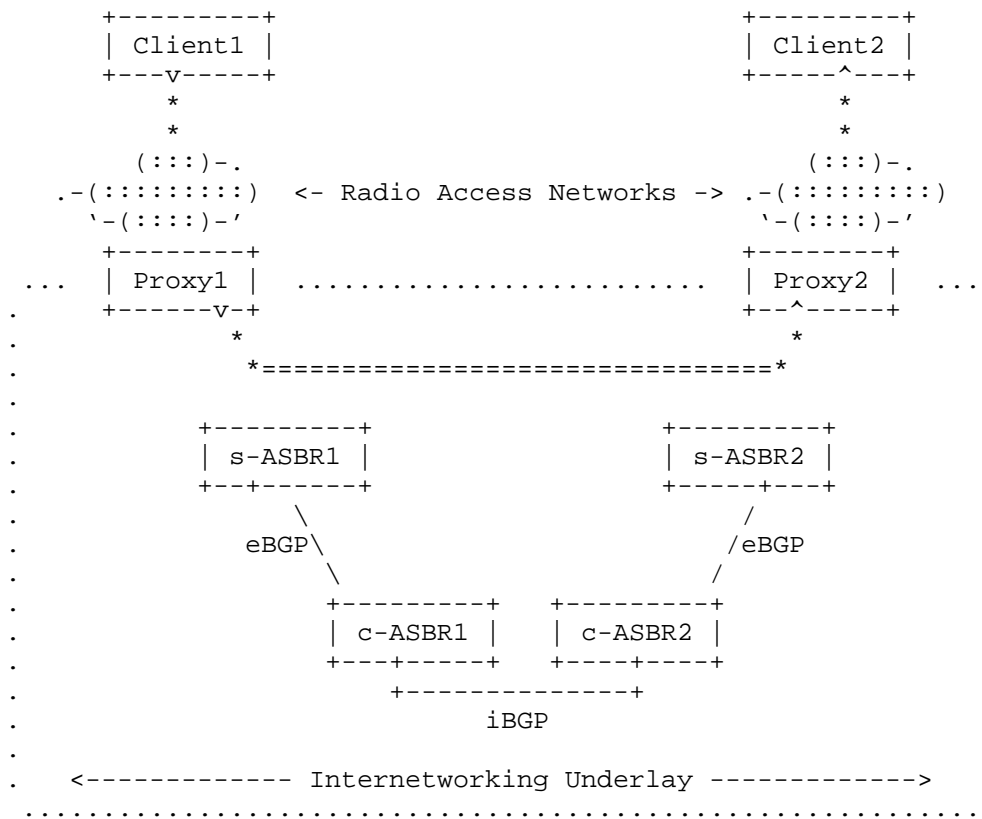


Figure 4: Optimized Route

## 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)



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 10, 2019

R. Bush  
Arrcus & IIJ  
R. Austein  
K. Patel  
Arrcus  
November 6, 2018

Link State Over Ethernet  
draft-ymbk-lsvr-lsoe-03

Abstract

Used in Massive Data Centers (MDCs), BGP-SPF and similar protocols need link neighbor discovery, link encapsulation data, and Layer 2 liveness. The Link State Over Ethernet protocol provides link discovery, exchanges supported encapsulations (IPv4, IPv6, ...), discovers encapsulation addresses (Layer 3 / MPLS identifiers) over raw Ethernet, and provides layer 2 liveness checking. The interface data are pushed directly to a BGP-LS API, obviating the need for centralized controller architectures. This protocol is intended to be more widely applicable to other upper layer routing protocols which need link discovery and characterisation.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in [RFC2119] only when they appear in all upper case. They may also appear in lower or mixed case as English words, without normative meaning. See [RFC8174].

Status of This Memo

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

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

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

This Internet-Draft will expire on May 10, 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 . . . . .	3
2. Terminology . . . . .	4
3. Background . . . . .	4
4. Top Level Overview . . . . .	5
5. Ethernet to Ethernet Protocols . . . . .	6
5.1. Inter-Link Ether Protocol Overview . . . . .	6
6. Transport Layer . . . . .	8
7. The Checksum . . . . .	8
8. TLV PDUs . . . . .	10
9. HELLO . . . . .	10
10. OPEN . . . . .	11
11. ACK . . . . .	13
11.1. Retransmission . . . . .	13
12. The Encapsulations . . . . .	13
12.1. The Encapsulation PDU Skeleton . . . . .	14
12.2. Prim/Loop Flags . . . . .	15
12.3. IPv4 Encapsulation . . . . .	15
12.4. IPv6 Encapsulation . . . . .	16
12.5. MPLS Label List . . . . .	16
12.6. MPLS IPv4 Encapsulation . . . . .	16
12.7. MPLS IPv6 Encapsulation . . . . .	17
13. KEEPALIVE - Layer 2 Liveness . . . . .	18
14. Layers 2.5 and 3 Liveness . . . . .	19
15. The North/South Protocol . . . . .	19
15.1. Use BGP-LS as Much as Possible . . . . .	19
15.2. Extensions to BGP-LS . . . . .	20
16. Discussion . . . . .	20
16.1. HELLO Discussion . . . . .	20
16.2. HELLO versus KEEPALIVE . . . . .	20
17. Open Issues . . . . .	21
18. Security Considerations . . . . .	21

19. IANA Considerations . . . . .	21
20. IEEE Considerations . . . . .	22
21. Acknowledgments . . . . .	22
22. References . . . . .	22
22.1. Normative References . . . . .	22
22.2. Informative References . . . . .	23
Authors' Addresses . . . . .	24

## 1. Introduction

The Massive Data Center (MDC) environment presents unusual problems of scale, e.g.  $O(10,000)$  devices, while its homogeneity presents opportunities for simple approaches. Approaches such as Jupiter Rising [JUPITER] use a central controller to deal with scaling, while BGP-SPF [I-D.ietf-lsvr-bgp-spf] provides massive scale-out without centralization using a tried and tested scalable distributed control plane, offering a scalable routing solution in Clos and similar environments. But BGP-SPF and similar higher level device-spanning protocols need link state and addressing data from the network to build the routing topology. LLDP has scaling issues, e.g. in extending a message beyond 1,500 bytes.

Link State Over Ethernet (LSOE) provides brutally simple mechanisms for devices to

- o Discover each other's Layer 2 (MAC) Addresses,
- o Run Layer 2 keep-alive messages for liveness continuity,
- o Discover each other's unique IDs (ASN, RouterID, ...),
- o Discover mutually supported encapsulations, e.g. IP/MPLS,
- o Discover Layer 3 and/or MPLS addressing of interfaces of the link encapsulations,
- o Enable layer 3 link liveness such as BFD, and finally
- o Present these data, using a very restricted profile of a BGP-LS [RFC7752] API, to BGP-SPF which computes the topology and builds routing and forwarding tables.

This protocol may be more widely applicable to a range of routing and similar protocols which need link discovery and characterisation.

## 2. Terminology

Even though it concentrates on the Ethernet layer, this document relies heavily on routing terminology. The following are some possibly confusing terms:

Association: An established, vis OPEN PDUs, session between two LSOE capable devices,

ASN: Autonomous System Number [RFC4271], a BGP identifier for an originator of Layer 3 routes, particularly BGP announcements.

BGP-LS: A mechanism by which link-state and TE information can be collected from networks and shared with external components using the BGP routing protocol. See [RFC7752].

BGP-SPF: A hybrid protocol using BGP transport but a Dijkstra SPF decision process. See [I-D.ietf-lsvr-bgp-spf].

Clos: A hierarchic subset of a crossbar switch topology commonly used in data centers.

Datagram: The LSOE content of a single Ethernet frame. A full LSOE PDU may be packaged in multiple Datagrams.

Encapsulation: Address Family Indicator and Subsequent Address Family Indicator (AFI/SAFI). I.e. classes of addresses such as IPv4, IPv6, MPLS, ...

Frame: An Ethernet Layer 2 packet.

MAC Address: Media Access Control Address, essentially an Ethernet address, six octets.

MDC: Massive Data Center, commonly thousands of TORs.

PDU: Protocol Data Unit, an LSOE application layer message. A PDU may need to be broken into multiple Datagrams to make it through MTU or other restrictions.

RouterID: An 32-bit identifier unique in the current routing domain, see [RFC4271] updated by [RFC6286].

SPF: Shortest Path First, an algorithm for finding the shortest paths between nodes in a graph; AKA Dijkstra's algorithm.

TOR: Top Of Rack switch, aggregates the servers in a rack and connects to aggregation layers of the Clos tree, AKA the Clos spine.

ZTP: Zero Touch Provisioning gives devices initial addresses, credentials, etc. on boot/restart.

## 3. Background

LSOE assumes a datacenter scale and topology, but can accommodate richer topologies which contain potential cycles.

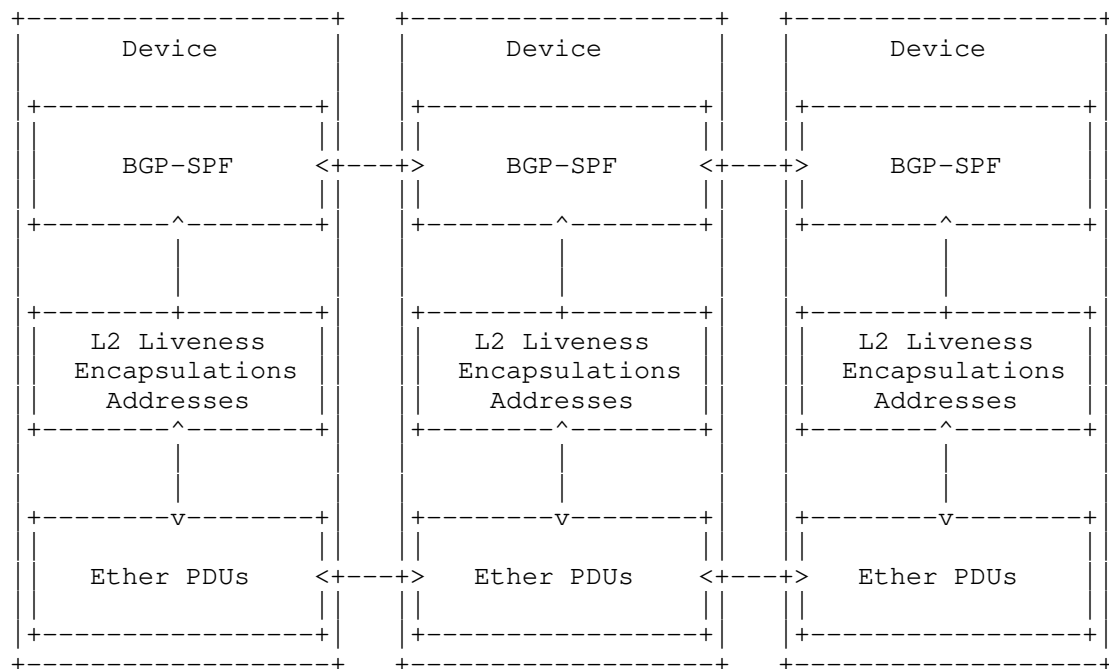
While LSOE is designed for the MDC, there are no inherent reasons it could not run on a WAN; though, as it is simply a discovery protocol, it is not clear that this would be useful. The authentication and

authorisation needed to run safely on the WAN are not provided in detail in this version of the protocol, although future versions/extensions could expend on them.

LSOE assumes a new IEEE assigned EtherType (TBD).

#### 4. Top Level Overview

- o Devices discover each other on Ethernet links
- o MAC addresses and Link State are exchanged over Ethernet
- o Layer 2 Liveness Checks are begun
- o Encapsulation data are exchanged and IP-Level Liveness Checks done
- o A BGP-like protocol is assumed to use these data to discover and build a topology database



There are two protocols, the Ethernet discovery and the interface to the upper level BGP-like protocol:

- o Layer 2 Ethernet protocols are used to exchange Layer 2 data, i.e. MAC addresses, and layer 2.5 and 3 identifiers (not payloads), i.e. ASNs, Encapsulations, and interface addresses.
- o A Link Layer to BGP API presents these data up the stack to a BGP protocol or an other device-spanning upper layer protocol, presenting them using the BGP-LS BGP-like data format.

The upper layer BGP family routing protocols cross all the devices, though they are not part of these LSOE protocols.

To simplify this document, Layer 2 Ethernet framing is not shown.

## 5. Ethernet to Ethernet Protocols

Two devices discover each other and their respective MAC addresses by sending multicast HELLO PDUs (Section 9). To allow discovery of new devices coming up on a multi-link topology, devices send periodic HELLOs forever, see Section 16.1.

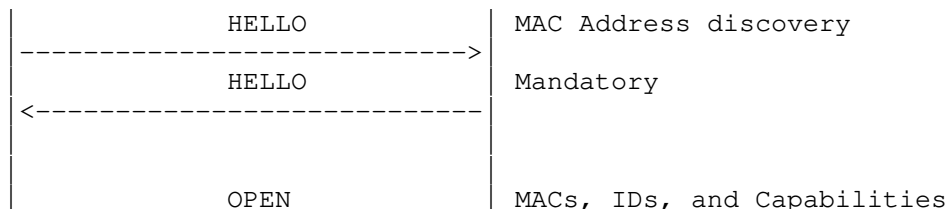
Once a new device is recognized, both devices attempt to negotiate and establish peering by sending unicast OPEN PDUs (Section 10). In an established peering, Encapsulations (Section 12) may be announced and modified. When two devices on a link have compatible Encapsulations and addresses, i.e. the same AFI/SAFI and the same subnet, the link is announced via the BGP-LS API.

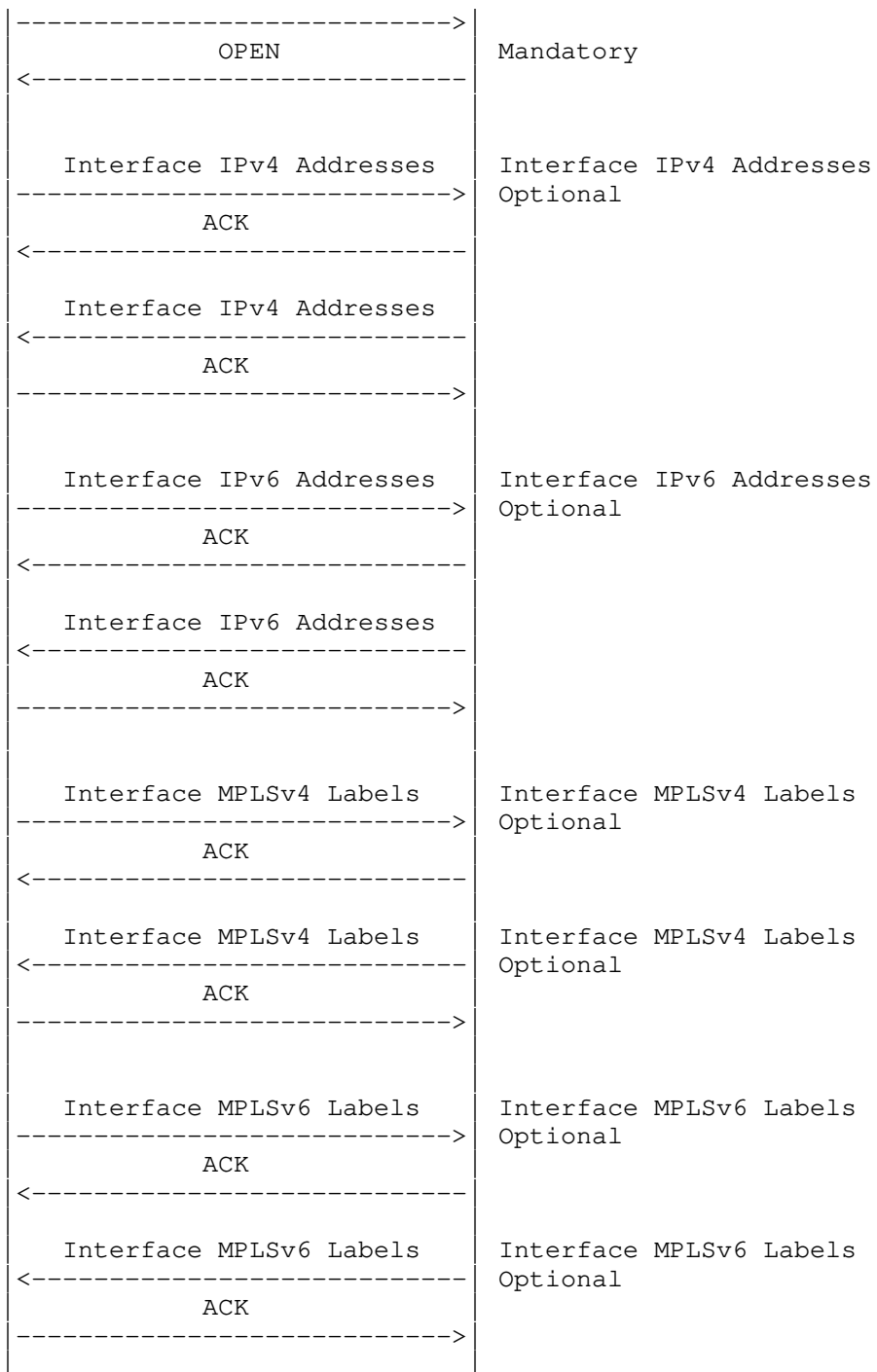
### 5.1. Inter-Link Ether Protocol Overview

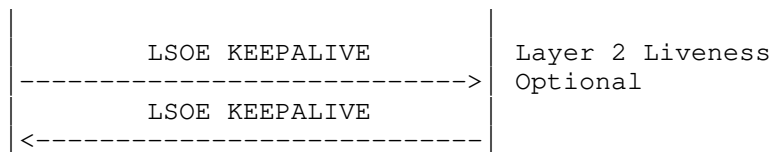
The HELLO, Section 9, is a priming message. It is an Ethernet multicast frame with a small LSOE PDU with the simple goal of discovering the Ethernet MAC address(es) of devices reachable via an interface.

The HELLO and OPEN, Section 10, PDUs, which are used to discover and exchange MAC address and IDs, are mandatory; other PDUs are optional; though at least one encapsulation MUST be agreed at some point.

The following is a ladder-style sketch of the Ethernet protocol exchanges:



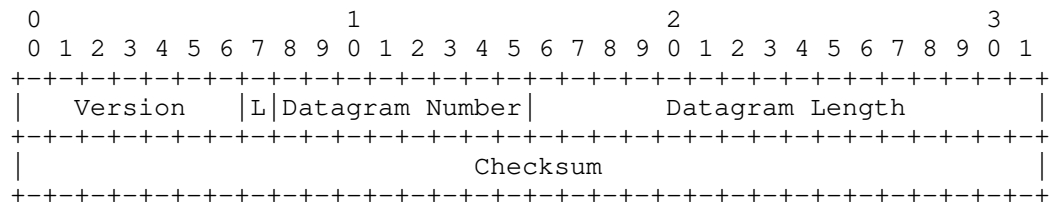




## 6. Transport Layer

LSOE PDU are carried by a simple transport layer which allows long PDUs to occupy multiple Ethernet frames. The LSOE data in each frame is referred to as a Datagram.

The LSOE Transport Layer encapsulates each Datagram using a common transport header.



The fields of the LSOE Transport Header are as follows:

**Version:** Version number of the protocol, currently 0. Values other than 0 are treated as failure.

**Datagram Number:** 0..255, a monotonically increasing value, modulo 256, see [RFC1982].

**L:** A bit that set to 1 if this Datagram is the last Datagram of the PDU. For a PDU which fits in only one Datagram, it is set to one.

**PDU Length:** Total number of octets in the Datagram including all payloads and fields.

**Checksum:** A 32 bit hash over the Datagram to detect bit flips, see Section 7.

## 7. The Checksum

There is a reason conservative folk use a checksum in UDP. And as many operators stretch to jumbo frames (over 1,500 octets) longer checksums are the conservative approach.

For the purpose of computing a checksum, the checksum field itself is assumed to be zero.



Sum up 32-bit unsigned ints in a 64-bit long, then take the high-order section, shift it right, rotate, add it in, repeat until zero.

```
#include <stdint.h>
#include <stdint.h>

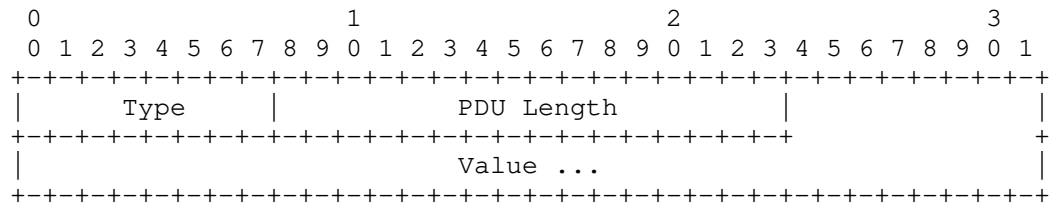
/* The F table from Skipjack, and it would work for the S-Box. */
static const uint8_t sbox[256] = {
0xa3,0xd7,0x09,0x83,0xf8,0x48,0xf6,0xf4,0xb3,0x21,0x15,0x78,
0x99,0xb1,0xaf,0xf9,0xe7,0x2d,0x4d,0x8a,0xce,0x4c,0xca,0x2e,
0x52,0x95,0xd9,0x1e,0x4e,0x38,0x44,0x28,0x0a,0xdf,0x02,0xa0,
0x17,0xf1,0x60,0x68,0x12,0xb7,0x7a,0xc3,0xe9,0xfa,0x3d,0x53,
0x96,0x84,0x6b,0xba,0xf2,0x63,0x9a,0x19,0x7c,0xae,0xe5,0xf5,
0xf7,0x16,0x6a,0xa2,0x39,0xb6,0x7b,0x0f,0xc1,0x93,0x81,0x1b,
0xee,0xb4,0x1a,0xea,0xd0,0x91,0x2f,0xb8,0x55,0xb9,0xda,0x85,
0x3f,0x41,0xbf,0xe0,0x5a,0x58,0x80,0x5f,0x66,0x0b,0xd8,0x90,
0x35,0xd5,0xc0,0xa7,0x33,0x06,0x65,0x69,0x45,0x00,0x94,0x56,
0x6d,0x98,0x9b,0x76,0x97,0xfc,0xb2,0xc2,0xb0,0xfe,0xdb,0x20,
0xe1,0xeb,0xd6,0xe4,0xdd,0x47,0x4a,0x1d,0x42,0xed,0x9e,0x6e,
0x49,0x3c,0xcd,0x43,0x27,0xd2,0x07,0xd4,0xde,0xc7,0x67,0x18,
0x89,0xcb,0x30,0x1f,0x8d,0xc6,0x8f,0xaa,0xc8,0x74,0xdc,0xc9,
0x5d,0x5c,0x31,0xa4,0x70,0x88,0x61,0x2c,0x9f,0x0d,0x2b,0x87,
0x50,0x82,0x54,0x64,0x26,0x7d,0x03,0x40,0x34,0x4b,0x1c,0x73,
0xd1,0xc4,0xfd,0x3b,0xcc,0xfb,0x7f,0xab,0xe6,0x3e,0x5b,0xa5,
0xad,0x04,0x23,0x9c,0x14,0x51,0x22,0xf0,0x29,0x79,0x71,0x7e,
0xff,0x8c,0x0e,0xe2,0x0c,0xef,0xbc,0x72,0x75,0x6f,0x37,0xa1,
0xec,0xd3,0x8e,0x62,0x8b,0x86,0x10,0xe8,0x08,0x77,0x11,0xbe,
0x92,0x4f,0x24,0xc5,0x32,0x36,0x9d,0xcf,0xf3,0xa6,0xbb,0xac,
0x5e,0x6c,0xa9,0x13,0x57,0x25,0xb5,0xe3,0xbd,0xa8,0x3a,0x01,
0x05,0x59,0x2a,0x46
};

/* non-normative example C code, constant time even */

uint32_t sbox_checksum_32(const uint8_t *b, const size_t n)
{
    uint32_t sum[4] = {0, 0, 0, 0};
    uint64_t result = 0;
    for (size_t i = 0; i < n; i++)
        sum[i & 3] += sbox[*b++];
    for (int i = 0; i < sizeof(sum)/sizeof(*sum); i++)
        result = (result << 8) + sum[i];
    result = (result >> 32) + (result & 0xFFFFFFFF);
    result = (result >> 32) + (result & 0xFFFFFFFF);
    return (uint32_t) result;
}
```

## 8. TLV PDUs

The basic LSOE application layer PDU is a typical TLV (Type Length Value) PDU. It may be broken into multiple Datagrams, see Section 6



The fields of the basic LSOE header are as follows:

Type: An integer differentiating PDU payload types

- 0 - HELLO
- 1 - OPEN
- 2 - KEEPALIVE
- 3 - ACK
- 4 - IPv4 Announcement
- 5 - IPv6 Announcement
- 6 - MPLS IPv4 Announcement
- 7 - MPLS IPv6 Announcement
- 8-255 Reserved

PDU Length: Total number of octets in the PDU including all payloads and fields

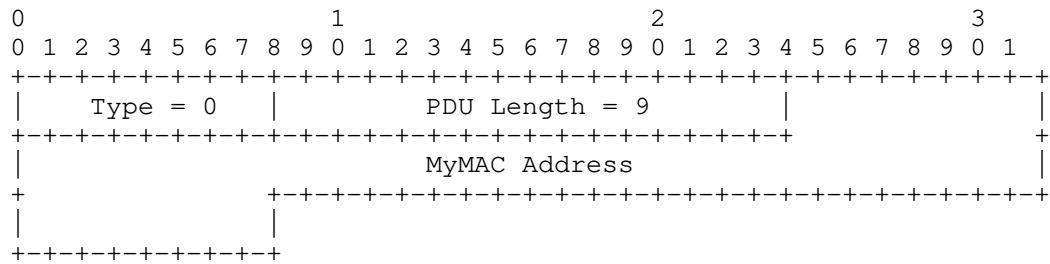
Value: Any application layer content of the LSOE PDU beyond the type.

## 9. HELLO

The HELLO PDU is unique in that it is a multicast Ethernet frame. It solicits response(s) from other device(s) on the link. See Section 16.1 for why multicast is used.

All other LSOE PDUs are unicast Ethernet frames, as the peer's MAC Address is known after the HELLO exchange.

When an interface is turned up on a device, it SHOULD issue a HELLO periodically. The interval is set by configuration.



If more than one device responds, one adjacency is formed for each unique (MAC address) response. LSOE treats the adjacencies as separate links.

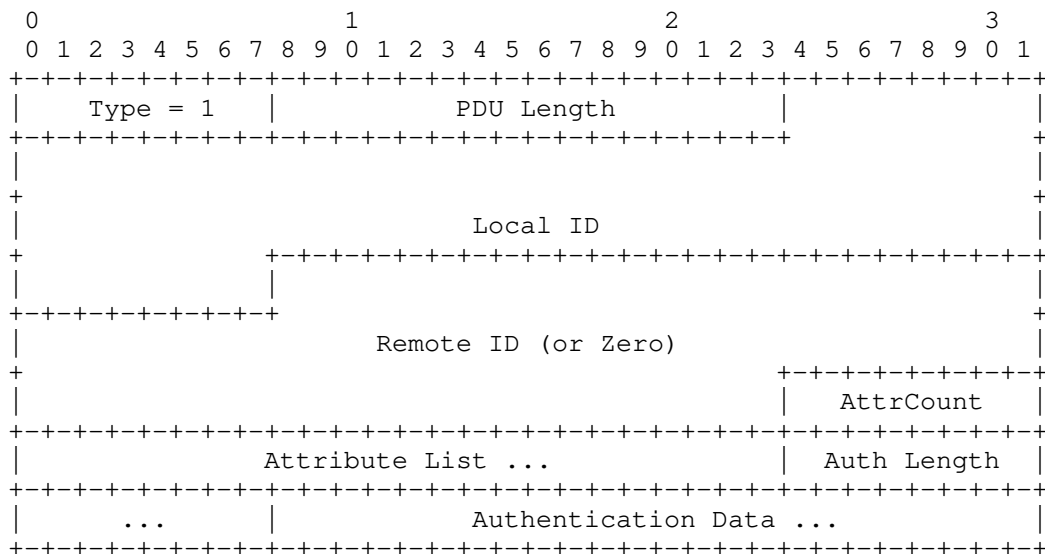
When a HELLO is received from a MAC address where there is no established LSOE adjacency, the receiver SHOULD respond with an OPEN PDU. The two devices establish an LSOE adjacency by exchanging OPEN PDUs.

The PDU Length is the octet count of the entire PDU, including the Type, the Datagram Length field itself, and the MyMAC Address payload.

A particular MAC address SHOULD arrive on frames from only one interface.

#### 10. OPEN

Each device has learned the other's MAC address from the HELLO exchange, see Section 9. Therefore the OPEN and subsequent PDUs are unicast, as opposed to the HELLO's multicast, Ethernet frames.



An ID can be an ASN with high order bits zero, a classic RouterID with high order bits zero, a catenation of the two, a 80-bit ISO System-ID, or any other identifier unique to a single device in the current routing space. IDs are big-endian.

When the local device sends an OPEN without knowing the remote device's ID, the Remote ID MUST be zero. The Local ID MUST NOT be zero.

AttrCount is the number of attributes in the Attribute List. Attributes are single octets whose semantics are user-defined.

A node may have zero or more user-defined attributes, e.g. spine, leaf, backbone, route reflector, arabica, ...

Attribute syntax and semantics are local to an operator or datacenter; hence there is no global registry. Nodes exchange their attributes only in the OPEN PDU.

Auth Length is a 16-bit field denoting the length in octets of the Authentication Data, not including the Auth Length itself. If there are no Authentication Data, the Auth Length MUST BE zero.

The Authentication Data are specific to the operational environment. A failure to authenticate is a failure to start the LSOE association, and HELLOs MUST BE restarted.

Once two devices know each other's MAC addresses, and have ACKed each other's OPEN PDUs, Layer 2 KEEPALIVES (see Section 13) SHOULD be started to ensure Layer 2 liveness and keep the association semantics alive. The timing and acceptable drop of the KEEPALIVE PDUs SHOULD be configured.

If a properly authenticated OPEN arrives from a device with which the receiving device believes it already has an LSOE association (OPENs have already been exchanged), the receiver MUST assume that the sending device has been reset. All discovered data MUST BE withdrawn via the BGP-LS API and the recipient MUST respond with a new OPEN.

## 11. ACK

```

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

```

The ACK acknowledges receipt of an OPEN or an Encapsulation PDU.

The PDU Type is the Type of the PDU being acknowledged, OPEN or one of the Encapsulations.

### 11.1. Retransmission

If a PDU sender expects an ACK, e.g. for an OPEN or an Encapsulation, and does not receive the ACK for a configurable time (default one second), the sender resends the PDU. This cycle MAY be repeated a configurable number of times (default three) before it is considered a failure. The session is considered closed in case of an ACK failure.

## 12. The Encapsulations

Once the devices know each other's MAC addresses, know each other's upper layer identities, have means to ensure link state, etc., the LSOE 'association' is considered established, and the devices SHOULD announce their interface encapsulation, addresses, (and labels).

The Encapsulation types the peers exchange may be IPv4 Announcement (Section 12.3), IPv6 Announcement (Section 12.4), MPLS IPv4 Announcement (Section 12.6), MPLS IPv6 Announcement (Section 12.7), and/or possibly others not defined here.

The sender of an Encapsulation PDU MUST NOT assume that the peer is capable of the same Encapsulation Type. An ACK (Section 11) merely

acknowledges receipt. Only if both peers have sent the same Encapsulation Type is it safe to assume that they are compatible for that type.

Further, to consider a link of a type to formally be established so that it may be pushed up to upper layer protocols, the addressing for the type must be compatible, i.e. on the same IPvX subnet.

## 12.1. The Encapsulation PDU Skeleton

The header for all encapsulation PDUs is as follows:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Type										PDU Length										Count																			
...										Encapsulation List...																													

The 16-bit Count is the number of Encapsulations in the Encapsulation list.

If the length of an Encapsulation PDU exceeds the Datagram size limit on media, the PDU is broken into multiple Datagrams. See Section 8.

The Receiver MUST acknowledge the Encapsulation PDU with a Type=3, ACK PDU (Section 11) with the Encapsulation Type being that of the encapsulation being announced, see Section 11.

If the Sender does not receive an ACK in one second, they SHOULD retransmit. After a user configurable number of failures, the LSOE association should be considered dead and the OPEN process SHOULD be restarted.

An Encapsulation PDU describes zero or more addresses of the encapsulation type.

An Encapsulation PDU of Type T replaces all previous encapsulations of Type T.

To remove all encapsulations of Type T, the sender uses a Count of zero.

If an interface has multiple addresses for an encapsulation type, one address SHOULD be marked as primary, see Section 12.2.

If a loopback address needs to be exposed, e.g. for iBGP peering, then it should be marked as such, see Section 12.2.

If a sender has multiple links on the same interface, separate data, ACKs, etc. must be kept for each peer.

Over time, multiple Encapsulation PDUs may be sent for an interface as configuration changes.

## 12.2. Prim/Loop Flags

0	1	2	3	...	7
-----	-----	-----	-----	-----	-----
Primary	Loopback	Reserved ...			
-----	-----	-----	-----	-----	-----

Each Encapsulation interface address MAY be marked as a primary address, and/or a loopback, in which case the respective bit is set to one.

Only one address MAY be marked as primary for an encapsulation type.

## 12.3. IPv4 Encapsulation

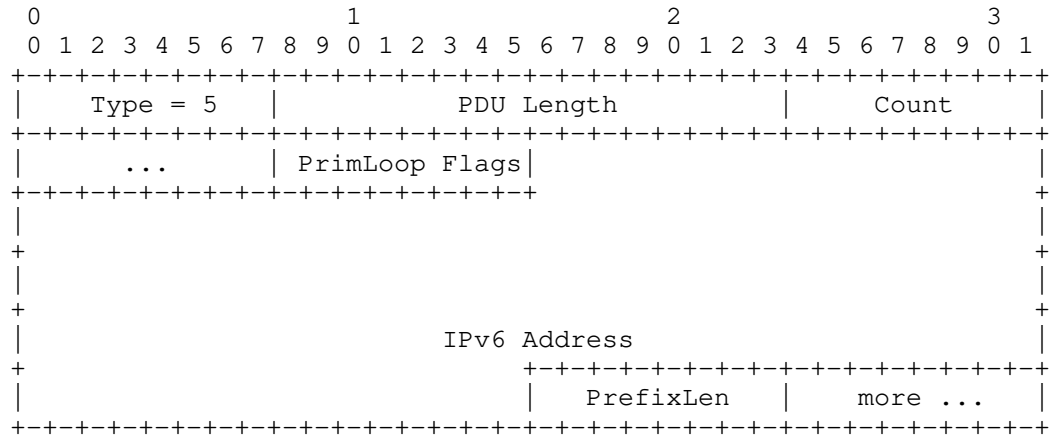
The IPv4 Encapsulation describes a device's ability to exchange IPv4 packets on one or more subnets. It does so by stating the interface's address and the prefix length.

0	1	2	3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1			
-----	-----	-----	-----
Type = 4	PDU Length	Count	
-----	-----	-----	-----
...	PrimLoop Flags	IPv4 Address	
-----	-----	-----	-----
	PrefixLen	PrimLoop Flags	
-----	-----	-----	-----
	IPv4 Address		
-----	-----	-----	-----
PrefixLen	more ...		
-----	-----	-----	-----

The 16-bit Count is the number of IPv4 Encapsulations.

#### 12.4. IPv6 Encapsulation

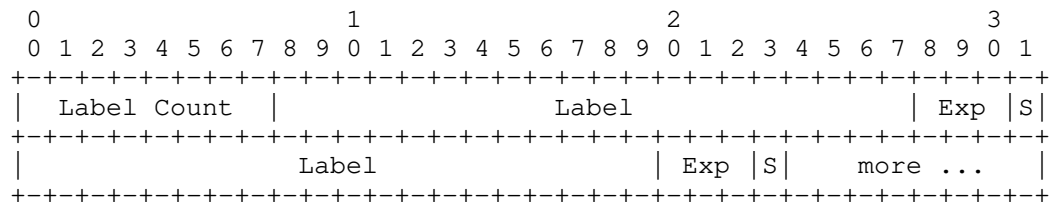
The IPv6 Encapsulation describes a device's ability to exchange IPv6 packets on one or more subnets. It does so by stating the interface's address and the prefix length.



The 16-bit Count is the number of IPv6 Encapsulations.

#### 12.5. MPLS Label List

As an MPLS enabled interface may have a label stack, see [RFC3032], a variable length list of labels is needed.

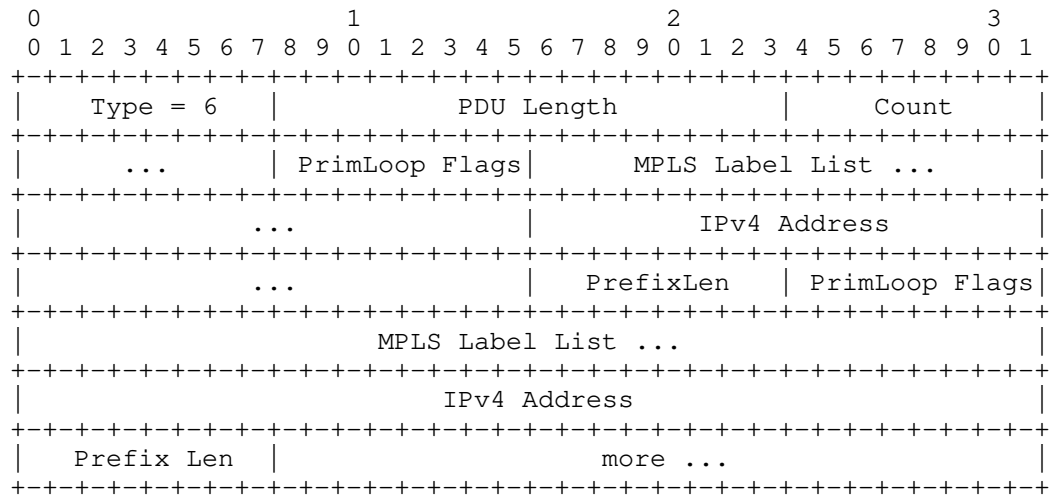


A Label Count of zero is an implicit withdraw of all labels for that prefix on that interface.

#### 12.6. MPLS IPv4 Encapsulation

The MPLS IPv4 Encapsulation describes a device's ability to exchange labeled IPv4 packets on one or more subnets. It does so by stating the interface's address and the prefix length.

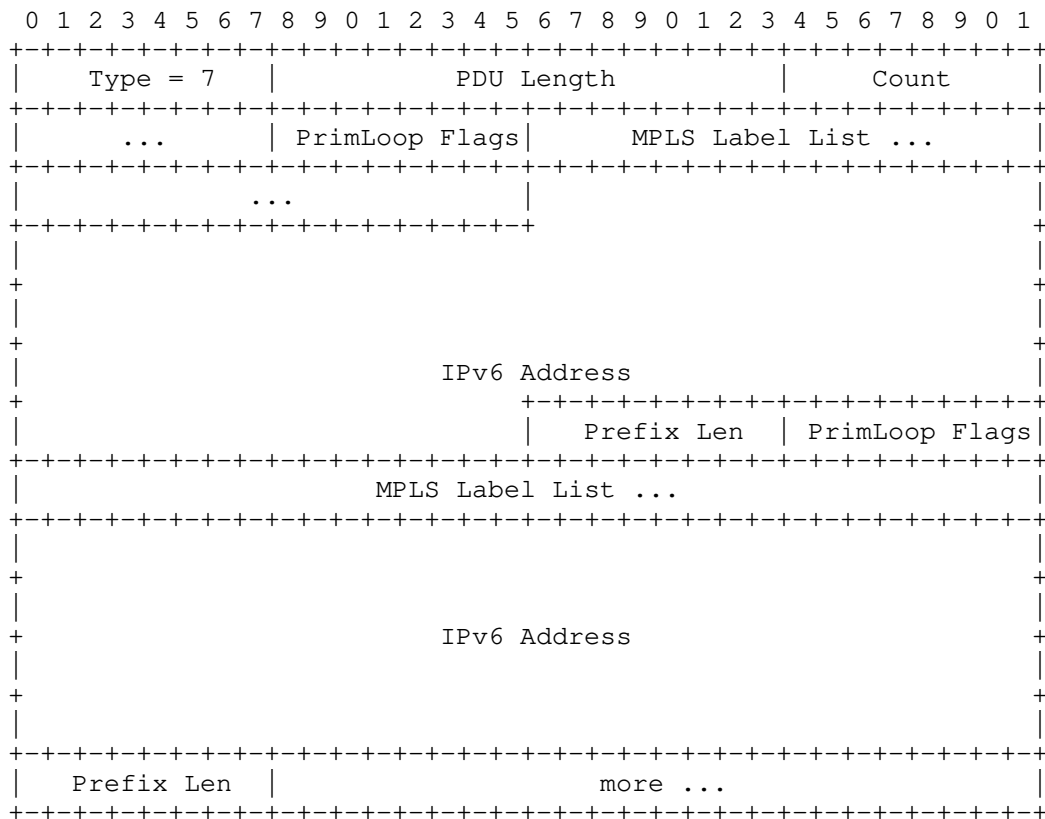




The 16-bit Count is the number of MPLSv6 Encapsulations.

#### 12.7. MPLS IPv6 Encapsulation

The MPLS IPv6 Encapsulation describes a device's ability to exchange labeled IPv6 packets on one or more subnets. It does so by stating the interface's address and the prefix length.



The 16-bit Count is the number of MPLSv6 Encapsulations.

### 13. KEEPALIVE - Layer 2 Liveness

LSOE devices MUST beacon occasional Layer 2 KEEPALIVE PDUs to ensure association continuity.

They SHOULD be beacons at a configured frequency. One per second is the default. Layer 3 liveness, such as BFD, will likely be more aggressive.

If a KEEPALIVE is not received from a peer with which a receiver has an open session for a configurable time (default one minute), the session SHOULD BE presumed closed. The devices MAY keep configuration state until a new session is established and new Encapsulation PDUs are received.

```

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

```

#### 14. Layers 2.5 and 3 Liveness

Ethernet liveness is continuously tested by KEEPALIVE PDUs, see Section 13. As layer 2.5 or layer 3 connectivity could still break, liveness above layer 2 SHOULD be frequently tested using BFD ([RFC5880]) or a similar technique.

This protocol assumes that one or more Encapsulation addresses will be used to ping, BFD, or whatever the operator configures.

#### 15. The North/South Protocol

Thus far, a one-hop point-to-point link discovery protocol has been defined.

The nodes know the unique node identifiers (ASNs, RouterIDs, ...) and Encapsulations on each link interface.

Full topology discovery is not appropriate at the Ethernet layer, so Dijkstra a la IS-IS etc. is assumed to be done by higher level protocols.

Therefore the node identifiers, link Encapsulations, and state changes are pushed North via a small subset of the BGP-LS API. The upper layer routing protocol(s), e.g. BGP-SPF, learn and maintain the topology, run Dijkstra, and build the routing database(s).

For example, if a neighbor's IPv4 Encapsulation address changes, the devices seeing the change push that change Northbound.

##### 15.1. Use BGP-LS as Much as Possible

BGP-LS [RFC7752] defines BGP-like Datagrams describing link state (links, nodes, link prefixes, and many other things), and a new BGP path attribute providing Northbound transport, all of which can be ingested by upper layer protocols such as BGP-SPF; see Section 4 of [I-D.ietf-lsvr-bgp-spf].

For IPv4 links, TLVs 259 and 260 are used. For IPv6 links, TLVs 261 and 262. If there are multiple addresses on a link, multiple TLV pairs are pushed North, having the same ID pairs.

## 15.2. Extensions to BGP-LS

The Northbound protocol needs a few minor extensions to BGP-LS. Luckily, others have needed the same extensions.

Similarly to BGP-SPF, the BGP protocol is used in the Protocol-ID field specified in table 1 of [I-D.ietf-idr-bgppls-segment-routing-epe]. The local and remote node descriptors for all NLRI are the ID's described in Section 10. This is equivalent to an adjacency SID or a node SID if the address is a loopback address.

Label Sub-TLVs from [I-D.ietf-idr-bgp-ls-segment-routing-ext] Section 2.1.1, are used to associate one or more MPLS Labels with a link.

## 16. Discussion

This section explores some trade-offs taken and some considerations.

### 16.1. HELLO Discussion

There is the question of whether to allow an intermediate switch to be transparent to discovery. We consider that an interface on a device is a Layer 2 or a Layer 3 interface. In theory it could be a Layer 3 interface with no encapsulation or Layer 3 addressing currently configured.

A device with multiple Layer 2 interfaces, traditionally called a switch, may be used to forward frames and therefore packets from multiple devices to one interface, I, on an LSOE speaking device. Interface I could discover a peer J across the switch. Later, a prospective peer K could come up across the switch. If I was not still sending and listening for HELLOs, the potential peering with K could not be discovered. Therefore, interfaces MUST continue to send HELLOs as long as they are turned up.

### 16.2. HELLO versus KEEPALIVE

Both HELLO and KEEPALIVE are periodic. KEEPALIVE might be eliminated in favor of keeping only HELLOs. But currently KEEPALIVE is unicast, has a checksum, is acknowledged, and thus more firmly verifies association existence.

This warrants discussion.

## 17. Open Issues

VLANs/SVIs/Subinterfaces

## 18. Security Considerations

The protocol as is MUST NOT be used outside a datacenter or similarly closed environment due to lack of formal definition of the authentication and authorisation mechanism. These will be worked on in a later effort, likely using credentials configured using ZTP or similar configuration automation.

Many MDC operators have a strange belief that physical walls and firewalls provide sufficient security. This is not credible. All MDC protocols need to be examined for exposure and attack surface.

It is generally unwise to assume that on the wire Ethernet is secure. Strange/unauthorized devices may plug into a port. Mis-wiring is very common in datacenter installations. A poisoned laptop might be plugged into a device's port.

Malicious nodes/devices could mis-announce addressing, form malicious associations, etc.

For these reasons, the OPEN PDU's authentication data exchange SHOULD be used. [ A mandatory to implement authentication is in development. ]

## 19. IANA Considerations

This document requests the IANA create a registry for LSOE PDU Type, which may range from 0 to 255. The name of the registry should be LSOE-PDU-Type. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

PDU Code	PDU Name
0	HELLO
1	OPEN
2	KEEPALIVE
3	ACK
4	IPv4 Announce / Withdraw
5	IPv6 Announce / Withdraw
6	MPLS IPv4 Announce / Withdraw
7	MPLS IPv6 Announce / Withdraw
8-255	Reserved

This document requests the IANA create a registry for LSOE PL Flag Bits, which may range from 0 to 7. The name of the registry should be LSOE-PL-Flag-Bits. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

Bit	Bit Name
----	-----
0	Primary
1	Loopback
2-7	Reserved

## 20. IEEE Considerations

This document requires a new EtherType.

## 21. Acknowledgments

The authors thank Cristel Pelsser for multiple reviews, Jeff Haas for review and comments, Joe Clarke for a useful review, John Scudder deeply serious review and comments, Larry Kreeger for a lot of layer 2 clue, Martijn Schmidt for his contribution, Russ Housley for checksum discussion and sBox, and Steve Bellovin for checksum advice.

## 22. References

### 22.1. Normative References

[I-D.ietf-idr-bgp-ls-segment-routing-ext]

Previdi, S., Talaulikar, K., Filsfils, C., Gredler, H., and M. Chen, "BGP Link-State extensions for Segment Routing", draft-ietf-idr-bgp-ls-segment-routing-ext-11 (work in progress), October 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]

Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-17 (work in progress), October 2018.

[I-D.ietf-lsvr-bgp-spf]

Patel, K., Lindem, A., Zandi, S., and W. Henderickx, "Shortest Path Routing Extensions for BGP Protocol", draft-ietf-lsvr-bgp-spf-03 (work in progress), September 2018.

- [RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, DOI 10.17487/RFC1982, August 1996, <<http://www.rfc-editor.org/info/rfc1982>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <<http://www.rfc-editor.org/info/rfc3032>>.
- [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, <<http://www.rfc-editor.org/info/rfc4271>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<http://www.rfc-editor.org/info/rfc5880>>.
- [RFC6286] Chen, E. and J. Yuan, "Autonomous-System-Wide Unique BGP Identifier for BGP-4", RFC 6286, DOI 10.17487/RFC6286, June 2011, <<http://www.rfc-editor.org/info/rfc6286>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<http://www.rfc-editor.org/info/rfc7752>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<http://www.rfc-editor.org/info/rfc8174>>.

## 22.2. Informative References

[JUPITER] Singh, A., Germano, P., Kanagala, A., Liu, H., Provost, J., Simmons, J., Tanda, E., Wanderer, J., HAP.1zle, U., Stuart, S., Vahdat, A., Ong, J., Agarwal, A., Anderson, G., Armistead, A., Bannon, R., Boving, S., Desai, G., and B. Felderman, "Jupiter rising", Communications of the ACM Vol. 59, pp. 88-97, DOI 10.1145/2975159, August 2016.

#### Authors' Addresses

Randy Bush  
Arrcus & IIJ  
5147 Crystal Springs  
Bainbridge Island, WA 98110  
United States of America

Email: randy@psg.com

Rob Austein  
Arrcus, Inc

Email: sra@hactrn.net

Keyur Patel  
Arrcus  
2077 Gateway Place, Suite #400  
San Jose, CA 95119  
United States of America

Email: keyur@arrcus.com