Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 6, 2021

Y. Gu
S. Chen
Huawei
H. Chen
China Telecom
Z. Li
Huawei

November 2, 2020

Network Monitoring For IGP
draft-gu-opsawg-network-monitoring-igp-00

Abstract

To evolve towards automated network OAM (Operations, administration and management), the monitoring of control plane protocols is a fundamental necessity. This document proposes network monitoring for IGP to facilitate troubleshooting by collecting the IGP monitoring data and reporting it to the network monitoring server in real-time. In this document, the operations of network monitoring for ISIS are described, and the corresponding network monitoring message types and message formats are defined.

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 May 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

1.1. Motivation

The requirement for better network OAM approaches has been greatly driven by the network evolvement. The concept of network Telemetry has been proposed to meet the current and future OAM demands w.r.t., massive and real-time data storage, collection, process, export, and analysis, and an architectural framework of existing Telemetry approaches is introduced in [I-D.song-ntf]. Network Telemetry provides visibility to the network health conditions, and is beneficial for faster network troubleshooting, network OpEx (operating expenditure) reduction, and network optimization. Telemetry can be applied to the data plane, control plane and management plane. There have been various methods proposed for each plane:

- o Management plane: For example, SNMP (Simple Network Management Protocol) [RFC1157], NETCONF (Network Configuration Protocol) [RFC6241] and gNMI (gRPC Network Management Interface) [I-D.openconfig-rtgwg-gnmi-spec] are three typical widely adopted management plane Telemetry approaches. Various YANG modules are defined for network operational state retrieval and configuration management. Subscription to specific YANG datastore can be realized in combination with gRPC/NETCONF.

- o Data plane: For example, In-situ OAM (iOAM) [I-D.brockners-inband-oam-requirements] embeds an instruction header to the user data packets, and collects the requested data and adds it to the use packet at each network node along the forwarding path. Applications such as path verification, SLA (service-level agreement) assurance can be enabled with iOAM.

- o Control Plane: BGP monitoring protocol (BMP) [RFC7854] is proposed to monitor BGP sessions and intended to provide a convenient interface for obtaining BGP route views. Date collected using BMP can be further analyzed with big data platforms for network health condition visualization, diagnose and prediction applications.

The general idea of most Telemetry approaches is to collect various information from devices and export to the centralized server for further analysis, and thus providing more network insight. It should not be surprising that any future and even current Telemetry applications may require the fusion of data acquired from more than one single approach/one single plane. For example, for network troubleshooting purposes, it requires the collection of comprehensive information from devices, such system ID/router ID, interface status, PDUs (protocol data units), device/protocol statistics and so on. Information such as system ID/router ID can be reported by management plane Telemetry approaches, while the protocol related data (especially PDUs) are more fit to be monitored using the control plane Telemetry. With rich information collected in real time at the centralized server, network issues can be localized faster and more accurately, and the root cause analysis can be also provided.

The conventional troubleshooting logic is to log in a faulty router, physically or through Telnet, and by using CLI to display related information/logs for fault source localization and further analysis. There are several concerns with the conventional troubleshooting methods:

1. It requires rich OAM experience for the OAM operator to know what information to check on the device, and the operation is complex;

2. In a multi-vendor network, it requires the understanding and familiarity of vendor specific operations and configurations;

3. Locating the fault source device could be non-trivial work, and is often realized through network-wide device-by-device check, which is both time-consuming and labor-consuming; and finally,

4. The acquisition of troubleshooting data can be difficult under some cases, e.g., when auto recovery is used.

This document proposes the network monitoring for IGP to monitor the running state of IGP, e.g., PDUs, protocol statistics and peer status, which have not been systematically covered by any other Telemetry approach, to facilitate network troubleshooting.

1.2. Overview

Like BMP, a networking monitoring session is established between each monitored router (NM client) and the NM monitoring station (NM server) through TCP connection. Information are collected directly from each monitored router and reported to the NM server. The NM message can be both periodic and event-triggered, depending on the message type.

IS-IS [RFC1195], as one of the most commonly adopted network layer protocols, builds the fundamental network connectivity of an autonomous system (AS). The disfunction of IS-IS, e.g., IS-IS neighbor down, route flapping, MTU mismatch, and so on, could lead to network-wide instability and service interruption. Thus, it is critical to keep track of the health condition of IS-IS, and the availability of information, related to IS-IS running status, is the fundamental requirement. In this document, typical network issues are identified as the use cases of network monitoring. Then the operations and the message formats of network monitoring for IS-IS are defined. Network monitoring for OSPF will be included in the future version.

2. Terminology

IGP: Interior Gateway Protocol

IS-IS: Intermediate System to Intermediate System

NM: Network Monitoring

IMP: Network Monitoring for IGP

BMP: BGP monitoring protocol

IIH: IS-IS Hello Packet

LSP: Link State Packet

CSNP: Complete Sequence Number Packet

NSNP: Partial Sequence Number Packet

3. Use Cases

We have identified two typical network issues due to IS-IS disfunction that are currently difficult to detect or localize.

3.1. IS-IS Route Flapping

The IS-IS Route Flapping refers to the situation that one or more routes appear and then disappear in the routing table repeatedly. Route flapping usually comes with massive PDUs interactions (e.g., LSP, LSP purge...), which consume excessive network bandwidth, and excessive CPU processing. In addition, the impact is often network-wide. The localizing of the flapping source and the identifying of root causes haven't been easy work due to various reasons.

The flapping can be caused by system ID conflict, IS-IS neighborship flapping, route source flapping (caused by import route policy misconfiguration) , device clock dis-function with abnormal LSP purge (e.g., 100 times faster) and so on.

- o The system ID conflict check is a network-wide work. If such information is collected centrally to a controller/server, the issues can be identified in seconds, and more importantly, in advance of the actual flapping event.

- o The IS-IS neighborship flapping is typically caused by interface flapping, BFD flapping, CPU high and so on. Conventionally, to located the issue, operators typically identify the target device(s), and then log in the devices to check related statistics, parsed protocol PDU data and configurations. The manual check often requires a combination of multiple CLIs (check cost/next hop/exit interface/LSP age...) in a repeated manner, which is time-consuming and requires rich OAM experience. If such statistics and configuration data were collected at the server in real-time, the server may analyze them automatically or semi-automatically with troubleshooting algorithms implemented at the server.

- In the case that route policies are misconfigured, which then causes the route flapping, it's typically difficult to directly identify the responsible policy in a short time. Thus, if the route change history is recorded in correlation with the route policy, then with such record collected at the server, the server can directly identify the responsible policy with the one-to-one mapping between policy processing and the route attribute change.

- In the case that flapping comes with abnormal LSP purges, it may be due to continuous LSP corruptions with falsified shorter Remaining Lifetime, or the clock running 100 times faster with 100 times more purge LSPs generated. In order to identify the purge originator, RFC 6232 [RFC6232] proposes to carry the Purge Orginator Identification (POI) TLV in IS-IS. However, to analyze the root cause of such abnormal purges, the collection and analysis of LSP PDUs are needed.

3.2. IS-IS LSDB Synchronization Failure

During the IS-IS flooding, sometimes the LSP synchronization failure happens. The synchronization failure causes can be generally classified into three cases:

- Case 1, the LSP is not correctly advertised. For example, an LSP sent by Router A fails to be synchronized at Router B. It can be due to incorrect route export policy, or too many prefixes being advertised which exceeds the LSP/MTU threshold, and so on at Router A.

- Case 2, LSP transmission error, which is typically caused by IS-IS adjacency failure, .e.g., link down/BFD down/authentication failure.

- Case 3, the LSP is received but not correctly processed. The problem that happens at Router B can be faulty route import policy, or Router B being in Overload mode, or the hardware/software bugs.

With sufficient ISIS PDU related statistics and parsed PDU information recorded at the device, the neighborship failure in Case 2 can be typically diagnosed at Router A or Router B independently. With such diagnosing information collected (e.g., in the format of reason code) in real-time, the server can identify the root synchronization issue with much less time and labor consumption compared with conventional methods. In Case 1 & 3, the failure is mostly caused by incorrect route policy and software/hardware issue. By comparing the LSDB with the sent/received LSP, differences can be recognized. Then the difference may further guide the localization of the root cause. Thus, by collecting the LSDBs and sent/received LSPs from the two affected neighbors, the server can have more insights at the synchronization failure.

4. Message Format

4.1. Protocol Selection Options

Regarding the network monitoring data export, BMP has been a good option. First of all, BMP serves similar purposes of network monitoring for IGP that reports routes, route statistics and peer status. In addition, BMP has already been implemented in major vendor devices and utilized by operator.

4.2. Message Types

The variety of IS-IS troubleshooting use cases requires a systematic information report of network monitoring, so that the NM server or any third party analyzer could efficiently utilize the reported messages to localize and recover various network issues. We define NM messages for IS-IS uses the following types:

- Initiation Message: A message used for the monitored device to inform the NM monitoring station of its capabilities, vendor, software version and so on. For example, the link MTU can be included within the message. The initiation message is sent once the TCP connection between the monitoring station and monitored router is set up. During the monitoring session, any change of the initiation message could trigger an Initiation Message update.

- Adjacency Status Change Notification Message: A message used to inform the NM monitoring station of the adjacency status change of the monitored device, i.e., from up to down, from down/initiation to up, with possible alarms/logs recorded in the device. This message notifies the NM server of the ongoing IS-IS adjacency change event and possible reasons. If no reason is provided or the provided reason is not specific enough, the NM server can further analyze the IS-IS PDU or the IS-IS statistics.

- o Statistic Report Message: A message used to report the statistics of the ongoing IS-IS process at the monitored device. For example, abnormal LSP count of the monitored device can be a sign of route flapping. This message can be sent periodically or event triggered. If sent periodically, the frequency can be configured by the operator depending on the monitoring requirement. If it's event triggered, it could be triggered by a counter/timer exceeding the threshold.

- o IS-IS PDU Monitoring Message: A message used to update the NM server of any PDU sent from and received at the monitored device. For example, the IIHs collected from two neighbors can be used for analyzing the adjacency set up failure issue. The LSPs collected from two neighbors can be analyzed for the LSP synchronization issue.

- o Termination Message: A message for the monitored router to inform the monitoring station of why it is closing the NM session. This message is sent when the monitoring session is to be closed.

4.3. Message Format

4.3.1. Common Header

The common header is encapsulated in all messages of network monitoring for IGP. It includes the Version, Message Length and Message Type fields. The common header can reuse the common header of BMP and new message types should defined for IGP monitoring.

- o Type = TBD: Adjacency Status Change Notification

- o Type = TBD: ISIS Statistic Report

- o Type = TBD: IS-IS PDU Monitoring

4.3.2. Per Adjacency Header

Except the Initiation and Termination Message, all the rest messages are per adjacency based. Thus, a per adjacency header is defined 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
+---------------------------------------------------------------+
|          Reserved           |CT|     Neighbor System ID        |
+---------------------------------------------------------------+
|                        Neighbor System ID                      |
+-------------------------------+-------------------------------+
|        Neighbor Area ID       |                               |
+-------------------------------+-------------------------------+
|                       Timestamp (seconds)                     |
+---------------------------------------------------------------+
|                     Timestamp (microseconds)                  |
+---------------------------------------------------------------+
```

o Adjacency Flag (2 bytes): The Circuit Type (2 bits) flag specifies if the router is an L1(01), L2(10), or L1/L2(11). If both bits are zeroes (00), the Per Adjacency Header SHALL be ignored. This configuration is used when the statistic is not per-adjacency based, e.g., when reporting the number of adjacencies.

o Neighbor System ID (6 bytes): identifies the system ID of the remote router.

o Neighbor Area ID (2 bytes): identifies the area ID of the remote router.

o Timestamp (4 bytes): records the time when the message is sent/ received, expressed in seconds and microseconds since midnight (zero hour), January 1, 1970 (UTC).

4.3.3. Initiation Message

Three new types of Router Capability TLVs should be defined for IGP monitoring:

o Type = TBD: Local System ID. The corresponding Router Capability Value field SHALL indicate the router’s System ID

o Type = TBD: Link MTU. The corresponding Router Capability Value field SHALL indicate the router’s link MTU.

4.3.4. Adjacency Status Change Notification

The Adjacency Status Change Notification Message indicates an IS-IS adjacency status change: from up to down or from initiation/down to up. It consists of the Common Header, Per Adjacency Header and the Reason TLV. The Notification is triggered whenever the status changes. The Reason TLV is optional, and is defined as follows. More Reason types can be defined if necessary.

```
 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
+-------------------------------+-------------------------------+
|    Reserved |S|   Reason Type |         Reason Length         |
+-------------------------------+-------------------------------+
+                        Reason Value (variable)                +
~                                                               ~
+---------------------------------------------------------------+
```

o Reason Flags (1 byte): The S flag (1 bit) indicates if the Adjacency status is from up to down (set to 0) or from down/ initial to up (set to 1). The rest bits of the Flag field are reserved. When the S flag is set to 1, the Reason Type SHALL be set to all zeroes (i.e., Type 0), the Reason Length fields SHALL be set to all zeroes, and the Reason Value field SHALL be set empty.

o Reason Type (1 byte): indicates the possible reason that caused the adjacency status change. Currently defined types are:

  * Type = 0: Adjacency Up. This type indicates the establishment of an adjacency. For this reason type, the S flag MUST be set to 1, indicating it's a adjacency-up event. There's no further reason to be provided. The reason Length field SHALL be set to all zeroes, and the Reason Value field SHALL be set empty.

  * Type = 1: Circuit Down. For this data type, the S flag MUST be set to 0, indicating it's a adjacency-down event. The length field is set to all zeroes, and the value field is set empty.

  * Type = 2: Memory Low. For this data type, the S flag MUST be set to 0, indicating it's a adjacency-down event. The length field is set to all zeroes, and the value field is set empty.

  * Type = 3: Hold timer expired. For this data type, the S flag MUST be set to 0, indicating it's a adjacency-down event. The length field is set to all zeroes, and the value field is set empty.

  * Type = 4: String. For this data type, the S flag MUST be set to 0, indicating it's a adjacency-down event. The corresponding Reason Value field indicates the reason specified by the monitored router in a free-form UTF-8 string whose length is given by the Reason Length field.

o Reason Length (2 bytes): indicates the length of the Reason Value field.

o Reason Value (variable): includes the possible reason why the Adjacency is down.

4.3.5. ISIS Statistic Report Message

The ISIS Statistic Report Message reports the statistics of the parameters that are of interest to the operator. The message consists of the Common Header, the Per Adjacency Header and the Statistic TLV. The message include both per-adjacency based statistics and non per- adjacency based statistics. For example, the received/sent LSP counts are per-adjacency based statistics, and the local LSP change times count and the number of established adjacencies are non per- adjacency based statistics. For the non per-adjacency based statistics, the CT Flag (2 bits) in the Per Adjacency Header MUST be set to 00. Upon receiving any message with CT flag set to 00, the Per Adjacency Header SHALL be ignored (the total length of the Per Adjacency Header is 18 bytes as defined in Section 3.2.2, and the message reading/analysis SHALL resume from the Statistic TLV part.

The Statistic TLV is defined 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
+-----------------------------------------------------------------+
|   Reserved   |T| Statistic Type|        Statistic Length        |
+-----------------------------------------------------------------+
|                        Statistic  Value                         |
+-----------------------------------------------------------------+
```

o Statistic Flags (1 byte): provides information for the reported statistics.

  * T flag (1 bit): indicates if the statistic is for the received-from direction (set to 1) or sent-to direction the neighbor (set to 0)

o Statistic Type (1 byte): specifies the statistic type of the counter. Currently defined types are:

  * Type = 0: IIH count. The T flag indicates if it's a sent or received Hello PDU. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 1: Incorrect IIH received count. For this type, the T flag MUST be set to 1. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 2: LSP count. The T flag indicates if it's a sent or received LSP. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 3: Incorrect LSP received count. For this type, the T flag MUST be set to 1. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

*

*

*

* Type = 4: Retransmitted LSP count. For this type, the T flag MUST be set to 0. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 5: CSNP count. The T flag indicates if it's a sent or received CSNP. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 6: PSNP count. The T flag indicates if it's a sent or received PSNP. It is a per-adjacency based statistic type, and the CT flag in the Per Adjacency Header MUST NOT be set to 00.

* Type = 7: Number of established adjacencies. It's a non per-adjacency based statistic type, and thus for the monitoring station to recognize this type, the CT flag in the Per Adjacency Header MUST be set to 00.

* Type = 8: LSP change time count. It's a non per-adjacency based statistic type, and thus for the monitoring station to recognize this type, the CT flag in the Per Adjacency Header MUST be set to 00.

o Statistic Length (2 bytes): indicates the length of the Statistic Value field.

o Statistic Value (4 bytes): specifies the counter value, which is a non-negative integer.

4.3.6. IS-IS PDU Monitoring Message

The IS-IS PDU Monitoring Message is used to update the monitoring station of any PDU sent from and received at the monitored device per neighbor. Following the Common Header and the Per Adjacency Header is the IS-IS PDU. To tell whether it's a sent or received PDU, the monitoring station can analyze the source and destination addresses in the reported PDUs.

4.3.7. Termination Message

This document does not change the Termination Message defined by RFC7854.

5. IANA

TBD

6. Contributors

TBD

7. Acknowledgments

TBD

8. References

[I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., Lapukhov, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

[I-D.chen-npm-use-cases]
Chen, H., Li, Z., Xu, F., Gu, Y., and Z. Li, "Network-wide Protocol Monitoring (NPM): Use Cases", draft-chen-npm-use-cases-00 (work in progress), March 2019.

[I-D.ietf-netconf-yang-push]
Clemm, A. and E. Voit, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-25 (work in progress), May 2019.

[I-D.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.song-ntf]
Song, H., Zhou, T., Li, Z., Fioccola, G., Li, Z., Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Toward a Network Telemetry Framework", draft-song-ntf-02 (work in progress), July 2018.

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol (SNMP)", RFC 1157, DOI 10.17487/RFC1157, May 1990, <https://www.rfc-editor.org/info/rfc1157>.

[RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <https://www.rfc-editor.org/info/rfc1195>.

[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>.

[RFC6232] Wei, F., Qin, Y., Li, Z., Li, T., and J. Dong, "Purge Originator Identification TLV for IS-IS", RFC 6232, DOI 10.17487/RFC6232, May 2011, <https://www.rfc-editor.org/info/rfc6232>.

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

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

[RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <https://www.rfc-editor.org/info/rfc7854>.

Authors' Addresses

Yunan Gu
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China

Email: guyunan@huawei.com

Shuanglong Chen
Huawei
156 Beiqing Road
Beijing,100095
P.R. China

Email: chenshuanglong@huawei.com

Huanan Chen
China Telecom
109 West Zhongshan Ave
Guangzhou
China

Email: chenhuanan@gsta.com

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

Email: lizhenbin@huawei.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: May 6, 2021

S. Barguil, Ed.
O. Gonzalez de Dios, Ed.
Telefonica
M. Boucadair
Orange
L. Munoz
Vodafone
L. Jalil
Verizon
J. Ma
China Unicom
November 02, 2020

A Layer 2 VPN Network YANG Model
draft-ietf-opsawg-l2nm-01

Abstract

This document defines a YANG Data model (called, L2NM) that can be used to manage the provisioning of Layer 2 VPN services within a Service Provider Network. This YANG module provides representation of the Layer 2 VPN Service from a network standpoint. The module is meant to be used by a Network Controller to derive the configuration information that will be sent to relevant network devices.

The L2NM YANG Data model complements the Layer 2 Service Model (RFC8466) by providing a network-centric view of the service that is internal to a Service Provider.

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 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

[RFC8466] defines a L2VPN Service Model (L2SM) YANG data model that can be used for L2VPN service ordering matters between customers and Service Providers (SPs). This document complements the L2SM model by creating a network-centric view of the service which can be exposed by a Network to a Service Controller within the Service Provider Network. In particular, the model can be used in the communication between the entity that interacts directly with the customer, the service orchestrator, (either fully automated or a human operator) and the entity in charge of network orchestration and control (a.k.a., network controller/orchestrator).

The data model defined in this document is called the L2VPN Network Model (L2NM), playing the role of Service Delivery Model (Figure 3 of [RFC8466]. The module supports additional capabilities, such as exposing operational parameters, transport protocols selection and precedence. It also serves as a multi-domain orchestration interface, because this model can transport resources (i.e., VCID) between domains. The data model keeps minimum customer-related information.

This document uses the common VPN YANG module defined in [I-D.ietf-opsawg-vpn-common].

The YANG data model in this document conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

1.1. Terminology

This document assumes that the reader is familiar with the contents of [RFC6241], [RFC7950], [RFC8466], [RFC8309], and [RFC8453] and uses terminology from those documents. The meaning of the symbols in YANG tree diagrams is [RFC8340].

This document makes use of the following terms:

- L2 VPN Customer Service Model (L2SM): Describes the service characterization (requirements) of a L2 VPN that interconnects a set of sites from the perspective of the customer. The customer service model does not provide details on the Service Provider Network. The L2 VPN Customer Service model is defined in [RFC8466].

- L2 VPN Service Network Model (L2NM): Refers to the YANG module that describes a L2 VPN Service with a network-centric view. It contains information of the Service Provider network and might include allocated resources. It can be used by network controllers to manage the Layer 2 VPN Service configuration in the Service Provider network. The YANG module can be consumed by a Service Orchestrator to request a VPN Service to a Network controller or to expose the list of active L2VPN services.

- o Service Orchestrator: Refers to a functional entity that interacts with the customer of a L2 VPN relying upon, e.g. L2SM. The Service Orchestrator is responsible of the CE-PE attachment circuits, the PE selection, and requesting the activation of the L2 VPN service to a network controller.

- o Network Controller: Denotes a functional entity responsible for the management of the service provider network.

- o VPN node (vpn-node): Is an abstraction that represents a set of policies applied on a PE and that belong to a single VPN service (vpn-service). A VPN service involves one or more VPN nodes. The VPN node will identify the Service Provider node on which the VPN is deployed.

- o VPN network access (vpn-network-access): Is an abstraction that represents the network interfaces that are associated to a given VPN node. Traffic coming from the VPN network access belongs to the VPN. The attachment circuits (bearers) between CEs and PEs are terminated in the VPN network access.

- o VPN Service Provider (SP): Is a Service Provider that offers VPN-related services.

- o Service Provider Network (SP Network): Is a network able to provide VPN-related services.

1.2. Requirements Language

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

2. Reference architecture

Figure 1 illustrates how L2NM is used. As a reminder, this figure is an expansion of the architecture presented in Section 3 of [RFC8466] and decomposes the box marked "orchestration" in that figure into three separate functional components called "Service Orchestration", "Network Orchestration", and "Domain Orchestration".

The reader may refer to [RFC8309] for the distinction between the "Customer Service Model", the "Service Delivery Model", the "Network Configuration Model", and the "Device Configuration Model". The "Domain Orchestration" and "Config Manager" roles may be performed by "SDN Controllers".

```
                             +---------------+
                             |   Customer    |
                             +---------------+
             Customer Service Model   |
                    l2vpn-svc         |
                             +---------------+
                             |    Service    |
                             | Orchestration |
                             +---------------+
       Service Delivery Model        |
             L2NM                    |
            (l2vpn-ntw)     |
                             +---------------+
                             |   Network     |
                             | Orchestration |
                             +---------------+
       Network Configuration Model   |
                          ___________|____________
                         |                       |
                +---------------+       +---------------+
                |    Domain     |       |    Domain     |
                | Orchestration |       | Orchestration |
                +---------------+       +---------------+
     Device         |        |                   |
     Configuration  |        |                   |
     Model          |        |                   |
               +---------+   |                   |
               | Config  |   |                   |
               | Manager |   |                   |
               +---------+   |                   |
                    |        |                   |
                    | NETCONF/CLI..................
                    |        |                   |
             +------------------------------------------------+
                                 Network

                            +++++++
                            + AAA +
                            +++++++

    ++++++++   Bearer    ++++++++           ++++++++      ++++++++
    + CE A + ----------- + PE A +           + PE B + ---- + CE B +
    ++++++++  Connection ++++++++           ++++++++      ++++++++

               Site A                               Site B
```

Figure 1: L2SM and L2NM Interaction

Figure 2 shows how L2SM and L2NM may be used in the context of the ACTN architecture [RFC8453]. Figure 2 shows the Customer Network Controller (CNC), the Multi-Domain Service Coordinator (MDSC), and the Provisioning Network Controller (PNC). It shows the interfaces between these functional units: the CNC-MDSC Interface (CMI), the MDSC-PNC Interface (MPI), and the Southbound Interface (SBI).

```
                   ----------------------------------
                  | Customer                         |
                  |  -----------------------------   |
                  | |             CNC             |  |
                  |  -----------------------------   |
                   ----:-----------------------:-----
                       :                       :
                       : L2SM                  : L2SM
                       :                       :
              ---------:---------     -------------------
             | MDSC    :         |   |       MDSC        |
             |  ---------------  |   |     (parent)      |
             | |    Service    | |    -------------------
             | | Orchestration | |             :
             |  ---------------  |             : L2NM
             |         :         |             :
             |         : L2NM    |    -------------------
             |         :         |   |       MDSC        |
             |  ---------------  |   |      (child)      |
             | |    Network    | |    -------------------
             | | Orchestration | |             :
             |  ---------------  |             :
              ---------:---------              :
                       :                       :
                       : Network Configuration :
                       :                       :
           ------------:-------       ---------:------------
          | Domain     :       |     |         : Domain     |
          | Controller :       |     |         : Controller |
          |        ---------   |     |    ---------         |
          |       |   PNC   |  |     |   |   PNC   |       |
          |        ---------   |     |    ---------         |
           ------------:-------       ---------:------------
                       :                       :
                       : Device Configuration  :
                       :                       :
                   --------                --------
                  | Device |              | Device |
                   --------                --------
```

Figure 2: L2SM and L2NM in the Context of ACTN

3. Description of the L2NM YANG Module

The L2NM module ('ietf-l2vpn-ntw') is meant to manage L2 VPNs within a service provider network. In particular, the 'ietf-l2vpn-ntw' module can be used to create, modify, and retrieve L2VPN Services in a Network Controller. The module is not aimed at maintaining customer-related information.

Editor's note: Next version of the document will include the full description of the parameters. When the parameters match with L2SM, the exact reference will be done

3.1. Structure of the Module

The 'ietf-l2vpn-ntw' module uses two main containers: 'vpn-services' and 'vpn-profiles'. The 'vpn-services' container maintains a set of L2 VPN Services managed in the service provider network. The module allows to create a new l2 VPN service by adding a new instance of 'vpn-service'. The 'vpn-service' is the data structure that abstracts the VPN Service.

```
module: ietf-l2vpn-ntw
+--rw l2vpn-ntw
 +--rw vpn-profiles
 |      .......
 +--rw vpn-services
     +--rw vpn-service* [vpn-id]
        ........
```

Figure 3: Simplified L2NM Tree Structure

3.2. VPN Profiles

The 'vpn-profiles' container (Figure 4) allows the network provider to define and maintain a set of common VPN profiles [I-D.ietf-opsawg-vpn-common] that apply to one or several VPN services. The exact definition of the profiles is local to each network provider.

This document does not make any assumption about the exact definition of these profiles. How such profiles are defined is deployment specific. The model only includes an identifier to these profiles to ease identifying local policies when building a VPN service. As shown in Figure 4, the following identifiers can be included:

o 'cloud-identifier': This identifier refers to a cloud service.

o 'encryption-profile-identifier': An encryption profile refers to a set of policies related to the encryption scheme(s) and setup that can be applied when building and offering a VPN service.

o 'qos-profile-identifier': A QoS profile refers to as set of policies such as classification, marking, and actions (e.g., [RFC3644]).

o 'bfd-profile-identifier': A Bidirectional Forwarding Detection (BFD) profile refers to a set of BFD [RFC5880] policies that can be invoked when building a VPN service.

o 'forwarding-profile-identifier': A forwarding profile refers to the policies that apply to the forwarding of packets conveyed within a VPN. Such policies may consist at applying Access Control Lists (ACLs).

o 'routing-profile-identifier': A routing profile refers to a set of routing policies that will be invoked (e.g., BGP policies).

```
+--rw l2vpn-ntw
   +--rw vpn-profiles
   |  +--rw valid-provider-identifiers
   |     +--rw cloud-identifier* [id] {cloud-access}?
   |     |  +--rw id    string
   |     +--rw encryption-profile-identifier* [id]
   |     |  +--rw id    string
   |     +--rw qos-profile-identifier* [id]
   |     |  +--rw id    string
   |     +--rw bfd-profile-identifier* [id]
   |     |  +--rw id    string
   |     +--rw forwarding-profile-identifier* [id]
   |     |  +--rw id    string
   |     +--rw routing-profile-identifier* [id]
   |        +--rw id    string
   +--rw vpn-services
      ...
```

Figure 4: VPN Profiles Subtree Structure

3.3. L2 VPN Service

The 'vpn-service' is the data structure that abstracts a L2 VPN Service within the SP Network. Every 'vpn-service' has a unique identifier: vpn-id. Such vpn-id is only meaningful locally within the Network controller. In order to facilitate the recognition of the service, a 'customer-name' and a 'description' may be included.

The topology of the VPN service is expressed in the 'vpn-service-topology' leaf.

A VPN Service is built by adding instances of 'vpn-node' to the 'vpn-nodes' container. The 'vpn-node' is an abstraction that represents a set of policies/configurations applied to a network node and that belong to a single 'vpn-service'. A 'vpn-node' contains 'vpn-network-accesses', which are the interfaces involved in the creation of the VPN. The customer sites are connected to the 'vpn_network_accesses'. Note that, as this is a network data model, the information about customers site is not needed. Such information, is only relevant in the L2SM model.

```
+--rw vpn-services
   +--rw vpn-service* [vpn-id]
      +--rw status
      |  +--rw admin-status
      |  |  +--rw status?         identityref
      |  |  +--rw last-updated?   yang:date-and-time
      |  +--ro oper-status
      |     +--ro status?         identityref
      |     +--ro last-updated?   yang:date-and-time
      +--rw vpn-id                      vpn-id
      +--rw vpn-name?                   string
      +--rw vpn-description?            string
      +--rw customer-name?              string
      +--rw l2sm-vpn-id?                vpn-common:vpn-id
      +--rw vpn-svc-type?               identityref
      +--rw svc-topo?                   identityref
      +--rw multicast-like {vpn-common:multicast}?
      |  +--rw enabled?                 boolean
      |  +--rw customer-tree-flavors
      |     +--rw tree-flavor*   identityref
      +--rw extranet-vpns {vpn-common:extranet-vpn}?
      |  +--rw extranet-vpn* [vpn-id]
      |     +--rw vpn-id              vpn-common:vpn-id
      |     +--rw local-sites-role?   identityref
      +--rw svc-mtu?                    uint32
      +--rw ce-vlan-preservation?       boolean
      +--rw ce-vlan-cos-perservation?   boolean
      +--rw underlay-transport
      |  +--rw type*   identityref
      +--rw vpn-nodes
              .....
```

Figure 5

3.3.1. L2 VPN Service Types

The L2 VPN Service types directly matches with the L2VPN Service types defined in Section 5.1.3 of [RFC8466]:

- o Point-to-point VPWSs.
- o Point-to-point or point-to-multipoint VPWSs [RFC8214].
- o Multipoint VPLSs.
- o Multipoint VPLSs connecting one or more root sites and a set of leaf sites but preventing inter-leaf-site communication.
- o EVPN services [RFC7432].
- o EVPN VPWSs between two customer sites or a set of customer sites as specified in [RFC8214].

3.3.2. Underlying Transport Selection

The model enables network operators to select the type of transport protocol underlay. Also, in scenarios with multiple domains and NNI types, the selection of the transport protocol underlay is required. The Service Provider Network might have several underlay possibilities available. If no underlay transport protocol is specified, the Network Controller will take care of the transport decision. The following options are supported in the "underlay-transport" container:

LDP: MPLS with LDP (martini encapsulation).

GRE: A mesh of GRE tunnels is established between vpn-nodes.

BGP: BGP tunnels (kompella encapsulation) are preferred to route traffic between VPN nodes.

TE: TE tunnels (either RSVP-TE or SR) are prefered. The mapping details will be specified in draft-ietf-te-service-mapping.

SR: Non-TE SR is preferred to route traffic.

3.3.3. VPN Node

The 'vpn-node' is an abstraction that represents a set of policies/ configurations applied to a network node and that belong to a single 'vpn-service'. A 'vpn-node' contains 'vpn-network-accesses', which

are the interfaces involved in the creation of the VPN. The customer
sites are connected to the 'vpn_network_accesses'.

```
                     +--rw vpn-nodes
                +--rw vpn-node* [vpn-node-id ne-id]
                   +--rw vpn-node-id              vpn-common:vpn-id
                   +--rw description?             string
                   +--rw node-role?               identityref
                   +--rw ne-id                    string
                   +--rw port-id?                 string
                   +--rw status
                   |  +--rw admin-status
                   |  |  +--rw status?         identityref
                   |  |  +--rw last-updated?   yang:date-and-time
                   |  +--ro oper-status
                   |     +--ro status?         identityref
                   |     +--ro last-updated?   yang:date-and-time
                   +--rw signaling-options* [type]
                   |  +--rw type         identityref
                   |  +--rw l2vpn-bgp
                   |  |  +--rw pwe-encapsulation-type?   identityref
                   |  |  +--rw vpn-target* [id]
                   |  |  |  +--rw id                    int8
                   |  |  |  +--rw route-targets* [route-target]
                   |  |  |  |  +--rw route-target
                   |  |  |  |          rt-types:route-target
                   |  |  |  +--rw route-target-type
                   |  |  |          rt-types:route-target-type
                   |  |  +--rw vpn-policies
                   |  |  |  +--rw import-policy?   string
                   |  |  |  +--rw export-policy?   string
                   |  |  +--rw pwe-mtu
                   |  |  |  +--rw allow-mtu-mismatch?   boolean
                   |  |  +--rw address-family?
                   |  |          vpn-common:address-family
                   |  +--rw evpn-bgp
                   |  |  +--rw vpn-id?              leafref
                   |  |  +--rw type?                identityref
                   |  |  +--rw address-family?
                   |  |  |       vpn-common:address-family
                   |  |  +--rw mac-learning-mode?   identityref
                   |  |  +--rw arp-suppress?        boolean
                   |  +--rw t-ldp-pwe
                   |  |  +--rw type?                     identityref
                   |  |  +--rw pwe-encapsulation-type?   identityref
                   |  |  +--rw pwe-mtu?                  uint16
                   |  |  +--rw ac-pw-list* [peer-addr vc-id]
                   |  |  |  +--rw peer-addr      inet:ip-address
```

```
|  |  |  +--rw vc-id          vpn-common:vpn-id
|  |  |  +--rw pw-type?       identityref
|  |  |  +--rw pw-priority?   uint32
|  |  +--rw qinq
|  |     +--rw s-tag?   uint32
|  |     +--rw c-tag?   uint32
|  +--rw l2tp-pwe
|     +--rw type?                   identityref
|     +--rw encapsulation-type?   identityref
|     +--rw ac-pw-list* [peer-addr vc-id]
|        +--rw peer-addr      inet:ip-address
|        +--rw vc-id          string
|        +--rw pw-priority?   uint32
+--rw vpn-network-accesses
```

Figure 6

3.3.3.1. Signaling options

This sub-tree defines the L2VPN service type, according to the several signalling options to exchange membership information between the PE that is used. There are some common parameters inside each of them (e.g encapsulation type, MTU) but some others are:

l2vpn-bgp. The service is a Multipoint VPLSs that use a BGP control plane as described in [RFC4761] and [RFC6624]. The VPLS members exchange Route Targets with related import/export policies.

evpn-bgp. The service is a Multipoint VPLSs that use also a BGP control plane but also includes the additional features and related parameters described in [RFC7432] and [RFC7209].

t-ldp-pwe. A Multipoint VPLSs that use a mesh of LDP-signaled Pseudowires [RFC6074], including as parameters the list of Pseudowires that constitute the mesh, with their details (VC-IDs and endpoints).

L2tp-pwe. Multipoint VPLSs that use L2TP-signaled Pseudowires [RFC6074].

```
               +--rw signaling-options* [type]
                 +--rw type         identityref
                 +--rw l2vpn-bgp
                 |  +--rw pwe-encapsulation-type?   identityref
                 |  +--rw vpn-target* [id]
                 |  |  +--rw id                   int8
                 |  |  +--rw route-targets* [route-target]
                 |  |  |  +--rw route-target
                 |  |  |          rt-types:route-target
                 |  |  +--rw route-target-type
                 |  |          rt-types:route-target-type
                 |  +--rw vpn-policies
                 |  |  +--rw import-policy?   string
                 |  |  +--rw export-policy?   string
                 |  +--rw pwe-mtu
                 |  |  +--rw allow-mtu-mismatch?   boolean
                 |  +--rw address-family?
                 |          vpn-common:address-family
                 +--rw evpn-bgp
                 |  +--rw vpn-id?              leafref
                 |  +--rw type?                identityref
                 |  +--rw address-family?
                 |  |       vpn-common:address-family
                 |  +--rw mac-learning-mode?   identityref
                 |  +--rw arp-suppress?        boolean
                 +--rw t-ldp-pwe
                 |  +--rw type?                     identityref
                 |  +--rw pwe-encapsulation-type?   identityref
                 |  +--rw pwe-mtu?                  uint16
                 |  +--rw ac-pw-list* [peer-addr vc-id]
                 |  |  +--rw peer-addr      inet:ip-address
                 |  |  +--rw vc-id          vpn-common:vpn-id
                 |  |  +--rw pw-type?       identityref
                 |  |  +--rw pw-priority?   uint32
                 |  +--rw qinq
                 |     +--rw s-tag?   uint32
                 |     +--rw c-tag?   uint32
                 +--rw l2tp-pwe
                    +--rw type?                 identityref
                    +--rw encapsulation-type?   identityref
                    +--rw ac-pw-list* [peer-addr vc-id]
                       +--rw peer-addr      inet:ip-address
                       +--rw vc-id          string
                       +--rw pw-priority?   uint32
```

Figure 7

3.3.3.2. VPN Network Access

A 'vpn-network-access' represents an entry point to a VPN service . In other words, this container encloses the parameters that describe the access information for the traffic that belongs to a particular L2VPN. As such, every 'vpn-network-access' MUST belong to one and only one 'vpn-node'.

A 'vpn-network-access' includes information such as the connection on which the access is defined , the specific layer 2 service requirements, etc.

The Site Network Access is comprised of:

id: Identifier of the vpn network access.

description: Text describing the vpn network access.

interface-mtu: maximum transmission unit or maximum frame size of the interface belonging to the vpn network access. When a frame is larger than the MTU, it is broken down, or fragmented, into smaller pieces by the network protocol to accommodate the MTU of the network"

status: Administrative and operational status of the service.

ethernet-service-oam: Carries information about the service OAM

```
          +--rw vpn-network-accesses
     +--rw vpn-network-access* [id]
        +--rw id
        |       vpn-common:vpn-id
        +--rw description?
        |       string
        +--rw Interface-mtu?
        |       uint32
        +--rw status
        |  +--rw admin-status
        |  |  +--rw status?         identityref
        |  |  +--rw last-updated?   yang:date-and-time
        |  +--ro oper-status
        |     +--ro status?         identityref
        |     +--ro last-updated?   yang:date-and-time
        +--rw access-diversity
        |       {vpn-common:placement-diversity}?
        |  +--rw groups
        |  |  +--rw fate-sharing-group-size?   uint16
        |  |  +--rw group-color?               string
```

```
       |  |  +--rw group* [group-id]
       |  |     +--rw group-id    string
       |  +--rw constraints
       |     +--rw constraint* [constraint-type]
       |        +--rw constraint-type    identityref
       |        +--rw target
       |           +--rw (target-flavor)?
       |              +--:(id)
       |              |  +--rw group* [group-id]
       |              |     +--rw group-id    string
       |              +--:(all-accesses)
       |              |  +--rw all-other-accesses?
       |              |          empty
       |              +--:(all-groups)
       |                 +--rw all-other-groups?
       |                         empty
       +--rw connection
       |      ....
       +--rw availability
       |  +--rw access-priority?       uint32
       |  +--rw (redundancy-mode)?
       |     +--:(single-active)
       |     |  +--rw single-active?   boolean
       |     +--:(all-active)
       |        +--rw all-active?      boolean
       +--rw service
       |      .....
       +--rw broadcast-unknown-unicast-multicast
       |  +--rw multicast-site-type?
       |  |       enumeration
       |  +--rw multicast-gp-address-mapping* [id]
       |  |  +--rw id                  uint16
       |  |  +--rw vlan-id?            uint32
       |  |  +--rw mac-gp-address?
       |  |  |       yang:mac-address
       |  |  +--rw port-lag-number?   uint32
       |  +--rw bum-overall-rate?
       |          uint32
       +--rw ethernet-service-oam
       |  +--rw md-name?        string
       |  +--rw md-level?       uint8
       |  +--rw cfm-802.1-ag
       |  |  +--rw n2-uni-c* [maid]
       |  |  |  +--rw maid                  string
       |  |  |  +--rw mep-id?               uint32
       |  |  |  +--rw mep-level?            uint32
       |  |  |  +--rw mep-up-down?          enumeration
       |  |  |  +--rw remote-mep-id?        uint32
```

```
|  |  |  +--rw cos-for-cfm-pdus?   uint32
|  |  |  +--rw ccm-interval?       uint32
|  |  |  +--rw ccm-holdtime?       uint32
|  |  |  +--rw ccm-p-bits-pri?
|  |  |          vpn-common:ccm-priority-type
|  |  +--rw n2-uni-n* [maid]
|  |     +--rw maid                string
|  |     +--rw mep-id?             uint32
|  |     +--rw mep-level?          uint32
|  |     +--rw mep-up-down?        enumeration
|  |     +--rw remote-mep-id?      uint32
|  |     +--rw cos-for-cfm-pdus?   uint32
|  |     +--rw ccm-interval?       uint32
|  |     +--rw ccm-holdtime?       uint32
|  |     +--rw ccm-p-bits-pri?
|  |             vpn-common:ccm-priority-type
|  +--rw y-1731* [maid]
|     +--rw maid
|     |       string
|     +--rw mep-id?
|     |       uint32
|     +--rw type?
|     |       identityref
|     +--rw remote-mep-id?
|     |       uint32
|     +--rw message-period?
|     |       uint32
|     +--rw measurement-interval?
|     |       uint32
|     +--rw cos?
|     |       uint32
|     +--rw loss-measurement?
|     |       boolean
|     +--rw synthethic-loss-measurement?
|     |       boolean
|     +--rw delay-measurement
|     |  +--rw enable-dm?   boolean
|     |  +--rw two-way?     boolean
|     +--rw frame-size?
|     |       uint32
|     +--rw session-type?
|             enumeration
+--rw mac-loop-prevention
|  +--rw frequency?         uint32
|  +--rw protection-type?   identityref
|  +--rw number-retries?    uint32
+--rw access-control-list
|  +--rw mac* [mac-address]
```

```
                         |     +--rw mac-address    yang:mac-address
                         +--rw mac-addr-limit
                            +--rw mac-num-limit?   uint16
                            +--rw time-interval?   uint32
                            +--rw action?          identityref
```

Figure 8

3.3.3.2.1. Connection

The connection container is used to configure the relevant properties of the interface that is attached to the VPN, for example the encapsulation type, the physical interface or creating a lag.

```
                  +--rw connection
                     +--rw encapsulation-type?    identityref
                     +--rw eth-inf-type*          identityref
                     +--rw dot1q-interface
                     |  +--rw l2-access-type?   identityref
                     |  +--rw dot1q {vpn-common:dot1q}?
                     |  |  +--rw physical-inf?   string
                     |  |  +--rw c-vlan-id?      uint32
                     |  +--rw qinq {vpn-common:qinq}?
                     |  |  +--rw s-vlan-id?   uint32
                     |  |  +--rw c-vlan-id?   uint32
                     |  +--rw qinany {vpn-common:qinany}?
                     |  |  +--rw s-vlan-id?   uint32
                     |  +--rw vxlan {vxlan}?
                     |     +--rw vni-id?      uint32
                     |     +--rw peer-mode?   identityref
                     |     +--rw peer-list* [peer-ip]
                     |        +--rw peer-ip    inet:ip-address
                     +--rw phy-interface
                     |  +--rw port-number?           uint32
                     |  +--rw port-speed?            uint32
                     |  +--rw mode?
                     |  |       vpn-common:neg-mode
                     |  +--rw phy-mtu?               uint32
                     |  +--rw flow-control?          string
                     |  +--rw oam-802.3ah-link {oam-3ah}?
                     |  |  +--rw enable?   boolean
                     |  +--rw uni-loop-prevention?   boolean
                     +--rw lag-interface
                     |       {vpn-common:lag-interface}?
                     |  +--rw lag-interface*
                     |          [lag-interface-number]
                     |     +--rw lag-interface-number    uint32
                     |     +--rw lacp
```

```
                  |         +--rw lacp-state?         boolean
                  |         +--rw lacp-mode?          boolean
                  |         +--rw lacp-speed?         boolean
                  |         +--rw mini-link?          uint32
                  |         +--rw system-priority?    uint16
                  |         +--rw member-link-list
                  |         |  +--rw member-link* [name]
                  |         |     +--rw name
                  |         |     |       string
                  |         |     +--rw port-speed?
                  |         |     |       uint32
                  |         |     +--rw mode?
                  |         |     |       vpn-common:neg-mode
                  |         |     +--rw link-mtu?
                  |         |     |       uint32
                  |         |     +--rw oam-802.3ah-link
                  |         |             {oam-3ah}?
                  |         |        +--rw enable?   boolean
                  |         +--rw flow-control?       string
                  |         +--rw lldp?               boolean
                  +--rw cvlan-id-to-svc-map* [svc-id]
                  |  +--rw svc-id      leafref
                  |  +--rw cvlan-id* [vid]
                  |     +--rw vid    uint32
                  +--rw split-horizon
                     +--rw group-name?   string
```

Figure 9

3.3.3.2.2. Layer 2 service requirements

This container is usedd to indicate the details of the ethernet service such as bandwidth or qos.

```
+--rw service
   |  +--rw svc-input-bandwidth
   |  |       {vpn-common:input-bw}?
   |  |  +--rw input-bandwidth* [type]
   |  |     +--rw type      identityref
   |  |     +--rw cos-id?   uint8
   |  |     +--rw cir?      uint64
   |  |     +--rw cbs?      uint64
   |  |     +--rw eir?      uint64
   |  |     +--rw ebs?      uint64
   |  |     +--rw pir?      uint64
   |  |     +--rw pbs?      uint64
   |  +--rw svc-output-bandwidth {output-bw}?
   |  |  +--rw output-bandwidth* [type]
```

```
  |  |     +--rw type      identityref
  |  |     +--rw cos-id?   uint8
  |  |     +--rw cir?      uint64
  |  |     +--rw cbs?      uint64
  |  |     +--rw eir?      uint64
  |  |     +--rw ebs?      uint64
  |  |     +--rw pir?      uint64
  |  |     +--rw pbs?      uint64
  |  +--rw qos {vpn-common:qos}?
  |  |  +--rw qos-classification-policy
  |  |  |  +--rw rule* [id]
  |  |  |     +--rw id
  |  |  |     |       string
  |  |  |     +--rw (match-type)?
  |  |  |     |  +--:(match-flow)
  |  |  |     |  |  +--rw (l3)?
  |  |  |     |  |  |  +--:(ipv4)
  |  |  |     |  |  |  |  +--rw ipv4
  |  |  |     |  |  |  |     +--rw dscp?
  |  |  |     |  |  |  |     |       inet:dscp
  |  |  |     |  |  |  |     +--rw ecn?
  |  |  |     |  |  |  |     |       uint8
  |  |  |     |  |  |  |     +--rw length?
  |  |  |     |  |  |  |     |       uint16
  |  |  |     |  |  |  |     +--rw ttl?
  |  |  |     |  |  |  |     |       uint8
  |  |  |     |  |  |  |     +--rw protocol?
  |  |  |     |  |  |  |     |       uint8
  |  |  |     |  |  |  |     +--rw ihl?
  |  |  |     |  |  |  |     |       uint8
  |  |  |     |  |  |  |     +--rw flags?
  |  |  |     |  |  |  |     |       bits
  |  |  |     |  |  |  |     +--rw offset?
  |  |  |     |  |  |  |     |       uint16
  |  |  |     |  |  |  |     +--rw identification?
  |  |  |     |  |  |  |     |       uint16
  |  |  |     |  |  |  |     +--rw (destination-network)?
  |  |  |     |  |  |  |     |  +--:(destination-ipv4-network)
  |  |  |     |  |  |  |     |     +--rw destination-ipv4-network?
  |  |  |     |  |  |  |     |             inet:ipv4-prefix
  |  |  |     |  |  |  |     +--rw (source-network)?
  |  |  |     |  |  |  |        +--:(source-ipv4-network)
  |  |  |     |  |  |  |           +--rw source-ipv4-network?
  |  |  |     |  |  |  |                   inet:ipv4-prefix
  |  |  |     |  |  |  |
  |  |  |     |  |  |  +--:(ipv6)
  |  |  |     |  |  |     +--rw ipv6
  |  |  |     |  |  |        +--rw dscp?
  |  |  |     |  |  |        |       inet:dscp
```

```
  |  |  |     |  |  |        +--rw ecn?
  |  |  |     |  |  |        |       uint8
  |  |  |     |  |  |        +--rw length?
  |  |  |     |  |  |        |       uint16
  |  |  |     |  |  |        +--rw ttl?
  |  |  |     |  |  |        |       uint8
  |  |  |     |  |  |        +--rw protocol?
  |  |  |     |  |  |        |       uint8
  |  |  |     |  |  |        +--rw (destination-network)?
  |  |  |     |  |  |        |  +--:(destination-ipv6-network)
  |  |  |     |  |  |        |     +--rw destination-ipv6-network?
  |  |  |     |  |  |        |             inet:ipv6-prefix
  |  |  |     |  |  |        +--rw (source-network)?
  |  |  |     |  |  |        |  +--:(source-ipv6-network)
  |  |  |     |  |  |        |     +--rw source-ipv6-network?
  |  |  |     |  |  |        |             inet:ipv6-prefix
  |  |  |     |  |  |        +--rw flow-label?
  |  |  |     |  |  |                inet:ipv6-flow-label
  |  |  |     |  |  +--rw (l4)?
  |  |  |     |  |     +--:(tcp)
  |  |  |     |  |     |  +--rw tcp
  |  |  |     |  |     |     +--rw sequence-number?
  |  |  |     |  |     |     |       uint32
  |  |  |     |  |     |     +--rw acknowledgement-number?
  |  |  |     |  |     |     |       uint32
  |  |  |     |  |     |     +--rw data-offset?
  |  |  |     |  |     |     |       uint8
  |  |  |     |  |     |     +--rw reserved?
  |  |  |     |  |     |     |       uint8
  |  |  |     |  |     |     +--rw flags?
  |  |  |     |  |     |     |       bits
  |  |  |     |  |     |     +--rw window-size?
  |  |  |     |  |     |     |       uint16
  |  |  |     |  |     |     +--rw urgent-pointer?
  |  |  |     |  |     |     |       uint16
  |  |  |     |  |     |     +--rw options?
  |  |  |     |  |     |     |       binary
  |  |  |     |  |     |     +--rw (source-port)?
  |  |  |     |  |     |     |  +--:(source-port-range-or-operator)
  |  |  |     |  |     |     |     +--rw source-port-range-or-operator
  |  |  |     |  |     |     |        +--rw (port-range-or-operator)?
  |  |  |     |  |     |     |           +--:(range)
  |  |  |     |  |     |     |           |  +--rw lower-port
  |  |  |     |  |     |     |           |  |       inet:port-number
  |  |  |     |  |     |     |           |  +--rw upper-port
  |  |  |     |  |     |     |           |          inet:port-number
  |  |  |     |  |     |     |           +--:(operator)
  |  |  |     |  |     |     |              +--rw operator?
```

```
   |  |  |     |  |     |     |              |       operator
   |  |  |     |  |     |     |              +--rw port
   |  |  |     |  |     |     |                      inet:port-number
   |  |  |     |  |     |     +--rw (destination-port)?
   |  |  |     |  |     |        +--:(destination-port-range-or-operator)
   |  |  |     |  |     |           +--rw destination-port-range-or-operator
   |  |  |     |  |     |              +--rw (port-range-or-operator)?
   |  |  |     |  |     |                 +--:(range)
   |  |  |     |  |     |    |  +--rw lower-port
   |  |  |     |  |     |    |  |       inet:port-number
   |  |  |     |  |     |    |  +--rw upper-port
   |  |  |     |  |     |    |          inet:port-number
   |  |  |     |  |     |                 +--:(operator)
   |  |  |     |  |     |                    +--rw operator?
   |  |  |     |  |     |       |       operator
   |  |  |     |  |     |                    +--rw port
   |  |  |     |  |     |                            inet:port-number
   |  |  |     |  |     +--:(udp)
   |  |  |     |  |        +--rw udp
   |  |  |     |  |           +--rw length?
   |  |  |     |  |           |       uint16
   |  |  |     |  |           +--rw (source-port)?
   |  |  |     |  |           |  +--:(source-port-range-or-operator)
   |  |  |     |  |           |     +--rw source-port-range-or-operator
   |  |  |     |  |           |        +--rw (port-range-or-operator)?
   |  |  |     |  |           |           +--:(range)
   |  |  |     |  |           |           |  +--rw lower-port
   |  |  |     |  |           |           |  |       inet:port-number
   |  |  |     |  |           |           |  +--rw upper-port
   |  |  |     |  |           |           |          inet:port-number
   |  |  |     |  |           |           +--:(operator)
   |  |  |     |  |           |              +--rw operator?
   |  |  |     |  |           |              |       operator
   |  |  |     |  |           |              +--rw port
   |  |  |     |  |           |                      inet:port-number
   |  |  |     |  |           +--rw (destination-port)?
   |  |  |     |  |              +--:(destination-port-range-or-operator)
   |  |  |     |  |                 +--rw destination-port-range-or-operator
   |  |  |     |  |                    +--rw (port-range-or-operator)?
   |  |  |     |  |                       +--:(range)
   |  |  |     |  |          |  +--rw lower-port
   |  |  |     |  |          |  |       inet:port-number
   |  |  |     |  |          |  +--rw upper-port
   |  |  |     |  |          |          inet:port-number
   |  |  |     |  |                       +--:(operator)
   |  |  |     |  |                          +--rw operator?
   |  |  |     |  |             |       operator
   |  |  |     |  |                          +--rw port
```

```
   |  |  |     |  |                                 inet:port-number
   |  |  |     |  +--:(match-application)
   |  |  |     |     +--rw match-application?
   |  |  |     |             identityref
   |  |  |     +--rw target-class-id?
   |  |  |             string
   |  |  +--rw qos-profile
   |  |     +--rw qos-profile* [profile]
   |  |        +--rw profile      leafref
   |  |        +--rw direction?   identityref
   |  +--rw precedence
   |     +--rw precedence?   identityref
```

Figure 10

4. Relation with other YANG Models

The L2NM model, aimed at managing the L2VPN Services in a Service Provider Network controller/orchestrator has relations with other YANG modules.

4.1. Relation with L2SM

[RFC8466] defines a L2VPN Service YANG data Model (L2SM) that can be used for communication between customers and VPN service providers. Hence, the model provides inputs to the Network Operator to deliver such service to the customer. Hence, most parts of the model can be directly mapped into L2NM.

- o Service requirements: The service requirements can be directly taken from L2SM to L2NM.
- o Sites: The sites from L2SM are used to select the Service Prodider node. The site information is NOT maintained in L2NM

4.2. Relation with Network Topology

The L2NM model manages VPN Services running over Service Provider Backbone network. The set of nodes over which it is possible to deploy a L2 VPN Service MAY be part of the topology contained in an ietf-network module.

4.3. Relation with Device Models

Creating services in the l2vpn-ntw module will will lead at some point to the configuration of devices. Hence, it is foreseen that the data for the device yang modules will be derived partially from the L2NM vpn-service container. Note that L2NM is NOT a device model.

5. YANG Module

```
<CODE BEGINS>file "ietf-l2vpn-ntw@2020-11-02.yang"
module ietf-l2vpn-ntw {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l2vpn-ntw";
  prefix l2vpn-ntw;

  import ietf-inet-types {
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }
  import ietf-vpn-common {
    prefix vpn-common;
    reference
      "RFC CCCC: A Layer 2/3 VPN Common YANG Model";
  }

  organization
    "IETF OPSA (Operations and Management Area) Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>

     Editor:    Samier Barguil
               <mailto:samier.barguilgiraldo.ext@telefonica.com>
     Editor:    Oscar Gonzalez de Dios
               <mailto:oscar.gonzalezdedios@telefonica.com>
     Author:    Mohamed Boucadair
               <mailto:mohamed.boucadair@orange.com>
     Author:    Luis Angel Munoz
                <mailto:luis-angel.munoz@vodafone.com>
     Author:    Luay Jalil
                <mailto:luay.jalil@verizon.com>
     Author:    Jichun Ma
                <mailto:majc16@chinaunicom.cn>
    ";
  description
    "The YANG module defines a generic network configuration
```

```
       model for Layer 2 VPN services common across all of the
       vendor implementations.

       Copyright (c) 2020 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

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

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

    revision 2020-11-02 {
      description
        "Initial version.";
      reference
        "RFC XXXX: A Layer 2 VPN Network YANG Model.";
    }

    /* Features */

    feature multicast-like {
      description
        "Indicates the support of multicast-like capabilities
         in a L2VPN.";
    }

    feature target-sites {
      description
        "Indicates the support of 'target-sites' match flow
         parameter.";
    }

    feature l2cp-control {
      description
        "Indicates the support of L2CP control.";
    }

    feature output-bw {
      description
        "Indicates the support of Output Bandwidth in
         a VPN";
    }
```

```
  feature uni-list {
    description
      "Indicates thesupport of UNI list in a VPN.";
  }

  feature oam-3ah {
    description
      "Indicates the support of OAM 802.3ah.";
  }

  feature micro-bfd {
    description
      "Indicates the support of Micro-BFD.";
  }

  feature signaling-options {
    description
      "Indicates the support of signalling option.";
  }

  feature always-on {
    description
      "Indicates the support for always-on access
       constraint.";
  }

  feature requested-type {
    description
      "Indicates the support for requested-type access
       constraint.";
  }

  feature vlan {
    description
      "Indicates the support of VLAN.";
  }

  feature sub-inf {
    description
      "Indicates the support of Sub Interface.";
  }

  feature atm {
    description
      "Indicates the support of ATM.";
  }

  feature vxlan {
```

```
    description
      "Indicates the support of VxLAN.";
  }

  feature lan-tag {
    description
      "Indicates the LAN Tag support in a VPN.";
  }

  /* Typedefs */
  /* Identities */

  identity mapping-type {
    base vpn-common:multicast-gp-address-mapping;
    description
      "Identity mapping-type.";
  }

  identity protection-mode {
    description
      "Identity of protection mode";
  }

  identity oneplusone {
    base protection-mode;
    description
      "In this scheme, the primary circuit will be
       protected by a backup circuit, typically meeting certain
       diverse path/fiber/site/node criteria.  Both primary and
       protection circuits are provisioned to be in the active
       forward ing state. The subscriber may choose to send the
       same service frames across both circuits simultaneously.";
  }

  identity one-to-one {
    base protection-mode;
    description
      "In this scheme, a backup circuit to the primary
       circuit is provisioned.  Depending on the implementation
       agreement, the protection circuits may either always be
       in active forwarding state, or may only become active when
       a faulty state is detected on the primary circuit.";
  }

  identity bundling-type {
    description
      "The base identity for the bundling type.  It supports
       multiple CE-VLANs associated with an L2VPN service or
```

```
       all CE-VLANs associated with an L2VPN service.";
  }

  identity multi-svc-bundling {
    base bundling-type;
    description
      "Identity for multi-service bundling, i.e.,
       multiple CE-VLAN IDs can be associated with an
       L2VPN service at a site.";
  }

  identity one2one-bundling {
    base bundling-type;
    description
      "Identity for one-to-one service bundling, i.e.,
       each L2VPN can be associated with only one CE-VLAN ID
       at a site.";
  }

  identity all2one-bundling {
    base bundling-type;
    description
      "Identity for all-to-one bundling, i.e., all CE-VLAN IDs
       are mapped to one L2VPN service.";
  }

  identity color-id {
    description
      "Base identity of the color ID.";
  }

  identity color-id-cvlan {
    base color-id;
    description
      "Identity of the color ID based on a CVLAN.";
  }

  identity color-type {
    description
      "Identity of color types.";
  }

  identity green {
    base color-type;
    description
      "Identity of the 'green' color type.";
  }
```

```
  identity yellow {
    base color-type;
    description
      "Identity of the 'yellow' color type.";
  }

  identity red {
    base color-type;
    description
      "Identity of the 'red' color type.";
  }

  identity perf-tier-opt {
    description
      "Identity of performance tier option.";
  }

  identity metro {
    base perf-tier-opt;
    description
      "Identity of metro";
  }

  identity regional {
    base perf-tier-opt;
    description
      "Identity of regional";
  }

  identity continental {
    base perf-tier-opt;
    description
      "Identity of continental";
  }

  identity global {
    base perf-tier-opt;
    description
      "Identity of global";
  }

  identity policing {
    description
      "Identity of policing type";
  }

  identity one-rate-two-color {
    base policing;
```

```
    description
      "Identity of one-rate, two-color (1R2C)";
  }

  identity two-rate-three-color {
    base policing;
    description
      "Identity of two-rate, three-color (2R3C)";
  }

  identity loop-prevention-type {
    description
      "Identity of loop prevention.";
  }

  identity shut {
    base loop-prevention-type;
    description
      "Identity of shut protection.";
  }

  identity trap {
    base loop-prevention-type;
    description
      "Identity of trap protection.";
  }

  identity t-ldp-pwe-type {
    description
      "Identity for t-ldp-pwe-type.";
  }

  identity vpws-type {
    base t-ldp-pwe-type;
    description
      "Identity for VPWS";
  }

  identity vpls-type {
    base t-ldp-pwe-type;
    description
      "Identity for vpls";
  }

  identity hvpls {
    base t-ldp-pwe-type;
    description
      "Identity for h-vpls";
```

```
  }

  identity l2vpn-type {
    description
      "Layer 2 VPN types";
  }

  identity l2vpn-vpws {
    base l2vpn-type;
    description
      "VPWS L2VPN type.";
  }

  identity l2vpn-vpls {
    base l2vpn-type;
    description
      "VPLS L2VPN type.";
  }

  identity distribute-vpls {
    base l2vpn-type;
    description
      "distribute VPLS L2VPN type.";
  }

  identity evpn-type {
    description
      "Ethernet VPN types";
  }

  identity evpn-vpws {
    base evpn-type;
    description
      "VPWS support in EVPN.";
  }

  identity evpn-pbb {
    base evpn-type;
    description
      " Provider Backbone Bridging Support in EVPN.";
  }

  identity pm-type {
    description
      "Performance-monitoring type.";
  }

  identity loss {
```

```
    base pm-type;
    description
      "Loss measurement.";
  }

  identity delay {
    base pm-type;
    description
      "Delay measurement.";
  }

  identity mac-learning-mode {
    description
      "MAC learning mode.";
  }

  identity data-plane {
    base mac-learning-mode;
    description
      "User MAC addresses are learned through ARP broadcast.";
  }

  identity control-plane {
    base mac-learning-mode;
    description
      "User MAC addresses are advertised through EVPN-BGP.";
  }

  identity mac-action {
    description
      "Base identity for a MAC action.";
  }

  identity drop {
    base mac-action;
    description
      "Identity for dropping a packet.";
  }

  identity flood {
    base mac-action;
    description
      "Identity for packet flooding.";
  }

  identity warning {
    base mac-action;
    description
```

```
      "Identity for sending a warning log message.";
  }

  identity load-balance-method {
    description
      "Base identity for load balance method.";
  }

  identity fat-pw {
    base load-balance-method;
    description
      "Identity for Fat PW. Fat label is
       applied to Pseudowires across MPLS
       network.";
  }

  identity entropy-label {
    base load-balance-method;
    description
      "Identity for entropy label.Entropy label
       is applied to IP forwarding,
       L2VPN or L3VPN across MPLS network";
  }

  identity vxlan-source-port {
    base load-balance-method;
    description
      "Identity for vxlan source port.VxLAN
       Source Port is one load balancing method.";
  }

  identity precedence-type {
    description
      "Redundancy type. The service can be created
       with active and bakcup signalization.";
  }

  identity primary {
    base precedence-type;
    description
      "Identifies the Main L2VPN.";
  }

  identity backup {
    base precedence-type;
    description
      "Identifies the Backup L2VPN.";
  }
```

```
  /* Groupings */

  grouping cfm-802-grouping {
    leaf maid {
      type string;
      description
        "MA ID";
    }
    leaf mep-id {
      type uint32;
      description
        "Local MEP ID";
    }
    leaf mep-level {
      type uint32;
      description
        "MEP level";
    }
    leaf mep-up-down {
      type enumeration {
        enum up {
          description
            "MEP up";
        }
        enum down {
          description
            "MEP down";
        }
      }
      description
        "MEP up/down";
    }
    leaf remote-mep-id {
      type uint32;
      description
        "Remote MEP ID";
    }
    leaf cos-for-cfm-pdus {
      type uint32;
      description
        "COS for CFM PDUs";
    }
    leaf ccm-interval {
      type uint32;
      description
        "CCM interval";
    }
    leaf ccm-holdtime {
```

```
      type uint32;
      description
        "CCM hold time";
    }
    leaf ccm-p-bits-pri {
      type vpn-common:ccm-priority-type;
      description
        "The priority parameter for CCMs transmitted by the MEP";
    }
    description
      "Grouping for 802.1ag CFM attribute";
  }

  grouping y-1731 {
    list y-1731 {
      key "maid";
      leaf maid {
        type string;
        description
          "MA ID ";
      }
      leaf mep-id {
        type uint32;
        description
          "Local MEP ID";
      }
      leaf type {
        type identityref {
          base pm-type;
        }
        description
          "Performance monitor types";
      }
      leaf remote-mep-id {
        type uint32;
        description
          "Remote MEP ID";
      }
      leaf message-period {
        type uint32;
        description
          "Defines the interval between OAM messages. The message
           period is expressed in milliseconds";
      }
      leaf measurement-interval {
        type uint32;
        description
          "Specifies the measurement interval for statistics. The
```

```
             measurement interval is expressed in seconds";
        }
        leaf cos {
          type uint32;
          description
            "Class of service";
        }
        leaf loss-measurement {
          type boolean;
          description
            "Whether enable loss measurement";
        }
        leaf synthethic-loss-measurement {
          type boolean;
          description
            "Indicate whether enable synthetic loss measurement";
        }
        container delay-measurement {
          leaf enable-dm {
            type boolean;
            description
              "Whether to enable delay measurement";
          }
          leaf two-way {
            type boolean;
            description
              "Whether delay measurement is two-way (true) of one-
               way (false)";
          }
          description
            "Container for delay measurement";
        }
        leaf frame-size {
          type uint32;
          description
            "Frame size";
        }
        leaf session-type {
          type enumeration {
            enum proactive {
              description
                "Proactive mode";
            }
            enum on-demand {
              description
                "On demand mode";
            }
          }
```

```
        description
          "Session type";
      }
      description
        "List for y-1731.";
    }
    description
      "Grouping for y.1731";
  }

  /* MAIN L2VPN SERVICE */

  container l2vpn-ntw {
    container vpn-profiles {
      uses vpn-common:vpn-profile-cfg;
      description
        "Container for VPN Profiles.";
    }
    container vpn-services {
      list vpn-service {
        key "vpn-id";
        uses vpn-common:service-status;
        uses vpn-common:vpn-description;
        leaf l2sm-vpn-id {
          type vpn-common:vpn-id;
          description
            "Pointer to the L2SM service.";
        }
        leaf vpn-svc-type {
          type identityref {
            base vpn-common:vpn-signaling-type;
          }
          description
            "Service type";
        }
        leaf svc-topo {
          type identityref {
            base vpn-common:vpn-topology;
          }
          description
            "Defining service topology, such as
             any-to-any, hub-spoke, etc.";
        }
        container multicast-like {
          if-feature "vpn-common:multicast";
          leaf enabled {
            type boolean;
            default "false";
```

```
            description
              "Enables multicast.";
          }
          container customer-tree-flavors {
            leaf-list tree-flavor {
              type identityref {
                base vpn-common:multicast-tree-type;
              }
              description
                "Type of tree to be used.";
            }
            description
              "Type of trees used by customer.";
          }
          description
            "Multicast like container";
        }
        container extranet-vpns {
          if-feature "vpn-common:extranet-vpn";
          list extranet-vpn {
            key "vpn-id";
            leaf vpn-id {
              type vpn-common:vpn-id;
              description
                "Identifies the target VPN.";
            }
            leaf local-sites-role {
              type identityref {
                base vpn-common:role;
              }
              default "vpn-common:any-to-any-role";
              description
                "This describes the role of the
                 local sites in the target VPN topology.";
            }
            description
              "List of extranet VPNs the local VPN is attached to.";
          }
          description
            "Container for extranet VPN configuration.";
        }
        leaf svc-mtu {
          type uint32;
          description
            "SVC MTU, it is also known as the maximum transmission unit
             or maximum frame size,When a frame is larger than the MTU,
             it is broken down, or fragmented, into smaller pieces by the
             network protocol to accommodate the MTU of the network";
```

```
        }
        leaf ce-vlan-preservation {
          type boolean;
          description
            "Preserve the CE-VLAN ID from ingress to egress,i.e.,
             CE-VLAN tag of the egress frame are identical to
             those of the ingress frame that yielded this egress
             service frame. If All-to-One bundling within a site
             is Enabled, then preservation applies to all Ingress
             service frames. If All-to-One bundling is Disabled,
             then preservation applies to tagged Ingress service
             frames having CE-VLAN ID 1 through 4094.";
        }
        leaf ce-vlan-cos-perservation {
          type boolean;
          description
            "CE vlan CoS preservation. PCP bits in the CE-VLAN tag
             of the egress frame are identical to those of the ingress
             frame that yielded this egress service frame.";
        }
        uses vpn-common:svc-transport-encapsulation;
        container vpn-nodes {
          list vpn-node {
            key "vpn-node-id ne-id";
            leaf vpn-node-id {
              type vpn-common:vpn-id;
              description
                "";
            }
            leaf description {
              type string;
              description
                "Textual description of a VPN node.";
            }
            leaf node-role {
              type identityref {
                base vpn-common:role;
              }
              default "vpn-common:any-to-any-role";
              description
                "Role of the vpn-node in the IP VPN.";
            }
            leaf ne-id {
              type string;
              description
                "NE IP address";
            }
            leaf port-id {
```

```
              type string;
              description
                "NE Port-id";
            }
            uses vpn-common:service-status;
            list signaling-options {
              key "type";
              leaf type {
                type identityref {
                  base vpn-common:vpn-signaling-type;
                }
                description
                  "VPN signaling types";
              }
              container l2vpn-bgp {
                when "/l2vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/sign
aling-options/type = 'vpn-common:l2vpn-bgp'" {
                  description
                    "Only applies when vpn signaling type is l2vpn
                     BGP protocol.";
                }
                leaf pwe-encapsulation-type {
                  type identityref {
                    base vpn-common:encapsulation-type;
                  }
                  description
                    "PWE Encapsulation Type";
                }
                uses vpn-common:vpn-route-targets;
                container pwe-mtu {
                  leaf allow-mtu-mismatch {
                    type boolean;
                    description
                      "Allow MTU mismatch";
                  }
                  description
                    "Container of PWE MTU configurations";
                }
                leaf address-family {
                  type vpn-common:address-family;
                  description
                    "Address family used for router-id information.";
                }
                description
                  "Container for MP BGP L2VPN";
              }
              container evpn-bgp {
                when "/l2vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/sign
aling-options/type = 'vpn-common:evpn-bgp'" {
                  description
```

```
                    "Only applies when vpn signaling type is EVPN
                     BGP protocol.";
                }
                leaf vpn-id {
                  type leafref {
                    path "/l2vpn-ntw/vpn-services/vpn-service/vpn-id";
                  }
                  description
                    "Identifies the target EVPN";
                }
                leaf type {
                  type identityref {
                    base evpn-type;
                  }
                  description
                    "L2VPN types";
                }
                leaf address-family {
                  type vpn-common:address-family;
                  description
                    "Address family used for router-id information.";
                }
                leaf mac-learning-mode {
                  type identityref {
                    base mac-learning-mode;
                  }
                  description
                    "Indicates through which plane MAC addresses are
                     advertised.";
                }
                leaf arp-suppress {
                  type boolean;
                  default "false";
                  description
                    "Indicates whether to suppress ARP broadcast.";
                }
                description
                  "Container for MP BGP L2VPN";
              }
              container t-ldp-pwe {
                when "/l2vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/sign
aling-options/type = 'vpn-common:t-ldp'" {
                  description
                    "Only applies when vpn signaling type is Target LDP.";
                }
                leaf type {
                  type identityref {
                    base t-ldp-pwe-type;
                  }
```

```
                  description
                    "T-LDP PWE type";
                }
                leaf pwe-encapsulation-type {
                  type identityref {
                    base vpn-common:encapsulation-type;
                  }
                  description
                    "PWE Encapsulation Type.";
                }
                leaf pwe-mtu {
                  type uint16;
                  description
                    "Allow MTU mismatch";
                }
                list ac-pw-list {
                  key "peer-addr vc-id";
                  leaf peer-addr {
                    type inet:ip-address;
                    description
                      "Peer IP address.";
                  }
                  leaf vc-id {
                    type vpn-common:vpn-id;
                    description
                      "VC lable used to identify PW.";
                  }
                  leaf pw-type {
                    type identityref {
                      base vpn-common:vpn-topology;
                    }
                    description
                      "PW topology type";
                  }
                  leaf pw-priority {
                    type uint32;
                    description
                      "Defines the priority for the PW.
                       The higher the pw-priority value,
                       the higher the preference of the PW will be.";
                  }
                  description
                    "List of AC and PW bindings.";
                }
                container qinq {
                  when "/l2vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/si
gnaling-options/type = 'vpn-common:h-vpls'" {
                    description
                      "Only applies when t-ldp pwe type is h-vpls.";
```

```
                  }
                  leaf s-tag {
                    type uint32;
                    description
                      "S-TAG";
                  }
                  leaf c-tag {
                    type uint32;
                    description
                      "C-TAG";
                  }
                  description
                    "Container for QinQ";
                }
                description
                  "Container of T-LDP PWE configurations";
              }
              container l2tp-pwe {
                when "/l2vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/sign
aling-options/type = 'vpn-common:l2tp'" {
                  description
                    "Applies when vpn signaling type is L2TP protocol.";
                }
                leaf type {
                  type identityref {
                    base t-ldp-pwe-type;
                  }
                  description
                    "T-LDP PWE type";
                }
                leaf encapsulation-type {
                  type identityref {
                    base vpn-common:encapsulation-type;
                  }
                  description
                    "Encapsulation type";
                }
                list ac-pw-list {
                  key "peer-addr vc-id";
                  leaf peer-addr {
                    type inet:ip-address;
                    description
                      "Peer IP address.";
                  }
                  leaf vc-id {
                    type string;
                    description
                      "VC lable used to identify PW.";
                  }
```

```
                 leaf pw-priority {
                   type uint32;
                   description
                     "PW priority";
                 }
                 description
                   "List of AC and PW bindings.";
               }
               description
                 "Container for l2tp pw";
             }
             description
               "List of VPN Signaling Option.";
           }
           container vpn-network-accesses {
             list vpn-network-access {
               key "id";
               leaf id {
                 type vpn-common:vpn-id;
                 description
                   "Identifier of network access";
               }
               leaf description {
                 type string;
                 description
                   "String to describe the element.";
               }
               leaf Interface-mtu {
                 type uint32;
                 description
                   "Interface MTU, it is also known as the maximum
                    transmission unit or maximum frame size. When a
                    frame is larger than the MTU, it is broken down,
                    or fragmented, into smaller pieces by the
                    network protocol to accommodate the MTU of the
                    network";
               }
               uses vpn-common:service-status;
               container access-diversity {
                 if-feature "vpn-common:placement-diversity";
                 container groups {
                   leaf fate-sharing-group-size {
                     type uint16;
                     description
                       "Fate sharing group size.";
                   }
                   leaf group-color {
                     type string;
```

```
          description
            "Group color associated with a particular VPN.";
        }
        list group {
          key "group-id";
          leaf group-id {
            type string;
            description
              "Group-id the site network access
               is belonging to";
          }
          description
            "List of group-id";
        }
        description
          "Groups the fate sharing group member
           is belonging to";
      }
      container constraints {
        list constraint {
          key "constraint-type";
          leaf constraint-type {
            type identityref {
              base vpn-common:placement-diversity;
            }
            description
              "Diversity constraint type.";
          }
          container target {
            choice target-flavor {
              case id {
                list group {
                  key "group-id";
                  leaf group-id {
                    type string;
                    description
                      "The constraint will apply
                       against this particular
                       group-id";
                  }
                  description
                    "List of groups";
                }
              }
              case all-accesses {
                leaf all-other-accesses {
                  type empty;
                  description
```

```
                            "The constraint will apply
                             against all other site network
                             access of this site";
                        }
                      }
                      case all-groups {
                        leaf all-other-groups {
                          type empty;
                          description
                            "The constraint will apply
                             against all other groups the
                             customer is managing";
                        }
                      }
                      description
                        "Choice for the group definition";
                    }
                    description
                      "The constraint will apply against
                       this list of groups";
                  }
                  description
                    "List of constraints";
                }
                description
                  "Constraints for placing this site
                    network access";
              }
              description
                "Diversity parameters.";
            }
            container connection {
              leaf encapsulation-type {
                type identityref {
                  base vpn-common:encapsulation-type;
                }
                description
                  "Encapsulation Type";
              }
              leaf-list eth-inf-type {
                type identityref {
                  base vpn-common:encapsulation-type;
                }
                description
                  "Ethernet Interface Type";
              }
              container dot1q-interface {
                leaf l2-access-type {
```

```
                    type identityref {
                      base vpn-common:encapsulation-type;
                    }
                    description
                      "L2 Access Encapsulation Type";
                  }
                  container dot1q {
                    when "../l2-access-type='vpn-common:dot1q'";
                    if-feature "vpn-common:dot1q";
                    leaf physical-inf {
                      type string;
                      description
                        "Physical Interface";
                    }
                    leaf c-vlan-id {
                      type uint32;
                      description
                        "VLAN identifier";
                    }
                    description
                      "Qot1q";
                  }
                  container qinq {
                    when "../l2-access-type='vpn-common:qinq'";
                    if-feature "vpn-common:qinq";
                    leaf s-vlan-id {
                      type uint32;
                      description
                        "S-VLAN Identifier";
                    }
                    leaf c-vlan-id {
                      type uint32;
                      description
                        "C-VLAN Identifier";
                    }
                    description
                      "QinQ";
                  }
                  container qinany {
                    if-feature "vpn-common:qinany";
                    leaf s-vlan-id {
                      type uint32;
                      description
                        "S-Vlan ID";
                    }
                    description
                      "Container for Q in Any";
                  }
```

```
                      container vxlan {
                        when "../l2-access-type='vpn-common:vxlan'";
                        if-feature "vxlan";
                        leaf vni-id {
                          type uint32;
                          description
                            "VNI Identifier";
                        }
                        leaf peer-mode {
                          type identityref {
                            base vpn-common:vxlan-peer-mode;
                          }
                          description
                            "specify the vxlan access mode";
                        }
                        list peer-list {
                          key "peer-ip";
                          leaf peer-ip {
                            type inet:ip-address;
                            description
                              "Peer IP";
                          }
                          description
                            "List for peer IP";
                        }
                        description
                          "QinQ";
                      }
                      description
                        "Container for dot1Q Interface";
                    }
                    container phy-interface {
                      leaf port-number {
                        type uint32;
                        description
                          "Port number";
                      }
                      leaf port-speed {
                        type uint32;
                        description
                          "Port speed";
                      }
                      leaf mode {
                        type vpn-common:neg-mode;
                        description
                          "Negotiation mode";
                      }
                      leaf phy-mtu {
```

```
                    type uint32;
                    description
                      "PHY MTU";
                  }
                  leaf flow-control {
                    type string;
                    description
                      "Flow control";
                  }
                  container oam-802.3ah-link {
                    if-feature "oam-3ah";
                    leaf enable {
                      type boolean;
                      description
                        "Indicate whether support oam 802.3 ah link";
                    }
                    description
                      "Container for oam 802.3 ah link.";
                  }
                  leaf uni-loop-prevention {
                    type boolean;
                    description
                      "If this leaf set to truth that the port automatically
                       goes down when a physical loopback is detect.";
                  }
                  description
                    "Container of PHY Interface Attributes configurations";
                }
                container lag-interface {
                  if-feature "vpn-common:lag-interface";
                  list lag-interface {
                    key "lag-interface-number";
                    leaf lag-interface-number {
                      type uint32;
                      description
                        "LAG interface number";
                    }
                    container lacp {
                      leaf lacp-state {
                        type boolean;
                        description
                          "LACP on/off";
                      }
                      leaf lacp-mode {
                        type boolean;
                        description
                          "LACP mode";
                      }
```

```
                        leaf lacp-speed {
                          type boolean;
                          description
                            "LACP speed";
                        }
                        leaf mini-link {
                          type uint32;
                          description
                            "The minimum aggregate bandwidth for a LAG";
                        }
                        leaf system-priority {
                          type uint16;
                          description
                            "Indicates the LACP priority for the system.
                             The range is from 0 to 65535.
                             The default is 32768.";
                        }
                        container member-link-list {
                          list member-link {
                            key "name";
                            leaf name {
                              type string;
                              description
                                "Member link name";
                            }
                            leaf port-speed {
                              type uint32;
                              description
                                "Port speed";
                            }
                            leaf mode {
                              type vpn-common:neg-mode;
                              description
                                "Negotiation mode";
                            }
                            leaf link-mtu {
                              type uint32;
                              description
                                "Link MTU size.";
                            }
                            container oam-802.3ah-link {
                              if-feature "oam-3ah";
                              leaf enable {
                                type boolean;
                                description
                                  "Indicate whether support oam 802.3 ah link";
                              }
                              description
```

```
                         "Container for oam 802.3 ah link.";
                     }
                     description
                       "Member link";
                   }
                   description
                     "Container of Member link list";
                 }
                 leaf flow-control {
                   type string;
                   description
                     "Flow control";
                 }
                 leaf lldp {
                   type boolean;
                   description
                     "LLDP";
                 }
                 description
                   "LACP";
               }
               description
                 "List of LAG interfaces";
             }
             description
               "Container of LAG interface attributes configuration";
           }
           list cvlan-id-to-svc-map {
             key "svc-id";
             leaf svc-id {
               type leafref {
                 path "/l2vpn-ntw/vpn-services/vpn-service/vpn-id";
               }
               description
                 "VPN Service identifier";
             }
             list cvlan-id {
               key "vid";
               leaf vid {
                 type uint32;
                 description
                   "CVLAN ID";
               }
               description
                 "List of CVLAN-ID to SVC Map configurations";
             }
             description
               "List for cvlan-id to L2VPn Service map configurations";
```

```
                }
                container split-horizon {
                  leaf group-name {
                    type string;
                    description
                      "group-name of the Split Horizon";
                  }
                  description
                    "Configuration with split horizon enabled";
                }
                description
                  "Container for bearer";
              }
              container availability {
                leaf access-priority {
                  type uint32;
                  description
                    "Access priority";
                }
                choice redundancy-mode {
                  case single-active {
                    leaf single-active {
                      type boolean;
                      description
                        "Single active";
                    }
                    description
                      "Single active case";
                  }
                  case all-active {
                    leaf all-active {
                      type boolean;
                      description
                        "All active";
                    }
                    description
                      "All active case";
                  }
                  description
                    "Redundancy mode choice";
                }
                description
                  "Container of availability optional configurations";
              }
              container service {
                container svc-input-bandwidth {
                  if-feature "vpn-common:input-bw";
                  list input-bandwidth {
```

```
                      key "type";
                      leaf type {
                        type identityref {
                          base vpn-common:bw-type;
                        }
                        description
                          "Bandwidth Type";
                      }
                      leaf cos-id {
                        type uint8;
                        description
                          "Identifier of Class of Service
                           , indicated by DSCP or a CE-CLAN
                           CoS(802.1p)value in the service frame.";
                      }
                      leaf cir {
                        type uint64;
                        description
                          "Committed Information Rate. The maximum number of
                           bits that a port can receive or send during
                           one-second over an interface.";
                      }
                      leaf cbs {
                        type uint64;
                        description
                          "Committed Burst Size.CBS controls the bursty nature
                           of the traffic. Traffic that does not use the
                           configured CIR accumulates credits until the credits
                           reach the configured CBS.";
                      }
                      leaf eir {
                        type uint64;
                        description
                          "Excess Information Rate,i.e.,Excess frame delivery
                           allowed not subject to SLA.The traffic rate can be
                           limited by eir.";
                      }
                      leaf ebs {
                        type uint64;
                        description
                          "Excess Burst Size. The bandwidth available for burst
                           traffic from the EBS is subject to the amount of
                           bandwidth that is accumulated during periods when
                           traffic allocated by the EIR policy is not used.";
                      }
                      leaf pir {
                        type uint64;
                        description
```

```
                "Peak Information Rate, i.e., maixmum frame delivery
                 allowed. It is equal to or less than sum of cir and
                 eir.";
            }
            leaf pbs {
              type uint64;
              description
                "Peak Burst Size. It is measured in bytes per second.";
            }
            description
              "List for input bandwidth";
          }
          description
            "From the PE perspective, the service input
             bandwidth of the connection.";
        }
        container svc-output-bandwidth {
          if-feature "output-bw";
          list output-bandwidth {
            key "type";
            leaf type {
              type identityref {
                base vpn-common:bw-type;
              }
              description
                "Bandwidth Type";
            }
            leaf cos-id {
              type uint8;
              description
                "Identifier of Class of Service
                 , indicated by DSCP or a CE-CLAN
                 CoS(802.1p)value in the service frame.";
            }
            leaf cir {
              type uint64;
              description
                "Committed Information Rate. The maximum number of
                 bits that a port can receive or send during
                 one-second over an interface.";
            }
            leaf cbs {
              type uint64;
              description
                "Committed Burst Size.CBS controls the bursty nature
                 of the traffic. Traffic that does not use the
                 configured CIR accumulates credits until the credits
                 reach the configured CBS.";
```

```
              }
              leaf eir {
                type uint64;
                description
                  "Excess Information Rate,i.e.,Excess frame delivery
                   allowed not subject to SLA.The traffic rate can be
                   limited by eir.";
              }
              leaf ebs {
                type uint64;
                description
                  "Excess Burst Size. The bandwidth available for burst
                   traffic from the EBS is subject to the amount of
                   bandwidth that is accumulated during periods when
                   traffic allocated by the EIR policy is not used.";
              }
              leaf pir {
                type uint64;
                description
                  "Peak Information Rate, i.e., maixmum frame delivery
                   allowed. It is equal to or less than sum of cir and
                   eir.";
              }
              leaf pbs {
                type uint64;
                description
                  "Peak Burst Size. It is measured in bytes per second.";
              }
              description
                "List for output bandwidth";
            }
            description
              "From the PE perspective, the service output
               bandwidth of the connection.";
          }
          container qos {
            if-feature "vpn-common:qos";
            container qos-classification-policy {
              uses vpn-common:qos-classification-policy;
              description
                "Configuration of the traffic classification
                 policy.";
            }
            container qos-profile {
              list qos-profile {
                key "profile";
                description
                  "QoS profile.
```

```
                     Can be standard profile or customized
                     profile.";
                  leaf profile {
                    type leafref {
                      path "/l2vpn-ntw/vpn-profiles"
                         + "/valid-provider-identifiers"
                         + "/qos-profile-identifier/id";
                    }
                    description
                      "QoS profile to be used.";
                  }
                  leaf direction {
                    type identityref {
                      base vpn-common:qos-profile-direction;
                    }
                    default "vpn-common:both";
                    description
                      "The direction to which the QoS profile
                       is applied.";
                  }
                }
                description
                  "QoS profile configuration.";
              }
              description
                "QoS configuration.";
            }
            container precedence {
              leaf precedence {
                type identityref {
                  base precedence-type;
                }
                description
                  "Defining service redundancy in transport
                   network.";
              }
              description
                "Transport netowrk precedence selector
                 Primary or Secondary tunnel.";
            }
            description
              "Container for service";
          }
          container broadcast-unknown-unicast-multicast {
            leaf multicast-site-type {
              type enumeration {
                enum receiver-only {
                  description
```

```
                   "The site only has receivers.";
               }
               enum source-only {
                 description
                   "The site only has sources.";
               }
               enum source-receiver {
                 description
                   "The site has both sources and receivers.";
               }
             }
             default "source-receiver";
             description
               "Type of multicast site.";
           }
           list multicast-gp-address-mapping {
             key "id";
             leaf id {
               type uint16;
               description
                 "Unique identifier for the mapping.";
             }
             leaf vlan-id {
               type uint32;
               description
                 "The VLAN ID of the Multicast group.";
             }
             leaf mac-gp-address {
               type yang:mac-address;
               description
                 "The MAC address of the Multicast group.";
             }
             leaf port-lag-number {
               type uint32;
               description
                 "The ports/LAGs belonging to the Multicast group.";
             }
             description
               "List of Port to group mappings.";
           }
           leaf bum-overall-rate {
             type uint32;
             description
               "overall rate for BUM";
           }
           description
             "Container of broadcast, unknown unicast, and multicast
              configurations";
```

```
                }
                container ethernet-service-oam {
                  leaf md-name {
                    type string;
                    description
                      "Maintenance domain name";
                  }
                  leaf md-level {
                    type uint8;
                    description
                      "Maintenance domain level";
                  }
                  container cfm-802.1-ag {
                    list n2-uni-c {
                      key "maid";
                      uses cfm-802-grouping;
                      description
                        "List of UNI-N to UNI-C";
                    }
                    list n2-uni-n {
                      key "maid";
                      uses cfm-802-grouping;
                      description
                        "List of UNI-N to UNI-N";
                    }
                    description
                      "Container of 802.1ag CFM configurations.";
                  }
                  uses y-1731;
                  description
                    "Container for Ethernet service OAM.";
                }
                container mac-loop-prevention {
                  leaf frequency {
                    type uint32;
                    description
                      "Frequency";
                  }
                  leaf protection-type {
                    type identityref {
                      base loop-prevention-type;
                    }
                    description
                      "Protection type";
                  }
                  leaf number-retries {
                    type uint32;
                    description
```

```
                  "Number of retries";
              }
              description
                "Container of MAC loop prevention.";
            }
            container access-control-list {
              list mac {
                key "mac-address";
                leaf mac-address {
                  type yang:mac-address;
                  description
                    "MAC address.";
                }
                description
                  "List for MAC.";
              }
              description
                "Container for access control List.";
            }
            container mac-addr-limit {
              leaf mac-num-limit {
                type uint16;
                description
                  "maximum number of MAC addresses learned from
                   the subscriber for a single service instance.";
              }
              leaf time-interval {
                type uint32;
                units "milliseconds";
                description
                  "The aging time of the mac address.";
              }
              leaf action {
                type identityref {
                  base mac-action;
                }
                description
                  "specify the action when the upper limit is
                   exceeded: drop the packet, flood the
                   packet, or simply send a warning log message.";
              }
              description
                "Container of MAC-Addr limit configurations";
            }
            description
              "List of VPN Network Accesses.";
          }
          description
```

```
                "List of VPN Nodes.";
            }
            description
              "Container of VPN Nodes.";
          }
          description
            "List of vpn-svc";
        }
        description
          "Container of port configurations";
      }
      description
        "Container for L2VPN service";
    }
    description
      "Container for VPN services.";
  }
}

<CODE ENDS>
```

Figure 11

6. Acknowledgements

The authors would like to thank Tom Petch for the comments to improve the document.

7. Contributors

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

Victor Lopez
Telefonica
Email: victor.lopezalvarez@telefonica.com

Zhang Guiyu
China Unicom
Email: zhanggy113@chinaunicom.cn

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

8. IANA Considerations

This document requests IANA to register the following URI in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-l2vpn-ntw

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

This document requests IANA to register the following YANG module in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

name: ietf-l2vpn-ntw

namespace: urn:ietf:params:xml:ns:yang:ietf-l2vpn-ntw

maintained by IANA: N

prefix: l2vpn-ntw

reference: RFC XXXX

9. 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 [RFC8466].

The Network Configuration Access Control Model (NACM) [RFC8341] 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.

The ietf-l2vpn-ntw module is used to manage L2 VPNs in a service provider backbone network. Hence, the module can be used to request, modify, or retrieve L2VPN services. 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) and delete operations to these data nodes without proper protection or authentication can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability in the ietf-l2vpn-ntw module:

- o vpn-service: An attacker who is able to access network nodes can undertake various attacks, such as deleting a running L2 VPN Service, interrupting all the traffic of a client.

Some of the readable data nodes in this 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:

- o customer-name: An attacker can retrieve privacy-related information which can be used to track a customer. Disclosing such information may be considered as a violation of the customer-provider trust relationship.

10. References

10.1. Normative References

[I-D.ietf-opsawg-vpn-common]
barguil, s., Dios, O., Boucadair, M., and Q. WU, "A Layer 2/3 VPN Common YANG Model", draft-ietf-opsawg-vpn-common-02 (work in progress), October 2020.

[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>.

[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>.

[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>.

[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>.

[RFC8214] Boutros, S., Sajassi, A., Salam, S., Drake, J., and J. Rabadan, "Virtual Private Wire Service Support in Ethernet VPN", RFC 8214, DOI 10.17487/RFC8214, August 2017, <https://www.rfc-editor.org/info/rfc8214>.

[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>.

[RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October 2018, <https://www.rfc-editor.org/info/rfc8466>.

10.2. Informative References

[RFC3644] Snir, Y., Ramberg, Y., Strassner, J., Cohen, R., and B. Moore, "Policy Quality of Service (QoS) Information Model", RFC 3644, DOI 10.17487/RFC3644, November 2003, <https://www.rfc-editor.org/info/rfc3644>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for Abstraction and Control of TE Networks (ACTN)", RFC 8453, DOI 10.17487/RFC8453, August 2018, <https://www.rfc-editor.org/info/rfc8453>.

Authors' Addresses

Samier Barguil (editor)
Telefonica
Madrid
ES

Email: samier.barguilgiraldo.ext@telefonica.com

Oscar Gonzalez de Dios (editor)
Telefonica
Madrid
ES

Email: oscar.gonzalezdedios@telefonica.com

Mohamed Boucadair
Orange
France

Email: "mohamed.boucadair@orange.com

Luis Angel Munoz
Vodafone
ES

Email: luis-angel.munoz@vodafone.com

Luay Jalil
Verizon
USA

Email: luay.jalil@verizon.com

Jichun Ma
China Unicom
China

Email: majc16@chinaunicom.cn

OPSAWG S. Barguil
Internet-Draft O. Gonzalez de Dios, Ed.
Intended status: Standards Track Telefonica
Expires: April 19, 2021 M. Boucadair, Ed.
Orange
L. Munoz
Vodafone
A. Aguado
Nokia
October 16, 2020

A Layer 3 VPN Network YANG Model
draft-ietf-opsawg-l3sm-l3nm-05

Abstract

This document defines a L3VPN Network YANG Model (L3NM) that can be used to manage the provisioning of Layer 3 Virtual Private Network (VPN) services within a Service Provider's network. The model provides a network-centric view of L3VPN services.

L3NM is meant to be used by a Network Controller to derive the configuration information that will be sent to relevant network devices. The model can also facilitate the communication between a service orchestrator and a network controller/orchestrator.

Editorial Note (To be removed by RFC Editor)

Please update these statements within the document with the RFC number to be assigned to this document:

o "This version of this YANG module is part of RFC XXXX;"

o "RFC XXXX: Layer 3 VPN Network Model";

o reference: RFC XXXX

Also, please update the "revision" date of the YANG module.

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 19, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

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

This document defined a YANG module called L3VPN Network Model (L3NM). The L3NM is aimed at providing a network-centric view of Layer 3 (L3) VPN Services. This data model can be used to facilitate communication between the service orchestrator (or a network operator) and the network controller/orchestrator by allowing for more network-centric information to be included. It enables further capabilities, such as resource management or to serve as a multi-domain orchestration interface, where logical resources (such as route targets or route distinguishers) must be synchronized.

This document uses the common VPN YANG module defined in [I-D.ietf-opsawg-vpn-common].

This document does not obsolete, but uses, the definitions in [RFC8299]. These two modules are used for similar objectives but with different scopes and views.

The L3NM YANG module is initially built with a prune and extend approach, taking as a starting points the YANG module described in [RFC8299]. Nevertheless, this module is not defined as an augment to L3SM because a specific structure is required to meet network-oriented L3 needs.

Some of the information captured in the L3SM can be passed by the Orchestrator in the L3NM (e.g., customer) or be used to fed some of the L3NM attributes (e.g., actual forwarding policies). Some of the information captured in L3SM may be maintained locally within the Orchestrator; which is in charge of maintaining the correspondence between a Customer view and its network instantiation. Likewise, some of the information captured and exposed using L3NM can fed the service layer (e.g., capabilities) to derive L3SM and drive VPN service order handling.

The L3NM does not attempt to address all deployment cases especially those where the L3VPN connectivity is supported through the coordination of different VPNs in different underlying networks. More complex deployment scenarios involving the coordination of different VPN instances and different technologies to provide end-to-end VPN connectivity are addressed by a complementary YANG model defined in [I-D.evenwu-opsawg-yang-composed-vpn].

L3NM focuses on BGP PE-based Layer 3 VPNs as described in [RFC4026][RFC4110][RFC4364] and Multicast VPNs as described in [RFC6037][RFC6513][RFC7988].

The YANG data model in this document conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

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

This document assumes that the reader is familiar with the contents of [RFC6241], [RFC7950], [RFC8299], [RFC8309], and [RFC8453] and uses the terminology defined in those documents.

The meaning of the symbols in tree diagrams is defined in [RFC8340].

The document is aimed at modeling BGP PE-based VPNs in a service provider network, so the terms defined in [RFC4026] and [RFC4176] are used.

This document makes use of the following terms:

- Layer 3 VPN Customer Service Model (L3SM): A YANG module that describes the requirements of a L3VPN that interconnects a set of sites from the point of view of the customer. The customer service model does not provide details on the service provider network. The L3VPN Customer Service model is defined in [RFC8299].

- Layer 3 VPN Service Network Model (L3NM): A YANG module that describes a VPN Service in the service provider network. It contains information of the Service Provider network and might include allocated resources. It can be used by network controllers to manage and control the VPN Service configuration in the Service Provider network. The YANG module can be consumed by a Service Orchestrator to request a VPN Service to a Network controller.

- Service Orchestrator: A functional entity that interacts with the customer of a L3VPN. The Service Orchestrator interacts with the customer using L3SM. The Service Orchestrator is responsible of the Customer Edge (CE) - the Provider Edge (PE) attachment circuits, the PE selection, and requesting the VPN service to the network controller.

- Network Orchestrator: A functional entity that is hierarchically intermediate between Service Orchestrator and Network Controllers. A network orchestrator can manage one or several Network Controllers.

- Network Controller: A functional entity responsible for the control and management of the service provider network.

- VPN node: An abstraction that represents a set of policies applied on a PE and that belong to a single VPN service. A VPN service involves one or more VPN nodes. As it is an abstraction, the network controller will take on how to implement a VPN node. For example, typically, in a BGP-based VPN, a VPN node could be mapped into a Virtual Routing and Forwarding (VRF).

- VPN network access: An abstraction that represents the network interfaces that are associated to a given VPN node. Traffic coming from the VPN network access belongs to the VPN. The attachment circuits (bearers) between CEs and PEs are terminated in the VPN network access. A reference to the bearer is maintained to allow keeping the link between L3SM and L3NM.

- o VPN Site: A VPN customer’s location that is connected to the Service Provider network via a CE-PE link, which can access at least one VPN [RFC4176].

- o VPN Service Provider (SP): A Service Provider that offers VPN-related services [RFC4176].

- o Service Provider (SP) Network: A network that is able to provide VPN-related services.

4. L3NM Reference Architecture

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

Although some deployments may choose to construct a monolithic orchestration component (covering both service and network matters), this document advocates for a clear separation between service and network orchestration components for the sake of better flexibility. Such design adheres to the L3VPN reference architecture defined in Section 1.3 of [RFC4176]. The above separation relies upon a dedicated communication interface between these components and appropriate YANG module that reflect network-related information (that is hidden to customers).

The intelligence for translating customer-facing information into network-centric one is implementation specific.

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

```
                                     +----------------+
                                     |   Customer     |
                                     +----------------+
                     Customer Service Model  |
                            l3vpn-svc        |
                                     +----------------+
                                     |    Service     |
                                     | Orchestration  |
                                     +----------------+
                       L3NM Network Model    |
                          l3vpn-ntw          |
                                     +----------------+
                                     |   Network      |
                                     | Orchestration  |
                                     +----------------+
               Network Configuration Model   |
                                 +-----------+-----------+
                                 |                       |
                        +----------------+      +----------------+
                        |    Domain      |      |     Domain     |
                        | Orchestration  |      | Orchestration  |
                        +----------------+      +----------------+
             Device             |        |                   |
             Configuration      |        |                   |
             Model              |        |                   |
                       +---------+       |                   |
                       | Config  |       |                   |
                       | Manager |       |                   |
                       +---------+       |                   |
                                |        |                   |
                                | NETCONF/CLI..................
                                |        |                   |
                     +------------------------------------------------+
                                         Network
```

Figure 1: Reference Architecture

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

```
                  +----------------------------------+
                  | Customer                         |
                  | +-----------------------------+  |
                  | |             CNC             |  |
                  | +-----------------------------+  |
                  +----:-----------------------:-----+
                       :                       :
                       : L3SM                  : L3SM
                       :                       :
             +---------:---------+   +-------------------+
             | MDSC    :         |   |       MDSC        |
             | +---------------+ |   |     (parent)      |
             | |    Service    | |   +-------------------+
             | | Orchestration | |             :
             | +---------------+ |             : L3NM
             |         :         |             :
             |         : L3NM    |   +-------------------+
             |         :         |   |       MDSC        |
             | +---------------+ |   |      (child)      |
             | |    Network    | |   +-------------------+
             | | Orchestration | |             :
             | +---------------+ |             :
             +---------:---------+             :
                       :                       :
                       : Network Configuration :
                       :                       :
          +------------:-------+     +---------:------------+
          | Domain     :       |     |         : Domain     |
          | Controller :       |     |         : Controller |
          |       +---------+  |     |    +---------+       |
          |       |   PNC   |  |     |    |   PNC   |       |
          |       +---------+  |     |    +---------+       |
          +------------:-------+     +---------:------------+
                       :                       :
                       : Device Configuration  :
                       :                       :
                  +--------+              +--------+
                  | Device |              | Device |
                  +--------+              +--------+
```

Figure 2: L3SM and L3NM in the Context of ACTN

5. Relation with other YANG Models

The "ietf-vpn-common" module [I-D.ietf-opsawg-vpn-common] includes a set of identities, types, and groupings that are meant to be reused by VPN-related YANG modules independently of the layer (e.g., Layer 2, Layer 3) and the type of the module (e.g., network model, service model) including future revisions of existing models (e.g., [RFC8299] or [RFC8466]). The L3NM reuses these common types and grouping.

In order to avoid data duplication and to ease passing data between layers when required (service layer to network layer and vice versa), early versions of the L3NM reused many of the data nodes that are defined in [RFC8299]. Nevertheless, that approach was abandoned in favor of the "ietf-vpn-common" module because that design was interpreted as if the deployment of L3NM depends on L3SM, while this is not the case. For example, a Service Provider may decide to use the L3NM to build its L3VPN services without exposing the L3SM.

As discussed in Section 4, the L3NM YANG module is meant to manage L3VPN services within a Service Provider network. The module provides a network view of the service. Such view is only visible within the Service Provider and is not exposed outside (to customers, for example). The following discusses how L3NM interfaces with other YANG modules:

L3SM: L3NM is not a Customer Service Model.

The internal view of the service (L3NM) may be mapped to an external view which is visible to Customers : L3VPN Service YANG data Model (L3SM) [RFC8299].

Typically, the L3NM can be fed with inputs that are requested by Customers, typically, relying upon a L3SM template. Concretely, some parts of the L3SM module can be directly mapped into L3NM while other parts are generated as a function of the requested service and local guidelines. Some other parts are local to the Service Provider and do not map directly to L3SM.

Note that the use of L3NM within a Service Provider does not assume nor preclude exposing the VPN service via L3SM. This is deployment-specific. Nevertheless, the design of L3NM tries to align as much as possible with the features supported by the L3SM to ease grafting both L3NM and L3SM for the sake of highly automated VPN service provisioning and delivery.

Network Topology Modules: A L3VPN involves nodes that are part of a topology managed by the Service Provider Backbone network. Such topology can be represented as using the network topology module in [RFC8345].

Device Modules: L3NM is not a device model.

Once a global VPN service is captured by means of L3NM, the actual activation and provisioning of the VPN service will involve a variety of device modules to tweak the required functions for the delivery of the service. These functions are supported by the VPN nodes and can be managed using device YANG modules. A non-comprehensive list of such device YANG modules is provided below:

* Routing management [RFC8349].

* BGP [I-D.ietf-idr-bgp-model].

* PIM [I-D.ietf-pim-yang].

* NAT management [RFC8512].

* QoS management [I-D.ietf-rtgwg-qos-model].

* ACLs [RFC8519].

How L3NM is used to derive device-specific actions is implementation-specific.

6. Sample Uses of the L3NM Data Model

6.1. Enterprise Layer 3 VPN Services

Enterprise L3VPNs are one of the most demanded services for carriers, and therefore, L3NM can be useful to automate the tasks of provisioning and maintenance of these VPNs. Templates and batch processes can be built, and as a result many parameters are needed for the creation from scratch of a VPN that can be abstracted to the upper SDN layer and little manual intervention will be still required.

Also common addition/removal of sites of an existing customer VPN can benefit of using L3NM, by creation of workflows that either prune or add nodes as required from the network data model object.

6.2. Multi-Domain Resource Management

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

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

6.3. Management of Multicast Services

Multicast services over L3VPN can be implemented either using dual PIM MVPNs (also known as, Draft Rosen model) [RFC4364] or multiprotocol BGP (MBGP)-based MVPNs[RFC6513][RFC6514]. Both methods are supported and equally effective, but the main difference is that MBGP-based MVPN does not require multicast configuration on the service provider backbone. MBGP MVPNs employ the intra-autonomous system BGP control plane and PIM sparse mode as the data plane. The PIM state information is maintained between the PE routers using the same architecture that is used for unicast VPNs.

On the other hand, Draft Rosen has limitations such as reduced options for transport, control plane scalability, availability, operational inconsistency, and the need of maintaining state in the backbone. Because of this, MBGP MVPN is the architectural model that has been taken as the base for implementing multicast service on L3VPN. In this scenario, BGP auto discovery is used to discover MVPN PE members and the customer PIM signaling is sent across provider core through MP-BGP. The multicast traffic is transported on MPLS P2MP LSPs. All of the previous information is carried in the MCAST-VPN BGP NRLI.

7. Description of the L3NM YANG Module

The L3NM ('ietf-l3vpn-ntw') is defined to manage L3VPNs in a service provider network. In particular, the 'ietf-l3vpn-ntw' module can be used to create, modify, and retrieve L3VPN Services of a network.

7.1. Overall Structure of the Module

The 'ietf-l3vpn-ntw' module uses two main containers: 'vpn-services' and 'vpn-profiles' (see Figure 3).

The 'vpn-services' container maintains the set of VPN services managed within the service provider's network. 'vpn-service' is the data structure that abstracts a VPN service (Section 7.3).

The 'vpn-profiles' container is used by the provider to maintain a set of common VPN profiles that apply to one or several VPN services (Section 7.2).

```
module: ietf-l3vpn-ntw
  +--rw l3vpn-ntw
     +--rw vpn-profiles
     |  ...
     +--rw vpn-services
        +--rw vpn-service* [vpn-id]
           ...
```

Figure 3: Overall L3NM Tree Structure

7.2. VPN Profiles

The 'vpn-profiles' container (Figure 4) allows the network provider to define and maintain a set of common VPN profiles [I-D.ietf-opsawg-vpn-common] that apply to one or several VPN services. The exact definition of the profiles is local to each network provider.

This document does not make any assumption about the exact definition of these profiles. How such profiles are defined is deployment specific. The model only includes an identifier to these profiles to ease identifying local policies when building a VPN service. As shown in Figure 4, the following identifiers can be included:

- 'cloud-identifier': This identifier refers to a cloud service.

- 'encryption-profile-identifier': An encryption profile refers to a set of policies related to the encryption scheme(s) and setup that can be applied when building and offering a VPN service.

- 'qos-profile-identifier': A QoS profile refers to as set of policies such as classification, marking, and actions (e.g., [RFC3644]).

- 'bfd-profile-identifier': A Bidirectional Forwarding Detection (BFD) profile refers to a set of BFD [RFC5880] policies that can be invoked when building a VPN service.

- 'forwarding-profile-identifier': A forwarding profile refers to the policies that apply to the forwarding of packets conveyed within a VPN. Such policies may consist at applying Access Control Lists (ACLs).

o 'routing-profile-identifier': A routing profile refers to a set of routing policies that will be invoked (e.g., BGP policies).

```
        +--rw l3vpn-ntw
           +--rw vpn-profiles
           |  +--rw valid-provider-identifiers
           |     +--rw cloud-identifier* [id] {cloud-access}?
           |     |  +--rw id    string
           |     +--rw encryption-profile-identifier* [id]
           |     |  +--rw id    string
           |     +--rw qos-profile-identifier* [id]
           |     |  +--rw id    string
           |     +--rw bfd-profile-identifier* [id]
           |     |  +--rw id    string
           |     +--rw forwarding-profile-identifier* [id]
           |     |  +--rw id    string
           |     +--rw routing-profile-identifier* [id]
           |        +--rw id    string
           +--rw vpn-services
              ...
```

Figure 4: VPN Profiles Subtree Structure

7.3. Modeling a Layer 3 VPN Service

The 'vpn-service' is the data structure that abstracts a VPN service in the service provider network. Each 'vpn-service' is uniquely identified by an identifier: 'vpn-id'. Such 'vpn-id' is only meaningful locally within the Network controller.

In order to facilitate the identification of the service, 'customer-name' and 'description' attributes may be provided.

The main 'vpn-service' parameters are:

o 'status': Allows the control of the operative and administrative status of the service as a whole.

o 'vpn-id': Is an identifier that is used to uniquely identify the L3VPN Service within L3NM scope.

o 'l3sm-vpn-id': Refers to an identifier of L3SM service. This identifier allows to easily correlate the (network) service as built in the network with a service order.

o 'vpn-service-topology': Indicates the network topology for the service: Hub-Spoke, Any-to-Any, and Custom. The deployment on the network is defined by the correct usage of import and export profiles

o 'vpn-type': Indicate the VPN service signaling type.

o 'ie-profiles': Defines reusable import/export policies for the same 'vpn-service'. More details are provided in Section 7.3.2.

o 'underlay-transport': Describes the preference for the transport technology to carry the traffic of the VPN service (Section 7.3.3).

The 'vpn-node' is an abstraction that represents a set of policies applied to a network node and that belong to a single 'vpn-service'. A VPN service is typically built by adding instances of 'vpn-node' to the 'vpn-nodes' container.

A 'vpn-node' contains 'vpn-network-accesses', which are the interfaces attached to the VPN by which the customer traffic is received. Therefore, the customer sites are connected to the 'vpn-network-accesses'.

Note that, as this is a network data model, the information about customers sites is not required in the model. Such information is rather relevant in the L3SM.

```
+--rw l3vpn-ntw
   +--rw vpn-profiles
   |  ...
   +--rw vpn-services
      +--rw vpn-service* [vpn-id]
         +--rw status
         |  ...
         +--rw vpn-id                   vpn-common:vpn-id
         +--rw vpn-name?                string
         +--rw vpn-description?         string
         +--rw customer-name?           string
         +--rw l3sm-vpn-id?             vpn-common:vpn-id
         +--rw vpn-type?                identityref
         +--rw vpn-service-topology?    identityref
         +--rw ie-profiles
         |  ...
         +--rw underlay-transport
         |  ...
         +--rw vpn-nodes
            ...
```

Figure 5: VPN Services Subtree Structure

7.3.1. Service Status

The L3NM allows to track service status ('status') of a given VPN service (Figure 6). Both operational and administrative status are maintained together with a timestamp. For example, a service can be created but not put into effect.

'admin' and 'ops' status can be used as trigger to detect service anomalies. For example, a service that is declared at the service layer as active but still inactive at the network layer is an indication that network provision actions are needed to align the observed service with the expected service status.

```
+--rw l3vpn-ntw
   +--rw vpn-profiles
   |  ...
   +--rw vpn-services
      +--rw vpn-service* [vpn-id]
         +--rw status
         |  +--rw admin-status
         |  |  +--rw status?         identityref
         |  |  +--rw last-updated?   yang:date-and-time
         |  +--ro oper-status
         |     +--ro status?         identityref
         |     +--ro last-updated?   yang:date-and-time
         ...
```

Figure 6: VPN Service Status Subtree Structure

7.3.2. Concept of Import/Export Profiles

The import and export profiles construct contains a list with information related with route target and distinguishers (RTs and RDs), grouped and identified by 'ie-profile-id'. The identifier is then referenced in one or multiple 'vpn-nodes' so the controller can identify RTs and RDs to be configured for a given VRF. The subtree is shown in Figure 7.

```
     +--rw l3vpn-ntw
        +--rw vpn-profiles
        |  ...
        +--rw vpn-services
           +--rw vpn-service* [vpn-id]
              +--rw vpn-id                  vpn-common:vpn-id
              +  ...
              +--rw ie-profiles
              |  +--rw ie-profile* [ie-profile-id]
              |     +--rw ie-profile-id    string
              |     +--rw rd?              union
              |     +--rw vpn-targets
              |        +--rw vpn-target* [id]
              |        |  +--rw id                    int8
              |        |  +--rw route-targets* [route-target]
              |        |  |  +--rw route-target    rt-types:route-target
              |        |  +--rw route-target-type
              |        |          rt-types:route-target-type
              |        +--rw vpn-policies
              |           +--rw import-policy?   string
              |           +--rw export-policy?   string
              +--rw vpn-nodes
                 +--rw vpn-node* [ne-id]
                    +--rw ne-id                       string
                    ...
                    +--rw vpn-targets
                    |  +--rw vpn-target* [id]
                    |  |  +--rw id                    int8
                    |  |  +--rw route-targets* [route-target]
                    |  |  |  +--rw route-target    rt-types:route-target
                    |  |  +--rw route-target-type
                    |  |          rt-types:route-target-type
                    |  +--rw vpn-policies
                    |     +--rw import-policy?   string
                    |     +--rw export-policy?   string
                    ...
```

Figure 7: Subtree Structure of Import/Export Profiles

7.3.3. Underlay Transport

The model allows to indicate a preference for the underlay transport technology when activating a L3VPN service (Figure 8). This preference is especially useful in networks with multiple domains and NNI types. This version of the YANG module supports these options: BGP, LDP, GRE, SR, SR-TE, and RSVP-TE as underlay transport mechanisms. Other profiles can be defined in the future.

```
     +--rw l3vpn-ntw
        +--rw vpn-profiles
        |  ...
        +--rw vpn-services
           +--rw vpn-service* [vpn-id]
              +--rw vpn-id                  vpn-common:vpn-id
              +  ...
              +--rw underlay-transport
              |  +--rw type*   identityref
              +--rw vpn-nodes
                 +--rw vpn-node* [ne-id]
                 ...
```

Figure 8: Subtree Structure of the Underlying Transport

7.3.4. VPN Node

The 'vpn-node' is an abstraction that represents a set of common policies applied on a given network node (tipcally, a PE) and belong to one L3VPN service. In order to indicate the network nodes where the 'vpn-node' applies, the 'ne-id' must be indicated. The 'vpn-node' includes a parameter to indicate the network node on which it is applied. In the case that the 'ne-id' points to a specific PE, the 'vpn-node' will likely be mapped into a VRF in the node. However, the model also allows to point to an abstract node. In this case, the network controller will decide how to split the 'vpn-node' into VRFs. Some 'vpn-node' parameters are listed below:

o local-autonomous-system: Refers to the autonomous system number that is locally configured in the instance. It can be overwritten for specific purposes in the CE-PE BGP session.

o maximum-routes: Set the maximum number of prefixes allowed in the 'vpn-node' instance. This value is typically set in the service request.

o 'rd' and 'vpn-targets': For the cases the logical resources are managed outside the network controller, the model allows to explicitely indicate the logical resources such as Route targets (RTs) and Route Distinguishers (RDs) (RT,RD).

o Multicast: Enable multicast traffic inside the VPN. Refer to Section 7.3.4.3.

Under the VPN Node ('vpn-node') container, VPN Network Accesses ('vpn-network-access') can be created. The VPN Network Access represents the point to which sites are connected. Note that, unlike in L3SM, the L3NM does not need to model the customer site, only the points where the traffic from the site are received (i.e., the PE side of PE-CE connections). Hence, the VPN Network access contains the connectivity information between the provider's network and the customer premises. The VPN profiles ('vpn-profiles') have a set of routing policies than can be applied during the service creation.

The L3NM allows to track the status ('status') of the nodes involved in a VPN service. Both operational and administrative status are maintained. Mismatch between an administrative status vs. the operational status can be used as trigger to detect anomalies.

```
  +--rw l3vpn-ntw
     +--rw vpn-profiles
     |  ...
     +--rw vpn-services
        +--rw vpn-service* [vpn-id]
           +--rw vpn-id                  vpn-common:vpn-id
           ...
           +--rw vpn-nodes
              +--rw vpn-node* [ne-id]
                 +--rw vpn-node-id?                union
                 +--rw local-autonomous-system?   inet:as-number
                 +--rw description?                string
                 +--rw ne-id                       string
                 +--rw router-id?                  inet:ip-address
                 +--rw address-family?
                 |       vpn-common:address-family
                 +--rw node-role?                  identityref
                 +--rw rd?                         union
                 +--rw vpn-targets
                 |  +--rw vpn-target* [id]
                 |  |  +--rw id                    int8
                 |  |  +--rw route-targets* [route-target]
                 |  |  |  +--rw route-target    rt-types:route-target
                 |  |  +--rw route-target-type
                 |  |          rt-types:route-target-type
                 |  +--rw vpn-policies
                 |     +--rw import-policy?   string
                 |     +--rw export-policy?   string
                 +--rw status
                 |  +--rw admin-status
                 |  |  +--rw status?         identityref
                 |  |  +--rw last-updated?   yang:date-and-time
                 |  +--ro oper-status
                 |     +--ro status?         identityref
                 |     +--ro last-updated?   yang:date-and-time
                 +--rw node-ie-profile?           leafref
                 +--rw groups
```

```
           |  +--rw group* [group-id]
           |     +--rw group-id    string
           +--rw vpn-network-accesses
           |  +--rw vpn-network-access* [id]
           |     ...
           +--rw maximum-routes
           |  +--rw address-family* [af]
           |     +--rw af
           |     |       vpn-common:address-family
           |     +--rw maximum-routes?   uint32
           +--rw multicast {vpn-common:multicast}?
              ...
```

Figure 9: VPN Node Subtree Structure

7.3.4.1. RT/RD Assignment/auto-assignment

For the cases the logical resources are managed outside the network controller, the model allows to explicitely indicate the logical resources such as Route targets (RTs) and Route Distinguishers (RDs) (RT,RD).

Three possible behaviors are needed to fulfil the identified use cases:

- o The network controller auto-assigns logical resources (RTs, RDs). This can apply for new services deployment.
- o The Network Operator/Service orchestrator assigns explicitly the RTs and RDs. This case will fit with a brownfield scenario where some existing services needs to be updated by the network operators.
- o The Network Operator/Service orchestrator explicitly wants NO RT/ RD to be assigned. This case will fit in VRF-Lite scenarios, CE testing inside the Network or just for troubleshooting pruposes.

Thus a union between two yang data types are included in order to support this scenarios. So, if the leaf is not created in the Yang the expected behavior is the auto-assigns. If the Leaf is created with a valid rd value it will be explicitly assign in the VPN Node and if the leaf is created with an empty value, the RD value will not be deployed in the VPN node.

7.3.4.2. VPN Network Access

A 'vpn-network-access' represents an entry point to a VPN service (Figure 10). In other words, this container encloses the parameters that describe the access information for the traffic that belongs to a particular L3VPN. As such, every 'vpn-network-access' MUST belong to one and only one 'vpn-node'.

A 'vpn-network-access' includes information such as the connection on which the access is defined (see Section 7.3.4.2.1), the encapsulation of the traffic, policies that are applied on the access, etc.

Each 'vpn-network-access' SHOULD have a 'vpn-network-access-type' to select the type of network interface to be deployed in the devices. The available options are:

- Point-to-Point: The point-to-point type represent a direct connection between the end-points. It implies the controller must keep the association between a logical or physical interface on the device with the 'id' of the vpn-network-access.

- Multipoint: This option represents a broadcast connection between end-points. It implies the controller must keep the association between a logical or physical interface on the device with the 'id' of the 'vpn-network-access'.

- Pseudowire: Represent a connection coming from an L2VPN service. It implies the controller must keep the relationship between the logical tunnels or bridges on the devices with the 'id' of the' vpn-network-access'.

- Loopback: It represents the creation of a logical interface on the devices.

A PNC [RFC8453] will accept VPN requests containing this construct, using the enclosed data to: configure the router's interface to include the parameters described at the 'vpn-network-access', include the given interface into a VRF, configuring policies or schedulers for processing the incoming traffic, etc.

```
...
+--rw vpn-services
   +--rw vpn-service* [vpn-id]
      +--rw vpn-id                  vpn-common:vpn-id
      +  ...
       +--rw vpn-node* [ne-id]
          +--rw ne-id                     string
          + ...
          +--rw vpn-network-accesses
          |  +--rw vpn-network-access* [id]
          |     +--rw id
          |     |       vpn-common:vpn-id
          |     +--rw port-id?
          |     |       vpn-common:vpn-id
          |     +--rw description?              string
          |     +--rw status
          |     |  +--rw admin-enabled?   boolean
          |     |  +--ro oper-status?     operational-type
          |     +--rw vpn-network-access-type?   identityref
          |     +--rw connection
          |     |  ...
          |     +--rw ip-connection
          |     |  ...
          |     +--rw security
          |     |  ...
          |     +--rw routing-protocols
          |     |  ...
          |     +--rw service
          |        ...
          ...
```

Figure 10: VPN Network Access Tree Structure

7.3.4.2.1. Connection

The definition of a L3VPN is commonly specified not only at the IP layer, but also requires to identify parameters at the Ethernet layer, such as encapsulation type (e.g., VLAN, QinQ, QinAny, VxLAN, etc.). The 'connection' container represents and groups the set of Layer 2 connectivity from where the traffic of the L3VPN in a particular VPN Network access is coming.

Ethernet encapsulation description is not supported in [RFC8299]. However, this parameters are mandatory to configure the PE interfaces. Thus, in the L3NM, these parameters uses the connection container inside the 'vpn-network-access'. This container defines protocols and parameters to enable connectivity at Layer 2.

```
    ...
    +--rw vpn-services
       +--rw vpn-service* [vpn-id]
          +--rw vpn-id                 vpn-common:vpn-id
          +  ...
           +--rw vpn-node* [ne-id]
              +--rw ne-id                     string
              + ...
              +--rw vpn-network-accesses
              |  +--rw vpn-network-access* [id]
              |     +--rw id
              |     |       vpn-common:vpn-id
              |     ...
              |     +--rw connection
              |     |  +--rw encapsulation-type?   identityref
              |     |  +--rw logical-interface
              |     |  |  +--rw peer-reference?   uint32
              |     |  +--rw tagged-interface
              |     |  |  +--rw type?                identityref
              |     |  |  +--rw dot1q-vlan-tagged
              |     |  |  |       {vpn-common:dot1q}?
              |     |  |  |  +--rw tag-type?   identityref
              |     |  |  |  +--rw cvlan-id?   uint16
              |     |  |  +--rw priority-tagged
              |     |  |  |  +--rw tag-type?   identityref
              |     |  |  +--rw qinq {vpn-common:qinq}?
              |     |  |  |  +--rw tag-type?   identityref
              |     |  |  |  +--rw svlan-id    uint16
              |     |  |  |  +--rw cvlan-id    uint16
              |     |  |  +--rw qinany {vpn-common:qinany}?
              |     |  |  |  +--rw tag-type?   identityref
              |     |  |  |  +--rw svlan-id    uint16
              |     |  |  +--rw vxlan {vpn-common:vxlan}?
              |     |  |     +--rw vni-id       uint32
              |     |  |     +--rw peer-mode?   identityref
              |     |  |     +--rw peer-list* [peer-ip]
              |     |  |        +--rw peer-ip    inet:ip-address
              |     |  +--rw bearer
              |     |     ...
              ...
```

Figure 11: Encapsulation Subtree Structure

Additionally, the 'bearer-reference' and the pseudowire termination are supported (see Figure 12). A site, as per [RFC4176] represents a VPN customer's location that is connected to the Service Provider network via a CE-PE link, which can access at least one VPN. The connection from the site to the Service Provider network is the bearer. Every site is associated with a list of bearers. A bearer is the layer two connections with the site. In the module it is assumed that the bearer has been allocated by the Service Provider at the service orchestration step. The bearer is associated to a network element and a port. Hence, a bearer is just a bearer-reference to allow the translation between L3SM and L3NM.

```
...
+--rw vpn-network-accesses
|  +--rw vpn-network-access* [id]
|     +--rw id
|     |       vpn-common:vpn-id
|     ...
|     +--rw vpn-network-access-type?   identityref
|     +--rw connection
|     |  ...
|     |  +--rw bearer
|     |     +--rw bearer-reference?   string
|     |     |       {vpn-common:bearer-reference}?
|     |     +--rw pseudowire
|     |     |  +--rw vcid?      uint32
|     |     |  +--rw far-end?   union
|     |     +--rw vpls
|     |        +--rw vcid?      union
|     |        +--rw far-end?   union
|  ...
```

Figure 12: Bearer Subtree Structure

7.3.4.2.2. IP Connections

IP connection container (Figure 13) has the parameters of the 'vpn-network-access' addressing information. The address allocated in this container would represent the PE interface address configuration. The IP connection container is designed to support both IPv4 and IPv6. It also supports three IP address assignment modes: SLAAC [RFC7527], Provider DHCP, DHCP relay, and static addressing. Only one of them is enabled for a given service.

```
...
+--rw vpn-network-accesses
|  +--rw vpn-network-access* [id]
|     +--rw id
|     |       vpn-common:vpn-id
|     ...
|     +--rw vpn-network-access-type?   identityref
|     +--rw connection
|     |  ...
```

```
     |     +--rw ip-connection
     |     |  +--rw ipv4 {vpn-common:ipv4}?
     |     |  |  +--rw address-allocation-type?
     |     |  |  |       identityref
     |     |  |  +--rw (allocation-type)?
     |     |  |     +--:(provider-dhcp)
     |     |  |     |  +--rw provider-address?
     |     |  |     |  |       inet:ipv4-address
     |     |  |     |  +--rw prefix-length?
     |     |  |     |  |       uint8
     |     |  |     |  +--rw (address-assign)?
     |     |  |     |     +--:(number)
     |     |  |     |     |  +--rw number-of-dynamic-address?
     |     |  |     |     |          uint16
     |     |  |     |     +--:(explicit)
     |     |  |     |        +--rw customer-addresses
     |     |  |     |           +--rw address-group*
     |     |  |     |                   [group-id]
     |     |  |     |              +--rw group-id
     |     |  |     |              |       string
     |     |  |     |              +--rw start-address?
     |     |  |     |              |       inet:ipv4-address
     |     |  |     |              +--rw end-address?
     |     |  |     |                      inet:ipv4-address
     |     |  |     +--:(dhcp-relay)
     |     |  |     |  +--rw dr-provider-address?
     |     |  |     |  |       inet:ipv4-address
     |     |  |     |  +--rw dr-prefix-length?
     |     |  |     |  |       uint8
     |     |  |     |  +--rw customer-dhcp-servers
     |     |  |     |     +--rw server-ip-address*
     |     |  |     |             inet:ipv4-address
     |     |  |     +--:(static-addresses)
     |     |  |        ...
     |     |  +--rw ipv6 {vpn-common:ipv6}?
     |     |  |  +--rw address-allocation-type?
     |     |  |  |       identityref
     |     |  |  +--rw (allocation-type)?
     |     |  |     +--:(provider-dhcp)
     |     |  |     |  +--rw (provider-dhcp)?
     |     |  |     |     +--:(provider-address)
     |     |  |     |     |  +--rw provider-address?
     |     |  |     |     |          inet:ipv6-address
     |     |  |     |     +--:(prefix-length)
     |     |  |     |     |  +--rw prefix-length?
     |     |  |     |     |          uint8
     |     |  |     |     +--:(address-assign)
     |     |  |     |        +--rw (address-assign)?
```

```
     |        |   |      |           +--:(number)
     |        |   |      |           |  +--rw number-of-dynamic-address?
     |        |   |      |           |          uint16
     |        |   |      |           +--:(explicit)
     |        |   |      |              +--rw customer-addresses
     |        |   |      |                 +--rw address-group*
     |        |   |      |                         [group-id]
     |        |   |      |                    +--rw group-id
     |        |   |      |                    |       string
     |        |   |      |                    +--rw start-address?
     |        |   |      |                    |       inet:ipv6-address
     |        |   |      |                    +--rw end-address?
     |        |   |      |                            inet:ipv6-address
     |        |   |      +--:(dhcp-relay)
     |        |   |      |  +--rw dr-provider-address?
     |        |   |      |  |       inet:ipv6-address
     |        |   |      |  +--rw dr-prefix-length?
     |        |   |      |  |       uint8
     |        |   |      |  +--rw customer-dhcp-servers
     |        |   |      |     +--rw server-ip-address*
     |        |   |      |             inet:ipv6-address
     |        |   |      +--:(static-addresses)
     |        |   |         ...
```

Figure 13: IP Connection Subtree Structure

In the case of the static addressing (Figure 14), the model supports the assignment of several IP addresses in the same 'vpn-network-access'. To identify which of the addresses is the primary address of a connection ,the 'primary-address' reference MUST be set with the corresponding 'address-id'.

```
     ...
     +--rw vpn-network-accesses
        |  +--rw vpn-network-access* [id]
        |     +--rw id
        |     |       vpn-common:vpn-id
        |     ...
        |     +--rw vpn-network-access-type?   identityref
        |     +--rw connection
        |     |  ...
        |     +--rw ip-connection
        |     |  +--rw ipv4 {vpn-common:ipv4}?
        |     |  |  +--rw address-allocation-type?
        |     |  |  |       identityref
        |     |  |  +--rw (allocation-type)?
        |     |  |     ...
        |     |  |     +--:(static-addresses)
        |     |  |        +--rw primary-address?
        |     |  |        |       -> ../address/address-id
        |     |  |        +--rw address* [address-id]
        |     |  |           +--rw address-id
        |     |  |           |       string
        |     |  |           +--rw s-provider-address?
        |     |  |           |       inet:ipv4-address
        |     |  |           +--rw s-customer-address?
        |     |  |           |       inet:ipv4-address
        |     |  |           +--rw s-prefix-length?
        |     |  |                   uint8
        |     |  +--rw ipv6 {vpn-common:ipv6}?
        |     |  |  +--rw address-allocation-type?
        |     |  |  |       identityref
        |     |  |  +--rw (allocation-type)?
        |     |  |     ...
        |     |  |     +--:(static-addresses)
        |     |  |        +--rw s-primary-address?
        |     |  |        |       -> ../s-address/address-id
        |     |  |        +--rw s-address* [address-id]
        |     |  |           +--rw address-id
        |     |  |           |       string
        |     |  |           +--rw provider-address?
        |     |  |           |       inet:ipv6-address
        |     |  |           +--rw customer-address?
        |     |  |           |       inet:ipv6-address
        |     |  |           +--rw prefix-length?      uint8
        ...
```

Figure 14: IP Connection Subtree Structure: Static Mode

7.3.4.2.3. Security

The 'security' container specifies the authentication and the encryption to be applied for a given VPN network access (Figure 15).

```
              ...
              +--rw vpn-network-accesses
              |  +--rw vpn-network-access* [id]
              |     +--rw id
              |     |       vpn-common:vpn-id
              |     +  ...
              |     +--rw connection
              |     |  ...
              |     +--rw ip-connection
              |     |  ...
              |     +--rw security
              |     |  +--rw encryption {vpn-common:encryption}?
              |     |  |  +--rw enabled?   boolean
              |     |  |  +--rw layer?     enumeration
              |     |  +--rw encryption-profile
              |     |     +--rw (profile)?
              |     |     |  +--:(provider-profile)
              |     |     |  |  +--rw profile-name?    leafref
              |     |     |  +--:(customer-profile)
              |     |     |     +--rw algorithm?       string
              |     |     +--rw (key-type)?
              |     |        +--:(psk)
              |     |           +--rw preshared-key?   string
              |     +--rw routing-protocols
              |     |  ...
              |     +--rw service
              |        ...
              |  ...
```

Figure 15: Security Subtree Structure

7.3.4.2.4. CE-PE Routing Protocols

The model allows the Provider to configure one or more routing protocols associated with a particular 'vpn-network-access' (Figure 16). This protocol will run between the PE and the CE. A routing protocol instance MUST have a type (e.g., bgp, ospf) and an identifier. The identifier is necessary when multiple instances of the same protocol have to be configured.

```
                    ...
                    +--rw vpn-network-accesses
                    |  +--rw vpn-network-access* [id]
                    |     +--rw id
                    |     |       vpn-common:vpn-id
                    |     ...
                    |     +--rw ip-connection
                    |     |  ...
                    |     +--rw routing-protocols
                    |     |  +--rw routing-protocol* [id]
                    |     |     +--rw id                  string
                    |     |     +--rw type?               identityref
                    |     |     +--rw routing-profiles* [id]
                    |     |     |  +--rw id      leafref
                    |     |     |  +--rw type?   identityref
                    |     |     +--rw ospf {vpn-common:rtg-ospf}?
                    |     |     |  ...
                    |     |     +--rw bgp {vpn-common:rtg-bgp}?
                    |     |     |  ...
                    |     |     +--rw isis {vpn-common:rtg-isis}?
                    |     |     |  ...
                    |     |     +--rw static
                    |     |     |  ...
                    |     |     +--rw rip {vpn-common:rtg-rip}?
                    |     |     |  +--rw address-family*
                    |     |     |          vpn-common:address-family
                    |     |     +--rw vrrp {vpn-common:rtg-vrrp}?
                    |     |        +--rw address-family*
                    |     |                vpn-common:address-family
                    |     +--rw service
                    |        ...
                    ...
```

Figure 16: Routing Subtree Structure

Routing configuration does not include low-level policies. These policies are low level device configurations that must not be part of an abstracted model. A provider's internal policies (such as security filters) will be implemented as part of the device configuration but does not require any input from this model. Some policies like primary/backup or load-balancing can be inferred from 'access-priority'.

When configuring multiple instances of the same routing protocol, this does not automatically imply that, from a device configuration perspective, there will be parallel instances (multiple processes) running. It will be up to the implementation to use the most appropriate deployment model. As an example, when multiple BGP peers need to be implemented, multiple instances of BGP must be configured as part of this model. However, from a device configuration point of view, this could be implemented as:

- o Multiple BGP processes with a single neighbor running in each process.

- o A single BGP process with multiple neighbors running.

- o A combination of both.

To be aligned with [RFC8299], this model supports the following CE-PE routing protocols:

- o OSPF: The model (Figure 17) allows the user to configure OSPF to run as routing protocol on the 'vpn-network-access interface'. An OSPF instance can be bound to IPv4, IPv6 or both. When only IPv4 address-family is requested, it will be up to the implementation to drive whether OSPFv2 or OSPFv3 is used.

```
                 ...
                 +--rw vpn-network-accesses
                 |  +--rw vpn-network-access* [id]
                 |     +--rw id
                 |     |       vpn-common:vpn-id
                 |     ...
                 |     +--rw ip-connection
                 |     |  ...
                 |     +--rw routing-protocols
                 |     |  +--rw routing-protocol* [id]
                 |     |     +--rw id                  string
                 |     |     +--rw type?               identityref
                 |     |     +--rw routing-profiles* [id]
                 |     |     |  +--rw id      leafref
                 |     |     |  +--rw type?   identityref
                 |     |     +--rw ospf {vpn-common:rtg-ospf}?
                 |     |     |  +--rw address-family*
                 |     |     |  |       vpn-common:address-family
                 |     |     |  +--rw area-address
                 |     |     |  |       yang:dotted-quad
                 |     |     |  +--rw metric?           uint16
                 |     |     |  +--rw mtu?              uint16
                 |     |     |  +--rw process-id?       uint16
                 |     |     |  +--rw security
                 |     |     |  |  +--rw auth-key?   string
                 |     |     |  +--rw sham-links
                 |     |     |          {vpn-common:rtg-ospf-sham-link}?
                 |     |     |     +--rw sham-link* [target-site]
                 |     |     |        +--rw target-site
                 |     |     |        |       vpn-common:vpn-id
                 |     |     |        +--rw metric?        uint16
                 |     |     +--rw bgp {vpn-common:rtg-bgp}?
                 |     |     |  ...
                 |     |     +--rw isis {vpn-common:rtg-isis}?
                 |     |     |  ...
                 |     |     +--rw static
                 |     |     |  ...
                 |     |     +--rw rip {vpn-common:rtg-rip}?
                 |     |     |  ...
                 |     |     +--rw vrrp {vpn-common:rtg-vrrp}?
                 |     |        ...
                 |     +--rw service
                 |        ...
                 ...
```

Figure 17: OPSF Routing Subtree Structure

o BGP: The model (Figure 18) allows to configure a BGP neighbor, including a set for parameters that are pertinent to be tweaked at the network level for service customization purposes. This container does not aim to include every BGP parameter; a comprehensive set of parameters belongs more to the BGP device model. The following parameters are captured in Figure 18. It is up to the implementation to drive the corresponding BGP device configuration.

  * 'peer-autonomous-system': This parameter conveys the Customer's AS Number (ASN).

  * 'local-autonomous-system': This parameter is set of AS override is activated for this peer.

  * 'address-family': This attribute indicates the address-family of the peer. It can be set to IPv4, IPv6, or both address-families.

  * 'neighbor': The module supports supplying two neighbors (each for a given address-family) or one neighbor (if 'address-family' attribute is set to both IPv4 and IPv6 address-families). A list of IP address(es) of the BGP neighbor can be then conveyed in this parameter.

  * 'multihop': This attribute indicates the number of allowed IP hops between a BGP peer and a PE.

  * 'security': The authentication type will be driven by the implementation but the module supports any authentication that uses a key as a parameter.

  * 'as-override': If set, this parameter indicates whether AS override is enabled, i.e., replace the ASN of the peer specified in the AS Path attribute with the ASN identified by the 'local-autonomous-system' attribute.

  * 'default-route': This attribute controls whether default route(s) can be advertised to the peer.

  * 'bgp-max-prefix': This attribute is used to control how many prefixes can be received from a neighbor. If reached, the BGP session will be teared down.

  * 'bgp-timer': Two timers can be captured in this container: (1) 'hold-time' which is the time interval that will be used for the HoldTimer (Section 4.2 of [RFC4271]) when establishing a BGP session. (2) 'keep-alive' which is the time interval for

the KeepAlive timer between a PE and a BGP peer (Section 4.4 of [RFC4271]).

```
...
+--rw vpn-network-accesses
|  +--rw vpn-network-access* [id]
|     +--rw id
|     |       vpn-common:vpn-id
|     ...
|     +--rw ip-connection
|     |  ...
|     +--rw routing-protocols
|     |  +--rw routing-protocol* [id]
|     |     +--rw id                   string
|     |     +--rw type?                identityref
|     |     +--rw routing-profiles* [id]
|     |     |  +--rw id      leafref
|     |     |  +--rw type?   identityref
|     |     +--rw ospf {vpn-common:rtg-ospf}?
|     |     |  ...
|     |     +--rw bgp {vpn-common:rtg-bgp}?
|     |     |  +--rw peer-autonomous-system
|     |     |  |          inet:as-number
|     |     |  +--rw local-autonomous-system?
|     |     |  |       inet:as-number
|     |     |  +--rw address-family*
|     |     |  |      vpn-common:address-family
|     |     |  +--rw neighbor*
|     |     |  |       inet:ip-address
|     |     |  +--rw multihop?            uint8
|     |     |  +--rw security
|     |     |  |  +--rw auth-key?   string
|     |     |  +--rw status
|     |     |  |  +--rw admin-status
|     |     |  |  |  +--rw status?       identityref
|     |     |  |  |  +--rw last-updated?
|     |     |  |  |  |            yang:date-and-time
|     |     |  |  +--ro oper-status
|     |     |  |     +--ro status?       identityref
|     |     |  |     +--ro last-updated?
|     |     |  |                  yang:date-and-time
|     |     |  +--rw description?             string
|     |     |  +--rw as-override?            boolean
|     |     |  +--rw default-route?          boolean
|     |     |  +--rw bgp-max-prefix
|     |     |  |  +--rw max-prefix?           uint32
|     |     |  |  +--rw warning-threshold? decimal64
|     |     |  |  +--rw violate-action?  enumeration
```

```
           |     |     |  |  +--rw restart-interval?     uint16
           |     |     |  +--rw bgp-timer
           |     |     |     +--rw keep-alive?   uint16
           |     |     |     +--rw hold-time?    uint16
           |     |     +--rw isis {vpn-common:rtg-isis}?
           |     |     |  ...
           |     |     +--rw static
           |     |     |  ...
           |     |     +--rw rip {vpn-common:rtg-rip}?
           |     |     |  ...
           |     |     +--rw vrrp {vpn-common:rtg-vrrp}?
           |     |        ...
           |     +--rw service
           |        ...
           ...
```

Figure 18: BGP Routing Subtree Structure

o IS-IS: The model (Figure 19) allows the user to configure IS-IS to run on the 'vpn-network-access' interface. An IS-IS instance can run L1, L2, or both levels.

```
  ...
  +--rw vpn-network-accesses
  |  +--rw vpn-network-access* [id]
  |     +--rw id
  |     |       vpn-common:vpn-id
  |     ...
  |     +--rw ip-connection
  |     |  ...
  |     +--rw routing-protocols
  |     |  +--rw routing-protocol* [id]
  |     |     +--rw id                  string
  |     |     +--rw type?
  |     |     |       identityref
  |     |     +--rw routing-profiles* [id]
  |     |     |  +--rw id      leafref
  |     |     |  +--rw type?   identityref
  |     |     +--rw ospf {vpn-common:rtg-ospf}?
  |     |     |  ...
  |     |     +--rw bgp {vpn-common:rtg-bgp}?
  |     |     |  ...
  |     |     +--rw isis {vpn-common:rtg-isis}?
  |     |     |  +--rw address-family*
  |     |     |  |       vpn-common:address-family
  |     |     |  +--rw area-address
  |     |     |  |       yang:dotted-quad
  |     |     |  +--rw level?            identityref
```

```
                     |     |     |  +--rw metric?           uint16
                     |     |     |  +--rw process-id?       uint16
                     |     |     |  +--rw mode?             enumeration
                     |     |     |  +--rw status
                     |     |     |     +--rw admin-status
                     |     |     |     |  +--rw status?
                     |     |     |     |  |       identityref
                     |     |     |     |  +--rw last-updated?
                     |     |     |     |          yang:date-and-time
                     |     |     |     +--ro oper-status
                     |     |     |        +--ro status?
                     |     |     |        |       identityref
                     |     |     |        +--ro last-updated?
                     |     |     |                yang:date-and-time
                     |     |     +--rw static
                     |     |     |  ...
                     |     |     +--rw rip {vpn-common:rtg-rip}?
                     |     |     |  ...
                     |     |     +--rw vrrp {vpn-common:rtg-vrrp}?
                     |     |        ...
                     |     +--rw service
                     |        ...
                     ...
```

Figure 19: IS-IS Routing Subtree Structure

o RIP: The module covers only a list of address-family as parameter.

o VRRP: The module covers only a list of address-family as parameter.

The module allows a user to configure one or more IPv4 and/or IPv6 static routes as depicted in Figure 20.

```
   ...
   +--rw vpn-network-accesses
   |  +--rw vpn-network-access* [id]
   |     +--rw id
   |     |       vpn-common:vpn-id
   |     ...
   |     +--rw ip-connection
   |     |  ...
   |     +--rw routing-protocols
   |     |  +--rw routing-protocol* [id]
   |     |     +--rw id              string
   |     |     +--rw type?      identityref
   |     |     +--rw routing-profiles* [id]
   |     |     |  +--rw id      leafref
   |     |     |  +--rw type?   identityref
   |     |     +--rw ospf {vpn-common:rtg-ospf}?
   |     |     |  ...
   |     |     +--rw bgp {vpn-common:rtg-bgp}?
   |     |     |  ...
   |     |     +--rw isis {vpn-common:rtg-isis}?
   |     |     |  ...
   |     |     +--rw static
   |     |     |  +--rw cascaded-lan-prefixes
   |     |     |     +--rw ipv4-lan-prefixes*
   |     |     |     |       [lan next-hop]
   |     |     |     |       {vpn-common:ipv4}?
   |     |     |     |  +--rw lan
   |     |     |     |  |       inet:ipv4-prefix
   |     |     |     |  +--rw lan-tag?    string
   |     |     |     |  +--rw next-hop
   |     |     |     |          inet:ipv4-address
   |     |     |     +--rw ipv6-lan-prefixes*
   |     |     |             [lan next-hop]
   |     |     |             {vpn-common:ipv6}?
   |     |     |        +--rw lan
   |     |     |        |       inet:ipv6-prefix
   |     |     |        +--rw lan-tag?    string
   |     |     |        +--rw next-hop
   |     |     |                inet:ipv6-address
   |     |     +--rw rip {vpn-common:rtg-rip}?
   |     |     |  ...
   |     |     +--rw vrrp {vpn-common:rtg-vrrp}?
   |     |        ...
   |     +--rw service
   |        ...
   ...
```

Figure 20: Static Routing Subtree Structure

7.3.4.2.5. Services

The 'services' container specifies the service parameters to apply for a given VPN network access (Figure 21).

```
      ...
      +--rw vpn-network-accesses
      |  +--rw vpn-network-access* [id]
      |     +--rw id
      |     |       vpn-common:vpn-id
      |     ...
      |     +--rw service
      |        +--rw svc-input-bandwidth     uint64
      |        +--rw svc-output-bandwidth    uint64
      |        +--rw svc-mtu                 uint16
      |        +--rw qos {vpn-common:qos}?
      |        |  +--rw qos-classification-policy
      |        |  |  +--rw rule* [id]
      |        |  |     +--rw id
      |        |  |     |       string
      |        |  |     +--rw (match-type)?
      |        |  |     |  +--:(match-flow)
      |        |  |     |  |  +--rw (l3)?
      |        |  |     |  |  |  +--:(ipv4)
      |        |  |     |  |  |  |  ...
      |        |  |     |  |  |  +--:(ipv6)
      |        |  |     |  |  |     ...
      |        |  |     |  |  +--rw (l4)?
      |        |  |     |  |     +--:(tcp)
      |        |  |     |  |     |  ...
      |        |  |     |  |     +--:(udp)
      |        |  |     |  |        ...
      |        |  |     |  +--:(match-application)
      |        |  |     |     +--rw match-application?
      |        |  |     |             identityref
      |        |  |     +--rw target-class-id?
      |        |  |             string
      |        |  +--rw qos-profile
      |        |     +--rw qos-profile* [profile]
      |        |        +--rw profile      leafref
      |        |        +--rw direction?   identityref
      |        +--rw carrierscarrier
      |        |       {vpn-common:carrierscarrier}?
      |        |  +--rw signalling-type?   enumeration
      |        +--rw multicast {vpn-common:multicast}?
      |           +--rw site-type?        enumeration
      |           +--rw address-family?
      |           |       vpn-common:address-family
      |           +--rw protocol-type?    enumeration
      |           +--rw remote-source?    boolean
      ...
```

Figure 21: Services Subtree Structure

The following attributes are defined:

o 'svc-input-bandwidth': Indicates the inbound bandwidth of the connection (i.e., download bandwidth from the SP to the site).

o 'svc-output-bandwidth': Indicates the outbound bandwidth of the connection (i.e., upload bandwidth from the site to the SP).

o 'svc-mtu': Indicates the MTU at service level. It can be the IP MTU or MPLS MTU, for example.

o 'carrierscarrier': Groups a set of parameters that are used when CsC is enabled such the use of BGP for signalling purposes [RFC8277].

o 'multicast': Specifies the multicast mode and other service-related attributes such as the address-family.

o 'qos': Is used to define QoS policies to apply on a given connection. Classification can be based on many criteria such as:

  * Layer 3: As shown in Figure 23, the model allow to classify based on any IP header field or a combination thereof. Both IPv4 and IPv6 are supported.

```
 +--rw qos {vpn-common:qos}?
 |  +--rw qos-classification-policy
 |  |  +--rw rule* [id]
 |  |     +--rw id
 |  |     |       string
 |  |     +--rw (match-type)?
 |  |     |  +--:(match-flow)
 |  |     |  |  +--rw (l3)?
 |  |     |  |  |  +--:(ipv4)
 |  |     |  |  |  |  ...
 |  |     |  |  |  +--:(ipv6)
 |  |     |  |  |     ...
 |  |     |  |  +--rw (l4)?
 |  |     |  |  +--rw (l3)?
 |  |     |  |  |  +--:(ipv4)
 |  |     |  |  |  |  +--rw ipv4
 |  |     |  |  |  |     +--rw dscp?
 |  |     |  |  |  |     |       inet:dscp
 |  |     |  |  |  |     +--rw ecn?
 |  |     |  |  |  |     |       uint8
 |  |     |  |  |  |     +--rw length?
 |  |     |  |  |  |     |       uint16
 |  |     |  |  |  |     +--rw ttl?
```

```
       |  |     |  |  |  |     |       uint8
       |  |     |  |  |  |     +--rw protocol?
       |  |     |  |  |  |     |       uint8
       |  |     |  |  |  |     +--rw ihl?
       |  |     |  |  |  |     |       uint8
       |  |     |  |  |  |     +--rw flags?
       |  |     |  |  |  |     |       bits
       |  |     |  |  |  |     +--rw offset?
       |  |     |  |  |  |     |       uint16
       |  |     |  |  |  |     +--rw identification?
       |  |     |  |  |  |     |       uint16
       |  |     |  |  |  |     +--rw (destination-network)?
       |  |     |  |  |  |     |  +--:(destination-ipv4-network)
       |  |     |  |  |  |     |     +--rw destination-ipv4-network?
       |  |     |  |  |  |     |             inet:ipv4-prefix
       |  |     |  |  |  |     +--rw (source-network)?
       |  |     |  |  |  |        +--:(source-ipv4-network)
       |  |     |  |  |  |           +--rw source-ipv4-network?
       |  |     |  |  |  |  inet:ipv4-prefix
       |  |     |  |  |  +--:(ipv6)
       |  |     |  |  |     +--rw ipv6
       |  |     |  |  |        +--rw dscp?
       |  |     |  |  |        |       inet:dscp
       |  |     |  |  |        +--rw ecn?
       |  |     |  |  |        |       uint8
       |  |     |  |  |        +--rw length?
       |  |     |  |  |        |       uint16
       |  |     |  |  |        +--rw ttl?
       |  |     |  |  |        |       uint8
       |  |     |  |  |        +--rw protocol?
       |  |     |  |  |        |       uint8
       |  |     |  |  |        +--rw (destination-network)?
       |  |     |  |  |        |  +--:(destination-ipv6-network)
       |  |     |  |  |        |     +--rw destination-ipv6-network?
       |  |     |  |  |        |             inet:ipv6-prefix
       |  |     |  |  |        +--rw (source-network)?
       |  |     |  |  |        |  +--:(source-ipv6-network)
       |  |     |  |  |        |     +--rw source-ipv6-network?
       |  |     |  |  |        |             inet:ipv6-prefix
       |  |     |  |  |        +--rw flow-label?
       |  |     |  |  |                   inet:ipv6-flow-label
       |  |     |  |  +--rw (l4)?
       |  |     |  |     +--:(tcp)
       |  |     |  |     |  ...
       |  |     |  |     +--:(udp)
       |  |     |  |        ...
       ...
```

                  Figure 22: QoS Subtree Structure (L3)

   * Layer 4: As shown in Figure 23, TCP or UDP-related match
      crietria can be specified.

```
+--rw qos {vpn-common:qos}?
|  +--rw qos-classification-policy
|  |  +--rw rule* [id]
|  |     +--rw id
|  |     |       string
|  |     +--rw (match-type)?
|  |     |  +--:(match-flow)
|  |     |  |  +--rw (l3)?
|  |     |  |  |  +--:(ipv4)
|  |     |  |  |  |  ...
|  |     |  |  |  +--:(ipv6)
|  |     |  |  |     ...
|  |     |  |  +--rw (l4)?
|  |     |  |     +--:(tcp)
|  |     |  |     |  +--rw tcp
|  |     |  |     |     +--rw sequence-number?
|  |     |  |     |     |       uint32
|  |     |  |     |     +--rw acknowledgement-number?
|  |     |  |     |     |       uint32
|  |     |  |     |     +--rw data-offset?
|  |     |  |     |     |       uint8
|  |     |  |     |     +--rw reserved?
|  |     |  |     |     |       uint8
|  |     |  |     |     +--rw flags?
|  |     |  |     |     |       bits
|  |     |  |     |     +--rw window-size?
|  |     |  |     |     |       uint16
|  |     |  |     |     +--rw urgent-pointer?
|  |     |  |     |     |       uint16
|  |     |  |     |     +--rw options?
|  |     |  |     |     |       binary
|  |     |  |     |     +--rw (source-port)?
|  |     |  |     |     |  +--:(source-port-range-or-operator)
|  |     |  |     |     |     +--rw source-port-range-or-operator
|  |     |  |     |     |        +--rw (port-range-or-operator)?
|  |     |  |     |     |           +--:(range)
|  |     |  |     |     |           |  +--rw lower-port
|  |     |  |     |     |           |  |       inet:port-number
|  |     |  |     |     |           |  +--rw upper-port
|  |     |  |     |     |           |          inet:port-number
|  |     |  |     |     |           +--:(operator)
|  |     |  |     |     |              +--rw operator?
|  |     |  |     |     |              |       operator
```

```
   |  |     |  |     |     |              +--rw port
   |  |     |  |     |     |                      inet:port-number
   |  |     |  |     |     +--rw (destination-port)?
   |  |     |  |     +--:(destination-port-range-or-operator)
   |  |     |  |     |          +--rw destination-port-range-or-operator
   |  |     |  |     |             +--rw (port-range-or-operator)?
   |  |     |  |     |                +--:(range)
   |  |     |  |     |                |  +--rw lower-port
   |  |     |  |     |                |  |       inet:port-number
   |  |     |  |     |                |  +--rw upper-port
   |  |     |  |     |                |          inet:port-number
   |  |     |  |     |                +--:(operator)
   |  |     |  |     |                   +--rw operator?
   |  |     |  |     |                   |       operator
   |  |     |  |     |                   +--rw port
   |  |     |  |     |                           inet:port-number
   |  |     |  |     +--:(udp)
   |  |     |  |        +--rw udp
   |  |     |  |           +--rw length?
   |  |     |  |           |       uint16
   |  |     |  |           +--rw (source-port)?
   |  |     |  |           |  +--:(source-port-range-or-operator)
   |  |     |  |           |     +--rw source-port-range-or-operator
   |  |     |  |           |        +--rw (port-range-or-operator)?
   |  |     |  |           |           +--:(range)
   |  |     |  |           |           |  +--rw lower-port
   |  |     |  |           |           |  |       inet:port-number
   |  |     |  |           |           |  +--rw upper-port
   |  |     |  |           |           |          inet:port-number
   |  |     |  |           |           +--:(operator)
   |  |     |  |           |              +--rw operator?
   |  |     |  |           |              |       operator
   |  |     |  |           |              +--rw port
   |  |     |  |           |                      inet:port-number
   |  |     |  |           +--rw (destination-port)?
   |  |     |  |              +--:(destination-port-range-or-operator)
   |  |     |  |                +--rw destination-port-range-or-operator
   |  |     |  |                    +--rw (port-range-or-operator)?
   |  |     |  |                       +--:(range)
   |  |     |  |                       |  +--rw lower-port
   |  |     |  |                       |  |       inet:port-number
   |  |     |  |                       |  +--rw upper-port
   |  |     |  |                       |          inet:port-number
   |  |     |  |                       +--:(operator)
   |  |     |  |                          +--rw operator?
   |  |     |  |                          |       operator
   |  |     |  |                          +--rw port
   |  |     |  |                                  inet:port-number
```

```
                  ...
```

Figure 23: QoS Subtree Structure (L4)

* Application match

7.3.4.3. Multicast

Multicast MAY be enabled for a particular vpn-network-node (see Figure 24).

The model supports a single type of tree (Any-Source Multicast (ASM), Source-Specific Multicast (SSM), or bidirectional).

When ASM is used, the model supports the configuration of rendez-vous points (RPs). RP discovery may be 'static', 'bsr-rp', or 'auto-rp'. When set to 'static', RP to multicast grouping mapping MUST be configured as part of the 'rp-group-mappings' container. The RP MAY be a provider node or a customer node. When the RP is a customer node, the RP address must be configured using the 'rp-address' leaf otherwise no RP address is needed.

The model supports RP redundancy through the 'rp-redundancy' leaf. How the redundancy is achieved is out of scope and is up to the implementation.

When a particular VPN using ASM requires a more optimal traffic delivery, 'optimal-traffic-delivery' can be set. When set to 'true', the implementation must use any mechanism to provide a more optimal traffic delivery for the customer. Anycast is one of the mechanisms to enhance RPs redundancy, resilience against failures, and to recover from failures quickly.

For redundancy purposes, Multicast Source Discovery Protocol (MSDP) [RFC3618] may be enabled and used to share the state about sources between multiple RPs. The purpose of MSDP in this context is to enhance the robustness of the multicast service. MSDP may be configured on Non-RP routers, which is useful in a domain that does not support multicast sources, but does support multicast transit.

```
...
+--rw vpn-network-accesses
|  +--rw vpn-network-access* [id]
|     +--rw id
|     ..
+--rw multicast {vpn-common:multicast}?
   +--rw enabled?       boolean
   +--rw tree-flavor*   identityref
```

```
             +--rw rp
             |  +--rw rp-group-mappings
             |  |  +--rw rp-group-mapping* [id]
             |  |     +--rw id                  uint16
             |  |     +--rw provider-managed
             |  |     |  +--rw enabled?
             |  |     |  |       boolean
             |  |     |  +--rw rp-redundancy?
             |  |     |  |       boolean
             |  |     |  +--rw optimal-traffic-delivery?
             |  |     |  |       boolean
             |  |     |  +--rw anycast
             |  |     |     +--rw local-address?
             |  |     |     |       inet:ip-address
             |  |     |     +--rw rp-set-address*
             |  |     |             inet:ip-address
             |  |     +--rw rp-address
             |  |     |       inet:ip-address
             |  |     +--rw groups
             |  |        +--rw group* [id]
             |  |           +--rw id
             |  |           |       uint16
             |  |           +--rw (group-format)
             |  |              +--:(group-prefix)
             |  |              |  +--rw group-address?
             |  |              |          inet:ip-prefix
             |  |              +--:(startend)
             |  |                 +--rw group-start?
             |  |                 |       inet:ip-address
             |  |                 +--rw group-end?
             |  |                         inet:ip-address
             |  +--rw rp-discovery
             |     +--rw rp-discovery-type?   identityref
             |     +--rw bsr-candidates
             |        +--rw bsr-candidate-address*
             |                inet:ip-address
             +--rw msdp {msdp}?
                +--rw enabled?         boolean
                +--rw peer?            inet:ip-address
                +--rw local-address?   inet:ip-address
```

Figure 24: Multicast Subtree Structure

8. Layer 3 Network Model

This module uses types defined in [RFC6991] and groupings defined in [RFC8519].

```
<CODE BEGINS>  file "ietf-l3vpn-ntw@2020-10-16.yang"
module ietf-l3vpn-ntw {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l3vpn-ntw";
  prefix l3nm;

  import ietf-vpn-common {
    prefix vpn-common;
    reference
      "RFC UUUU: A Layer 2/3 VPN Common YANG Model";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }
  import ietf-packet-fields {
    prefix pf;
    reference
      "RFC 8519: YANG Data Model for Network Access
                 Control Lists (ACLs)";
  }

  organization
    "IETF OPSA (Operations and Management Area) Working Group ";
  contact
    "WG Web:   <http://tools.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>
     Editor:   Samier Barguil
               <mailto:samier.barguilgiraldo.ext@telefonica.com>
     Editor:   Oscar Gonzalez de Dios
               <mailto:oscar.gonzalezdedios@telefonica.com>
     Editor:   Mohamed Boucadair
              <mailto:mohamed.boucadair@orange.com>
     Author:   Luis Angel Munoz
               <mailto:luis-angel.munoz@vodafone.com>
     Author:   Alejandro Aguado
               <mailto:alejandro.aguado_martin@nokia.com>
    ";
  description
    "This YANG module defines a generic network-oriented model
     for the configuration of Layer 3 Virtual Private Networks.
```

```
     Copyright (c) 2020 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

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

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

  revision 2020-10-16 {
    description
      "Initial revision.";
    reference
      "RFC XXXX: A Layer 3 VPN Network YANG Model";
  }

  /* Features */

  feature msdp {
    description
      "This feature indicates that Multicast Source
       Discovery Protocol (MSDP) capabilities are
       supported by the VPN.";
    reference
      "RFC 3618: Multicast Source Discovery Protocol (MSDP)";
  }

  /* Typedefs */

  typedef area-address {
    type string {
      pattern '[0-9A-Fa-f]{2}(\.[0-9A-Fa-f]{4}){0,6}';
    }
    description
      "This type defines the area address format.";
  }

  /* Identities */

  identity address-allocation-type {
    description
      "Base identity for address-allocation-type for
       PE-CE link.";
```

```
  }

  identity provider-dhcp {
    base address-allocation-type;
    description
      "The Provider's network provides a DHCP service
       to the customer.";
  }

  identity provider-dhcp-relay {
    base address-allocation-type;
    description
      "The Provider's network provides a DHCP relay service
       to the customer.";
  }

  identity provider-dhcp-slaac {
    base address-allocation-type;
    description
      "The Provider's network provides a DHCP service to
       the customer, as well as IPv6 Stateless Address
       Autoconfiguration (SLAAC).";
    reference
      "RFC 7527: IPv6 Stateless Address Autoconfiguration";
  }

  identity static-address {
    base address-allocation-type;
    description
      "The Provider-to-customer addressing is static.";
  }

  identity slaac {
    base address-allocation-type;
    description
      "Use IPv6 SLAAC.";
    reference
      "RFC 7527: IPv6 Stateless Address Autoconfiguration";
  }

  identity isis-level {
    description
      "Defines the IS-IS level for interface
       and system.";
  }

  identity level1 {
    base isis-level;
```

```
    description
      "IS-IS level 1.";
  }

  identity level2 {
    base isis-level;
    description
      "IS-IS level 2.";
  }

  identity level1-2 {
    base isis-level;
    description
      "IS-IS levels 1 and 2.";
  }

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

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

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

  /* Groupings */

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

```
      "Grouping to define a set of security parameters";
  }

  grouping ports {
    choice source-port {
      container source-port-range-or-operator {
        uses pf:port-range-or-operator;
        description
          "Source port definition.";
      }
      description
        "Choice of specifying the source port or
          referring to a group of source port numbers.";
    }
    choice destination-port {
      container destination-port-range-or-operator {
        uses pf:port-range-or-operator;
        description
          "Destination port definition.";
      }
      description
        "Choice of specifying a destination port or
         referring to a group of destination port
         numbers.";
    }
    description
      "Choice of specifying a source or destination
       port numbers.";
  }

  /* Main Blocks */
  /* Main l3nm */

  container l3vpn-ntw {
    container vpn-profiles {
      uses vpn-common:vpn-profile-cfg;
      description
        "Contains a set of valid VPN Profiles to
         reference in the VPN service.";
    }
    container vpn-services {
      list vpn-service {
        key "vpn-id";
        uses vpn-common:service-status;
        uses vpn-common:vpn-description;
        leaf l3sm-vpn-id {
          type vpn-common:vpn-id;
          description
```

```
            "Pointer to the parent L3SM service,
             if any.";
        }
        leaf vpn-type {
          type identityref {
            base vpn-common:vpn-signaling-type;
          }
          description
            "Indicates the service type";
        }
        leaf vpn-service-topology {
          type identityref {
            base vpn-common:vpn-topology;
          }
          default "vpn-common:any-to-any";
          description
            "VPN service topology.";
        }
        container ie-profiles {
          list ie-profile {
            key "ie-profile-id";
            leaf ie-profile-id {
              type string;
              description
                "IE profile id.";
            }
            uses vpn-common:rt-rd;
            description
              "List for Imort/Export profile.";
          }
          description
            "Container for Import/Export profiles.";
        }
        uses vpn-common:svc-transport-encapsulation;
        container vpn-nodes {
          description
            "Container for VPN nodes.";
          list vpn-node {
            key "vpn-node-id";
            leaf vpn-node-id {
              type union {
                type vpn-common:vpn-id;
                type uint32;
              }
              description
                "Type STRING or NUMBER Service-Id.";
            }
            leaf local-autonomous-system {
```

```
              type inet:as-number;
              description
                "Provider's AS number in case the customer
                 requests BGP routing.";
            }
            leaf description {
              type string;
              description
                "Textual description of the VPN node.";
            }
            leaf ne-id {
              type string;
              description
                "Unique identifier of the network element
                 where the VPN node is deployed.";
            }
            leaf router-id {
              type inet:ip-address;
              description
                "The router-id information can be an IPv4
                 or IPv6 address.";
            }
            leaf address-family {
              type vpn-common:address-family;
              description
                "The address family used for router-id
                 information.";
            }
            leaf node-role {
              type identityref {
                base vpn-common:role;
              }
              default "vpn-common:any-to-any-role";
              description
                "Role of the VPN node in the IP VPN.";
            }
            uses vpn-common:rt-rd;
            uses vpn-common:service-status;
            leaf node-ie-profile {
              type leafref {
                path "/l3vpn-ntw/vpn-services/"
                   + "vpn-service/ie-profiles/"
                   + "ie-profile/ie-profile-id";
              }
              description
                "Node's Import/Export profile.";
            }
            uses vpn-common:vpn-node-group;
```

```
            container vpn-network-accesses {
              list vpn-network-access {
                key "id";
                leaf id {
                  type vpn-common:vpn-id;
                  description
                    "Identifier for the access.";
                }
                leaf port-id {
                  type vpn-common:vpn-id;
                  description
                    "Identifier for the network access.";
                }
                leaf description {
                  type string;
                  description
                    "Textual description of a network access.";
                }
                uses vpn-common:service-status;
                leaf vpn-network-access-type {
                  type identityref {
                    base vpn-common:site-network-access-type;
                  }
                  default "vpn-common:point-to-point";
                  description
                    "Describes the type of connection, e.g.,
                     point-to-point or multipoint.";
                }
                container connection {
                  leaf encapsulation-type {
                    type identityref {
                      base vpn-common:encapsulation-type;
                    }
                    default "vpn-common:untagged-int";
                    description
                      "Encapsulation type.  By default,
                       the encapsulation type is set to
                       'untagged'.";
                  }
                  container logical-interface {
                    leaf peer-reference {
                      type uint32;
                      description
                        "Specify the associated logical peer
                         interface";
                    }
                    description
                      "Reference of a logical interface
```

```
                      type.";
                 }
                 container tagged-interface {
                   leaf type {
                     type identityref {
                       base vpn-common:encapsulation-type;
                     }
                     default "vpn-common:priority-tagged";
                     description
                       "Tagged interface type.  By default,
                        the type of the tagged interface is
                        'priority-tagged'.";
                   }
                   container dot1q-vlan-tagged {
                     when "derived-from-or-self(../type, "
                        + "'vpn-common:dot1q')" {
                       description
                         "Only applies when the type of the
                          tagged interface is 'dot1q'.";
                     }
                     if-feature "vpn-common:dot1q";
                     leaf tag-type {
                       type identityref {
                         base vpn-common:tag-type;
                       }
                       default "vpn-common:c-vlan";
                       description
                         "Tag type.  By default, the tag
                          type is 'c-vlan'.";
                     }
                     leaf cvlan-id {
                       type uint16;
                       description
                         "VLAN identifier.";
                     }
                     description
                       "Tagged interface.";
                   }
                   container priority-tagged {
                     when "derived-from-or-self(../type, "
                        + "'vpn-common:priority-tagged')" {
                       description
                         "Only applies when the type of the
                          tagged interface is
                          'priority-tagged'.";
                     }
                     leaf tag-type {
                       type identityref {
```

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

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

```
                  description
                    "Peer IP.";
                }
                description
                  "List of peer IP addresses.";
              }
              description
                "QinQ.";
            }
            description
              "Container for tagged interfaces.";
          }
          container bearer {
            leaf bearer-reference {
              if-feature "vpn-common:bearer-reference";
              type string;
              description
                "This is an internal reference for*
                 the SP.";
            }
            container pseudowire {
              leaf vcid {
                type uint32;
                description
                  "PW or VC identifier.";
              }
              leaf far-end {
                type union {
                  type uint32;
                  type inet:ipv4-address;
                }
                description
                  "SDP/Far End/LDP neighbour reference.";
              }
              description
                "Pseudowire termination parameters";
            }
            container vpls {
              leaf vcid {
                type union {
                  type uint32;
                  type string;
                }
                description
                  "VCID identifier, IRB/RVPPLs interface
                   supported using string
                   format.";
              }
```

```
                  leaf far-end {
                    type union {
                      type uint32;
                      type inet:ipv4-address;
                    }
                    description
                      "SDP/Far End/LDP Neighbour reference.";
                  }
                  description
                    "Pseudowire termination parameters";
                }
                description
                  "Defines physical properties of a site
                   attachment.";
              }
              description
                "Encapsulation types";
            }
            container ip-connection {
              container ipv4 {
                if-feature "vpn-common:ipv4";
                leaf address-allocation-type {
                  type identityref {
                    base address-allocation-type;
                  }
                  must "not(derived-from-or-self(current(), "
                    + "'slaac') or derived-from-or-self(current(),"
                    + " 'provider-dhcp-slaac'))" {
                    error-message "SLAAC is only applicable to
                                 IPv6";
                  }
                  description
                    "Defines how addresses are allocated.
                     If there is no value for the address
                     allocation type, then IPv4 is not enabled.";
                }
                choice allocation-type {
                  case provider-dhcp {
                    when "derived-from-or-self(./address-"
                       + "allocation-type, 'provider-dhcp')" {
                      description
                        "Only applies when addresses are
                         allocated by DHCP.";
                    }
                    leaf provider-address {
                      type inet:ipv4-address;
                      description
                        "Address of provider side.
```

```
                        If provider-address is not specified,
                        then prefix length should not be
                        specified either.

                        It also implies provider-dhcp
                        allocation is not enabled.

                        If provider-address is specified,
                        then the prefix length may or
                        may not be specified.";
                   }
                   leaf prefix-length {
                     type uint8 {
                       range "0..32";
                     }
                     must '(../provider-address)' {
                       error-message
                         "If the prefix length is specified,
                          provider-address must also be
                          specified.";
                       description
                         "If the prefix length is specified,
                          provider-address must also be
                          specified.";
                     }
                     description
                       "Subnet prefix length expressed in bits.
                        If not specified, or specified as zero,
                        this means the customer leaves the actual
                        prefix length value to the provider.";
                   }
                   choice address-assign {
                     default "number";
                     case number {
                       leaf number-of-dynamic-address {
                         type uint16;
                         default "1";
                         description
                           "Describes the number of IP
                            addresses the customer requires.";
                       }
                     }
                     case explicit {
                       container customer-addresses {
                         list address-group {
                           key "group-id";
                           leaf group-id {
                             type string;
```

```
                    description
                      "Group-id for the address range
                       from start-address to
                       end-address.";
                  }
                  leaf start-address {
                    type inet:ipv4-address;
                    description
                      "First address.";
                  }
                  leaf end-address {
                    type inet:ipv4-address;
                    description
                      "Last address.";
                  }
                  description
                    "Describes IP addresses allocated by
                     DHCP.

                     When only start-address or only
                     end-address is present, it
                     represents a single address.
                     When both start-address and
                     end-address are specified, it
                     implies a range inclusive of
                     both addresses.  If no address
                     is specified, it implies customer
                     addresses group is not supported.";
                }
                description
                  "Container for customer addresses is
                   allocated by DHCP.";
              }
            }
            description
              "Choice for the way to assign
               addresses.";
          }
          description
            "DHCP allocated addresses related
             parameters.";
        }
        case dhcp-relay {
          when "derived-from-or-self(./address-allocation"
             + "-type, 'provider-dhcp-relay')" {
            description
              "Only applies when provider is required to
               implement DHCP relay function.";
```

```
                        }
                        leaf dr-provider-address {
                          type inet:ipv4-address;
                          description
                            "Address of provider side.

                             If provider-address is
                             not specified, then prefix length
                             should not be specified either.

                             It also implies provider-dhcp
                             allocation is not enabled.

                             If provider-address is specified,
                             then prefix length may or may
                             not be specified.";
                        }
                        leaf dr-prefix-length {
                          type uint8 {
                            range "0..32";
                          }
                          must '(../dr-provider-address)' {
                            error-message
                              "If prefix length is specified,
                               provider-address must also be
                               specified.";
                            description
                              "If prefix length is specified,
                               provider-address must also be
                               specified.";
                          }
                          description
                            "Subnet prefix length expressed in bits.

                             If not specified, or specified as zero,
                             this means the customer leaves the
                             actual prefix  length value
                             to the provider.";
                        }
                        container customer-dhcp-servers {
                          leaf-list server-ip-address {
                            type inet:ipv4-address;
                            description
                              "IP address of customer DHCP
                               server.";
                          }
                          description
                            "Container for list of customer
```

```
                       DHCP servers.";
                  }
                  description
                    "DHCP relay provided by operator.";
                }
                case static-addresses {
                  when "derived-from-or-self(./address-allocation"
                     + "-type, 'static-address')" {
                    description
                      "Only applies when address allocation
                      type is static.";
                  }
                  leaf primary-address {
                    type leafref {
                      path "../address/address-id";
                    }
                    description
                      "Principal address of the connection.";
                  }
                  list address {
                    key "address-id";
                    leaf address-id {
                      type string;
                      description
                        "IPv4 Address";
                    }
                    leaf s-provider-address {
                      type inet:ipv4-address;
                      description
                        "IPv4 Address List of the provider side.
                         When the protocol allocation type is
                         static, the provider address must be
                         configured.";
                    }
                    leaf s-customer-address {
                      type inet:ipv4-address;
                      description
                        "IPv4 Address of customer side.";
                    }
                    leaf s-prefix-length {
                      type uint8 {
                        range "0..32";
                      }
                      description
                        "Subnet prefix length expressed
                         in bits. It is applied to both
                         provider-address and customer-address.";
                    }
```

```
                description
                  "Describes IPv4 addresses used.";
              }
              description
                "Describes IPv4 addresses used.";
            }
            description
              "Choice the address allocation.";
          }
          description
            "IPv4-specific parameters.";
        }
        container ipv6 {
          if-feature "vpn-common:ipv6";
          leaf address-allocation-type {
            type identityref {
              base address-allocation-type;
            }
            description
              "Defines how addresses are allocated.
               If there is no value for the address
               allocation type, then IPv6 is
               not enabled.";
          }
          choice allocation-type {
            choice provider-dhcp {
              when "derived-from-or-self(./address-allo"
                 + "cation-type, 'provider-dhcp') "
                 + "or derived-from-or-self(./address-allo"
                 + "cation-type, 'provider-dhcp-slaac')" {
                description
                  "Only applies when addresses are
                   allocated by DHCP.";
              }
              leaf provider-address {
                type inet:ipv6-address;
                description
                  "Address of the provider side.

                   If provider-address is not specified,
                   then prefix length should not be
                   specified either.  It also implies
                   provider-dhcp allocation is not
                   enabled.

                   If provider-address is
                   specified, then prefix length may
                   or may not be specified.";
```

```
                        }
                        leaf prefix-length {
                          type uint8 {
                            range "0..128";
                          }
                          must '(../provider-address)' {
                            error-message
                              "If prefix length is specified,
                               provider-address
                               must also be specified.";
                            description
                              "If prefix length is specified,
                                provider-address
                               must also be specified.";
                          }
                          description
                            "Subnet prefix length expressed in
                             bits.

                             If not specified, or specified as
                             zero, this means the customer leaves
                             the actual prefix length value to
                             the provider.";
                        }
                        choice address-assign {
                          default "number";
                          case number {
                            leaf number-of-dynamic-address {
                              type uint16;
                              default "1";
                              description
                                "Describes the number of IP
                                 addresses required by the
                                 customer.";
                            }
                          }
                          case explicit {
                            container customer-addresses {
                              list address-group {
                                key "group-id";
                                leaf group-id {
                                  type string;
                                  description
                                    "Group-id for the address range
                                     from start-address to
                                     end-address.";
                                }
                                leaf start-address {
```

```
                      type inet:ipv6-address;
                      description
                        "First address.";
                    }
                    leaf end-address {
                      type inet:ipv6-address;
                      description
                        "Last address.";
                    }
                    description
                      "Describes IP addresses allocated
                       by DHCP.

                       When only start-address or only
                       end-address is present, it
                       represents a single address.

                       When both start-address and
                       end-address are specified, it
                       implies a range inclusive of
                       both addresses.

                       If no address is specified, it
                       implies customer addresses group
                       is not supported.";
                  }
                  description
                    "Container for customer addresses
                     allocated by DHCP.";
                }
              }
              description
                "Choice for the way to assign addresses.";
            }
            description
              "DHCP allocated addresses related
               parameters.";
          }
          case dhcp-relay {
            when "derived-from-or-self(./address-allo"
               + "cation-type, 'provider-dhcp-relay')" {
              description
                "Only applies when the provider is required
                 to implement DHCP relay function.";
            }
            leaf dr-provider-address {
              type inet:ipv6-address;
              description
```

```
                    "Address of the provider side.

                     If provider-address is not specified,
                     then prefix length should not be
                     specified either.  It also implies
                     provider-dhcp allocation is not enabled.

                     If provider address is specified, then
                     prefix length may or may not be
                     specified.";
                }
                leaf dr-prefix-length {
                  type uint8 {
                    range "0..128";
                  }
                  must '(../dr-provider-address)' {
                    error-message
                      "If prefix length is specified,
                       provider-address must also be
                       specified.";
                    description
                      "If prefix length is specified,
                       provider-address must also be
                       specified.";
                  }
                  description
                    "Subnet prefix length expressed in bits.

                     If not specified, or specified as zero,
                     this means the customer leaves the
                     actual prefix length value to the
                     provider.";
                }
                container customer-dhcp-servers {
                  leaf-list server-ip-address {
                    type inet:ipv6-address;
                    description
                      "This node contains the IP address of
                       the customer DHCP server.  If the DHCP
                       relay function is implemented by the
                       provider, this node contains the
                       configured value.";
                  }
                  description
                    "Container for list of customer DHCP
                     servers.";
                }
                description
```

```
                    "DHCP relay provided by operator.";
                }
                case static-addresses {
                  when "derived-from-or-self(./address-allocation"
                     + "-type, 'static-address')" {
                    description
                      "Only applies when protocol allocation type
                       is static.";
                  }
                  leaf s-primary-address {
                    type leafref {
                      path "../s-address/address-id";
                    }
                    description
                      "Principal address of the connection";
                  }
                  list s-address {
                    key "address-id";
                    leaf address-id {
                      type string;
                      description
                        "IPv4 Address";
                    }
                    leaf provider-address {
                      type inet:ipv6-address;
                      description
                        "IPv6 Address of the provider side.  When
                         the protocol allocation type is static,
                         the provider address must be
                         configured.";
                    }
                    leaf customer-address {
                      type inet:ipv6-address;
                      description
                        "The IPv6 Address of the customer side.";
                    }
                    leaf prefix-length {
                      type uint8 {
                        range "0..128";
                      }
                      description
                        "Subnet prefix length expressed in bits.
                         It is applied to both provider-address
                         and customer-address.";
                    }
                    description
                      "Describes IPv6 addresses used.";
                  }
```

```
                       description
                         "IPv6-specific parameters.";
                     }
                     description
                       "IPv6 allocation type.";
                   }
                   description
                     "IPv6-specific parameters.";
                 }
                 container oam {
                   container bfd {
                     if-feature "vpn-common:bfd";
                     leaf enabled {
                       type boolean;
                       default "false";
                       description
                         "If true, BFD activation is required.";
                     }
                     choice holdtime {
                       default "fixed";
                       case fixed {
                         leaf fixed-value {
                           type uint32;
                           units "msec";
                           description
                             "Expected BFD holdtime.

                              The customer may impose some fixed
                              values for the holdtime period if the
                              provider allows the customer use this
                              function.

                              If the provider doesn't allow the
                              customer to use this function,
                              the fixed-value will not be set.";
                         }
                       }
                       case profile {
                         leaf profile-name {
                           type leafref {
                             path "/l3vpn-ntw/vpn-profiles/"
                                + "valid-provider-identifiers/"
                                + "bfd-profile-identifier/id";
                           }
                           description
                             "Well-known SP profile name.

                              The provider can propose some profiles
```

```
                              to the customer, depending on the
                              service level the customer wants to
                              achieve.

                              Profile names must be communicated to
                              the customer.";
                         }
                         description
                           "Well-known SP profile.";
                       }
                       description
                         "Choice for holdtime flavor.";
                     }
                     description
                       "Container for BFD.";
                   }
                   description
                     "Defines the Operations, Administration,
                      and Maintenance (OAM)mechanisms used on
                      the connection.

                      BFD is set as a fault detection mechanism,
                      but the 'oam' container can easily
                      be augmented by other mechanisms";
                 }
                 description
                   "Defines connection parameters.";
               }
               container security {
                 container encryption {
                   if-feature "vpn-common:encryption";
                   leaf enabled {
                     type boolean;
                     default "false";
                     description
                       "If true, traffic encryption on the
                        connection is required. It is
                        disabled, otherwise.";
                   }
                   leaf layer {
                     when "../enabled = 'true'" {
                       description
                         "Require a value for layer when
                          enabled is true.";
                     }
                     type enumeration {
                       enum layer2 {
                         description
```

```
                      "Encryption will occur at Layer 2.";
                  }
                  enum layer3 {
                    description
                      "Encryption will occur at Layer 3.
                       For example, IPsec may be used when
                       a customer requests Layer 3
                       encryption.";
                  }
                }
                description
                  "Layer on which encryption is applied.";
              }
              description
                "Container for CE-PE security encryption.";
            }
            container encryption-profile {
              choice profile {
                case provider-profile {
                  leaf profile-name {
                    type leafref {
                      path "/l3vpn-ntw/vpn-profiles"
                         + "/valid-provider-identifiers"
                         + "/encryption-profile-identifier/id";
                    }
                    description
                      "Name of the SP profile to be applied.";
                  }
                }
                case customer-profile {
                  leaf algorithm {
                    type string;
                    description
                      "Encryption algorithm to be used.";
                  }
                }
                description
                  "Choice for encryption profile.";
              }
              choice key-type {
                default "psk";
                case psk {
                  leaf preshared-key {
                    type string;
                    description
                      "Pre-Shared Key (PSK) coming from the
                       customer.";
                  }
```

```
                    }
                    description
                      "Choice of encryption profile.
                       The encryption profile can be the
                       provider profile or customer profile.";
                  }
                  description
                    "Container for encryption profile.";
                }
                description
                  "Site-specific security parameters.";
              }
              container routing-protocols {
                list routing-protocol {
                  key "id";
                  leaf id {
                    type string;
                    description
                      "Unique identifier for routing protocol.";
                  }
                  leaf type {
                    type identityref {
                      base vpn-common:routing-protocol-type;
                    }
                    description
                      "Type of routing protocol.";
                  }
                  list routing-profiles {
                    key "id";
                    leaf id {
                      type leafref {
                        path "/l3vpn-ntw/vpn-profiles"
                           + "/valid-provider-identifiers"
                           + "/routing-profile-identifier/id";
                      }
                      description
                        "Routing profile to be used.";
                    }
                    leaf type {
                      type identityref {
                        base vpn-common:ie-type;
                      }
                      description
                        "Import, export or both.";
                    }
                    description
                      "Routing profiles.";
                  }
```

```
                   container ospf {
                     when "derived-from-or-self(../type, "
                        + "'vpn-common:ospf')" {
                       description
                         "Only applies when protocol is OSPF.";
                     }
                     if-feature "vpn-common:rtg-ospf";
                     leaf-list address-family {
                       type vpn-common:address-family;
                       min-elements 1;
                       description
                         "If OSPF is used on this site, this node
                          contains a configured value.  This node
                          contains at least one address family
                          to be activated.";
                     }
                     leaf area-address {
                       type yang:dotted-quad;
                       mandatory true;
                       description
                         "Area address.";
                     }
                     leaf metric {
                       type uint16;
                       default "1";
                       description
                         "Metric of the PE-CE link.  It is used
                          in the routing state calculation and
                          path selection.";
                     }
                     leaf mtu {
                       type uint16;
                       description
                         "Maximum transmission unit for a given
                          OSPF link.";
                     }
                     leaf process-id {
                       type uint16;
                       description
                         "Process id of the OSPF CE-PE connection.";
                     }
                     uses security-params;
                     container sham-links {
                       if-feature "vpn-common:rtg-ospf-sham-link";
                       list sham-link {
                         key "target-site";
                         leaf target-site {
                           type vpn-common:vpn-id;
```

```
                   description
                     "Target site for the sham link connection.
                      The site is referred to by its ID.";
                 }
                 leaf metric {
                   type uint16;
                   default "1";
                   description
                     "Metric of the sham link.  It is used in
                      the routing state calculation and path
                      selection.  The default value is set
                      to 1.";
                 }
                 description
                   "Creates a sham link with another site.";
               }
               description
                 "List of sham links.";
             }
             description
               "OSPF-specific configuration.";
           }
           container bgp {
             when "derived-from-or-self(../type, "
                + "'vpn-common:bgp')" {
               description
                 "Only applies when protocol is BGP.";
             }
             if-feature "vpn-common:rtg-bgp";
             leaf peer-autonomous-system {
               type inet:as-number;
               mandatory true;
               description
                 "Indicates the Customer's AS Number (ASN) in
                  case the Customer requests BGP routing.";
             }
             leaf local-autonomous-system {
               type inet:as-number;
               description
                 "Is set to the ASN to override a peers' ASN
                  if such feature is requested by the
                  Customer.";
             }
             leaf-list address-family {
               type vpn-common:address-family;
               min-elements 1;
               description
                 "This node contains at least one
```

```
                        address-family to be activated.";
                   }
                   leaf-list neighbor {
                     type inet:ip-address;
                     description
                       "IP address(es) of the BGP neighbor. IPv4
                        and IPv6 neighbors may be indicated if
                        two sessions will be used for IPv4 and
                        IPv6.";
                   }
                   leaf multihop {
                     type uint8;
                     description
                       "Describes the number of IP hops allowed
                        between a given BGP neighbor and the PE.";
                   }
                   uses security-params;
                   uses vpn-common:service-status;
                   leaf description {
                     type string;
                     description
                       "Includes a description of the BGP session.
                        Such description is meant to be used for
                        diagnosis purposes. The semantic of the
                        description is local to an
                        implementation.";
                   }
                   leaf as-override {
                     type boolean;
                     default "false";
                     description
                       "Defines whether AS override is enabled,
                        i.e., replace the ASN of the peer specified
                        in the AS Path attribute with the local
                        AS number.";
                   }
                   leaf default-route {
                     type boolean;
                     default "false";
                     description
                       "Defines whether default route(s) can be
                        advertised to its peer. If set, the
                        default route(s) is advertised to its
                        peer.";
                   }
                   container bgp-max-prefix {
                     leaf max-prefix {
                       type uint32;
```

```
                    default "5000";
                    description
                      "Indicates the maximimum number of BGP
                       prefixes allowed in the BGP session.

                       It allows to control how many prefixes
                       can be received from a neighbor.

                       If the limit is exceeded, the session
                        can be teared down.";
                    reference
                      "RFC4271, Section 8.2.2.";
                  }
                  leaf warning-threshold {
                    type decimal64 {
                      fraction-digits 5;
                      range "0..100";
                    }
                    units "percent";
                    default "75";
                    description
                      "When this value is reached, a warning
                       notification will be triggered.";
                  }
                  leaf violate-action {
                    type enumeration {
                      enum warning {
                        description
                          "Only a warning message is sent to
                           the peer when the limit is
                           exceeded.";
                      }
                      enum discard-extra-paths {
                        description
                          "Discards extra paths when the
                           limit is exceeded.";
                      }
                      enum restart {
                        description
                          "Restart after a time interval.";
                      }
                    }
                    description
                      "BGP neighbour max-prefix violate
                       action";
                  }
                  leaf restart-interval {
                    type uint16;
```

```
                        units "minutes";
                        description
                          "Time interval (min) after which the
                           BGP session will be reestablished.";
                      }
                      description
                        "Controls the behavior when a prefix
                         maximum is reached.";
                    }
                    container bgp-timer {
                      description
                        "Includes two BGP timers that can be
                         customized when building a VPN service
                         with BGP used as CE-PE routing
                         protocol.";
                      leaf keep-alive {
                        type uint16 {
                          range "0..21845";
                        }
                        units "seconds";
                        default "30";
                        description
                          "This timer indicates the KEEPALIVE
                           messages'  frequency between a PE
                           and a BGP peer.

                           If set to '0', it indicates KEEPALIVE
                           messages are disabled.

                           It is suggested that the maximum time
                           between  KEEPALIVEmessages would be
                           one third of the Hold Time interval.";
                        reference
                          "Section 4.4 of RFC 4271";
                      }
                      leaf hold-time {
                        type uint16 {
                          range "0 | 3..65535";
                        }
                        units "seconds";
                        default "90";
                        description
                          "It indicates the maximum number of
                          seconds that may elapse between the
                          receipt of successive KEEPALIVE
                          and/or UPDATE   messages from the peer.

                          The Hold Time must be either zero or
```

```
                         at least three seconds.";
                       reference
                         "Section 4.2 of RFC 4271";
                     }
                   }
                   description
                     "BGP-specific configuration.";
                 }
                 container isis {
                   when "derived-from-or-self(../type, "
                      + "'vpn-common:isis')" {
                     description
                       "Only applies when protocol is IS-IS.";
                   }
                   if-feature "vpn-common:rtg-isis";
                   leaf-list address-family {
                     type vpn-common:address-family;
                     min-elements 1;
                     description
                       "If ISIS is used on this site, this node
                        contains a configured value.  This node
                        contains at least one address family
                        to be activated.";
                   }
                   leaf area-address {
                     type yang:dotted-quad;
                     mandatory true;
                     description
                       "Area address.";
                   }
                   leaf level {
                     type identityref {
                       base isis-level;
                     }
                     description
                       "level1, level2 or level1-2";
                   }
                   leaf metric {
                     type uint16;
                     default "1";
                     description
                       "Metric of the PE-CE link.  It is used
                        in the routing state calculation and
                        path selection.";
                   }
                   leaf process-id {
                     type uint16;
                     description
```

```
                     "Process id of the IS-IS CE-PE
                      connection.";
                 }
                 leaf mode {
                   type enumeration {
                     enum active {
                       description
                         "Interface sends or receives IS-IS
                          protocol control packets.";
                     }
                     enum passive {
                       description
                         "Suppresses the sending of IS-IS
                          updates through the specified
                          interface.";
                     }
                   }
                   default "active";
                   description
                     "IS-IS interface mode type.";
                 }
                 uses vpn-common:service-status;
                 description
                   "IS-IS specific configuration.";
               }
               container static {
                 when "derived-from-or-self(../type, "
                    + "'vpn-common:static')" {
                   description
                     "Only applies when protocol is static.
                      BGP activation requires the SP to know
                      the address of the customer peer.  When
                      BGP is enabled, the 'static-address'
                      allocation type for the IP connection
                      must be used.";
                 }
                 container cascaded-lan-prefixes {
                   list ipv4-lan-prefixes {
                     if-feature "vpn-common:ipv4";
                     key "lan next-hop";
                     leaf lan {
                       type inet:ipv4-prefix;
                       description
                         "LAN prefixes.";
                     }
                     leaf lan-tag {
                       type string;
                       description
```

```
                        "Internal tag to be used in VPN
                         policies.";
                    }
                    leaf next-hop {
                      type inet:ipv4-address;
                      description
                        "Next-hop address to use on the
                         customer side.";
                    }
                    description
                      "List of LAN prefixes for the site.";
                  }
                  list ipv6-lan-prefixes {
                    if-feature "vpn-common:ipv6";
                    key "lan next-hop";
                    leaf lan {
                      type inet:ipv6-prefix;
                      description
                        "LAN prefixes.";
                    }
                    leaf lan-tag {
                      type string;
                      description
                        "Internal tag to be used in VPN
                         policies.";
                    }
                    leaf next-hop {
                      type inet:ipv6-address;
                      description
                        "Next-hop address to use on the
                         customer side.";
                    }
                    description
                      "List of LAN prefixes for the site.";
                  }
                  description
                    "LAN prefixes from the customer.";
                }
                description
                  "Configuration specific to static routing.";
              }
              container rip {
                when "derived-from-or-self(../type, "
                   + "'vpn-common:rip')" {
                  description
                    "Only applies when the protocol is RIP.
                     For IPv4, the model assumes that RIP
                     version 2 is used.";
```

```
                      }
                      if-feature "vpn-common:rtg-rip";
                      leaf-list address-family {
                        type vpn-common:address-family;
                        min-elements 1;
                        description
                          "If RIP is used on this site, this node
                           contains a configured value.  This node
                           contains at least one address family
                           to be activated.";
                      }
                      description
                        "Configuration specific to RIP routing.";
                    }
                    container vrrp {
                      when "derived-from-or-self(../type, "
                         + "'vpn-common:vrrp')" {
                        description
                          "Only applies when protocol is VRRP.";
                      }
                      if-feature "vpn-common:rtg-vrrp";
                      leaf-list address-family {
                        type vpn-common:address-family;
                        min-elements 1;
                        description
                          "If VRRP is used on this site, this node
                           contains a configured value.  This node
                           contains at least one address family to
                           be activated.";
                      }
                      leaf vrrp-group {
                        type uint8 {
                          range "1..255";
                        }
                        description
                          "VRRP group number";
                      }
                      leaf backup-peer {
                        type inet:ip-address;
                        description
                          "IP address of the peer";
                      }
                      leaf priority {
                        type uint8;
                        description
                          "Local priority of the VRRP speaker";
                      }
                      leaf ping-reply {
```

```
                  type boolean;
                  description
                    "Whether the VRRP speaker should answer
                     to ping requests";
                }
                description
                  "Configuration specific to VRRP.";
              }
              description
                "List of routing protocols used on
                 the site.  This list can be augmented.";
            }
            description
              "Defines routing protocols.";
          }
          container service {
            leaf svc-input-bandwidth {
              type uint64;
              units "bps";
              mandatory true;
              description
                "From the customer site's perspective, the
                 service input bandwidth of the connection
                 or download bandwidth from the SP to
                 the site.";
            }
            leaf svc-output-bandwidth {
              type uint64;
              units "bps";
              mandatory true;
              description
                "From the customer site's perspective,
                 the service output bandwidth of the
                 connection or upload bandwidth from
                 the site to the SP.";
            }
            leaf svc-mtu {
              type uint16;
              units "bytes";
              mandatory true;
              description
                "MTU at service level.  If the service is IP,
                 it refers to the IP MTU.  If CsC is enabled,
                 the requested 'svc-mtu' leaf will refer
                  to the MPLS MTU and not to the IP MTU.";
            }
            container qos {
              if-feature "vpn-common:qos";
```

```
                    container qos-classification-policy {
                      list rule {
                        key "id";
                        ordered-by user;
                        leaf id {
                          type string;
                          description
                            "A description identifying the
                             qos-classification-policy rule.";
                        }
                        choice match-type {
                          default "match-flow";
                          case match-flow {
                            choice l3 {
                              container ipv4 {
                                uses pf:acl-ip-header-fields;
                                uses pf:acl-ipv4-header-fields;
                                description
                                  "Rule set that matches IPv4 header.";
                              }
                              container ipv6 {
                                uses pf:acl-ip-header-fields;
                                uses pf:acl-ipv6-header-fields;
                                description
                                  "Rule set that matches IPv6 header.";
                              }
                              description
                                "Either IPv4 or IPv6.";
                            }
                            choice l4 {
                              container tcp {
                                uses pf:acl-tcp-header-fields;
                                uses ports;
                                description
                                  "Rule set that matches TCP header.";
                              }
                              container udp {
                                uses pf:acl-udp-header-fields;
                                uses ports;
                                description
                                  "Rule set that matches UDP header.";
                              }
                              description
                                "Can be TCP or UDP";
                            }
                          }
                          case match-application {
                            leaf match-application {
```

```
                      type identityref {
                        base vpn-common:customer-application;
                      }
                      description
                        "Defines the application to match.";
                    }
                  }
                  description
                    "Choice for classification.";
                }
                leaf target-class-id {
                  type string;
                  description
                    "Identification of the class of service.
                     This identifier is internal to the
                     administration.";
                }
                description
                  "List of marking rules.";
              }
              description
                "Configuration of the traffic classification
                 policy.";
            }
            container qos-profile {
              list qos-profile {
                key "profile";
                description
                  "QoS profile.
                   Can be standard profile or customized
                   profile.";
                leaf profile {
                  type leafref {
                    path "/l3vpn-ntw/vpn-profiles"
                       + "/valid-provider-identifiers"
                       + "/qos-profile-identifier/id";
                  }
                  description
                    "QoS profile to be used.";
                }
                leaf direction {
                  type identityref {
                    base vpn-common:qos-profile-direction;
                  }
                  default "vpn-common:both";
                  description
                    "The direction to which the QoS profile
                     is applied.";
```

```
                 }
               }
               description
                 "QoS profile configuration.";
             }
             description
               "QoS configuration.";
           }
           container carrierscarrier {
             if-feature "vpn-common:carrierscarrier";
             leaf signalling-type {
               type enumeration {
                 enum ldp {
                   description
                     "Use LDP as the signalling protocol
                      between the PE and the CE.  In this
                      case, an IGP routing protocol must
                      also be activated.";
                 }
                 enum bgp {
                   description
                     "Use BGP as the signalling protocol
                      between the PE and the CE.
                      In this case, BGP must also be configured
                      as the routing protocol.";
                   reference
                     "RFC 8277: Using BGP to Bind MPLS Labels
                                to Address Prefixes";
                 }
               }
               default "bgp";
               description
                 "MPLS signalling type.";
             }
             description
               "This container is used when the customer
                provides MPLS-based services.  This is
                only used in the  case of CsC (i.e., a
                customer builds an MPLSservice using an
                IP VPN to carry its traffic).";
           }
           container multicast {
             if-feature "vpn-common:multicast";
             leaf site-type {
               type enumeration {
                 enum receiver-only {
                   description
                     "The site only has receivers.";
```

```
                        }
                        enum source-only {
                          description
                            "The site only has sources.";
                        }
                        enum source-receiver {
                          description
                            "The site has both sources and
                             receivers.";
                        }
                      }
                      default "source-receiver";
                      description
                        "Type of multicast site.";
                    }
                    leaf address-family {
                      type vpn-common:address-family;
                      description
                        "Address family.";
                    }
                    leaf protocol-type {
                      type enumeration {
                        enum host {
                          description
                            "Hosts are directly connected to the
                             provider network.

                             Host protocols such as IGMP or MLD are
                             required.";
                        }
                        enum router {
                          description
                            "Hosts are behind a customer router.
                             PIM will be implemented.";
                        }
                        enum both {
                          description
                            "Some hosts are behind a customer router,
                             and some others are directly connected
                             to the provider network.  Both host and
                             routing protocols must be used.

                             Typically,  IGMP and PIM will be
                             implemented.";
                        }
                      }
                      default "both";
                      description
```

```
                    "Multicast protocol type to be used with
                     the customer site.";
                }
                leaf remote-source {
                  type boolean;
                  default "false";
                  description
                    "When true, there is no PIM adjacency on
                     the interface.";
                }
                description
                  "Multicast parameters for the site.";
              }
              description
                "Service parameters on the attachment.";
            }
            description
              "List of accesses for a site.";
          }
          description
            "List of accesses for a site.";
        }
        container maximum-routes {
          list address-family {
            key "af";
            leaf af {
              type vpn-common:address-family;
              description
                "Indicates the address family
                (IPv4 or IPv6).";
            }
            leaf maximum-routes {
              type uint32;
              description
                "Indicates the maximum prefixes the VRF
                 can accept for this address family.";
            }
            description
              "List of address families.";
          }
          description
            "Defines 'maximum-routes' for the VRF.";
        }
        container multicast {
          if-feature "vpn-common:multicast";
          leaf enabled {
            type boolean;
            default "false";
```

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

```
                    "Only applicable if
                     RP redundancy is
                     enabled and delivery through
                     optimal path is activated.";
                }
                leaf local-address {
                  type inet:ip-address;
                  description
                    "IP local address for PIM RP.
                     Usually, it corresponds to router
                     ID or primary address";
                }
                leaf-list rp-set-address {
                  type inet:ip-address;
                  description
                    "Address other RP routers
                     that share the same RP IP address.";
                }
                description
                  "PIM Anycast-RP parameters.";
              }
              description
                "Parameters for a provider-managed RP.";
            }
            leaf rp-address {
              when "../provider-managed/enabled = 'false'" {
                description
                  "Relevant when the RP is not
                   provider-managed.";
              }
              type inet:ip-address;
              mandatory true;
              description
                "Defines the address of the RP.
                 Used if the RP is customer-managed.";
            }
            container groups {
              list group {
                key "id";
                leaf id {
                  type uint16;
                  description
                    "Identifier for the group.";
                }
                choice group-format {
                  mandatory true;
                  case group-prefix {
                    leaf group-address {
```

```
                      type inet:ip-prefix;
                      description
                        "A single multicast group prefix.";
                    }
                  }
                  case startend {
                    leaf group-start {
                      type inet:ip-address;
                      description
                        "The first multicast group address in
                         the multicast group address range.";
                    }
                    leaf group-end {
                      type inet:ip-address;
                      description
                        "The last multicast group address in
                         the multicast group address range.";
                    }
                  }
                  description
                    "Choice for multicast group format.";
                }
                description
                  "List of multicast groups.";
              }
              description
                "Multicast groups associated with the RP.";
            }
            description
              "List of RP-to-group mappings.";
          }
          description
            "RP-to-group mappings parameters.";
        }
        container rp-discovery {
          leaf rp-discovery-type {
            type identityref {
              base vpn-common:multicast-rp-discovery-type;
            }
            default "vpn-common:static-rp";
            description
              "Type of RP discovery used.";
          }
          container bsr-candidates {
            when "derived-from-or-self(../rp-discovery-type, "
               + "'vpn-common:bsr-rp')" {
              description
                "Only applicable if discovery type
```

```
                     is BSR-RP.";
                }
                leaf-list bsr-candidate-address {
                  type inet:ip-address;
                  description
                    "Address of candidate Bootstrap Router
                    (BSR).";
                }
                description
                  "Container for List of Customer
                   BSR candidate’s addresses.";
              }
              description
                "RP discovery parameters.";
            }
            description
              "RP parameters.";
          }
          container msdp {
            if-feature "msdp";
            leaf enabled {
              type boolean;
              default "false";
              description
                "If true, MSDP is activated.";
            }
            leaf peer {
              type inet:ip-address;
              description
                "IP address of the MSDP peer.";
            }
            leaf local-address {
              type inet:ip-address;
              description
                "IP address of the local end. This local
                 address must be configured on the
                 node.";
            }
            description
              "MSDP parameters.";
          }
          description
            "Multicast global parameters for the VPN
             service.";
        }
        description
          "List for VPN node.";
      }
```

```
        }
        description
          "List of VPN services.";
      }
      description
        "Top-level container for the VPN services.";
    }
    description
      "Main container for L3VPN services management.";
  }
}
<CODE ENDS>
```

Figure 25

9. IANA Considerations

This document requests IANA to register the following URI in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

```
      URI: urn:ietf:params:xml:ns:yang:ietf-l3vpn-ntw
      Registrant Contact: The IESG.
      XML: N/A; the requested URI is an XML namespace.
```

This document requests IANA to register the following YANG module in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

```
      name: ietf-l3vpn-ntw
      namespace: urn:ietf:params:xml:ns:yang:ietf-l3vpn-ntw
      maintained by IANA: N
      prefix: l3nm
      reference: RFC XXXX
```

10. Security Considerations

The YANG module specified in this document defines 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 [RFC8466].

The Network Configuration Access Control Model (NACM) [RFC8341] 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.

The "ietf-l3vpn-ntw" module is used to manage Layer 3 VPNs in a service provider backbone network. Hence, the module can be used to request, modify, or retrieve L3VPN services. For example, the creation of a 'vpn-service' leaf instance triggers the creation of an L3VPN Service in a service provider network.

Due to the foreseen use of the "ietf-l3vpn-ntw" module, there are a number of data nodes defined in the 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) and delete operations to these data nodes without proper protection or authentication can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/ vulnerability in the "ietf-l3vpn-ntw" module:

o 'vpn-service': An attacker who is able to access network nodes can undertake various attacks, such as deleting a running L3VPN Service, interrupting all the traffic of a client. In addition, an attacker may modify the attributes of a running service (e.g., QoS, bandwidth, routing protocols), leading to malfunctioning of the service and therefore to SLA violations. In addition, an attacker could attempt to create a L3VPN Service or adding a new network access. Such activity can be detected by adequately monitoring and tracking network configuration changes.

Some of the readable data nodes in the "ietf-l3vpn-ntw" 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:

o 'customer-name' and 'ip-connection': An attacker can retrieve privacy-related information which can be used to track a customer. Disclosing such information may be considered as a violation of the customer-provider trust relationship.

The following summarizes the foreseen risks of using the "ietf-l3vpn-ntw" module can be classified into:

o Malicious clients attempting to delete or modify VPN services.

o Unauthorized clients attempting to create/modify/delete a VPN service.

o Unauthorized clients attempting to read VPN service related information.

11. Acknowledgements

Thanks to Adrian Farrel and Miguel Cros for the suggestions on the document. Thanks to Philip Eardlay for the review. Lots of thanks for the discussions on opsawg mailing list and at IETF meeting.

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

12. Contributors

Victor Lopez
Telefonica
Email: victor.lopezalvarez@telefonica.com

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

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

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

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

Stephane Litkowski
Cisco
Email: slitkows@cisco.com>

Manuel Julian
Vodafone
Email: manuel-julian.lopez@vodafone.com>

Lucia Oliva Ballega
Telefonica
Email: lucia.olivaballega.ext@telefonica.com>

Erez Segev
ECI Telecom
Email: erez.segev@ecitele.com>

13. References

13.1. Normative References

[I-D.ietf-opsawg-vpn-common]
barguil, s., Dios, O., Boucadair, M., and Q. WU, "A Layer 2/3 VPN Common YANG Model", draft-ietf-opsawg-vpn-common-01 (work in progress), September 2020.

[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>.

[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>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[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>.

[RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/ BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February 2012, <https://www.rfc-editor.org/info/rfc6513>.

[RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012, <https://www.rfc-editor.org/info/rfc6514>.

[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>.

[RFC7988] Rosen, E., Ed., Subramanian, K., and Z. Zhang, "Ingress Replication Tunnels in Multicast VPN", RFC 7988, DOI 10.17487/RFC7988, October 2016, <https://www.rfc-editor.org/info/rfc7988>.

[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>.

[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>.

[RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October 2018, <https://www.rfc-editor.org/info/rfc8466>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

13.2. Informative References

[I-D.evenwu-opsawg-yang-composed-vpn] Even, R., Bo, W., Wu, Q., and Y. Cheng, "YANG Data Model for Composed VPN Service Delivery", draft-evenwu-opsawg-yang-composed-vpn-03 (work in progress), March 2019.

[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-09 (work in progress), June 2020.

[I-D.ietf-pim-yang]
Liu, X., McAllister, P., Peter, A., Sivakumar, M., Liu, Y., and f. hu, "A YANG Data Model for Protocol Independent Multicast (PIM)", draft-ietf-pim-yang-17 (work in progress), May 2018.

[I-D.ietf-rtgwg-qos-model]
Choudhary, A., Jethanandani, M., Strahle, N., Aries, E., and I. Chen, "YANG Model for QoS", draft-ietf-rtgwg-qos-model-02 (work in progress), July 2020.

[RFC3618] Fenner, B., Ed. and D. Meyer, Ed., "Multicast Source Discovery Protocol (MSDP)", RFC 3618, DOI 10.17487/RFC3618, October 2003, <https://www.rfc-editor.org/info/rfc3618>.

[RFC3644] Snir, Y., Ramberg, Y., Strassner, J., Cohen, R., and B. Moore, "Policy Quality of Service (QoS) Information Model", RFC 3644, DOI 10.17487/RFC3644, November 2003, <https://www.rfc-editor.org/info/rfc3644>.

[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>.

[RFC4176] El Mghazli, Y., Ed., Nadeau, T., Boucadair, M., Chan, K., and A. Gonguet, "Framework for Layer 3 Virtual Private Networks (L3VPN) Operations and Management", RFC 4176, DOI 10.17487/RFC4176, October 2005, <https://www.rfc-editor.org/info/rfc4176>.

[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>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC6037] Rosen, E., Ed., Cai, Y., Ed., and IJ. Wijnands, "Cisco Systems' Solution for Multicast in BGP/MPLS IP VPNs", RFC 6037, DOI 10.17487/RFC6037, October 2010, <https://www.rfc-editor.org/info/rfc6037>.

[RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <https://www.rfc-editor.org/info/rfc6982>.

[RFC7527] Asati, R., Singh, H., Beebee, W., Pignataro, C., Dart, E., and W. George, "Enhanced Duplicate Address Detection", RFC 7527, DOI 10.17487/RFC7527, April 2015, <https://www.rfc-editor.org/info/rfc7527>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[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>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/info/rfc8345>.

[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>.

[RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for Abstraction and Control of TE Networks (ACTN)", RFC 8453, DOI 10.17487/RFC8453, August 2018, <https://www.rfc-editor.org/info/rfc8453>.

[RFC8512] Boucadair, M., Ed., Sivakumar, S., Jacquenet, C., Vinapamula, S., and Q. Wu, "A YANG Module for Network Address Translation (NAT) and Network Prefix Translation (NPT)", RFC 8512, DOI 10.17487/RFC8512, January 2019, <https://www.rfc-editor.org/info/rfc8512>.

Appendix A. L3VPN Examples

A.1. 4G VPN Provisioning Example

L3VPNs are widely used to deploy 3G/4G, fixed, and enterprise services mainly because several traffic discrimination policies can be applied within the network to deliver to the mobile customers a service that meets the SLA requirements.

As it is shown in the Figure 26, typically, an eNodeB (CE) is directly connected to the access routers of the mobile backhaul and their logical interfaces (one or many according to the Service type) are configured in a VPN that transports the packets to the mobile core platforms. In this example, a 'vpn-node' is created with two 'vpn-network-accesses'.

```
+-------------+                 +------------------+
|             |                 | PE               |
|             | 192.0.2.2       |  10.0.0.1        |
|   eNodeB    |>--------/------->|...........       |
|             |         Vlan 1  |          |       |
|             |>--------/------->|......    |       |
|             |         Vlan 2  |     |    |       |
|             | Direct          |  +-------------+ |
+-------------+ Routing         |  | vpn-node-id | |
                                |  | 44          | |
                                |  +-------------+ |
                                |                  |
                                +------------------+
```

Figure 26: Mobile Backhaul Example

To create a L3VPN service using the L3NM model, the following sample steps can be followed:

First: Create the 4G VPN Service (Figure 27).

```
POST: /restconf/data/ietf-l3vpn-ntw:l3vpn-ntw/vpn-services
Host: example.com
Content-Type: application/yang-data+json

{
  "ietf-l3vpn-ntw:vpn-services": {
    "vpn-service": [
      {
        "vpn-id": "4G",
        "customer-name": "mycustomer",
        "vpn-service-topology": "custom",
        "description": "VPN to deploy 4G services"
      }
    ]
  }
}
```

Figure 27: Create VPN Service

Second: Create a VPN Node as depicted in Figure 28. In this type of service, the VPN Node is equivalent to the VRF configured in the physical device ('ne-id'=10.0.0.1).

```
POST: /restconf/data/ietf-l3vpn-ntw:l3vpn-ntw/\
      vpn-services/vpn-service=4G
Host: example.com
Content-Type: application/yang-data+json
```


```
{
  "ietf-l3vpn-ntw:vpn-nodes": {
    "vpn-node": [
      {
        "vpn-node-id": "44",
        "ne-id": "10.0.0.1",
        "local-autonomous-system": "65550",
        "rd": "0:65550:1",
        "vpn-targets": {
          "vpn-target": [
            {
              "id": "1",
              "route-targets": [
                "0:65550:1"
              ],
              "route-target-type": "both"
            }
          ]
        }
      }
    ]
  }
}
```


Figure 28: Create VPN Node

Finally, two VPN Network Accesses are created using the same physical port ('port-id'=1/1/1). Each 'vpn-network-access' has a particular VLAN (1,2) to differentiate the traffic between: Sync and data (Figure 29).

```
POST: /restconf/data/ietf-l3vpn-ntw:l3vpn-ntw/\
      vpn-services/vpn-service=4G/vpn-nodes/vpn-node=44
content-type: application/yang-data+json
```


```
{
  "ietf-l3vpn-ntw:vpn-network-accesses": {
    "vpn-network-access": [
      {
        "vpn-network-access-id": "1/1/1.1",
        "port-id": "1/1/1",
        "description": "Interface SYNC to eNODE-B",
        "status": {
```

```
            "admin-enabled": "true"
          },
          "vpn-network-access-type": "vpn-common:point-to-point",
          "ip-connection": {
            "ipv4": {
              "address-allocation-type": "static-address",
              "static-addresses": {
                "primary-address": "1",
                "address": [
                  {
                    "address-id": "1",
                    "s-provider-address": "192.0.2.1",
                    "s-customer-address": "192.0.2.1",
                    "s-prefix-length": 32
                  }
                ]
              }
            }
          },
          "routing-protocols": {
            "routing-protocol": [
              {
                "id": "1",
                "type": "vpn-common:direct"
              }
            ]
          }
        },
        {
          "vpn-network-access-id": "1/1/1.2",
          "port-id": "1/1/1",
          "description": "Interface DATA to eNODE-B",
          "status": {
            "admin-enabled": "true"
          },
          "ip-connection": {
            "ipv4": {
              "address-allocation-type": "static-address",
              "static-addresses": {
                "primary-address": "1",
                "address": [
                  {
                    "address-id": "1",
                    "s-provider-address": "192.0.2.1",
                    "s-customer-address": "192.0.2.2",
                    "s-prefix-length": 32
                  }
                ]
```

```
              }
            }
          },
          "routing-protocols": {
            "routing-protocol": [
              {
                "id": "1",
                "type": "vpn-common:direct"
              }
            ]
          }
        }
      ]
    }
  }
```


Figure 29: Create VPN Network Access

A.2. Multicast VPN Provisioning Example

IPTV is mainly distributed through multicast over the LANs. In the following example, PIM-SM is enabled and functional between the PE and the CE. The PE receives multicast traffic from a CE that is directly connected to the multicast source. The signaling between PE and CE is achieved using BGP. Also, RP is statically configured for a multicast group.

```
+-----------+   +-------+     +-------+    +-----------+
| Multicast |---|  CE   |--/--|  PE   |----|  Backbone |
|  source   |   +-------+     +-------+    |   IP/MPLS |
+-----------+                              +-----------+
```

Figure 30: Multicast L3VPN Service Example

To configure a Multicast L3VPN service using the L3NM model the procedure and the JSON with the data structure is the following:

First, the multicast service is created (see the excerpt of the request message body shown in Figure 31)

```
{
  "ietf-l3vpn-ntw:vpn-services": {
    "vpn-service": [
      {
        "vpn-id": "Multicast-IPTV",
        "customer-name": "310",
        "vpn-service-topology": "vpn-common:hub-spoke",
        "description": "Multicast IPTV VPN service"
      }
    ]
  }
}
```


Figure 31: Create Multicast VPN Service (Excerpt of the Message Request Body)

Then, the VPN nodes are created (see the excerpt of the request message body shown in Figure 32). In this example, the VPN Node will represent VRF configured in the physical device.

```
{
  "ietf-l3vpn-ntw:vpn-node": [
    {
      "vpn-node-id": "500003105",
      "ne-id": "10.250.2.202",
      "autonomous-system": "3816",
      "description": "VRF_IPTV_MULTICAST",
      "router-id": "10.250.2.202",
      "address-family": "ipv4",
      "node-role": "vpn-common:hub-role",
      "rd": "3816:31050202",
      "multicast": {
        "enabled": "true",
        "rp": {
          "rp-group-mappings": {
            "rp-group-mapping": [
              {
                "id": "1",
                "rp-address": "172.19.48.17",
                "groups": {
                  "group": [
                    {
                      "id": "1",
                      "group-address": "239.130.0.0/15"
                    }
                  ]
                }
              }
            ]
          },
          "rp-discovery": {
            "rp-discovery-type": "vpn-common:static-rp"
          }
        }
      }
    }
  ]
}
```


Figure 32: Create Multicast VPN Node (Excerpt of the Message Request Body)

Finally, create the VPN Network Access with multicast enabled (see the excerpt of the request message body shown in Figure 33).


```
 {
  "ietf-l3vpn-ntw:vpn-network-access": {
    "vpn-network-access-id": "1/1/1",
```

```
       "description": "Connected_to_source",
       "status": {
         "admin-enabled": "true"
       },
       "vpn-network-access-type": "vpn-common:point-to-point",
       "ip-connection": {
         "ipv4": {
           "address-allocation-type": "static-address",
           "static-addresses": {
             "primary-address": "1",
             "address": [
               {
                 "address-id": "1",
                 "s-provider-address": "172.19.48.1",
                 "s-prefix-length": 30
               }
             ]
           }
         }
       },
       "routing-protocols": {
         "routing-protocol": [
           {
             "id": "1",
             "type": "vpn-common:bgp",
             "bgp": {
               "peer-autonomous-system": "6500",
               "local-autonomous-system": "3816",
               "address-family": "ipv4",
               "neighbor": "172.19.48.2",
               "description": "Connected to CE"
             }
           }
         ]
       },
       "service": {
         "multicast": {
           "multicast-site-type": "source-only",
           "multicast-address-family": {
             "ipv4": "true"
           },
           "protocol-type": "router"
         }
       }
     }
   }
```

Figure 33: Create VPN Network Access (Excerpt of the Message Request Body)

Appendix B. Implementation Status

This section records the status of known implementations of the Yang module defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

Note the RFC Editor: As per [RFC6982] guidelines, please remove this Implementation Status apendix prior publication.

B.1. Nokia Implementation

Details can be found at: https://github.com/IETF-OPSAWG-WG/l3nm/blob/master/Implementattion/Nokia.txt

B.2. Huawei Implementation

Details can be found at: https://github.com/IETF-OPSAWG-WG/l3nm/blob/master/Implementattion/Huawei.txt

B.3. Infinera Implementation

Details can be found at: https://github.com/IETF-OPSAWG-WG/l3nm/blob/master/Implementattion/Infinera.txt

B.4. Ribbon-ECI Implementation

Details can be found at: https://github.com/IETF-OPSAWG-WG/l3nm/blob/master/Implementattion/Ribbon-ECI.txt

Authors' Addresses

Samier Barguil
Telefonica
Madrid
ES

Email: samier.barguilgiraldo.ext@telefonica.com

Oscar Gonzalez de Dios (editor)
Telefonica
Madrid
ES

Email: oscar.gonzalezdedios@telefonica.com

Mohamed Boucadair (editor)
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Luis Angel Munoz
Vodafone
ES

Email: luis-angel.munoz@vodafone.com

Alejandro Aguado
Nokia
Madrid
ES

Email: alejandro.aguado_martin@nokia.com

OPSAWG WG T. Reddy
Internet-Draft McAfee
Intended status: Standards Track D. Wing
Expires: July 19, 2021 Citrix
B. Anderson
Cisco
January 15, 2021

Manufacturer Usage Description (MUD) (D)TLS Profiles for IoT Devices
draft-ietf-opsawg-mud-tls-04

Abstract

This memo extends the Manufacturer Usage Description (MUD) specification to incorporate (D)TLS profile parameters. This allows a network security service to identify unexpected (D)TLS usage, which can indicate the presence of unauthorized software or malware on an endpoint.

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 July 19, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

Encryption is necessary to enhance the privacy of end users using IoT devices. TLS [RFC8446] and DTLS [I-D.ietf-tls-dtls13] are the dominant protocols (counting all (D)TLS versions) providing encryption for IoT device traffic. Unfortunately, in conjunction with IoT applications' rise of encryption, malware authors are also using encryption which thwarts network-based analysis such as deep packet inspection (DPI). Other mechanisms are thus needed to help detecting malware running on an IoT device.

Malware frequently uses proprietary libraries for its activities, and those libraries are reused much like any other software engineering project. [malware] indicates that there are observable differences in how malware uses encryption compared with how non-malware uses encryption. There are several interesting findings specific to (D)TLS which were found common to malware:

- o Older and weaker cryptographic parameters (e.g., TLS_RSA_WITH_RC4_128_SHA).

- o TLS server name indication (SNI) extension [RFC6066] and server certificates are composed of subjects with characteristics of a domain generation algorithm (DGA) (e.g., 'www.33mhwt2j.net').

- o Higher use of self-signed certificates compared with typical legitimate software.

- o Discrepancies in the SNI TLS extension and the DNS names in the SubjectAltName (SAN) X.509 extension in the server certificate message.

- o Discrepancies in the key exchange algorithm and the client public key length in comparison with legitimate flows. As a reminder, the Client Key Exchange message has been removed from TLS 1.3.

- o Lower diversity in TLS client advertised extensions compared to legitimate clients.

- o Using privacy enhancing technologies like Tor, Psiphon, Ultrasurf (see [malware-tls]), and evasion techniques such as ClientHello randomization.

- o Using DNS-over-HTTPS (DoH) [RFC8484] to avoid detection by malware DNS filtering services [malware-doh]. Specifically, malware may not use the DoH server provided by the local network.

If observable (D)TLS profile parameters are used, the following functions are possible which have a positive impact on the local network security:

- o Permit intended DTLS or TLS use and block malicious DTLS or TLS use. This is superior to the layers 3 and 4 ACLs of Manufacturer Usage Description Specification (MUD) [RFC8520] which are not suitable for broad communication patterns.

- o Ensure TLS certificates are valid. Several TLS deployments have been vulnerable to active Man-In-The-Middle (MITM) attacks because of the lack of certificate validation or vulnerability in the certificate validation function (see [cryto-vulnerability]). By observing (D)TLS profile parameters, a network element can detect when the TLS SNI mismatches the SubjectAltName and when the server's certificate is invalid. In TLS 1.2, the ClientHello, ServerHello and Certificate messages are all sent in clear-text. This check is not possible with TLS 1.3, which encrypts the Certificate message thereby hiding the server identity from any intermediary. In TLS 1.3, the server certificate validation functions should be executed within an on-path TLS proxy, if such a proxy exists.

- Support new communication patterns. An IoT device can learn a new capability, and the new capability can change the way the IoT device communicates with other devices located in the local network and Internet. There would be an inaccurate policy if an IoT device rapidly changes the IP addresses and domain names it communicates with while the MUD ACLs were slower to update (see [clear-as-mud]). In such a case, observable (D)TLS profile parameters can be used to permit intended use and to block malicious behavior from the IoT device.

The YANG module specified in Section 5 of this document is an extension of YANG Data Model for Network Access Control Lists (ACLs) [RFC8519] to enhance MUD [RFC8520] to model observable (D)TLS profile parameters. Using these (D)TLS profile parameters, an active MUD-enforcing network security service (e.g., firewall) can identify MUD non-compliant (D)TLS behavior indicating outdated cryptography or malware. This detection can prevent malware downloads, block access to malicious domains, enforce use of strong ciphers, stop data exfiltration, etc. In addition, organizations may have policies around acceptable ciphers and certificates for the websites the IoT devices connect to. Examples include no use of old and less secure versions of TLS, no use of self-signed certificates, deny-list or accept-list of Certificate Authorities, valid certificate expiration time, etc. These policies can be enforced by observing the (D)TLS profile parameters. Network security services can use the IoT device's (D)TLS profile parameters to identify legitimate flows by observing (D)TLS sessions, and can make inferences to permit legitimate flows and to block malicious or insecure flows. The proposed technique is also suitable in deployments where decryption techniques are not ideal due to privacy concerns, non-cooperating end-points, and expense.

2. Terminology

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

"(D)TLS" is used for statements that apply to both Transport Layer Security [RFC8446] and Datagram Transport Layer Security [RFC6347]. Specific terms are used for any statement that applies to either protocol alone.

'DoH/DoT' refers to DNS-over-HTTPS and/or DNS-over-TLS.

3. Overview of MUD (D)TLS profiles for IoT devices

In Enterprise networks, protection and detection are typically done both on end hosts and in the network. Host security agents have deep visibility on the devices where they are installed, whereas the network has broader visibility. Installing host security agents may not be a viable option on IoT devices, and network-based security is an efficient means to protect such IoT devices. If the IoT device supports a MUD (D)TLS profile, the (D)TLS profile parameters of the IoT device can be used by a middlebox to detect and block malware communication, while at the same time preserving the privacy of legitimate uses of encryption. The middlebox need not proxy (D)TLS but can passively observe the parameters of (D)TLS handshakes from IoT devices and gain visibility into TLS 1.2 parameters and partial visibility into TLS 1.3 parameters.

Malicious agents can try to use the (D)TLS profile parameters of legitimate agents to evade detection, but it becomes a challenge to mimic the behavior of various IoT device types and IoT device models from several manufacturers. In other words, malware developers will have to develop malicious agents per IoT device type, manufacturer and model, infect the device with the tailored malware agent and will have keep up with updates to the device's (D)TLS profile parameters over time. Furthermore, the malware's command and control server certificates need to be signed by the same certifying authorities trusted by the IoT devices. Typically, IoT devices have an infrastructure that supports a rapid deployment of updates, and malware agents will have a near-impossible task of similarly deploying updates and continuing to mimic the TLS behavior of the IoT device it has infected. However, if the IoT device has reached end-of-life and the IoT manufacturer will not issue a firmware or software update to the Thing or will not update the MUD file, the "is-supported" attribute defined in Section 3.6 of [RFC8520] can be used by the MUD manager to identify the IoT manufacturer no longer supports the device.

The end-of-life of a device does not necessarily mean that it is defective; rather, it denotes a need to replace and upgrade the network to next-generation devices for additional functionality. The network security service will have to rely on other techniques discussed in Section 8 to identify malicious connections until the device is replaced.

Compromised IoT devices are typically used for launching DDoS attacks (Section 3 of [RFC8576]). For example, DDoS attacks like Slowloris and Transport Layer Security (TLS) re-negotiation can be blocked if the victim's server certificate is not be signed by the same certifying authorities trusted by the IoT device.

4. (D)TLS 1.3 Handshake

In (D)TLS 1.3, full (D)TLS handshake inspection is not possible since all (D)TLS handshake messages excluding the ClientHello message are encrypted. (D)TLS 1.3 has introduced new extensions in the handshake record layers called Encrypted Extensions. Using these extensions handshake messages will be encrypted and network security services (such as a firewall) are incapable to decipher the handshake, and thus cannot view the server certificate. However, the ClientHello and ServerHello still have some fields visible, such as the list of supported versions, named groups, cipher suites, signature algorithms and extensions in ClientHello, and chosen cipher in the ServerHello. For instance, if the malware uses evasion techniques like ClientHello randomization, the observable list of cipher suites and extensions offered by the malware agent in the ClientHello message will not match the list of cipher suites and extensions offered by the legitimate client in the ClientHello message, and the middlebox can block malicious flows without acting as a (D)TLS 1.3 proxy.

4.1. Full (D)TLS 1.3 Handshake Inspection

To obtain more visibility into negotiated TLS 1.3 parameters, a middlebox can act as a (D)TLS 1.3 proxy. A middlebox can act as a (D)TLS proxy for the IoT devices owned and managed by the IT team in the Enterprise network and the (D)TLS proxy must meet the security and privacy requirements of the organization. In other words, the scope of middlebox acting as a (D)TLS proxy is restricted to Enterprise network owning and managing the IoT devices. The middlebox would have to follow the behaviour detailed in Section 9.3 of [RFC8446] to act as a compliant (D)TLS 1.3 proxy.

To further increase privacy, Encrypted Client Hello (ECH) extension [I-D.ietf-tls-esni] prevents passive observation of the TLS Server Name Indication extension and other potentially sensitive fields, such as the ALPN [RFC7301]. To effectively provide that privacy protection, ECH extension needs to be used in conjunction with DNS encryption (e.g., DoH). A middlebox (e.g., firewall) passively inspecting ECH extension cannot observe the encrypted SNI nor observe the encrypted DNS traffic.

4.2. Encrypted DNS

A common usage pattern for certain type of IoT devices (e.g., light bulb) is for it to "call home" to a service that resides on the public Internet, where that service is referenced through a domain name (A or AAAA record). As discussed in Manufacturer Usage Description Specification [RFC8520], because these devices tend to require access to very few sites, all other access should be considered suspect. If an IoT device is pre-configured to use a public DoH/DoT server, the MUD policy enforcement point is moved to that public server, which cannot enforce the MUD policy based on domain names (Section 8 of [RFC8520]). If the DNS query is not accessible for inspection, it becomes quite difficult for the infrastructure to suspect anything. Thus the use of a public DoH/DoT server is incompatible with MUD in general. A local DoH/DoT server is necessary to allow MUD policy enforcement on the local network [I-D.reddy-add-enterprise].

5. (D)TLS Profile of a IoT device

This document specifies a YANG module for representing (D)TLS profile. The (D)TLS profile YANG module provides a method for network security services to observe the (D)TLS profile parameters in the (D)TLS handshake to permit intended use and to block malicious behavior. This module uses the cryptographic types defined in [I-D.ietf-netconf-crypto-types]. See [RFC7925] for (D)TLS 1.2 and [I-D.ietf-uta-tls13-iot-profile] for DTLS 1.3 recommendations related to IoT devices, and [RFC7525] for additional (D)TLS 1.2 recommendations.

A companion YANG module is defined to include a collection of (D)TLS parameters and (D)TLS versions maintained by IANA: "iana-tls-profile" (Section 5.3).

The (D)TLS parameters in each (D)TLS profile include the following:

- Profile name

- (D)TLS versions supported by the IoT device.

- List of supported cipher suites. For (D)TLS1.2, [RFC7925] recommends AEAD ciphers for IoT devices.

- List of supported extension types

- List of trust anchor certificates used by the IoT device. If the server certificate is signed by one of the trust anchors, the middlebox continues with the connection as normal. Otherwise, the middlebox will react as if the server certificate validation has failed and takes appropriate action (e.g, block the (D)TLS session). An IoT device can use a private trust anchor to validate a server's certificate (e.g., the private trust anchor can be preloaded at manufacturing time on the IoT device and the IoT device fetches the firmware image from the Firmware server whose certificate is signed by the private CA). This empowers the middlebox to reject TLS sessions to servers that the IoT device does not trust.

- List of SPKI pin set pre-configured on the client to validate self-signed server certificates or raw public keys. A SPKI pin set is a cryptographic digest to "pin" public key information in a manner similar to HTTP Public Key Pinning (HPKP) [RFC7469]. If SPKI pin set is present in the (D)TLS profile of a IoT device and the server certificate does not pass the PKIX certification path validation, the middlebox computes the SPKI Fingerprint for the public key found in the server's certificate (or in the raw public key, if the server provides that instead). If a computed fingerprint exactly matches one of the SPKI pin sets in the (D)TLS profile, the middlebox continues with the connection as normal. Otherwise, the middlebox will act on the SPKI validation failure and takes appropriate action.

- Cryptographic hash algorithm used to generate the SPKI pinsets

- List of pre-shared key exchange modes

- List of named groups (DHE or ECDHE) supported by the client

- List of signature algorithms the client can validate in X.509 server certificates

- List of signature algorithms the client is willing to accept for CertificateVerify message (Section 4.2.3 of [RFC8446]). For example, a TLS client implementation can support different sets of algorithms for certificates and in TLS to signal the capabilities in "signature_algorithms_cert" and "signature_algorithms" extensions.

- List of supported application protocols (e.g., h3, h2, http/1.1 etc.)

- List of certificate compression algorithms (defined in [I-D.ietf-tls-certificate-compression])

- List of the distinguished names [X501] of acceptable certificate authorities, represented in DER-encoded format [X690] (defined in Section 4.2.4 of [RFC8446])

GREASE [RFC8701] sends random values on TLS parameters to ensure future extensibility of TLS extensions. Similar random values might be extended to other TLS parameters. Thus, the (D)TLS profile parameters defined in the YANG module by this document MUST NOT include the GREASE values for extension types, named groups, signature algorithms, (D)TLS versions, pre-shared key exchange modes, cipher suites and for any other TLS parameters defined in future RFCs.

The (D)TLS profile does not include parameters like compression methods for data compression, [RFC7525] recommends disabling TLS-level compression to prevent compression-related attacks. In TLS 1.3, only the "null" compression method is allowed (Section 4.1.2 of [RFC8446]).

5.1. Tree Structure of the (D)TLS profile Extension to the ACL YANG Model

This document augments the "ietf-acl" ACL YANG module defined in [RFC8519] for signaling the IoT device (D)TLS profile. This document defines the YANG module "ietf-acl-tls", which has the following tree structure:

```
   module: ietf-acl-tls
     augment /acl:acls/acl:acl/acl:aces/acl:ace/acl:matches:
       +--rw client-profile {match-on-tls-dtls}?
          +--rw client-profile
             +--rw tls-dtls-profiles* [profile-name]
                +--rw profile-name                   string
                +--rw supported-tls-versions*        ianatp:tls-version
                +--rw supported-dtls-versions*       ianatp:dtls-version
                +--rw cipher-suites* [cipher hash]
                |  +--rw cipher    ianatp:cipher-algorithm
                |  +--rw hash      ianatp:hash-algorithm
                +--rw extension-types*
                |       ianatp:extension-type
                +--rw acceptlist-ta-certs*
                |       ct:trust-anchor-cert-cms
                +--rw spki
                |  +--rw spki-pin-sets*         ianatp:spki-pin-set
                |  +--rw spki-hash-algorithm?   iha:hash-algorithm-type
                +--rw psk-key-exchange-modes*
                |       ianatp:psk-key-exchange-mode
                |       {tls-1-3 or dtls-1-3}?
                +--rw supported-groups*
                |       ianatp:supported-group
                +--rw signature-algorithms-cert*
                |       ianatp:signature-algorithm
                |       {tls-1-3 or dtls-1-3}?
                +--rw signature-algorithms*
                |       ianatp:signature-algorithm
                +--rw application-protocols*
                |       ianatp:application-protocol
                +--rw cert-compression-algorithms*
                |       ianatp:cert-compression-algorithm
                |       {tls-1-3 or dtls-1-3}?
                +--rw certificate-authorities*
                        ianatp:certificate-authority
                        {tls-1-3 or dtls-1-3}?
```

5.2. The (D)TLS profile Extension to the ACL YANG Model

```
<CODE BEGINS> file "ietf-acl-tls@2020-10-07.yang"
module ietf-acl-tls {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-acl-tls";
  prefix ietf-acl-tls;

  import iana-tls-profile {
    prefix ianatp;
```

```
    reference
      "RFC XXXX: Manufacturer Usage Description (MUD) (D)TLS
                 Profiles for IoT Devices";
  }
  import ietf-crypto-types {
    prefix ct;
    reference
      "RFC CCCC: Common YANG Data Types for Cryptography";
  }
  import iana-hash-algs {
    prefix iha;
    reference
      "RFC IIII: Common YANG Data Types for
       Hash algorithms";
  }
  import ietf-access-control-list {
    prefix acl;
    reference
      "RFC 8519: YANG Data Model for Network Access
                 Control Lists (ACLs)";
  }

  organization
    "IETF OPSAWG (Operations and Management Area Working Group)";
  contact
    "WG Web: <https://datatracker.ietf.org/wg/opsawg/>
     WG List: opsawg@ietf.org

      Author: Konda, Tirumaleswar Reddy
                  TirumaleswarReddy_Konda@McAfee.com
    ";
  description
    "This YANG module defines a component that augments the
      IETF description of an access list to allow (D)TLS profile
      as matching criteria.

     Copyright (c) 2020 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 2020-11-02 {
    description
      "Initial revision";
    reference
      "RFC XXXX: Manufacturer Usage Description (MUD) (D)TLS
                 Profiles for IoT Devices";
  }

  feature tls-1-2 {
    description
      "TLS Protocol Version 1.2 is supported.";
    reference
      "RFC 5246: The Transport Layer Security (TLS) Protocol
                 Version 1.2";
  }

  feature tls-1-3 {
    description
      "TLS Protocol Version 1.3 is supported.";
    reference
      "RFC 8446: The Transport Layer Security (TLS) Protocol
                 Version 1.3";
  }

  feature dtls-1-2 {
    description
      "DTLS Protocol Version 1.2 is supported.";
    reference
      "RFC 6346: Datagram Transport Layer Security
                 Version 1.2";
  }

  feature dtls-1-3 {
    description
      "DTLS Protocol Version 1.3 is supported.";
    reference
      "draft-ietf-tls-dtls13: Datagram Transport Layer
                Security 1.3";
  }

  feature match-on-tls-dtls {
    description
      "The networking device can support matching on
       (D)TLS parameters.";
  }

  augment "/acl:acls/acl:acl/acl:aces/acl:ace/acl:matches" {
    if-feature "match-on-tls-dtls";
```

```
    description
      "(D)TLS specific matches.";
    container client-profile {
      description
        "A grouping for (D)TLS profiles.";
      container client-profile {
        description
          "A grouping for DTLS profiles.";
        list tls-dtls-profiles {
          key "profile-name";
          description
            "A list of (D)TLS version profiles supported by
             the client.";
          leaf profile-name {
            type string {
              length "1..64";
            }
            description
              "The name of (D)TLS profile; space and special
               characters are not allowed.";
          }
          leaf-list supported-tls-versions {
            type ianatp:tls-version;
            description
              "TLS versions supported by the client.";
          }
          leaf-list supported-dtls-versions {
            type ianatp:dtls-version;
            description
              "DTLS versions supported by the client.";
          }
          list cipher-suites {
            key "cipher hash";
            leaf cipher {
              type ianatp:cipher-algorithm;
              description
                "AEAD encryption algorithm as defined in RFC5116.";
            }
            leaf hash {
              type ianatp:hash-algorithm;
              description
                "Hash algorithm used with HKDF as defined in RFC5869.";
            }
            description
              "A list of Cipher Suites supported by the client.";
          }
          leaf-list extension-types {
            type ianatp:extension-type;
```

```
            description
              "A list of Extension Types supported by the client.";
          }
          leaf-list acceptlist-ta-certs {
            type ct:trust-anchor-cert-cms;
            description
              "A list of trust anchor certificates used by the client.";
          }
          container spki {
            description
              "A grouping for spki.";
            leaf-list spki-pin-sets {
              type ianatp:spki-pin-set;
              description
                "A list of SPKI pin sets pre-configured on the client
                 to validate self-signed server certificate or
                 raw public key.";
            }
            leaf spki-hash-algorithm {
              type iha:hash-algorithm-type;
              description
                "cryptographic hash algorithm used to generate the
                 SPKI pinset.";
            }
          }
          leaf-list psk-key-exchange-modes {
            if-feature "tls-1-3 or dtls-1-3";
            type ianatp:psk-key-exchange-mode;
            description
              "pre-shared key exchange modes.";
          }
          leaf-list supported-groups {
            type ianatp:supported-group;
            description
              "A list of named groups supported by the client.";
          }
          leaf-list signature-algorithms-cert {
            if-feature "tls-1-3 or dtls-1-3";
            type ianatp:signature-algorithm;
            description
              "A list signature algorithms the client can validate
               in X.509 certificates.";
          }
          leaf-list signature-algorithms {
            type ianatp:signature-algorithm;
            description
              "A list signature algorithms the client can validate
               in the CertificateVerify message.";
```

```
          }
          leaf-list application-protocols {
            type ianatp:application-protocol;
            description
              "A list application protocols supported by the client.";
          }
          leaf-list cert-compression-algorithms {
            if-feature "tls-1-3 or dtls-1-3";
            type ianatp:cert-compression-algorithm;
            description
              "A list certificate compression algorithms
               supported by the client.";
          }
          leaf-list certificate-authorities {
            if-feature "tls-1-3 or dtls-1-3";
            type ianatp:certificate-authority;
            description
              "A list of the distinguished names of certificate authorities
               acceptable to the client.";
          }
        }
      }
    }
  }
}
<CODE ENDS>
```

5.3. IANA (D)TLS profile YANG Module

The TLS and DTLS IANA registries are available from <https://www.iana.org/assignments/tls-parameters/tls-parameters.txt> and <https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.txt>.

The values for all the parameters in the "iana-tls-profile" YANG module are defined in the TLS and DTLS IANA registries excluding the tls-version, dtls-version, spki-pin-set, and certificate-authority parameters. The values of spki-pin-set and certificate-authority parameters will be specific to the IoT device.

The TLS and DTLS IANA registries do not maintain (D)TLS version numbers. In (D)TLS 1.2 and below, "legacy_version" field in the ClientHello message is used for version negotiation. However in (D)TLS 1.3, the "supported_versions" extension is used by the client to indicate which versions of (D)TLS it supports. TLS 1.3 ClientHello messages are identified as having a "legacy_version" of 0x0303 and a "supported_versions" extension present with 0x0304 as the highest version. DTLS 1.3 ClientHello messages are identified as having a "legacy_version" of 0xfefd and a "supported_versions" extension present with 0x0304 as the highest version.

In order to ease updating the "iana-tls-profile" YANG module with future (D)TLS versions, new (D)TLS version registries are defined in Section 10.2 and Section 10.3. Whenever a new (D)TLS protocol version is defined, the registry will be updated using expert review; the "iana-tls-profile" YANG module will be automatically updated by IANA.

The "iana-tls-profile" YANG module is defined as follows:

```
<CODE BEGINS> file "iana-tls-profile@2020-10-07.yang"

module iana-tls-profile {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:iana-tls-profile";
  prefix ianatp;

  organization
    "IANA";
  contact
    "        Internet Assigned Numbers Authority

     Postal: ICANN
             12025 Waterfront Drive, Suite 300
             Los Angeles, CA  90094-2536
             United States

     Tel:    +1 310 301 5800
     E-Mail: iana@iana.org>";
  description
    "This module contains YANG definition for the (D)TLS profile.

     Copyright (c) 2020 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 2020-11-02 {
```

```
       description
         "Initial revision";
       reference
         "RFC XXXX: Manufacturer Usage Description (MUD) (D)TLS Profiles
                    for IoT Devices";
     }

     typedef extension-type {
       type uint16;
       description
         "Extension type in the TLS ExtensionType Values registry as
          defined in Section 7 of RFC8447.";
     }

     typedef supported-group {
       type uint16;
       description
         "Supported Group in the TLS Supported Groups registry as
          defined in Section 9 of RFC8447.";
     }

     typedef spki-pin-set {
       type binary;
       description
         "Subject Public Key Info pin set as discussed in
          Section 2.4 of RFC7469.";
     }

     typedef signature-algorithm {
       type uint16;
       description
         "Signature algorithm in the TLS SignatureScheme registry as
          defined in Section 11 of RFC8446.";
     }

     typedef psk-key-exchange-mode {
       type uint8;
       description
         "Pre-shared key exchange mode in the TLS PskKeyExchangeMode
          registry as defined in Section 11 of RFC8446.";
     }

     typedef application-protocol {
       type string;
       description
         "Application-Layer Protocol Negotiation (ALPN) Protocol ID
          registry as defined in Section 6 of RFC7301.";
     }
```

```
     typedef cert-compression-algorithm {
       type uint16;
       description
         "Certificate compression algorithm in TLS Certificate
          Compression Algorithm IDs registry as defined in
          Section 7.3 of ietf-tls-certificate-compression";
     }

     typedef certificate-authority {
       type string;
       description
         "Distinguished Name of Certificate authority as discussed
          in Section 4.2.4 of RFC8446.";
     }

     typedef cipher-algorithm {
       type uint8;
       description
         "AEAD encryption algorithm in TLS Cipher Suites registry
          as discussed in Section 11 of RFC8446.";
     }

     typedef hash-algorithm {
       type uint8;
       description
         "Hash algorithm used with HMAC-based Extract-and-Expand Key
          Derivation Function (HKDF) in TLS Cipher Suites registry
          as discussed in Section 11 of RFC8446.";
     }

     typedef tls-version {
       type enumeration {
         enum tls-1.2 {
           value 1;
           description
             "TLS Protocol Version 1.2.

              TLS 1.2 ClientHello contains
              0x0303 in 'legacy_version'.";
           reference
             "RFC 5246: The Transport Layer Security (TLS) Protocol
                        Version 1.2";
         }
         enum tls-1.3 {
           value 2;
           description
             "TLS Protocol Version 1.3.
```

```
              TLS 1.3 ClientHello contains a
              supported_versions extension with 0x0304
              contained in its body and the ClientHello contains
              0x0303 in 'legacy_version'.";
           reference
             "RFC 8446: The Transport Layer Security (TLS) Protocol
                       Version 1.3";
         }
       }
       description
         "Indicates the TLS version.";
     }

     typedef dtls-version {
       type enumeration {
         enum dtls-1.2 {
           value 1;
           description
             "DTLS Protocol Version 1.2.

              DTLS 1.2 ClientHello contains
              0xfefd in 'legacy_version'.";
           reference
             "RFC 6346: Datagram Transport Layer Security 1.2";
         }
         enum dtls-1.3 {
           value 2;
           description
             "DTLS Protocol Version 1.3.

              DTLS 1.3 ClientHello contains a
              supported_versions extension with 0x0304
              contained in its body and the ClientHello contains
              0xfefd in 'legacy_version'.";
           reference
             "RFC DDDD: Datagram Transport Layer Security 1.3";
         }
       }
       description
         "Indicates the DTLS version.";
     }
   }
   <CODE ENDS>
```

5.4. MUD (D)TLS Profile Extension

This document augments the "ietf-mud" MUD YANG module to indicate whether the device supports (D)TLS profile. If the "ietf-mud-tls" extension is supported by the device, MUD file is assumed to implement the "match-on-tls-dtls" ACL model feature defined in this specification. Furthermore, only "accept" or "drop" actions SHOULD be included with the (D)TLS profile similar to the actions allowed in Section 2 of [RFC8520].

This document defines the YANG module "ietf-mud-tls", which has the following tree structure:

```
module: ietf-mud-tls
  augment /ietf-mud:mud:
    +--rw is-tls-dtls-profile-supported?   boolean
```

The model is defined as follows:

```
<CODE BEGINS> file "iana-tls-mud@2020-10-20.yang"

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

  import ietf-mud {
    prefix ietf-mud;
  }

  organization
    "IETF OPSAWG (Operations and Management Area Working Group)";
  contact
    "WG Web: <https://datatracker.ietf.org/wg/opsawg/>
     WG List: opsawg@ietf.org

     Author: Konda, Tirumaleswar Reddy
             TirumaleswarReddy_Konda@McAfee.com

    ";
   description
     "Extension to a MUD module to indicate (D)TLS
      profile support.

     Copyright (c) 2020 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 2020-10-19 {
        description
          "Initial revision.";
          reference
            "RFC XXXX: Manufacturer Usage Description (MUD) (D)TLS
             Profiles for IoT Devices";
      }

      augment "/ietf-mud:mud" {
        description
          "This adds a extension for a manufacturer
           to indicate whether (D)TLS profile is
           is supported by a device.";
        leaf is-tls-dtls-profile-supported {
          type boolean;
            description
              "This value will equal 'true' if a device supports
               (D)TLS profile.";
        }
      }
   }
   <CODE ENDS>
```

6. Processing of the MUD (D)TLS Profile

The following text outlines the rules for a network security service (e.g., firewall) to follow to process the MUD (D)TLS Profile:

- If the (D)TLS parameter observed in a (D)TLS session is not specified in the MUD (D)TLS profile and the parameter is recognized by the firewall, it can identify unexpected (D)TLS usage, which can indicate the presence of unauthorized software or malware on an endpoint. The firewall can take several actions like block the (D)TLS session or raise an alert to quarantine and remediate the compromised device. For example, if the cipher suite TLS_RSA_WITH_AES_128_CBC_SHA in the ClientHello message is not specified in the MUD (D)TLS profile and the cipher suite is recognized by the firewall, it can identify unexpected TLS usage.

- If the (D)TLS parameter observed in a (D)TLS session is not specified in the MUD (D)TLS profile and the (D)TLS parameter is not recognized by the firewall, it can ignore the unrecognized parameter and the correct behavior is not to block the (D)TLS session. The behaviour is functionally equivalent to the compliant TLS middlebox description in Section 9.3 of [RFC8446] to ignore all unrecognized cipher suites, extensions, and other parameters. For example, if the cipher suite TLS_CHACHA20_POLY1305_SHA256 in the ClientHello message is not specified in the MUD (D)TLS profile and the cipher suite is not recognized by the firewall, it can ignore the unrecognized cipher suite.

- Deployments update at different rates, so an updated MUD (D)TLS profile may support newer parameters. If the firewall does not recognize the newer parameters, an alert should be triggered to the firewall vendor and the IoT device owner or administrator. A firewall must be readily updatable, so that when new parameters in the MUD (D)TLS profile are discovered that are not recognized by the firewall, it can be updated quickly. Most importantly, if the firewall is not readily updatable, its protection efficacy to identify emerging malware will decrease with time. For example, if the cipher suite TLS_AES_128_CCM_8_SHA256 specified in the MUD (D)TLS profile is not recognized by the firewall, an alert will be triggered. Similarly, if the (D)TLS version specified in the MUD file is not recognized by the firewall, an alert will be triggered.

7. MUD File Example

The example below contains (D)TLS profile parameters for a IoT device used to reach servers listening on port 443 using TCP transport. JSON encoding of YANG modelled data [RFC7951] is used to illustrate the example.


```
{
   "ietf-mud:mud": {
     "mud-version": 1,
      "mud-url": "https://example.com/IoTDevice",
      "last-update": "2019-18-06T03:56:40.105+10:00",
      "cache-validity": 100,
      "extensions": [
           "ietf-mud-tls"
       ],
      "ietf-mud-tls:is-tls-dtls-profile-supported": "true",
      "is-supported": true,
      "systeminfo": "IoT device name",
      "from-device-policy": {
```

```
            "access-lists": {
              "access-list": [
                {
                  "name": "mud-7500-profile"
                }
              ]
            }
          },
        "ietf-access-control-list:acls": {
          "acl": [
            {
              "name": "mud-7500-profile",
              "type": "ipv6-acl-type",
              "aces": {
                "ace": [
                  {
                    "name": "cl0-frdev",
                    "matches": {
                      "ipv6": {
                        "protocol": 6
                      },
                      "tcp": {
                        "ietf-mud:direction-initiated": "from-device",
                        "destination-port": {
                          "operator": "eq",
                          "port": 443
                        }
                      },
                      "ietf-acl-tls:client-profile" : {
                        "tls-dtls-profiles" : [
                           {
                              "supported-tls-versions" : ["tls-1.3"],
                              "cipher-suites" : [
                                {
                                  "cipher": 19,
                                  "hash":  1
                                },
                                {
                                  "cipher": 19,
                                  "hash":  2
                                }
                               ],
                              "extension-types" : [10,11,13,16,24],
                              "supported-groups" : [29]
                           }
                         ]
                      },
                      "actions": {
```

```
                         "forwarding": "accept"
                      }
                  }
               }
             ]
           }
          }
         ]
       }
     }
  }
```


The following illustrates the example scenarios for processing the above profile:

- If the extension type "encrypt_then_mac" (code point 22) [RFC7366] in the ClientHello message is recognized by the firewall, it can identify unexpected TLS usage.

- If the extension type "token_binding" (code point 24) [RFC8472] in the MUD (D)TLS profile is not recognized by the firewall, it can ignore the unrecognized extension. Because the extension type "token_binding" is specified in the profile, an alert will be triggered to the firewall vendor and the IoT device owner or administrator to notify the firewall is not up to date.

8. Security Considerations

Security considerations in [RFC8520] need to be taken into consideration. The middlebox must adhere to the invariants discussed in Section 9.3 of [RFC8446] to act as a compliant proxy.

Although it is challenging for a malware to mimic the TLS behavior of various IoT device types and IoT device models from several manufacturers, malicious agents have a very low probability of using the same (D)TLS profile parameters as legitimate agents on the IoT device to evade detection. Network security services should also rely on contextual network data to detect false negatives. In order to detect such malicious flows, anomaly detection (deep learning techniques on network data) can be used to detect malicious agents using the same (D)TLS profile parameters as legitimate agent on the IoT device. In anomaly detection, the main idea is to maintain rigorous learning of "normal" behavior and where an "anomaly" (or an attack) is identified and categorized based on the knowledge about the normal behavior and a deviation from this normal behavior.

9. Privacy Considerations

Privacy considerations discussed in Section 16 of [RFC8520] to not reveal the MUD URL to an attacker need to be taken into consideration. The MUD URL can be stored in Trusted Execution Environment (TEE) for secure operation, enhanced data security, and prevent exposure to unauthorized software.

Full handshake inspection (Section 4.1) requires a TLS proxy device which needs to decrypt traffic between the IoT device and its server(s). There is a tradeoff between privacy of the data carried inside TLS (especially e.g., personally identifiable information and protected health information) and efficacy of endpoint security. It is strongly RECOMMENDED to avoid a TLS proxy whenever possible. For example, an enterprise firewall administrator can configure the middlebox to bypass TLS proxy functionality or payload inspection for connections destined to specific well-known services. Alternatively, a IoT device could be configured to reject all sessions that involve proxy servers to specific well-known services. In addition, mechanisms based on object security can be used by IoT devices to enable end-to-end security and the middlebox will not have any access to the packet data. For example, Object Security for Constrained RESTful Environments (OSCORE) [RFC8613] is a proposal that protects CoAP messages by wrapping them in the COSE format [RFC8152].

10. IANA Considerations

10.1. (D)TLS Profile YANG Modules

This document requests IANA to register the following URIs in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

```
URI: urn:ietf:params:xml:ns:yang:iana-tls-profile
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-acl-tls
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

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

IANA is requested to create an IANA-maintained YANG Module called "iana-tls-profile", based on the contents of Section 5.3, which will allow for new (D)TLS parameters and (D)TLS versions to be added to "client-profile". The registration procedure will be Expert Review, as defined by [RFC8126].

This document requests IANA to register the following YANG modules in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

```
name: iana-tls-profile
namespace: urn:ietf:params:xml:ns:yang:iana-tls-profile
maintained by IANA: Y
prefix: ianatp
reference: RFC XXXX

name: ietf-acl-tls
namespace: urn:ietf:params:xml:ns:yang:ietf-acl-tls
maintained by IANA: N
prefix: ietf-acl-tls
reference: RFC XXXX

name: ietf-mud-tls
namespace: urn:ietf:params:xml:ns:yang:ietf-mud-tls
maintained by IANA: N
prefix: ietf-mud-tls
reference: RFC XXXX
```

IANA is requested to create an the initial version of the IANA-maintained YANG Module called "iana-tls-profile", based on the contents of Section 5.3, which will allow for new (D)TLS parameters and (D)TLS versions to be added. IANA is requested to add this note:

- o tls-version and dtls-version values must not be directly added to the iana-tls-profile YANG module. They must instead be respectively added to the "ACL TLS Version Codes", and "ACL DTLS Version Codes" registries.

- o (D)TLS parameters must not be directly added to the iana-tls-profile YANG module. They must instead be added to the "ACL (D)TLS Parameters" registry.

When a 'tls-version' or 'dtls-version' value is respectively added to the "ACL TLS Version Codes" or "ACL DTLS Version Codes" registry, a new "enum" statement must be added to the iana-tls-profile YANG module. The following "enum" statement, and substatements thereof, should be defined:

"enum":   Replicates the label from the registry.

"value": Contains the IANA-assigned value corresponding to the 'tls-version' or 'dtls-version'.

"description": Replicates the description from the registry.

"reference": Replicates the reference from the registry and adds the title of the document.

When a (D)TLS parameter is added to "ACL (D)TLS Parameters" registry, a new "type" statement must be added to the iana-tls-profile YANG module. The following "type" statement, and substatements thereof, should be defined:

"derived type": Replicates the parameter name from the registry.

"built-in type": Contains the built-in YANG type.

"description": Replicates the description from the registry.

When the iana-tls-profile YANG module is updated, a new "revision" statement must be added in front of the existing revision statements.

IANA is requested to add this note to "ACL TLS Version Codes", "ACL DTLS Version Codes", and "ACL (D)TLS Parameters" registries:

> When this registry is modified, the YANG module iana-tls-profile must be updated as defined in [RFCXXXX].

The registration procedure for "ietf-acl-tls" YANG module will be Specification Required, as defined by [RFC8126].

10.2. ACL TLS Version registry

IANA is requested to create a new registry titled "ACL TLS Version Codes". Codes in this registry are used as valid values of 'tls-version' parameter. Further assignments are to be made through Expert Review [RFC8126].

Value	Label	Description	Reference
1	tls-1.2	TLS Version 1.2	[RFC5246]
2	tls-1.3	TLS Version 1.3	[RFC8446]

10.3. ACL DTLS version registry

IANA is requested to create a new registry titled "ACL DTLS Version Codes". Codes in this registry are used as valid values of 'dtls-version' parameter. Further assignments are to be made through Expert Review [RFC8126].

Value	Label	Description	Reference
1	dtls-1.2	DTLS Version 1.2	[RFC6346]
2	dtls-1.3	DTLS Version 1.3	[draft-ietf-tls-dtls13]

10.4. ACL (D)TLS Parameters registry

IANA is requested to create a new registry titled "ACL (D)TLS parameters".

The values for all the (D)TLS parameters in the registry are defined in the TLS and DTLS IANA registries (<https://www.iana.org/assignments/tls-parameters/tls-parameters.txt> and <https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.txt>) excluding the tls-version, dtls-version, spki-pin-set and certificate-authority parameters. Further assignments are to be made through Expert Review [RFC8126]. The registry is initially populated with the following parameters:

Parameter Name	YANG Type	JSON Type	Description
extension-type	uint16	Number	Extension type
supported-group	uint16	Number	Supported group
spki-pin-set	binary	String	Subject public key info pin set
signature-algorithm	uint16	Number	Signature algorithm
psk-key-exchange-mode	uint8	Number	pre-shared key exchange mode
application-protocol	string	String	Application protocol
cert-compression-algorithm	uint16	Number	Certificate compression algorithm
certificate-authority	string	String	Distinguished name of Certificate Authority
cipher-algorithm	uint8	Number	AEAD encryption algorithm
hash-algorithm	uint8	Number	Hash algorithm
tls-version	enumeration	String	TLS version
dtls-version	enumeration	String	DTLS version

10.5. MUD Extensions registry

IANA is requested to create a new MUD Extension Name "ietf-mud-tls" in the MUD Extensions IANA registry <https://www.iana.org/assignments/mud/mud.xhtml>.

11. Acknowledgments

Thanks to Flemming Andreasen, Shashank Jain, Michael Richardson, Piyush Joshi, Eliot Lear, Harsha Joshi, Qin Wu, Mohamed Boucadair, Ben Schwartz, Eric Rescorla, Panwei William, Nick Lamb and Nick Harper for the discussion and comments.

12. References

12.1. Normative References

[I-D.ietf-netconf-crypto-types]
Watsen, K., "YANG Data Types and Groupings for Cryptography", draft-ietf-netconf-crypto-types-18 (work in progress), August 2020.

[I-D.ietf-tls-certificate-compression]
Ghedini, A. and V. Vasiliev, "TLS Certificate Compression", draft-ietf-tls-certificate-compression-10 (work in progress), January 2020.

[I-D.ietf-tls-dtls13]
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", draft-ietf-tls-dtls13-39 (work in progress), November 2020.

[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>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

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

[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>.

[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>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

[RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <https://www.rfc-editor.org/info/rfc8701>.

[X690] ITU-T, "Information technology - ASN.1 encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ISO/IEC 8825-1:2002, 2002.

12.2. Informative References

[clear-as-mud] "Clear as MUD: Generating, Validating and Applying IoT Behaviorial Profiles", October 2019, <https://arxiv.org/pdf/1804.04358.pdf>.

[cryto-vulnerability] Perez, B., "Exploiting the Windows CryptoAPI Vulnerability", January 2020, <https://media.defense.gov/2020/Jan/14/2002234275/-1/-1/0/CSA-WINDOWS-10-CRYPT-LIB-20190114.PDF>.

[I-D.ietf-tls-esni] Rescorla, E., Oku, K., Sullivan, N., and C. Wood, "TLS Encrypted Client Hello", draft-ietf-tls-esni-09 (work in progress), December 2020.

[I-D.ietf-uta-tls13-iot-profile] Tschofenig, H. and T. Fossati, "TLS/DTLS 1.3 Profiles for the Internet of Things", draft-ietf-uta-tls13-iot-profile-00 (work in progress), June 2020.

[I-D.reddy-add-enterprise] Reddy.K, T. and D. Wing, "DNS-over-HTTPS and DNS-over-TLS Server Deployment Considerations for Enterprise Networks", draft-reddy-add-enterprise-00 (work in progress), June 2020.

[malware] Anderson, B., Paul, S., and D. McGrew, "Deciphering Malware's use of TLS (without Decryption)", July 2016, <https://arxiv.org/abs/1607.01639>.

[malware-doh]
Cimpanu, C., "First-ever malware strain spotted abusing new DoH (DNS over HTTPS) protocol", July 2019, <https://www.zdnet.com/article/first-ever-malware-strain-spotted-abusing-new-doh-dns-over-https-protocol/>.

[malware-tls]
Anderson, B. and D. McGrew, "TLS Beyond the Browser: Combining End Host and Network Data to Understand Application Behavior", October 2019, <https://dl.acm.org/citation.cfm?id=3355601>.

[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>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

[RFC7366] Gutmann, P., "Encrypt-then-MAC for Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7366, DOI 10.17487/RFC7366, September 2014, <https://www.rfc-editor.org/info/rfc7366>.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <https://www.rfc-editor.org/info/rfc7469>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[RFC7925] Tschofenig, H., Ed. and T. Fossati, "Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", RFC 7925, DOI 10.17487/RFC7925, July 2016, <https://www.rfc-editor.org/info/rfc7925>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

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

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[RFC8472] Popov, A., Ed., Nystroem, M., and D. Balfanz, "Transport Layer Security (TLS) Extension for Token Binding Protocol Negotiation", RFC 8472, DOI 10.17487/RFC8472, October 2018, <https://www.rfc-editor.org/info/rfc8472>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <https://www.rfc-editor.org/info/rfc8520>.

[RFC8576] Garcia-Morchon, O., Kumar, S., and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges", RFC 8576, DOI 10.17487/RFC8576, April 2019, <https://www.rfc-editor.org/info/rfc8576>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/info/rfc8613>.

[X501] "Information Technology - Open Systems Interconnection - The Directory: Models", ITU-T X.501, 1993.

Authors' Addresses

Tirumaleswar Reddy
McAfee, Inc.
Embassy Golf Link Business Park
Bangalore, Karnataka 560071
India

Email: kondtir@gmail.com

Dan Wing
Citrix Systems, Inc.
4988 Great America Pkwy
Santa Clara, CA 95054
USA

Email: danwing@gmail.com

Blake Anderson
Cisco Systems, Inc.
170 West Tasman Dr
San Jose, CA 95134
USA

Email: blake.anderson@cisco.com

Internet-Draft O. Gonzalez de Dios, Ed.
Intended status: Standards Track Telefonica
Expires: July 18, 2021 M. Boucadair, Ed.
Orange
Q. Wu
Huawei
January 14, 2021

A Layer 2/3 VPN Common YANG Model
draft-ietf-opsawg-vpn-common-03

Abstract

This document defines a common YANG module that is meant to be reused by various VPN-related modules such as Layer 3 VPN and Layer 2 VPN network models.

Editorial Note (To be removed by RFC Editor)

Please update these statements within the document with the RFC number to be assigned to this document:

- "This version of this YANG module is part of RFC XXXX;"
- "RFC XXXX: A Layer 2/3 VPN Common YANG Model";
- reference: RFC XXXX

Also, please update the "revision" date of the YANG module.

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 July 18, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

Various VPN-related YANG data modules were specified by the IETF (e.g., Layer 3 VPN Service Model (L3SM) [RFC8299] or Layer 2 VPN Service Model (L2SM) [RFC8466]). Others are also being specified (e.g., Layer 3 VPN Network Model (L3NM) [I-D.ietf-opsawg-l3sm-l3nm] or Layer 2 VPN Network Model (L2NM) [I-D.ietf-opsawg-l2nm]). These modules have data nodes and structures that are present in almost all these models or a subset of them. An example of such data nodes is depicted in Figure 1.

```
 module: ietf-l2vpn-ntw
    +--rw vpn-services
        +--rw vpn-service* [vpn-id]
              +--rw vpn-id                     svc-id
              +--rw vpn-svc-type?              identityref
              +--rw customer-name?             string
              +--rw svc-topo?                  identityref
              +-rw service-status
              |  +-rw admin
              |  |  +-rw status?      operational-type
              |  |  +-rw timestamp?   yang:date-and-time
              |  +-ro ops
              |     +-ro status?      operational-type
              |     +-ro timestamp?   yang:date-and-time
              |  ...

 module: ietf-l3vpn-ntw
    +--rw vpn-services
        +--rw vpn-service* [vpn-id]
              +-rw service-status
              |  +-rw admin
              |  |  +-rw status?      operational-type
              |  |  +-rw timestamp?   yang:date-and-time
              |  +-ro ops
              |     +-ro status?      operational-type
              |     +-ro timestamp?   yang:date-and-time
              +--rw vpn-id                 l3vpn-svc:svc-id
              +--rw l3sm-vpn-id?           l3vpn-svc:svc-id
              +--rw customer-name?         string
              +--rw vpn-service-topology?  identityref
              +--rw description?           string
              |  ...
```

Figure 1: Example of Common Data Nodes in Both L2NM/L3NM

In order to avoid data nodes duplication and to ease passing data among layers (service layer to network layer and vice versa), early versions of the L3NM reused many of the data nodes that are defined in the L3SM [RFC8299]. Nevertheless, that approach was abandoned because that design was interpreted as if the deployment of L3NM depends on L3SM, while this is not required. For example, a Service Provider may decide to use the L3NM to build its L3VPN services without exposing the L3SM.

Likewise, early versions of the L2NM reused many of the data nodes that are defined in both L2SM and L3NM. An example of L3NM groupings reused in L3NM is shown in Figure 2. Such data nodes reuse was interpreted as if the deployment of the L2NM requires the support of the L3NM; which is not required.

```
module ietf-l2vpn-ntw {
 ...
  import ietf-l3vpn-ntw {
    prefix l3vpn-ntw;
    reference
      "RFC NNNN: A Layer 3 VPN Network YANG Model";
  }
  ...
  container l2vpn-ntw {
    ...
    container vpn-services {
      list vpn-service {
        ...
        uses l3vpn-ntw:service-status;
        uses l3vpn-ntw:svc-transport-encapsulation;
        ...
      }
    }
    ...
  }
}
```

Figure 2: Excerpt from the L2NM YANG Module

To avoid the issues discussed above, this document defines a common YANG module that is meant to be reused by various VPN-related modules such as L3NM [I-D.ietf-opsawg-l3sm-l3nm] and L2NM [I-D.ietf-opsawg-l2nm]: "ietf-vpn-common" (Section 4).

The "ietf-vpn-common" module includes a set of identities, types, and groupings that are meant to be reused by other VPN-related YANG modules independently of their layer (e.g., Layer 2, Layer 3) and the type of the module (e.g., network model, service model) including future revisions (if any) of existing models (e.g., L3SM [RFC8299] or L2SM [RFC8466]).

2. Terminology

The terminology for describing YANG modules is defined in [RFC7950].

The meaning of the symbols in tree diagrams is defined in [RFC8340].

3. Description of the VPN Common YANG Module

The "ietf-vpn-common" module defines a set of common VPN-related features (e.g., encapsulation features such as [RFC7348], multicast [RFC6513], or routing features such as OSPF [RFC4577] or Bidirectional Forwarding Detection (BFD) [RFC5880]) and identities (e.g., service types [RFC4761][RFC4762][RFC8214][RFC7623][RFC7432][RFC8365], VPN signaling types [RFC6624][RFC5036] [RFC3931], protocol types [RFC1701][RFC1702] [RFC7676][RFC8660][RFC8663][RFC8754][RFC8426][RFC2205][RFC8277]).

The "ietf-vpn-common" module also contains the following reusable VPN-related groupings:

o 'vpn-description':

A YANG grouping that provides common administrative VPN information such as a name, a textual description, and a customer name.

o 'vpn-profile-cfg':

A YANG grouping that defines a set of profiles (encryption, routing, forwarding) valid for any Layer 2/3 VPN.

o 'status-timestamp':

A YANG grouping that defines the operational status updates of a VPN service component.

o 'service-status':

A YANG grouping that defines the administrative and operational status of a component. The grouping can be applied to the whole service or an endpoint.

o 'svc-transport-encapsulation':

A YANG grouping that defines the type of the underlay transport for a VPN service.

o 'rt-rd':

A YANG grouping that defines the set of route targets, defined as Route targets (RTs) and Route Distinguishers (RDs), to match for import and export routes to/from a Virtual Routing and Forwarding (VRF).

o 'vpn-route-targets':

A YANG grouping that A defines RT import-export rules used in a BGP-enabled VPN (e.g., [RFC4364][RFC4664]).

o 'vpn-components-group':

A YANG grouping that is used to group VPN nodes, VPN network accesses, or sites.

o 'placement-constraints':

A YANG grouping that is used to define the placement constraints of a VPN node, VPN network access, or site.

o 'ports':

A YANG grouping that defines ranges of source and destination port numbers and operators.

o 'qos-classification-policy':

A YANG grouping that defines a set of QoS classification policies based on various match Layer 3/4 criteria.

The tree diagram of the "ietf-vpn-common" module that depicts the common groupings is provided in Figure 3.

```
module: ietf-vpn-common

  grouping vpn-description
    +-- vpn-id?            vpn-id
    +-- vpn-name?          string
    +-- vpn-description?   string
    +-- customer-name?     string
  grouping vpn-profile-cfg
    +-- valid-provider-identifiers
       +-- cloud-identifier* [id] {cloud-access}?
       |  +-- id?   string
       +-- encryption-profile-identifier* [id]
       |  +-- id?   string
       +-- qos-profile-identifier* [id]
       |  +-- id?   string
       +-- bfd-profile-identifier* [id]
       |  +-- id?   string
       +-- forwarding-profile-identifier* [id]
       |  +-- id?   string
       +-- routing-profile-identifier* [id]
```

```
             +-- id?   string
     grouping status-timestamp
       +--ro status?         identityref
       +--ro last-updated?   yang:date-and-time
     grouping service-status
       +-- status
          +-- admin-status
          |  +-- status?         identityref
          |  +-- last-updated?   yang:date-and-time
          +-- oper-status
             +--ro status?         identityref
             +--ro last-updated?   yang:date-and-time
     grouping svc-transport-encapsulation
       +-- underlay-transport
          +-- type*   identityref
     grouping rt-rd
       +-- (rd-choice)?
       |  +--:(directly-assigned)
       |  |  +-- rd?              rt-types:route-distinguisher
       |  +--:(pool-assigned)
       |  |  +-- rd-pool-name?   string
       |  |  +--ro rd-from-pool?   rt-types:route-distinguisher
       |  +--:(full-autoasigned)
       |  |  +-- auto?           empty
       |  |  +--ro rd-auto?        rt-types:route-distinguisher
       |  +--:(no-rd)
       |     +-- no-rd?          empty
       +-- vpn-targets
          +-- vpn-target* [id]
          |  +-- id?                  int8
          |  +-- route-targets* [route-target]
          |  |  +-- route-target?   rt-types:route-target
          |  +-- route-target-type    rt-types:route-target-type
          +-- vpn-policies
             +-- import-policy?   string
             +-- export-policy?   string
     grouping vpn-route-targets
       +-- vpn-target* [id]
       |  +-- id?                  int8
       |  +-- route-targets* [route-target]
       |  |  +-- route-target?   rt-types:route-target
       |  +-- route-target-type    rt-types:route-target-type
       +-- vpn-policies
          +-- import-policy?   string
          +-- export-policy?   string
     grouping vpn-components-group
       +-- groups
          +-- group* [group-id]
```

```
             +-- group-id?   string
     grouping placement-constraints
       +-- constraint* [constraint-type]
          +-- constraint-type?   identityref
          +-- target
             +-- (target-flavor)?
                +--:(id)
                |  +-- group* [group-id]
                |     +-- group-id?   string
                +--:(all-accesses)
                |  +-- all-other-accesses?   empty
                +--:(all-groups)
                   +-- all-other-groups?     empty
     grouping ports
       +-- (source-port)?
       |  +--:(source-port-range-or-operator)
       |     +-- source-port-range-or-operator
       |        +-- (port-range-or-operator)?
       |           +--:(range)
       |           |  +-- lower-port    inet:port-number
       |           |  +-- upper-port    inet:port-number
       |           +--:(operator)
       |              +-- operator?     operator
       |              +-- port          inet:port-number
       +-- (destination-port)?
          +--:(destination-port-range-or-operator)
             +-- destination-port-range-or-operator
                +-- (port-range-or-operator)?
                   +--:(range)
                   |  +-- lower-port    inet:port-number
                   |  +-- upper-port    inet:port-number
                   +--:(operator)
                      +-- operator?     operator
                      +-- port          inet:port-number
     grouping qos-classification-policy
       +-- rule* [id]
          +-- id?                            string
          +-- (match-type)?
          |  +--:(match-flow)
          |  |  +-- (l3)?
          |  |  |  +--:(ipv4)
          |  |  |  |  +-- ipv4
          |  |  |  |     +-- dscp?                             inet:dscp
          |  |  |  |     +-- ecn?                              uint8
          |  |  |  |     +-- length?                           uint16
          |  |  |  |     +-- ttl?                              uint8
          |  |  |  |     +-- protocol?                         uint8
          |  |  |  |     +-- ihl?                              uint8
```

```
           |  |  |  |     +-- flags?                             bits
           |  |  |  |     +-- offset?                            uint16
           |  |  |  |     +-- identification?                    uint16
           |  |  |  |     +-- (destination-network)?
           |  |  |  |     |  +--:(destination-ipv4-network)
           |  |  |  |     |     +-- destination-ipv4-network?
           |  |  |  |     |             inet:ipv4-prefix
           |  |  |  |     +-- (source-network)?
           |  |  |  |        +--:(source-ipv4-network)
           |  |  |  |           +-- source-ipv4-network?
           |  |  |  |                   inet:ipv4-prefix
           |  |  |  +--:(ipv6)
           |  |  |     +-- ipv6
           |  |  |        +-- dscp?                              inet:dscp
           |  |  |        +-- ecn?                               uint8
           |  |  |        +-- length?                            uint16
           |  |  |        +-- ttl?                               uint8
           |  |  |        +-- protocol?                          uint8
           |  |  |        +-- (destination-network)?
           |  |  |        |  +--:(destination-ipv6-network)
           |  |  |        |     +-- destination-ipv6-network?
           |  |  |        |             inet:ipv6-prefix
           |  |  |        +-- (source-network)?
           |  |  |        |  +--:(source-ipv6-network)
           |  |  |        |     +-- source-ipv6-network?
           |  |  |        |             inet:ipv6-prefix
           |  |  |        +-- flow-label?
           |  |  |                inet:ipv6-flow-label
           |  |  +-- (l4)?
           |  |     +--:(tcp)
           |  |     |  +-- tcp
           |  |     |     +-- sequence-number?
           |  |     |     |       uint32
           |  |     |     +-- acknowledgement-number?
           |  |     |     |       uint32
           |  |     |     +-- data-offset?
           |  |     |     |       uint8
           |  |     |     +-- reserved?
           |  |     |     |       uint8
           |  |     |     +-- flags?
           |  |     |     |       bits
           |  |     |     +-- window-size?
           |  |     |     |       uint16
           |  |     |     +-- urgent-pointer?
           |  |     |     |       uint16
           |  |     |     +-- options?
           |  |     |     |       binary
           |  |     |     +-- (source-port)?
```

```
          |  |     |     |  +--:(source-port-range-or-operator)
          |  |     |     |     +-- source-port-range-or-operator
          |  |     |     |        +-- (port-range-or-operator)?
          |  |     |     |           +--:(range)
          |  |     |     |           |  +-- lower-port
          |  |     |     |           |  |       inet:port-number
          |  |     |     |           |  +-- upper-port
          |  |     |     |           |          inet:port-number
          |  |     |     |           +--:(operator)
          |  |     |     |              +-- operator?     operator
          |  |     |     |              +-- port
          |  |     |     |                      inet:port-number
          |  |     |     +-- (destination-port)?
          |  |     |        +--:(destination-port-range-or-operator)
          |  |     |           +-- destination-port-range-or-operator
          |  |     |              +-- (port-range-or-operator)?
          |  |     |                 +--:(range)
          |  |     |                 |  +-- lower-port
          |  |     |                 |  |       inet:port-number
          |  |     |                 |  +-- upper-port
          |  |     |                 |          inet:port-number
          |  |     |                 +--:(operator)
          |  |     |                    +-- operator?     operator
          |  |     |                    +-- port
          |  |     |                            inet:port-number
          |  |     +--:(udp)
          |  |        +-- udp
          |  |           +-- length?
          |  |           |       uint16
          |  |           +-- (source-port)?
          |  |           |  +--:(source-port-range-or-operator)
          |  |           |     +-- source-port-range-or-operator
          |  |           |        +-- (port-range-or-operator)?
          |  |           |           +--:(range)
          |  |           |           |  +-- lower-port
          |  |           |           |  |       inet:port-number
          |  |           |           |  +-- upper-port
          |  |           |           |          inet:port-number
          |  |           |           +--:(operator)
          |  |           |              +-- operator?     operator
          |  |           |              +-- port
          |  |           |                      inet:port-number
          |  |           +-- (destination-port)?
          |  |              +--:(destination-port-range-or-operator)
          |  |                 +-- destination-port-range-or-operator
          |  |                    +-- (port-range-or-operator)?
          |  |                       +--:(range)
          |  |                       |  +-- lower-port
```

```
          |  |                        |  |       inet:port-number
          |  |                        |  +-- upper-port
          |  |                        |          inet:port-number
          |  |                        +--:(operator)
          |  |                           +-- operator?     operator
          |  |                           +-- port
          |  |                                   inet:port-number
          |  +--:(match-application)
          |     +-- match-application?   identityref
          +-- target-class-id?           string
```

Figure 3: VPN Common Tree

4. Layer 2/3 VPN Common Module

This module uses types defined in [RFC6991], [RFC8294], and [RFC8519].

```
 <CODE BEGINS>  file "ietf-vpn-common@2021-01-14.yang"
 module ietf-vpn-common {
   yang-version 1.1;
   namespace "urn:ietf:params:xml:ns:yang:ietf-vpn-common";
   prefix vpn-common;

   import ietf-netconf-acm {
     prefix nacm;
     reference
       "RFC 8341: Network Configuration Access Control Model";
   }
   import ietf-routing-types {
     prefix rt-types;
     reference
       "RFC 8294: Common YANG Data Types for the Routing Area";
   }
   import ietf-yang-types {
     prefix yang;
     reference
       "Section 3 of RFC 6991";
   }
   import ietf-packet-fields {
     prefix packet-fields;
     reference
       "RFC 8519: YANG Data Model for Network Access
                  Control Lists (ACLs)";
   }

   organization
```

```
     "IETF OPSA (Operations and Management Area) Working Group";
   contact
     "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
      WG List:  <mailto:opsawg@ietf.org>

      Editor:  Samier Barguil
               <mailto:samier.barguilgiraldo.ext@telefonica.com>
      Editor:  Oscar Gonzalez de Dios
               <mailto:oscar.gonzalezdedios@telefonica.com>
      Editor:  Mohamed Boucadair
               <mailto:mohamed.boucadair@orange.com>
      Editor:  Qin Wu
               <mailto:bill.wu@huawei.com>";
   description
     "This YANG module defines a common module that is meant
      to be reused by various VPN-related modules (e.g.,
      Layer 3 VPN Service Model (L3SM), Layer 2 VPN Service
      Model (L2SM), Layer 3 VPN Network Model (L3NM), Layer 2
      VPN Network Model (L2NM)).

      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 of the IETF Trust's Legal Provisions
      Relating to IETF Documents
      (https://trustee.ietf.org/license-info).

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

   revision 2021-01-14 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A Layer 2/3 VPN Common YANG Model";
   }

   /******** Collection of VPN-related Features ********/

   /*
    * Features related to encapsulation schemes
    */

   feature dot1q {
```

```
     description
       "Indicates the support of the 'dot1q'
        encapsulation.";
     reference
       "IEEE Std 802.1Q: Bridges and Bridged Networks";
   }

   feature qinq {
     description
       "Indicates the support of the 'qinq'
        encapsulation.";
     reference
       "IEEE Std 802.1ad: Provider Bridges";
   }

   feature vxlan {
     description
       "Indicates the support of the 'vxlan'
        encapsulation.";
     reference
       "RFC 7348: Virtual eXtensible Local Area  Network (VXLAN):
                  A Framework for Overlaying Virtualized Layer 2
                  Networks over Layer 3 Networks";
   }

   feature qinany {
     description
       "Indicates the support of the 'qinany'
        encapsulation.";
   }

   feature lag-interface {
     description
       "Indicates the support of Link Aggregation Group (LAG)
        between VPN network accesses.";
   }

   /*
    * Features related to multicast and address family types
    */

   feature multicast {
     description
       "Indicates multicast capabilities support in a VPN.";
     reference
       "RFC 6513: Multicast in MPLS/BGP IP VPNs";
   }
```

```
   feature ipv4 {
     description
       "Indicates IPv4 support in a VPN.";
   }

   feature ipv6 {
     description
       "Indicates IPv6 support in a VPN.";
   }

   /*
    * Features related to routing protocols
    */

   feature rtg-ospf {
     description
       "Indicates support of the OSPF as the PE/CE protocol.";
     reference
       "RFC 4577: OSPF as the Provider/Customer Edge Protocol
                  for BGP/MPLS IP Virtual Private Networks(VPNs)";
   }

   feature rtg-ospf-sham-link {
     description
       "This feature indicates the support of OSPF sham links.";
     reference
       "Section 4.2.7 of RFC 4577";
   }

   feature rtg-bgp {
     description
       "Indicates support of BGP as the PE/CE protocol.";
   }

   feature rtg-rip {
     description
       "Indicates support of RIP as the PE/CE protocol.";
   }

   feature rtg-vrrp {
     description
       "Indicates support of the Virtual Router Redundancy
        Protocol (VRRP) between a cutsomer LAN and the PE.";
   }

   feature rtg-isis {
     description
       "Indicates the support of IS-IS as the PE-CE protocol.";
```

```
   }

   feature bfd {
     description
       "Indicates support of Bidirectional Forwarding Detection
        (BFD) between the CE and the PE.";
     reference
       "RFC 5880: Bidirectional Forwarding Detection (BFD)";
   }

   /*
    * Features related to VPN service constraints
    */

   feature bearer-reference {
     description
       "Indicates support of the bearer reference access
        constraint. That is, the reuse of a network connection
        that was already ordered to the SP apart from the IP VPN
        site.";
   }

   feature placement-diversity {
     description
       "Indicates the support of placement diversity
        constraints in the customer premises. An example
        of these constraints may be to avoid connecting
        a site network access to the same Provider
        Edge as a target site network access.";
   }

   /*
    * Features related to Quality of Service (QoS)
    */

   feature qos {
     description
       "Indicates support of Classes of Services (CoSes).";
   }

   feature input-bw {
     description
       "Indicates the support of the input bandwidth in a VPN.";
   }

   /*
    * Features related to security and resilience
    */
```

```
   feature encryption {
     description
       "Indicates support of encryption.";
   }

   feature fast-reroute {
     description
       "Indicates support of Fast Reroute (FRR).";
   }

   /*
    * Features related to advanced VPN options
    */

   feature cloud-access {
     description
       "Indicates support of the VPN to connect to a Cloud
        Service Provider (CSP).";
   }

   feature extranet-vpn {
     description
       "Indicates support of extranet VPNs. That is,
        the capability of a VPN to access a list of
        other VPNs.";
   }

   feature carrierscarrier {
     description
       "Indicates support of Carrier-of-Carrier VPNs.";
     reference
       "Section 9 of RFC 4364";
   }

   /*
    * Address family related identities
    */

   identity address-family {
     description
       "Defines a type for the address family.";
   }

   identity ipv4 {
     if-feature "ipv4";
     base address-family;
     description
       "IPv4 address family.";
```

```
   }

   identity ipv6 {
     if-feature "ipv6";
     base address-family;
     description
       "IPv6 address family.";
   }

   /*
    * Identities related to VPN topology
    */

   identity vpn-topology {
     description
       "Base identity of the VPN topology.";
   }

   identity any-to-any {
     base vpn-topology;
     description
       "Identity for any-to-any VPN topology.";
   }

   identity hub-spoke {
     base vpn-topology;
     description
       "Identity for Hub-and-Spoke VPN topology.";
   }

   identity hub-spoke-disjoint {
     base vpn-topology;
     description
       "Identity for Hub-and-Spoke VPN topology
        where Hubs cannot communicate with each other.";
   }

   identity custom {
     base vpn-topology;
     description
       "Identity for custom VPN topologies where the
        role of the nodes is not strictly hub or spoke.
        VPN topology controlled by the import/export
        policies. The custom topology reflects more complex
        VPN nodes such as VPN node that acts as Hub for
        certain nodes and Spoke to others.";
   }
```

```
   /*
    * Identities related to network access types
    */

   identity site-network-access-type {
     description
       "Base identity for site-network-access type.";
   }

   identity point-to-point {
     base site-network-access-type;
     description
       "Identity for point-to-point connections.";
   }

   identity multipoint {
     base site-network-access-type;
     description
       "Identity for multipoint connections.
        Example: Ethernet broadcast segment.";
   }

   identity irb {
     base site-network-access-type;
     description
       "Integrated Routing Bridge (IRB).
        Identity for pseudowire connections.";
   }

   identity loopback {
     base site-network-access-type;
     description
       "Identity for loopback connections.";
   }

   /*
    * Identities related to operational and administrative status
    */

   identity operational-status {
     description
       "Base identity for the operational status.";
   }

   identity operational-state-up {
     base operational-status;
     description
       "Operational status is UP/Enabled.";
```

```
   }

   identity operational-state-down {
     base operational-status;
     description
       "Operational status is DOWN/Disabled.";
   }

   identity operational-state-unknown {
     base operational-status;
     description
       "Operational status is UNKNOWN.";
   }

   identity administrative-status {
     description
       "Base identity for administrative status.";
   }

   identity administrative-state-up {
     base administrative-status;
     description
       "Administrative status is UP/Enabled.";
   }

   identity administrative-state-down {
     base administrative-status;
     description
       "Administrative status is DOWN/Disabled.";
   }

   identity administrative-state-testing {
     base administrative-status;
     description
       "Administrative status is up for testing purposes.";
   }

   identity administrative-state-pre-deployment {
     base administrative-status;
     description
       "Administrative status is pre-deployment phase.
        That is prior to the actual deployment of a service.";
   }

   /*
    * Identities related to site or node role
    */
```

```
   identity role {
     description
       "Base identity of a site or a node role.";
   }

   identity any-to-any-role {
     base role;
     description
       "Identity of any-to-any IP VPN.";
   }

   identity spoke-role {
     base role;
     description
       "A node or a site is acting as a Spoke IP VPN.";
   }

   identity hub-role {
     base role;
     description
       "A node or a site isacting as a Hub IP VPN.";
   }

   identity custom-role {
     base role;
     description
       "VPN-Node with custom or complex role in the VPN.
        For certain sources/destinations,  it can behave
        as a hub but for others it can act as a spoke
        depending on the configured policy.";
   }

   /*
    * Identities related to VPN service constraints
    */

   identity placement-diversity {
     description
       "Base identity for access placement constraints.";
   }

   identity bearer-diverse {
     base placement-diversity;
     description
       "Identity for bearer diversity.

        The bearers should not use common elements.";
   }
```

```
   identity pe-diverse {
     base placement-diversity;
     description
       "Identity for PE diversity.";
   }

   identity pop-diverse {
     base placement-diversity;
     description
       "Identity for POP diversity.";
   }

   identity linecard-diverse {
     base placement-diversity;
     description
       "Identity for linecard diversity.";
   }

   identity same-pe {
     base placement-diversity;
     description
       "Identity for having sites connected on the same PE.";
   }

   identity same-bearer {
     base placement-diversity;
     description
       "Identity for having sites connected using the same bearer.";
   }

   /*
    * Identities related to service types
    */

   identity service-type {
     description
       "Identity of service type.";
   }

   identity l3vpn {
     base service-type;
     description
       "Identity of L3VPN service.";
   }

   identity vpls {
     base service-type;
     description
```

```
       "Identity of the VPLS service type.";
     reference
       "RFC 4761: Virtual Private LAN Service (VPLS) Using
                  BGP for Auto-Discovery and Signaling
        RFC 4762: Virtual Private LAN Service (VPLS) Using
                  Label Distribution Protocol (LDP) Signaling";
   }

   identity vpws-evpn {
     base service-type;
     description
       "Identity of the Point-to-point Virtual Private
        Wire Service (VPWS) service type.";
     reference
       "RFC8214: Virtual Private Wire Service Support in
                 Ethernet VPN";
   }

   identity pbb-evpn {
     base service-type;
     description
       "Identity of Provider Backbone Bridging (PBB) EVPNs.";
     reference
       "RFC 7623: Provider Backbone Bridging Combined
                  with Ethernet VPN (PBB-EVPN)";
   }

   identity mpls-evpn {
     base service-type;
     description
       "Identity of MPLS based EVPNs.";
     reference
       "RFC 7432: BGP MPLS-Based Ethernet VPN";
   }

   identity vxlan-evpn {
     base service-type;
     description
       "Identity of VXLAN based EVPNs.";
     reference
       "RFC 8365: A Network Virtualization Overlay Solution
                  Using Ethernet VPN (EVPN)";
   }

   /*
    * Identities related to VPN signaling type
    */
```

```
   identity vpn-signaling-type {
     description
       "Identity of VPN signaling types";
   }

   identity bgp-signaling {
     base vpn-signaling-type;
     description
       "Identity of Layer 2 VPNs using BGP";
     reference
       "RFC 6624: Layer 2 Virtual Private Networks Using BGP for
                  Auto-Discovery and Signaling
        RFC 7432: BGP MPLS-Based Ethernet VPN";
   }

   identity ldp-signaling {
     base vpn-signaling-type;
     description
       "Identity of Targeted Label Distribution Protocol.";
     reference
       "RFC 5036: LDP Specification";
   }

   identity l2tp-signaling {
     base vpn-signaling-type;
     description
       "Identity of l2tp.";
     reference
       "RFC3931: Layer Two Tunneling Protocol - Version 3 (L2TPv3)";
   }

   /*
    * Identities related to routing protocols
    */

   identity routing-protocol-type {
     description
       "Base identity for routing protocol type.";
   }

   identity ospf {
     if-feature "rtg-ospf";
     base routing-protocol-type;
     description
       "Identity for OSPF protocol type.";
   }

   identity bgp {
```

```
     if-feature "rtg-bgp";
     base routing-protocol-type;
     description
       "Identity for BGP protocol type.";
   }

   identity static {
     base routing-protocol-type;
     description
       "Identity for static routing protocol type.";
   }

   identity rip {
     if-feature "rtg-rip";
     base routing-protocol-type;
     description
       "Identity for RIP protocol type.";
   }

   identity isis {
     if-feature "rtg-isis";
     base routing-protocol-type;
     description
       "Identity for IS-IS protocol type.";
   }

   identity vrrp {
     if-feature "rtg-vrrp";
     base routing-protocol-type;
     description
       "Identity for VRRP protocol type.

        This is to be used when LANs are directly connected
        to PE routers.";
   }

   identity direct {
     base routing-protocol-type;
     description
       "Identity for direct protocol type.

        This is to be used when LANs are directly connected
        to PE routers and and must be advertised in the VPN.";
   }

   /*
    * Identities related to Routes Import and Export
    */
```

```
   identity ie-type {
     description
       "Defines Import-Export routing profiles.
        Those profiles can be reused between VPN nodes.";
   }

   identity import {
     base ie-type;
     description
       "Import a routing profile.";
   }

   identity export {
     base ie-type;
     description
       "Export a routing profile.";
   }

   identity import-export {
     base ie-type;
     description
       "Import/Export a routing profile.";
   }

   /*
    * Identities related to bandwidth and QoS
    */

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

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

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

   identity bw-type {
     description
       "Identity of the bandwidth type.";
```

```
   }

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

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

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

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

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

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

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

   identity both {
     base qos-profile-direction;
     description
       "Identity for both WAN-to-Site and Site-to-WAN
```

```
        directions.";
   }

   /*
    *  Identities related to protocol types
    */

   identity protocol-type {
     description
       "Base identity for Protocol Type.";
   }

   identity gre {
     base protocol-type;
     description
       "GRE encapsulation.";
     reference
       "RFC 1701: Generic Routing Encapsulation (GRE)
        RFC 1702: Generic Routing Encapsulation over IPv4 networks
        RFC 7676: IPv6 Support for Generic Routing Encapsulation
                  (GRE)";
   }

   identity ldp {
     base protocol-type;
     description
       "Transport based on LDP.";
     reference
       "RFC 5086: LDP Specification";
   }

   identity sr {
     base protocol-type;
     description
       "Transport based on Segmnet Routing (SR).";
     reference
       "RFC 8660: Segment Routing with the MPLS Data Plane
        RFC 8663: MPLS Segment Routing over IP
        RFC 8754: IPv6 Segment Routing Header (SRH)";
   }

   identity sr-te {
     base protocol-type;
     description
       "Transport based on SR-TE.";
     reference
       "RFC 8426: Recommendations for RSVP-TE and Segment Routing (SR)
                  Label Switched Path (LSP) Coexistence";
```

```
   }

   identity rsvp-te {
     base protocol-type;
     description
       "Transport based on RSVP-TE.";
     reference
       "RFC 2205: Resource ReSerVation Protocol (RSVP) --
                  Version 1 Functional Specification";
   }

   identity bgp-lu {
     base protocol-type;
     description
       "Transport based on BGP-LU.";
     reference
       "RFC 8277: Using BGP to Bind MPLS Labels to Address
                  Prefixes";
   }

   identity unknown {
     base protocol-type;
     description
       "Not known protocol type.";
   }

   /*
    * Identities related to encapsulations
    */

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

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

   identity dot1q {
     if-feature "dot1q";
     base encapsulation-type;
     description
       "Identity for the support of the 'dot1q'
        encapsulation.";
   }
```

```
   identity qinq {
     if-feature "qinq";
     base encapsulation-type;
     description
       "Identity for the support of the 'qinq'
        encapsulation.";
   }

   identity qinany {
     if-feature "qinany";
     base encapsulation-type;
     description
       "Identity for the support of the 'qinany'
        encapsulation.";
   }

   identity vxlan {
     if-feature "vxlan";
     base encapsulation-type;
     description
       "Identity for the support of the 'vxlan'
        encapsulation.";
   }

   identity ethernet-type {
     base encapsulation-type;
     description
       "Identity of the Ethernet encapsulation type.";
   }

   identity vlan-type {
     base encapsulation-type;
     description
       "Identity of the VLAN encapsulation.";
   }

   identity untagged-int {
     base encapsulation-type;
     description
       "Identity of the untagged interface type.";
   }

   identity tagged-int {
     base encapsulation-type;
     description
       "Identity of the tagged interface type.";
   }
```

```
   identity lag-int {
     if-feature "lag-interface";
     base encapsulation-type;
     description
       "Identity of the LAG interface type.";
     reference
       "IEEE Std. 802.1AX: Link Aggregation";
   }

   /*
    * Identities related to VLAN Tag
    */

   identity tag-type {
     description
       "Base identity of the tag types.";
   }

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

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

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

   /*
    * Identities related to VXLAN
    */

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

   identity static-mode {
     base vxlan-peer-mode;
```

```
     description
       "Identity for VXLAN access in the static mode.";
   }

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

   /*
    * Identities related to multicast
    */

   identity multicast-gp-address-mapping {
     if-feature "multicast";
     description
       "Identity for multicast group mapping type.";
   }

   identity static-mapping {
     base multicast-gp-address-mapping;
     description
       "Identity for static mapping, i.e., attach the interface
        to the multicast group as a static member.";
   }

   identity dynamic-mapping {
     base multicast-gp-address-mapping;
     description
       "Identity for dynamic mapping, i.e., an interface was added
        to the multicast group as a result of snooping.";
   }

   identity multicast-tree-type {
     if-feature "multicast";
     description
       "Base identity for multicast tree type.";
   }

   identity ssm-tree-type {
     base multicast-tree-type;
     description
       "Identity for SSM tree type.";
   }

   identity asm-tree-type {
     base multicast-tree-type;
```

```
     description
       "Identity for ASM tree type.";
   }

   identity bidir-tree-type {
     base multicast-tree-type;
     description
       "Identity for bidirectional tree type.";
   }

   identity multicast-rp-discovery-type {
     if-feature "multicast";
     description
       "Base identity for RP discovery type.";
   }

   identity auto-rp {
     base multicast-rp-discovery-type;
     description
       "Base identity for Auto-RP discovery type.";
   }

   identity static-rp {
     base multicast-rp-discovery-type;
     description
       "Base identity for static type.";
   }

   identity bsr-rp {
     base multicast-rp-discovery-type;
     description
       "Base identity for BSR discovery type.";
   }

   /*
    * Identities related to traffic types
    */

   identity tf-type {
     description
       "Identity for the traffic type.";
   }

   identity multicast-traffic {
     if-feature "multicast";
     base tf-type;
     description
       "Identity for multicast traffic.";
```

```
   }

   identity broadcast-traffic {
     base tf-type;
     description
       "Identity for broadcast traffic.";
   }

   identity unknown-unicast-traffic {
     base tf-type;
     description
       "Identity for unknown unicast traffic.";
   }

   identity bundling-type {
     description
       "The base identity for the bundling type.  It supports
        multiple CE-VLANs associated with an L2VPN service or
        all CE-VLANs associated with an L2VPN service.";
   }

   /*
    * Identities related to service bundling
    */

   identity multi-svc-bundling {
     base bundling-type;
     description
       "Identity for multi-service bundling, i.e.,
        multiple CE-VLAN IDs can be associated with an
        L2VPN service at a site.";
   }

   identity one2one-bundling {
     base bundling-type;
     description
       "Identity for one-to-one service bundling, i.e.,
        each L2VPN can be associated with only one CE-VLAN ID
        at a site.";
   }

   identity all2one-bundling {
     base bundling-type;
     description
       "Identity for all-to-one bundling, i.e., all CE-VLAN IDs
        are mapped to one L2VPN service.";
   }
```

```
   /*
    * Identities related to customer applications
    */

   identity customer-application {
     description
       "Base identity for customer applications.";
   }

   identity web {
     base customer-application;
     description
       "Identity for a aWeb application (e.g., HTTP, HTTPS).";
   }

   identity mail {
     base customer-application;
     description
       "Identity for a mail application.";
   }

   identity file-transfer {
     base customer-application;
     description
       "Identity for a file transfer application
        (e.g., FTP, SFTP).";
   }

   identity database {
     base customer-application;
     description
       "Identity for a database application.";
   }

   identity social {
     base customer-application;
     description
       "Identity for a social-network application.";
   }

   identity games {
     base customer-application;
     description
       "Identity for a gaming application.";
   }

   identity p2p {
     base customer-application;
```

```
     description
       "Identity for a peer-to-peer application.";
   }

   identity network-management {
     base customer-application;
     description
       "Identity for a management application
        (e.g., Telnet, syslog, SNMP).";
   }

   identity voice {
     base customer-application;
     description
       "Identity for a voice application.";
   }

   identity video {
     base customer-application;
     description
       "Identity for a video conference application.";
   }

   identity embb {
     base customer-application;
     description
       "Identity for an enhanced Mobile Broadband (eMBB)
        application.  Note that an eMBB application demands
        network performance with a wide variety of
        characteristics, such as data rate, latency,
        loss rate, reliability, and many other parameters.";
   }

   identity urllc {
     base customer-application;
     description
       "Identity for an Ultra-Reliable and Low Latency
        Communications (URLLC) application.  Note that a
        URLLC application demands network performance
        with a wide variety of characteristics, such as latency,
        reliability, and many other parameters.";
   }

   identity mmtc {
     base customer-application;
     description
       "Identity for a massive Machine Type
        Communications (mMTC) application.  Note that an
```

```
        mMTC application demands network performance
        with a wide variety of characteristics, such as data
        rate, latency, loss rate, reliability, and many
        other parameters.";
  }

  /*
   * Identities related to Ethernet Services
   */

  identity control-mode {
    description
      "Defines the type of control mode on L2CP protocols.";
  }

  identity peer {
    base control-mode;
    description
      "'peer' mode, i.e., participate in the protocol towards
       the CE.  Peering is common for LACP and the Ethernet
       Local Management Interface (E-LMI) and, occasionally,
       for LLDP.  For VPLSs and VPWSs, the subscriber can also
       request that the SP peer enable spanning tree.";
  }

  identity tunnel {
    base control-mode;
    description
      "'tunnel' mode, i.e., pass to the egress or destination
       site.  For  Ethernet Private Lines (EPLs), the
       expectation is that L2CP frames are tunnelled.";
  }

  identity discard {
    base control-mode;
    description
      "'discard' mode, i.e., discard the frame.";
  }

  identity neg-mode {
    description
      "Defines the type of negotiation mode.";
  }

  identity full-duplex {
    base neg-mode;
    description
      "Refers to the auto-negotiation mode.";
```

```
   }

   identity auto-neg {
     base neg-mode;
     description
       "Indicates the auto-negotiation mode.";
   }

   /******** Collection of VPN-related Types & Identities ********/

   typedef vpn-id {
     type string;
     description
       "Defines an identifier that is used as
        a service identifier, for example.";
   }


   /*
    * Types related to Ethernet Services
    */

   typedef ccm-priority-type {
     type uint8 {
       range "0..7";
     }
     description
       "A 3-bit priority value to be used in the VLAN tag,
        if present in the transmitted frame.";
   }


   /******* VPN-related reusable groupings *******/

   grouping vpn-description {
     description
       "Provides common VPN information.";
     leaf vpn-id {
       type vpn-id;
       description
         "VPN identifier.
          This identifier has a local meaning.";
     }
     leaf vpn-name {
       type string;
       description
         "A name used to refer to the VPN.";
     }
```

```
     leaf vpn-description {
       type string;
       description
         "Textual description of a VPN service.";
     }
     leaf customer-name {
       type string;
       description
         "Name of the customer that actually uses the VPN service.";
     }
   }

   grouping vpn-profile-cfg {
     description
       "Grouping for VPN Profile configuration.";
     container valid-provider-identifiers {
       description
         "Container for Valid Provider Identifies.";
       list cloud-identifier {
         if-feature "cloud-access";
         key "id";
         description
           "List for Cloud Identifiers.";
         leaf id {
           type string;
           description
             "Identification of cloud service.
              Local administration meaning.";
         }
       }
       list encryption-profile-identifier {
         key "id";
         description
           "List for encryption profile identifiers.";
         leaf id {
           type string;
           description
             "Identification of the SP encryption profile
              to be used.  Local administration meaning.";
         }
       }
       list qos-profile-identifier {
         key "id";
         description
           "List for QoS Profile Identifiers.";
         leaf id {
           type string;
           description
```

```
              "Identification of the QoS Profile to be used.
               Local administration meaning.";
          }
        }
        list bfd-profile-identifier {
          key "id";
          description
            "List for BFD Profile identifiers.";
          leaf id {
            type string;
            description
              "Identification of the SP BFD Profile to be used.
               Local administration meaning.";
          }
        }
        list forwarding-profile-identifier {
          key "id";
          description
            "List for Forwrding Profile identifiers.";
          leaf id {
            type string;
            description
              "Identification of the Forwrding Profile Filter to be used.
               Local administration meaning.";
          }
        }
        list routing-profile-identifier {
          key "id";
          description
            "List for Routing Profile Identifiers.";
          leaf id {
            type string;
            description
              "Identification of the routing Profile to be used
               by the routing-protocols within sites, vpn-
               network-accesses or vpn-nodes for refering
               vrf-import/export policies.

               This identifier has a local meaning.";
          }
        }
        nacm:default-deny-write;
      }
    }

    grouping status-timestamp {
      description
        "This grouping defines some operational
```

```
            parameters for the service.";
      leaf status {
        type identityref {
          base operational-status;
        }
        config false;
        description
          "Operations status.";
      }
      leaf last-updated {
        type yang:date-and-time;
        config false;
        description
          "Indicates the actual date and time of the service
           status change.";
      }
    }

    grouping service-status {
      description
        "Service status grouping.";
      container status {
        description
          "Service status.";
        container admin-status {
          description
            "Administrative service status.";
          leaf status {
            type identityref {
              base administrative-status;
            }
            description
              "Administrative service status.";
          }
          leaf last-updated {
            type yang:date-and-time;
            description
              "Indicates the actual date and time of the service
               status change.";
          }
        }
        container oper-status {
          description
            "Operational service status.";
          uses status-timestamp;
        }
      }
    }
```

```
   grouping svc-transport-encapsulation {
     description
       "This grouping defines the type of underlay transport
        for VPN service.";
     container underlay-transport {
       description
         "Container for the Transport underlay.";
       leaf-list type {
         type identityref {
           base protocol-type;
         }
         ordered-by user;
         description
           "Protocols used to deliver a VPN service.";
       }
     }
   }

   grouping rt-rd {
     description
       "Grouping for RT and RD.";
     choice rd-choice {
       description
         "Route distinguisher choice between several options
          on providing the route distiniguisher value.";
       case directly-assigned {
         description
           "Explicitly assign a RD value";
         leaf rd {
           type rt-types:route-distinguisher;
           description
             "Explicitly assign a route distinguisher (RD) value.";
         }
       }
       case pool-assigned {
         leaf rd-pool-name {
           type string;
           description
             "The server will auto-assign a route
              distinguisher value and use that value operationally.
              The assignment will be selected from the pool
              identified by the rd-pool-name.";
         }
         leaf rd-from-pool {
           type rt-types:route-distinguisher;
           config false;
           description
             "The RD assigned from the pool name.";
```

```
          }
        }
        case full-autoasigned {
          leaf auto {
            type empty;
            description
              "Indicates an RD is fully auto assigned.";
          }
          leaf rd-auto {
            type rt-types:route-distinguisher;
            config false;
            description
              "Auto assigned RD.";
          }
        }
        case no-rd {
          description
            "Use the empty type to indicate RD has no value and
             is not to be auto-assigned.";
          leaf no-rd {
            type empty;
            description
              "No RD is assigned.";
          }
        }
      }
      container vpn-targets {
        description
          "Set of route-targets to match for import and export routes
           to/from VRF";
        uses vpn-route-targets;
      }
    }

    grouping vpn-route-targets {
      description
        "A grouping that specifies Route Target import-export rules
         used in a BGP-enabled VPN.";
      reference
        "RFC 4364: BGP/MPLS IP Virtual Private Networks (VPNs)
         RFC 4664: Framework for Layer 2 Virtual Private Networks
                   (L2VPNs)";
      list vpn-target {
        key "id";
        description
          "L3VPN route targets. AND/OR Operations are available
           based on the RTs assigment.";
        leaf id {
```

```
         type int8;
         description
           "Identifies each VPN Target";
       }
       list route-targets {
         key "route-target";
         description
           "List of Route Targets.";
         leaf route-target {
           type rt-types:route-target;
           description
             "Route Target value";
         }
       }
       leaf route-target-type {
         type rt-types:route-target-type;
         mandatory true;
         description
           "Import/export type of the Route Target.";
       }
     }
     container vpn-policies {
       description
         "VPN policies";
       leaf import-policy {
         type string;
         description
           "Defines the import policy.";
       }
       leaf export-policy {
         type string;
         description
           "Defines the export policy.";
       }
     }
   }

   grouping vpn-components-group {
     description
       "Grouping definition to assign
        group-ids to associate VPN nodes, sites,
        or network accesses.";
     container groups {
       description
         "Lists the groups to which a VPN node,
          a site, or a network access belongs to.";
       list group {
         key "group-id";
```

```
          description
            "List of group-ids.";
          leaf group-id {
            type string;
            description
              "Is the group-id to which a VPN node,
               a site, or a network access belongs to.";
          }
        }
      }
    }

    grouping placement-constraints {
      description
        "Constraints for placing a network
         access.";
      list constraint {
        key "constraint-type";
        description
          "List of constraints.";
        leaf constraint-type {
          type identityref {
            base placement-diversity;
          }
          description
            "Diversity constraint type.";
        }
        container target {
          description
            "The constraint will apply against
             this list of groups.";
          choice target-flavor {
            description
              "Choice for the group definition";
            case id {
              list group {
                key "group-id";
                description
                  "List of groups";
                leaf group-id {
                  type string;
                  description
                    "The constraint will apply
                     against this particular
                     group-id.";
                }
              }
            }
```

```
          case all-accesses {
            leaf all-other-accesses {
              type empty;
              description
                "The constraint will apply
                 against all other network
                 accesses of a site.";
            }
          }
          case all-groups {
            leaf all-other-groups {
              type empty;
              description
                "The constraint will apply
                 against all other groups the
                 customer is managing.";
            }
          }
        }
      }
    }
  }

  grouping ports {
    description
      "Choice of specifying a source or destination port numbers.";
    choice source-port {
      description
        "Choice of specifying the source port or referring to
         a group of source port numbers.";
      container source-port-range-or-operator {
        description
          "Source port definition.";
        uses packet-fields:port-range-or-operator;
      }
    }
    choice destination-port {
      description
        "Choice of specifying a destination port or referring
         to a group of destination port numbers.";
      container destination-port-range-or-operator {
        description
          "Destination port definition.";
        uses packet-fields:port-range-or-operator;
      }
    }
  }
```

```
   grouping qos-classification-policy {
     description
       "Configuration of the traffic classification
        policy.";
     list rule {
       key "id";
       ordered-by user;
       description
         "List of marking rules.";
       leaf id {
         type string;
         description
           "A description identifying the
            qos-classification-policy rule.";
       }
       choice match-type {
         default "match-flow";
         description
           "Choice for classification.";
         case match-flow {
           choice l3 {
             description
               "Either IPv4 or IPv6.";
             container ipv4 {
               description
                 "Rule set that matches IPv4 header.";
               uses packet-fields:acl-ip-header-fields;
               uses packet-fields:acl-ipv4-header-fields;
             }
             container ipv6 {
               description
                 "Rule set that matches IPv6 header.";
               uses packet-fields:acl-ip-header-fields;
               uses packet-fields:acl-ipv6-header-fields;
             }
           }
           choice l4 {
             description
               "Can be TCP or UDP";
             container tcp {
               description
                 "Rule set that matches TCP header.";
               uses packet-fields:acl-tcp-header-fields;
               uses ports;
             }
             container udp {
               description
                 "Rule set that matches UDP header.";
```

```
                uses packet-fields:acl-udp-header-fields;
                uses ports;
              }
            }
          }
          case match-application {
            leaf match-application {
              type identityref {
                base customer-application;
              }
              description
                "Defines the application to match.";
            }
          }
        }
        leaf target-class-id {
          type string;
          description
            "Identification of the class of service.
             This identifier is internal to the
             administration.";
        }
      }
    }
  }
  <CODE ENDS>
```

5. Security Considerations

The YANG modules specified in this document define schemas 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 Network Configuration Access Control Model (NACM) [RFC8341] 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.

The "ietf-vpn-common" module defines a set of identities, types, and groupings. These nodes are intended to be reused by other YANG modules. As such, the module does not expose by itself any data nodes which are writable, contain read-only state, or RPCs. As such, there are no additional security issues to be considered relating to the "ietf-vpn-common" module.

6. IANA Considerations

This document requests IANA to register the following URI in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

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

This document requests IANA to register the following YANG module in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

```
name: ietf-vpn-common
namespace: urn:ietf:params:xml:ns:yang:ietf-vpn-common
maintained by IANA: N
prefix: vpn-common
reference: RFC XXXX
```

7. Acknowledgements

Many thanks to Radek Krejči for the yangdoctors review.

8. Contributors

Italo Busi
Huawei Technologies
Email: Italo.Busi@huawei.com

Luis Angel Munoz
Vodafone
Email: luis-angel.munoz@vodafone.com

Victor Lopez Alvarez
Telefonica
Email: victor.lopezalvarez@telefonica.com

9. References

9.1. Normative References

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

[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>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[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>.

[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>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

9.2. Informative References

[I-D.ietf-opsawg-l2nm]
barguil, s., Dios, O., Boucadair, M., Munoz, L., Jalil, L., and J. Ma, "A Layer 2 VPN Network YANG Model", draft-ietf-opsawg-l2nm-01 (work in progress), November 2020.

[I-D.ietf-opsawg-l3sm-l3nm]
barguil, s., Dios, O., Boucadair, M., Munoz, L., and A. Aguado, "A Layer 3 VPN Network YANG Model", draft-ietf-opsawg-l3sm-l3nm-05 (work in progress), October 2020.

[RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, DOI 10.17487/RFC1701, October 1994, <https://www.rfc-editor.org/info/rfc1701>.

[RFC1702] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation over IPv4 networks", RFC 1702, DOI 10.17487/RFC1702, October 1994, <https://www.rfc-editor.org/info/rfc1702>.

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <https://www.rfc-editor.org/info/rfc2205>.

[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>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC4577] Rosen, E., Psenak, P., and P. Pillay-Esnault, "OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4577, DOI 10.17487/RFC4577, June 2006, <https://www.rfc-editor.org/info/rfc4577>.

[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>.

[RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <https://www.rfc-editor.org/info/rfc4762>.

[RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February 2012, <https://www.rfc-editor.org/info/rfc6513>.

[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>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[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>.

[RFC7623] Sajassi, A., Ed., Salam, S., Bitar, N., Isaac, A., and W. Henderickx, "Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)", RFC 7623, DOI 10.17487/RFC7623, September 2015, <https://www.rfc-editor.org/info/rfc7623>.

[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>.

[RFC8214] Boutros, S., Sajassi, A., Salam, S., Drake, J., and J. Rabadan, "Virtual Private Wire Service Support in Ethernet VPN", RFC 8214, DOI 10.17487/RFC8214, August 2017, <https://www.rfc-editor.org/info/rfc8214>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[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>.

[RFC8365] Sajassi, A., Ed., Drake, J., Ed., Bitar, N., Shekhar, R., Uttaro, J., and W. Henderickx, "A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)", RFC 8365, DOI 10.17487/RFC8365, March 2018, <https://www.rfc-editor.org/info/rfc8365>.

[RFC8426] Sitaraman, H., Ed., Beeram, V., Minei, I., and S. Sivabalan, "Recommendations for RSVP-TE and Segment Routing (SR) Label Switched Path (LSP) Coexistence", RFC 8426, DOI 10.17487/RFC8426, July 2018, <https://www.rfc-editor.org/info/rfc8426>.

[RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October 2018, <https://www.rfc-editor.org/info/rfc8466>.

[RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <https://www.rfc-editor.org/info/rfc8660>.

[RFC8663] Xu, X., Bryant, S., Farrel, A., Hassan, S., Henderickx, W., and Z. Li, "MPLS Segment Routing over IP", RFC 8663, DOI 10.17487/RFC8663, December 2019, <https://www.rfc-editor.org/info/rfc8663>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

Authors' Addresses

Samier Barguil
Telefonica
Madrid
Spain

Email: samier.barguilgiraldo.ext@telefonica.com

Oscar Gonzalez de Dios (editor)
Telefonica
Madrid
Spain

Email: oscar.gonzalezdedios@telefonica.com

Mohamed Boucadair (editor)
Orange
France

Email: mohamed.boucadair@orange.com

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Network Working Group E. Lear
Internet-Draft Cisco Systems
Intended status: Standards Track S. Rose
Expires: November 19, 2020 NIST
May 18, 2020

SBOM Extension for MUD
draft-lear-opsawg-mud-sbom-00

Abstract

Software bills of materials (SBOMs) are formal descriptions of what pieces of software are included in a product. This memo specifies a means for manufacturers to state how SBOMs may be retrieved through an extension to manufacturer usage descriptions (MUD).

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 November 19, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

Manufacturer Usage Descriptions (MUD) [RFC8520] provides a means for devices to identify what they are and what sort of network access they need. This memo specifies a YANG model [RFC6991] for reporting and a means for transmitting the report, and appropriate extensions to the MUD file to indicate how to report and how often.

Software bills of material (SBOMs) are descriptions of what software, including versionioning and dependencies, a device contains. There are different SBOM formats such as Software Package Data Exchange [SPDX] and Software Identity Tags [SWID].

This memo extends the MUD YANG schema to provide location information of an SBOM.

These SBOMs are typically found in one of three ways:

- on devices themselves
- on a web site (e.g., via URI)
- through direct contact with the manufacturer.

Some devices will have interfaces that permit direct SBOM retrieval. Examples of these interfaces might be 'ssh' or an HTTP endpoint for retrieval. There may also be private interfaces as well.

When a web site is used, versioning information about the SBOM is implicit based on the MUD file.

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.

1.1. How This Information Is Used

SBOMs are used for numerous purposes, including vulnerability assessment, license management, and inventory management. This memo provides means for either automated or semi-automated collection of that information. For devices that can output a MUD URL, the mechanism may be highly automated. For devices that have a MUD URL in either their documentation or within a QR code on a box, the mechanism is semi-automated (someone has to scan the QR code or enter the URL).

Note that SBOMs may change more frequently than access control requirements. A change to software does not necessarily mean a change to control channels that are used. Therefore, it is important to retrieve the MUD file as suggested by the manufacturer in the cache-validity period. In many cases, only the SBOM list will have been updated.

1.2. SBOM formats

There are multiple ways to express an SBOM. When these are retrieved either directly from the device or directly from a web server, tools will need to observe the content-type header to determine precisely which format is being transmitted. Because IoT devices in particular have limited capabilities, use of a specific Accept: header in HTTP or the Accept Option in CoAP is NOT RECOMMENDED. Instead, backend tooling MUST silently discard SBOM information sent with a media type that is not understood.

1.3. Discussion points

The following is discussion to be removed at time of RFC publication.

- o Is the model structured correctly?

- o Are there other retrieval mechanisms that need to be specified?

- o Do we need to be more specific in how to authenticate and retrieve SBOMs?

- o What are the implications if the MUD URL is an extension in a certificate (e.g. an IDevID cert)?

2. The mud-sbom extension model extension

We now formally define this extension. This is done in two parts. First, the extension name "sbom" is listed in the "extensions" array of the MUD file.

Second, the "mud" container is augmented with a list of SBOM sources.

This is done as follows:

```
module: ietf-mud-sbom
  augment /mud:mud:
    +--rw sboms* [version-info]
       +--rw version-info         string
       +--rw (sbom-type)?
          +--:(url)
          |  +--rw sbom-url?      inet:uri
          +--:(local-uri)
          |  +--rw sbom-local*    enumeration
          +--:(contact-info)
             +--rw contact-uri?   inet:uri
```

3. The mud-sbom augmentation to the MUD YANG model

```
<CODE BEGINS>file "ietf-mud-sbom@2020-03-06.yang"
module ietf-mud-sbom {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud-sbom";
  prefix mud-sbom;

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

  organization
    "IETF OPSAWG (Ops Area) Working Group";
  contact
```

```
    "WG
     Web: http://tools.ietf.org/wg/opsawg/
     WG List: opsawg@ietf.org
     Author: Eliot Lear lear@cisco.com ";
  description
    "This YANG module augments the ietf-mud model to provide for
     reporting of SBOMs.

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

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

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

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

  revision 2020-03-06 {
    description
      "Initial proposed standard.";
    reference
      "RFC XXXX: Extension for MUD Reporting";
  }

  grouping mud-sbom-extension {
    description
      "SBOM extension grouping";
    list sboms {
      key "version-info";
      leaf version-info {
        type string;
        description
          "A version string that is applicable for this SBOM list entry.
           The format of this string is left to the device manufacturer.
           How the network administrator determines the version of
           software running on the device is beyond the scope of this
           memo.";
```

```
      }
      choice sbom-type {
        case url {
          leaf sbom-url {
            type inet:uri;
            description
              "A statically located URI.";
          }
        }
        case local-uri {
          leaf-list sbom-local {
            type enumeration {
              enum coap {
                description
                  "Use COAP schema to retrieve SBOM";
              }
              enum coaps {
                description
                  "Use COAPS schema to retrieve SBOM";
              }
              enum http {
                description
                  "Use HTTP schema to retrieve SBOM";
              }
              enum https {
                description
                  "Use HTTPS schema to retrieve SBOM";
              }
            }
            description
              "The choice of sbom-local means that the SBOM resides at
              a location indicated by an indicted scheme for the
              device in question, at well known location
              '/.well-known/sbom'.  For example, if the MUD file
              indicates that coaps is to be used and the host is
              located at address 10.1.2.3, the SBOM could be retrieved
              at 'coaps://10.1.2.3/.well-known/sbom'.  N.B., coap and
              http schemes are NOT RECOMMENDED.";
          }
        }
        case contact-info {
          leaf contact-uri {
            type inet:uri;
            description
              "This MUST be either a tel, http, https, or
               mailto uri schema that customers can use to
               contact someone for SBOM information.";
          }
```

```
        }
        description
          "choices for SBOM retrieval.";
      }
      description
        "list of methods to get an SBOM.";
    }
  }

  augment "/mud:mud" {
    description
      "Add extension for SBOMs.";
    uses mud-sbom-extension;
  }
}

<CODE ENDS>
```

4. Examples

In this example MUD file that uses a cloud service, the Frobinator presents a location of the SBOM in a URL. Note, the ACLs in a MUD file are NOT required, although they are a very good idea for IP-based devices. The first MUD file demonstrates how to get the SBOM without ACLs, and the second has ACLs.

4.1. Without ACLS

```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://iot-device.example.com/dnsname",
    "last-update": "2019-01-15T10:22:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "device that wants to talk to a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://frobinator.example.com/doc/frob2000",
    "model-name": "Frobinator 2000",
    "extensions" : [
       "sbom"
      ],
    "sboms" : [
      {
         "version-info" : "FrobOS Release 1.1",
         "sbom-url" : "https://frobinator.example.com/sboms/f20001.1",
      }
    ]
  }
}
```


4.2. Located on the Device


```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://iot-device.example.com/dnsname",
    "last-update": "2019-01-15T10:22:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "device that wants to talk to a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://frobinator.example.com/doc/frob2000",
    "model-name": "Frobinator 2000",
    "extensions" : [
       "sbom"
      ],
    "sboms" : [
      {
         "version-info" : "FrobOS Release 1.1",
         "sbom-local" : "coaps:///.well-known/sbom",
      }
    ]
  }
}
```

4.3. SBOM Obtained from Contact Information


```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://iot-device.example.com/dnsname",
    "last-update": "2019-01-15T10:22:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "device that wants to talk to a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://frobinator.example.com/doc/frob2000",
    "model-name": "Frobinator 2000",
    "extensions" : [
       "sbom"
     ],
    "sboms" : [
      {
         "version-info" : "FrobOS Release 1.1",
         "contact-uri" : "mailto:sbom-requst@example.com",
      }
    ]
  }
}
```


4.4. With ACLS


```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://iot-device.example.com/dnsname",
    "last-update": "2019-01-15T10:22:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "device that wants to talk to a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://frobinator.example.com/doc/frob2000",
    "model-name": "Frobinator 2000",
    "extensions" : [
       "sbom"
     ],
    "sboms" : [
      {
         "version-info" : "FrobOS Release 1.1",
         "sbom-url" : "https://frobinator.example.com/sboms/f20001.1",
      }
    ],
    "from-device-policy": {
```

```
      "access-lists": {
        "access-list": [
          {
            "name": "mud-96898-v4fr"
          },
          {
            "name": "mud-96898-v6fr"
          }
        ]
      }
    },
    "to-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-96898-v4to"
          },
          {
            "name": "mud-96898-v6to"
          }
        ]
      }
    }
  },
  "ietf-access-control-list:acls": {
    "acl": [
      {
        "name": "mud-96898-v4to",
        "type": "ipv4-acl-type",
        "aces": {
          "ace": [
            {
              "name": "cl0-todev",
              "matches": {
                "ipv4": {
                  "ietf-acldns:src-dnsname": "cloud-service.example.com"
                }
              },
              "actions": {
                "forwarding": "accept"
              }
            }
          ]
        }
      },
      {
        "name": "mud-96898-v4fr",
        "type": "ipv4-acl-type",
```

```
        "aces": {
          "ace": [
            {
              "name": "cl0-frdev",
              "matches": {
                "ipv4": {
                  "ietf-acldns:dst-dnsname": "cloud-service.example.com"
                }
              },
              "actions": {
                "forwarding": "accept"
              }
            }
          ]
        }
      },
      {
        "name": "mud-96898-v6to",
        "type": "ipv6-acl-type",
        "aces": {
          "ace": [
            {
              "name": "cl0-todev",
              "matches": {
                "ipv6": {
                  "ietf-acldns:src-dnsname": "cloud-service.example.com"
                }
              },
              "actions": {
                "forwarding": "accept"
              }
            }
          ]
        }
      },
      {
        "name": "mud-96898-v6fr",
        "type": "ipv6-acl-type",
        "aces": {
          "ace": [
            {
              "name": "cl0-frdev",
              "matches": {
                "ipv6": {
                  "ietf-acldns:dst-dnsname": "cloud-service.example.com"
                }
              },
              "actions": {
```

```
                "forwarding": "accept"
              }
            }
          ]
        }
      }
    ]
  }
}
```


At this point, the management system can attempt to retrieve the SBOM, and determine which format is in use through the content-type header on the response to a GET request.

5. Security Considerations

SBOMs provide an inventory of software. If firmware is available to an attacker, the attacker may well already be able to derive this very same software inventory. Manufacturers MAY restrict access to SBOM information using appropriate authorization semantics within HTTP. In particular, if a system attempts to retrieve an SBOM via HTTP, if the client is not authorized, the server MUST produce an appropriate error, with instructions on how to register a particular client. One example may be to issue a certificate to the client for this purpose after a registration process has taken place. Another example would involve the use of OAUTH in combination with a federations of SBOM servers.

To further mitigate attacks against a device, manufacturers SHOULD recommend access controls through the normal MUD mechanism.

6. IANA Considerations

6.1. MUD Extension

The IANA is requested to add "controller-candidate" to the MUD extensions registry as follows:

```
Extension Name: sbom
Standard reference: This document
```

6.2. Well-Known Prefix

The following well known URI is requested in accordance with [RFC8615]:

URI suffix: "sbom"
Change controller: "IETF"
Specification document: This memo
Related information: See ISO/IEC 19970-2 and SPDX.org

7. References

7.1. 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>.

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

[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>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <https://www.rfc-editor.org/info/rfc8520>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

7.2. Informative References

[SPDX] The Linux Foundation, "SPDX Specification 2.1", 2016.

[SWID] ISO/IEC, "Information technology -- IT asset management -- Part 2: Software identification tag", ISO 19770-2:2015, 2015.

Appendix A. Changes from Earlier Versions

Draft -00:

- Initial revision

Authors' Addresses

Eliot Lear
Cisco Systems
Richtistrasse 7
Wallisellen CH-8304
Switzerland

Phone: +41 44 878 9200
Email: lear@cisco.com

Scott Rose
NIST
100 Bureau Dr
Gaithersburg MD 20899
USA

Phone: +1 301-975-8439
Email: scott.rose@nist.gov

IP Flow Information Export   C. Munukutla
Internet-Draft   S. Vaid
Intended status: Standards Track   Juniper Networks, Inc.
Expires: January 28, 2021   A. Mahale
                                                                   D. Patel
                                                               Google, Inc.
                                                              July 27, 2020

IP Flow Information Export (IPFIX) Information Elements Extension for Forwarding Exceptions

draft-mvmd-opsawg-ipfix-fwd-exceptions-00

Abstract

This draft proposes couple of new Forwarding exceptions related Information Elements (IEs) and Templates for the IP Flow Information Export (IPFIX) protocol. These new Information Elements and Exception Template can be used to export information about any forwarding errors in a network. This essential information is adequate to correlate packet drops to any control plane entity and map it to an impacted service. Once exceptions are correlated to a particular entity, an action can be assigned to mitigate such problems essentially enabling self-driving networks.

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 28, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

All networks are susceptible to traffic drops due to a number of factors. Traffic drops can go unnoticed unless they are service impacting. In a multi-layered network architecture, it is tedious manual work to localize and root cause traffic blackholing issues. Transient drops are even harder to detect. Existing methodologies that rely on periodically monitoring interfaces on several hosts in a network does not guarantee timely detection, and are not scalable for large networks.

In order to eliminate this tedious monitoring work-flow, objective is to simplify localization and build correlation of dropped packets to particular entity. The network entity shall identify the dropped packets by monitoring dropped counters or doing a deep packet inspection of the packet discarded by the forwarding ASIC. The implementation of the method used to detect the drop is outside the scope of this document. Dropped packets will be sampled in the forwarding-path and sent to a host or software queue along with type of exception, in/out interface information and other relevant meta data. This will be a push model where the node encountering the error will emit the information about dropped packets and associated meta-data. Techniques for IP Packet Selection [RFC5475] describes Sampling and Filtering techniques for IP packet selection either using Systematic Sampling or Random Sampling.

The IPFIX Protocol Specification [RFC7011] defines a generic exchange mechanism for collecting flow information. It supports source-triggered export of information via the push model approach. The IPFIX Information Model [IANA-IPFIX] defines a list of standard Information Elements (IEs) which can be carried by the IPFIX protocol.

This document focuses on telemetry information for dropped packet exceptions, and proposes an extension to IPFIX message format for collecting sampled exceptions. Some of the IPFIX Information Elements (IEs) already exist, some will be defined along with corresponding formats. It is also possible to achieve sampling of the dropped packets by using sampling methods like SFLOW but details of other sampling methods are outside the scope of this document.

1.1. Terminology

IPFIX-specific terminology (e.g. Information Element, Template, Template Record, Options Template Record, Template Set, Collector, Exporter, Data Record) used in this document is defined in Section 2 of [RFC7011]. As in [RFC7011] these IPFIX-specific terms have the first letter of a word capitalized. This document also makes use of the same terminology and definitions as Section 2 of [RFC5470].

1.2. Requirements Language

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

2. Scope

This document specifies the information model used for reporting packet-based forwarding exceptions. [RFC7011] provides guidance on the choices of the transport protocols used for IPFIX and their effects. Encoded IPFIX exception packets need to be reliably transported to the collector. The choice of the actual transport protocol is beyond the scope of this document.

This document assumes that all devices reporting exceptions will use existing IPFIX framework/module to send encoded packets to the collector. This would mean that the network device will specify the template that it is going to use for each of the events. The templates can be of varying length, and there could be multiple templates that a network device could use to encode the exceptions.

The implementation details of the collector application are beyond the scope of this document.

3. Information Elements

The Exception template could contain a subset of the IEs shown in Table 1, depending upon the exception reported.

Whenever packet drop happens inside forwarding plane, following information is key to understanding the issue: reason for packet drop, flow which encountered the drop (packet content), additional meta-data e.g. flow direction (ingress/egress), nexthop index, input interface, output interface, etc. on which this packet was flowing.

The following table includes all the existing IEs that a device reporting IPFIX Exceptions using Exception Template would typically need. The formats of IEs and IPFIX IDs are listed in the table below.

Field Name	Size (bits)	IANA IPFIX ID	Description
flowDirection	8	61	The direction of the Flow observed at Observation point.
ingressInterface	32	10	Index of IP interface where packets of this flow are being received.
egressInterface	32	14	Index of IP interface where packets of this flow are being sent.
dataLinkFrameSize	16	312	Specified length of data link frame.
dataLinkFrameSection	65535	315	Carries n octets from data link frame of selected frame.
commonPropertiesID	64	137	Identifier of a set of common properties that is unique per observation domain.

Table 1: Forwarding Exception Information Elements

Information Elements

4. New Information Elements

4.1. Proposed New Information Elements

The proposed new IEs that a device reporting Exceptions using Exception template would need are listed in Table 2 below.

Field Name	Abstract Data Type	Description
forwardingExceptionCode	unsigned32	Unique code for every exception
forwardingNextHopID	unsigned64	Forwarding NH - index associated with packet that encountered this exception

Table 2: New Information Elements

New Information Elements

The Information Elements defined in Figure 1 are proposed to be incorporated into the IANA IPFIX Information Elements registry [IANA-IPFIX]

4.2. Definition of Exceptions

Every network will encounter issues like packet loss, from time to time. Some of the causes for such a loss of traffic or a block in transmission of data packets include overloaded system conditions, misconfiguration, profiles and policies that restrict the bandwidth or priority of traffic, network outages, or disruption with physical cable faults. Packet loss could also happen because of incorrect stitching of the forwarding path or a mismatch between control plane and data plane state. Exception code entails the reason/error code due to which this packet has been dropped.

forwardingExceptionCode will be defined in "IPFIX Information Elements" registry. This list can be expanded in the future as necessary. The data record will have corresponding exception code value to indicate forwarding error that caused the traffic drop.

An implementation may choose to encode device internal exception codes as forwardingExceptionCode. In such scenarios, Enterprise Bit

MUST be set to 1 and corresponding Enterprise Number MUST be present as described in [RFC7011]

A list of commonly used forwarding Exception codes will be identified and listed as part of Table 3 below.

Forwarding Exception Code	Reason
1	FIREWALL_DISCARD
2	TTL_EXPIRY
3	DISCARD_ROUTE
4	BAD_IPV4_CHECKSUM
5	REJECT_ROUTE
6	BAD_IPV4_HEADER (Version incorrect or IHL < 5)
7	BAD_IPV6_HEADER (Version incorrect)
8	BAD_IPV4_HEADER_LENGTH (V4 frame is too short)
9	BAD_IPV6_HEADER_LENGTH
10	BAD_IPV6_OPTIONS_PACKET(too many option headers)
..	..

Table 3: Exception Codes

Reachability to any given destination inside the router is defined using a next-hop which is typically represented in the forwarding path as an index. The nexthop index uniquely identifies the egress path a packet would take to reach the destination. This could include information about the outgoing interface, forwarding features configured for the packet path etc.

An implementation may choose to report linecard and forwarding ASIC information on which an exception occurs, but mechanism to export these fields is out of the scope of this document.

5. Exception Templates

This section presents a list of templates for reporting exceptions using newly proposed IEs in addition to few existing IEs.

5.1. IPFIX Exception Template 1 for Forwarding Exceptions

Exception Template defined in Figure 1 may be used to export forwarding Exceptions.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Set ID = 2                 |      Length = N octets          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Template ID = 256            |       Field Count = N           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|   forwardingExceptionCode        |       Field Length = 4          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|   forwardingNextHopId            |       Field Length = 8          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|      flowDirection               |       Field Length = 1          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     ingressInterface             |       Field Length = 4          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     egressInterface              |       Field Length = 4          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     dataLinkFrameSize            |       Field Length = 2          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|    dataLinkFrameSection          |       Field Length = 65535      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Padding (opt)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPFIX Exception Template for Forwarding Exceptions

5.2. IPFIX Exception Template 2 for Forwarding Exceptions

Alternatively, Exception Template defined in Figure 2 may be used. This includes Information Element 137 to represent following fields: forwardingNexthopId, ingressInterface, underlyingIngressInterface and egressInterface.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Set ID = 2             |      Length = N octets           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Template ID = 256        |       Field Count = N            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|   forwardingExceptionCode    |       Field Length = 4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|      flowDirection           |       Field Length = 1           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     commonPropertiesId1      |       Field Length = 8           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     commonPropertiesId2      |       Field Length = 8           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     commonPropertiesId3      |       Field Length = 8           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     commonPropertiesId4      |       Field Length = 8           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|     dataLinkFrameSize        |       Field Length = 2           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|    dataLinkFrameSection      |       Field Length = 65535       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Padding (opt)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPFIX Exception Template 2 for Forwarding Exceptions

6. IANA Considerations

6.1. Information Elements

IANA manages the IPFIX Information Elements registry at [IANA-IPFIX]. This document introduces two new IPFIX Information Elements.

Name: forwardingExceptionCode
ElementID: TBD
Description: Exception code is an identifier uniquely describing cause of irregularity or traffic drop on a device.
Abstract Data Type: unsigned32
Data Type Semantics: identifier

Name: forwardingNexthopId
ElementID: TBD
Description: Nexthop ID is a unique identifier for a Nexthop on a device.
Abstract Data Type: unsigned64
Data Type Semantics: identifier

6.2. Forwarding Exception Codes

This document requests addition of a new registry for Forwarding Exception Codes.

Forwarding Exception Code	Reason
1	FIREWALL_DISCARD
2	TTL_EXPIRY
3	DISCARD_ROUTE
4	BAD_IPV4_CHECKSUM
5	REJECT_ROUTE
6	BAD_IPV4_HEADER (Version incorrect or IHL < 5)
7	BAD_IPV6_HEADER (Version incorrect)
8	BAD_IPV4_HEADER_LENGTH (V4 frame is too short)
9	BAD_IPV6_HEADER_LENGTH
10	BAD_IPV6_OPTIONS_PACKET(too many option headers)
..	..

Table 3: Exception Codes

All assignments in this registry are to be performed via Expert Review.

7. Security Considerations

Security Considerations listed in detail for IPFIX in [RFC7011] apply to this document as well. As described in [RFC7011], the IPFIX messages exchanged between network device and collector MUST be protected to provide confidentiality, integrity, and authenticity. Without those characteristics, the messages are subject to various kinds of attacks. These attacks are described in great detail in [RFC7011].

8. Contributors

Manikandan Musuvathi Poornachary
Juniper Networks, Inc.
Electra Exora Business Park˜Marathahalli-Sarjapur Outer Ring Road,
Bangalore, KA - 560103
India
Email: mpoornachary@juniper.net

Vishnu Pavan Beeram
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
USA
Email: vbeeram@juniper.net

Raveendra Torvi
Juniper Networks, Inc.
10 Technology Park Dr
Westford, MA 01886
USA
Email: rtorvi@juniper.net

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>.

[RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <https://www.rfc-editor.org/info/rfc5475>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[IANA-IPFIX]
IANA, "IP Flow Information Export (IPFIX) Entities", <https://www.iana.org/assignments/ipfix>.

[RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", RFC 5470, DOI 10.17487/RFC5470, March 2009, <https://www.rfc-editor.org/info/rfc5470>.

Authors' Addresses

Venkata Naga Chaitanya Munukutla
Juniper Networks, Inc.
10 Technology Park Dr
Westford, MA 01886
USA

Email: vmunuku@juniper.net

Shivam Vaid
Juniper Networks, Inc.
Electra, Exora Business Park- Marathahalli-Sarjapur Outer Ring Road
Bangalore, Karnataka 560103
India

Email: shivamv@juniper.net

Aditya Mahale
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
USA

Email: amahale@google.com

Devang Patel
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
USA

Email: pateldevang@google.com

OPSAWG Working Group B. Wu
Internet-Draft Q. Wu
Intended status: Standards Track Huawei
Expires: May 6, 2021 M. Boucadair
Orange
O. Gonzalez de Dios
Telefonica
B. Wen
Comcast
C. Liu
China Unicom
H. Xu
China Telecom
November 2, 2020

A YANG Model for Network and VPN Service Performance Monitoring
draft-www-opsawg-yang-vpn-service-pm-02

Abstract

The data model defined in RFC8345 introduces vertical layering relationships between networks that can be augmented to cover network/service topologies. This document defines a YANG model for both Network Performance Monitoring and VPN Service Performance Monitoring that can be used to monitor and manage network performance on the topology at higher layer or the service topology between VPN sites.

This document does not define metrics for network performance or mechanisms for measuring network performance. The YANG model defined in this document is designed as an augmentation to the network topology YANG model defined in RFC 8345 and draws on relevant YANG types defined in RFC 6991, RFC 8299, RFC 8345, and RFC 8532.

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 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

[RFC4176] provides a framework for L3VPN operations and management and specifies that performance management is required after service configuration. This document defines a YANG Model for both network performance monitoring and VPN service performance monitoring that can be used to monitor and manage network performance on the topology level or the service topology between VPN sites.

This document does not introduce new metrics for network performance or mechanisms for measuring network performance, but uses the existing mechanisms and statistics to show the performance monitoring statistics at the network and service layers. The YANG model defined in this document is designed as an augmentation to the network topology YANG model defined in [RFC8345] and draws on relevant YANG types defined in [RFC6991], [RFC8299], [RFC8345], and [RFC8532].

2. Terminology

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

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

3. Network and VPN Service performance Monitoring Model Usage

Models are key for automatic management operations. According to [I-D.ietf-opsawg-model-automation-framework] , together with service and network models, performance measurement telemetry model can monitor network performance to meet specific service SLA requirements. The model defined in this document is to derive VPN or network level performance data based on lower-level data collected via monitoring counters in the devices.

```
                      +----------------+
                      |   Customer     |
                      +----------------+
    Customer Service Models |
                            |
                      +------------------+
                      |     Service      |
                      |  Orchestration   |
                      +------------------+
    Service Network Models | | Network and VPN Service PM Model
                           | |
                      +------------------+
                      |     Network      |
                      |   Controller     |
                      +-------|----------+
                              |
    +-------------------------------------------------+
                            Network
```

Figure 1

As shown in figure 1, the Network and VPN Service PM Model can be used to expose some performance information to the above layer. The information can be used by the Orchestrator to subscribe to performance data. The Controller will then notify the Orchestrator of corresponding parameter changes.

Before using the Network and VPN Service PM Model, the mapping between the VPN Service topology and the underlying physical network has been setup, and the performance monitoring data per link in the underlying network can be collected using network performance measurement method such as MPLS Loss and Delay Measurement [RFC6374].

The performance monitoring information reflecting the quality of the Network or VPN service such as end to end network performance data between source node and destination node in the network or between VPN sites can be aggregated or calculated using, for example, PCEP solution [RFC8233] [RFC7471] [RFC7810] [RFC8571] or LMAP [RFC8194].

The measurement interval and report interval associated with these performance data usually depends on configuration parameters.

3.1. Retrieval via Pub/Sub Mechanism

Some applications such as service-assurance applications, which must maintain a continuous view of operational data and state, can use subscription model [I-D.ietf-netconf-yang-push] to subscribe to the specific Network performance data or VPN service performance data they are interested in, at the data source.

The data source can then use the Network and VPN service assurance model defined in this document and the YANG Push model [I-D.ietf-netconf-yang-push] to distribute specific telemetry data to target recipients.

3.2. On demand Retrieval via RPC Model

To obtain a snapshot of a large amount of performance data from a network element (including network controllers), service-assurance applications may use polling-based methods such as RPC model to fetch performance data on demand.

4. Description of the Data Model

This document defines the YANG module "ietf-network-vpn-pm", which is an augmentation to the "ietf-network" and "ietf-network-topology".

The performance monitoring data is augmented to service topology as shown in Figure 1.

```
+-----------------------+           +-----------------------+
|ietf-network           |           |Network and VPN Service|
|ietf-network-topology  |<----------|Performance Monitoring |
+-----------------------+ augments  |        Model          |
                                    +-----------------------+
```

Figure 1: Module Augmentation

[RFC8345]defines a YANG data model for network/service topologies and inventories. The service topology described in [RFC8345] includes the virtual topology for a service layer above Layer 1 (L1), Layer 2 (L2), and Layer 3 (L3). This service topology has the generic topology elements of node, link, and terminating point. One typical example of a service topology is described in Figure 3 of [RFC8345]: two VPN service topologies instantiated over a common L3 topology. Each VPN service topology is mapped onto a subset of nodes from the common L3 topology.

4.1. Layering Relationship Between Multiple Layers of Topology

The data model defined in [RFC8345] can describe vertical layering relationships between networks. That model can be augmented to cover network/service topologies.

Figure 2 illustrates an example of a topology mapping between the VPN service topology and an underlying network:

```
                VPN-SVC 1           VPN-SVC 2
                   /                     \
      VPN-Service-topology 1     VPN-Service-topology-2
          /     |      \            /     |      \
     Site-1A Site-1B Site1-C    Site-2A Site-2B Site-2C    Top-Down
       |          |       |       |         |     |     Service Topology
       CE         CE     CE      CE        CE     CE
       |          |       |       |         |     |
       PE         PE     PE      PE        PE     PE
   ====|==========|=======|=======|=========|=====|======================
       +-------+  |        \    /           /     |
    Bottom-up  |  |         \ /           /       |
    Network    |  |         /\           /        |
    topology   |  |       /    \        |         |
               |  |      |       |      |         |
           node1 node2 node3   node4   node5    node6
```

Figure 2: Example of topology mapping between VPN Service Topo and Underlying network

As shown in Figure 2, two VPN services topologies are both built on top of one common underlying physical network:

o VPN-SVC 1: supporting "hub-spoke" communications for Customer 1 connecting the customer’s access at 3 sites. Site-1A, Site-1B, and Site-1C are connected to PEs that are mapped to nodes 1, 2, and 3 in the underlying physical network.

  Site-1 A plays the role of hub while Site-2 B and C plays the role of spoke.

o VPN-SVC 2: supporting "hub-spoke disjoint" communications for Customer 2 connecting the customer’s access at 3 sites. Site-2A, Site-2B, and Site-2C are connected to PEs that are mapped to nodes 4, 5, and 6 in the underlying physical network.

  Site-2 A and B play the role of hub while Site-2 C plays the role of spoke.

4.2. Network Level

For network performance monitoring, the attributes of "Network Level" that defined in [RFC8345] do not need to be extended.

For VPN service performance monitoring, this document defines some new network type: "L3VPN, L2VPN". When a network topology data instance contains the L3VPN or L2VPN network type, it represents an VPN instance that can perform performance monitoring.

This model defines only the following minimal set of Network level network topology attributes:

- o "l3nm-vpn-id": Refers to an identifier of L3NM service. This identifier allows to correlate the performance status with the network service configuration.

- o "vpn-topo": The type of VPN service topology, this model supports "any-to-any", "Hub and Spoke" (where Hubs can exchange traffic), and "Hub and Spoke disjoint" (where Hubs cannot exchange traffic). [RFC8299] defines a YANG model for L3VPN Service Delivery. Three types of VPN service topologies are supported in : "any to any", "hub and spoke", and "hub and spoke disjoint". These VPN topology types can be used to describe how VPN sites communicate with each other.

```
module: ietf-network-vpn-pm
  augment /nw:networks/nw:network/nw:network-types:
    +--rw network-service-type!
       +--rw network-service-type?   identityref
  augment /nw:networks/nw:network:
    +--rw vpn-topo-attributes
       +--rw l3nm-vpn-id?    vpn-common:vpn-id
       +--rw vpn-topology?   identityref
```

Figure 3: Network Level View of the hierarchies

4.3. Node Level

For network performance monitoring, the attributes of "Node Level" that defined in [RFC8345] do not need to be extended.

For VPN service performance monitoring, this model defines only the following minimal set of Node level network topology attributes:

- o "node-type" (Attribute): Indicates the type of the node, such as PE or ASBR. This "node-type" can be used to report performance metric between any two nodes each with specific node-type.

- o "site-id" (Constraint): Uniquely identifies the site within the overall network infrastructure.

- "site-role" (Constraint): Defines the role of the site in a particular VPN topology.

- "vpn-summary-statistics": IPv4 statistics, and IPv6 statisticshave been specified separately. And MAC statistics could be extended for L2VPN.

```
augment /nw:networks/nw:network/nw:node:
  +--rw node-attributes
  |  +--rw node-type?   identityref
  |  +--rw site-id?     string
  |  +--rw site-role?   identityref
  +--rw vpn-summary-statistics
     +--rw ipv4
     |  +--rw total-routes?          uint32
     |  +--rw total-active-routes?   uint32
     +--rw ipv6
        +--rw total-routes?          uint32
        +--rw total-active-routes?   uint32
```

Figure 4: Node Level View of the hierarchies

4.4. Link and Termination Point Level

The link nodes are classified into two types: one is topology link defined in [RFC8345] , and the other is abstract link of a VPN between PEs.

The performance statistics of the topology links can be based on BGP-LS [RFC8571]. The statistics of the VPN abstract links can be collected based on VPN OAM mechanisms, e.g. TWAMP etc. Alternatively, the data cen based on the underlay technology OAM mechanism, for example, GRE tunnel OAM.

"link-type": Refers to an identifier of L3NM "underlay-transport". This identifier describes the transport technology to carry the traffic of the VPN service.

```
     augment /nw:networks/nw:network/nt:link:
       +--rw link-type?   identityref
     augment /nw:networks/nw:network/nt:link:
       +--rw low-percentile                   percentile
       +--rw high-percentile                  percentile
       +--rw middle-percentile                percentile
       +--ro reference-time                   yang:date-and-time
       +--ro measurement-interval             uint32
       +--ro link-telemetry-attributes
          +--ro loss-statistics
          |  +--ro packet-loss-count?         uint32
          |  +--ro loss-ratio?                percentage
          |  +--ro packet-reorder-count?      uint32
          |  +--ro packets-out-of-seq-count?  uint32
          |  +--ro packets-dup-count?         uint32
          +--ro delay-statistics
          |  +--ro direction?                identityref
          |  +--ro unit-value                identityref
          |  +--ro min-delay-value?          yang:gauge64
          |  +--ro max-delay-value?          yang:gauge64
          |  +--ro high-delay-percentile?    yang:gauge64
          |  +--ro middle-delay-percentile?  yang:gauge64
          |  +--ro low-delay-percentile?     yang:gauge64
          +--ro jitter-statistics
             +--ro unit-value                identityref
             +--ro min-jitter-value?         yang:gauge64
             +--ro max-jitter-value?         yang:gauge64
             +--ro low-jitter-percentile?    yang:gauge64
             +--ro high-jitter-percentile?   yang:gauge64
             +--ro middle-jitter-percentile? yang:gauge64
     augment /nw:networks/nw:network/nw:node/nt:termination-point:
       +--ro tp-telemetry-attributes
          +--ro in-octets?           uint32
          +--ro out-octets?          uint32
          +--ro inbound-unicast?     uint32
          +--ro inbound-nunicast?    uint32
          +--ro inbound-discards?    uint32
          +--ro inbound-errors?      uint32
          +--ro in-unknown-protocol?     uint32
          +--ro outbound-unicast?    uint32
          +--ro outbound-nunicast?   uint32
          +--ro outbound-discards?   uint32
          +--ro outbound-errors?     uint32
          +--ro outbound-qlen?       uint32
```

Figure 5: Link and Termination point Level View of the hierarchies

The Network and VPN service performance monitoring model defines only the following minimal set of Link level network topology attributes:

- o "link-type" (Attribute): Indicates the type of the link, such as GRE or IP-in-IP.

- o "low-percentile": Indicates low percentile to report. Setting low-percentile into 0.00 indicates the client is not interested in receiving low percentile.

- o "middle-percentile": Indicates middle percentile to report. Setting middle-percentile into 0.00 indicates the client is not interested in receiving middle percentile.

- o "high-percentile": Indicates high percentile to report. Setting low-percentile into 0.00 indicates the client is not interested in receiving high percentile.

- o Loss Statistics: A set of loss statistics attributes that are used to measure end to end loss between VPN sites or between any two network nodes.

- o Delay Statistics: A set of delay statistics attributes that are used to measure end to end latency between VPN sites or between any two network nodes..

- o Jitter Statistics: A set of IP Packet Delay Variation [RFC3393] statistics attributes that are used to measure end to end jitter between VPN sites or between any two network nodes..

The Network and VPN service performance monitoring defines the following minimal set of Termination point level network topology attributes:

- o Inbound statistics: A set of inbound statistics attributes that are used to measure the inbound statistics of the termination point, such as "the total number of octets received on the termination point", "The number of inbound packets which were chosen to be discarded", "The number of inbound packets that contained errors", etc.

- o Outbound statistics: A set of outbound statistics attributes that are used to measure the outbound statistics of the termination point, such as "the total number of octets transmitted out of the termination point", "The number of outbound packets which were chosen to be discarded", "The number of outbound packets that contained errors", etc.

5. Example of I2RS Pub/Sub Retrieval

This example shows the way for a client to subscribe for the Performance monitoring information between node A and node B in the L3 network topology built on top of the underlying network . The performance monitoring parameter that the client is interested in is end to end loss attribute.


```
 <rpc netconf:message-id="101"
    xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
    <establish-subscription
       xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
       <stream-subtree-filter>
          <networks xmlns="urn:ietf:params:xml:ns:yang:ietf-network-topo">
             <network>
              <network-id>l3-network</network-id>
              <network-technology-type xmlns="urn:ietf:params:xml:ns:yang:ietf
-network-vpn-pm">
              L3VPN
              </network-technology-type>
               <node>
                <node-id>A</node-id>
                <node-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-netwo
rk-vpn-pm">
                 <node-type>pe</node-type>
                </node-attribtues>
                <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-net
work-topology">
                 <tp-id>1-0-1</tp-id>
                <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ie
tf-network-vpn-pm">
                  <in-octets>100</in-octets>
                  <out-octets>150</out-octets>
                 </tp-telemetry-attributes>
                </termination-point>
               </node>
               <node>
                <node-id>B</node-id>
                <node-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-netwo
rk-vpn-pm">
                 <node-type>pe</node-type>
                </node-attribtues>
                <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-net
work-topology">
                 <tp-id>2-0-1</tp-id>
                <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ie
tf-network-vpn-pm">
                  <in-octets>150</in-octets>
                  <out-octets>100</out-octets>
                 </tp-telemetry-attributes>
                </termination-point>
               </node>
               <link xmlns="urn:ietf:params:xml:ns:yang:ietf-network-topology"
>
                <link-id>A-B</link-id>
                <source>
```

```
                    <source-node>A</source-node>
                   </source>
                   <destination>
                    <dest-node>B</dest-node>
                   </destination>
                    <link-type>mpls-te</link-type>
                    <link-telemetry-attributes
                     xmlns="urn:ietf:params:xml:ns:yang:ietf-network-vpn-pm">
                     <loss-statistics>
                      <packet-loss-count>100</packet-loss-count>
                     </loss-statistics>
                    </link-telemetry-attributes>
                   </link>
                </network>
             </networks>
          </stream-subtree-filter>
          <period xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push:1.0">500</per
iod>
       </establish-subscription>
    </rpc>
```

6. Example of RPC-based Retrieval

This example shows the way for the client to use RPC model to fetch performance data on demand, e.g., the client requests "packet-loss-count" between PE1 in site 1 and PE2 in site 2 belonging to the same VPN1.

```
    <rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
        message-id="1">
     <report xmlns="urn:ietf:params:xml:ns:yang:example-service-pm-report">
      <networks xmlns="urn:ietf:params:xml:ns:yang:ietf-network-topo">
        <network>
         <network-id>vpn1</network-id>
         <node>
          <node-id>A</node-id>
          <node-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-network-vpn-pm
">
          <node-type>pe</node-type>
          </node-attribtues>
          <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-network-topo
logy">
           <tp-id>1-0-1</tp-id>
           <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-netwo
rk-vpn-pm">
            <in-octets>100</in-octets>
            <out-octets>150</out-octets>
           </tp-telemetry-attributes>
          </termination-point>
         </node>
         <node>
          <node-id>B</node-id>
```

```
          <node-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-network-vpn-pm
">
          <node-type>pe</node-type>
          </node-attribtues>
          <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-network-topo
logy">
           <tp-id>2-0-1</tp-id>
           <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-netwo
rk-vpn-pm">
            <in-octets>150</in-octets>
            <out-octets>100</out-octets>
           </tp-telemetry-attributes>
          </termination-point>
         </node>
         <link-id>A-B</link-id>
          <source>
           <source-node>A</source-node>
          </source>
          <destination>
           <dest-node>B</dest-node>
          </destination>
           <link-type>mpls-te</link-type>
          <telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-network-p
m">
           <loss-statistics>
            <packet-loss-count>120</packet-loss-count>
           </loss-statistics>
          </telemetry-attributes>
         </link>
       </network>
     </report>
   </rpc>
```


7. Network and VPN Service Assurance YANG Module

This module uses types defined in [RFC8345], [RFC8299] and [RFC8532].

```
<CODE BEGINS> file "ietf-network-vpn-pm@2020-10-30.yang"
module ietf-network-vpn-pm {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-network-vpn-pm";
  prefix nvp;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Types.";
  }
  import ietf-vpn-common {
    prefix vpn-common;
  }
  import ietf-network {
```

```
    prefix nw;
    reference
      "Section 6.1 of RFC 8345: A YANG Data Model for Network
       Topologies";
  }
  import ietf-network-topology {
    prefix nt;
    reference
      "Section 6.2 of RFC 8345: A YANG Data Model for Network
       Topologies";
  }
  import ietf-lime-time-types {
    prefix lime;
    reference
      "RFC 8532: Generic YANG Data Model for the Management of
       Operations, Administration, and Maintenance (OAM) Protocols
       That Use Connectionless Communications";
  }

  organization
    "IETF OPSAWG Working Group";
  contact
    "Editor: Qin Wu
             <bill.wu@huawei.com>
     Editor: Mohamed Boucadair
             <mohamed.boucadair@orange.com>";
  description
    "This module defines a model for Network and VPN Service Performance
     monitoring.

     Copyright (c) 2020 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 2020-10-28 {
    description
      "Initial revision.";
    reference
      "RFC XXXX: A YANG Model for Network and VPN Service Performance
```

```
                 Monitoring";
  }

  identity pe {
    base vpn-common:role;
    description
      "Identity for PE type";
  }

  identity ce {
    base vpn-common:role;
    description
      "Identity for CE type";
  }

  identity asbr {
    base vpn-common:role;
    description
      "Identity for ASBR type";
  }

  identity p {
    base vpn-common:role;
    description
      "Identity for P type";
  }

  identity link-type {
    base vpn-common:protocol-type;
    description
      "Base identity for link type, e.g.,GRE, MPLS TE, VXLAN.";
  }

  identity VXLAN {
    base link-type;
    description
      "Base identity for VXLAN Tunnel.";
  }

  identity ip-in-ip {
    base link-type;
    description
      "Base identity for IP in IP Tunnel.";
  }

  identity direction {
    description
      "Base Identity for measurement direction including
```

```
       one way measurement and two way measurement.";
  }

  identity one-way {
    base direction;
    description
      "Identity for one way measurement.";
  }

  identity two-way {
    base direction;
    description
      "Identity for two way measurement.";
  }

  typedef percentage {
    type decimal64 {
      fraction-digits 5;
      range "0..100";
    }
    description
      "Percentage.";
  }

  typedef percentile {
    type decimal64 {
      fraction-digits 2;
    }
    description
      "The nth percentile of a set of data is the
       value at which n percent of the data is below it.";
  }

  grouping vpn-summary-statistics {
    description
      "VPN Statistics grouping used for network topology
       augmentation.";
    container vpn-summary-statistics {
      description
        "Container for VPN summary statistics.";
      container ipv4 {
        leaf total-routes {
          type uint32;
          description
            "Total routes in the RIB from all protocols.";
        }
        leaf total-active-routes {
          type uint32;
```

```
          description
            "Total active routes in the RIB.";
        }
        description
          "IPv4-specific parameters.";
      }
      container ipv6 {
        leaf total-routes {
          type uint32;
          description
            "Total routes in the RIB from all protocols.";
        }
        leaf total-active-routes {
          type uint32;
          description
            "Total active routes in the RIB.";
        }
        description
          "IPv6-specific parameters.";
      }
    }
  }

  grouping link-error-statistics {
    description
      "Grouping for per link error statistics.";
    container loss-statistics {
      description
        "Per link loss statistics.";
      leaf packet-loss-count {
        type uint32 {
          range "0..4294967295";
        }
        default "0";
        description
          "Total received packet drops count.
           The value of count will be set to zero (0)
           on creation and will thereafter increase
           monotonically until it reaches a maximum value
           of 2^32-1 (4294967295 decimal), when it wraps
           around and starts increasing again from zero.";
      }
      leaf loss-ratio {
        type percentage;
        description
          "Loss ratio of the packets. Express as percentage
           of packets lost with respect to packets sent.";
      }
```

```
      leaf packet-reorder-count {
        type uint32 {
          range "0..4294967295";
        }
        default "0";
        description
          "Total received packet reordered count.
           The value of count will be set to zero (0)
           on creation and will thereafter increase
           monotonically until it reaches a maximum value
           of 2^32-1 (4294967295 decimal), when it wraps
           around and starts increasing again from zero.";
      }
      leaf packets-out-of-seq-count {
        type uint32 {
          range "0..4294967295";
        }
        description
          "Total received out of sequence count.
           The value of count will be set to zero (0)
           on creation and will thereafter increase
           monotonically until it reaches a maximum value
           of 2^32-1 (4294967295 decimal), when it wraps
           around and starts increasing again from zero..";
      }
      leaf packets-dup-count {
        type uint32 {
          range "0..4294967295";
        }
        description
          "Total received packet duplicates count.
           The value of count will be set to zero (0)
           on creation and will thereafter increase
           monotonically until it reaches a maximum value
           of 2^32-1 (4294967295 decimal), when it wraps
           around and starts increasing again from zero.";
      }
    }
  }

  grouping link-delay-statistics {
    description
      "Grouping for per link delay statistics";
    container delay-statistics {
      description
        "Link delay summarised information. By default,
         one way measurement protocol (e.g., OWAMP) is used
         to measure delay.";
```

```
      leaf direction {
        type identityref {
          base direction;
        }
        default "one-way";
        description
          "Define measurement direction including one way
           measurement and two way measurement.";
      }
      leaf unit-value {
        type identityref {
          base lime:time-unit-type;
        }
        default "lime:milliseconds";
        description
          "Time units, where the options are s, ms, ns, etc.";
      }
      leaf min-delay-value {
        type yang:gauge64;
        description
          "Minimum delay value observed.";
      }
      leaf max-delay-value {
        type yang:gauge64;
        description
          "Maximum delay value observed.";
      }
      leaf low-delay-percentile {
        type yang:gauge64;
        description
          "Low percentile of the delay observed with
           specific measurement method.";
      }
      leaf middle-delay-percentile {
        type yang:gauge64;
        description
          "Middle percentile of the delay observed with
           specific measurement method.";
      }
      leaf high-delay-percentile {
        type yang:gauge64;
        description
          "High percentile of the delay observed with
           specific measurement method.";
      }
    }
  }
```

```
  grouping link-jitter-statistics {
    description
      "Grouping for per link jitter statistics";
    container jitter-statistics {
      description
        "Link jitter summarised information. By default,
         jitter is measured using IP Packet Delay Variation
         (IPDV).";
      leaf unit-value {
        type identityref {
          base lime:time-unit-type;
        }
        default "lime:milliseconds";
        description
          "Time units, where the options are s, ms, ns, etc.";
      }
      leaf min-jitter-value {
        type yang:gauge64;
        description
          "Minimum jitter value observed.";
      }
      leaf max-jitter-value {
        type yang:gauge64;
        description
          "Maximum jitter value observed.";
      }
      leaf low-jitter-percentile {
        type yang:gauge64;
        description
          "Low percentile of the jitter observed.";
      }
      leaf middle-jitter-percentile {
        type yang:gauge64;
        description
          "Middle percentile of the jitter observed.";
      }
      leaf high-jitter-percentile {
        type yang:gauge64;
        description
          "High percentile of the jitter observed.";
      }
    }
  }

  grouping tp-svc-telemetry {
    leaf in-octets {
      type uint32;
      description
```

```
          "The total number of octets received on the
           interface, including framing characters.";
      }
      leaf inbound-unicast {
        type uint32;
        description
          "Inbound unicast packets were received, and delivered
           to a higher layer during the last period.";
      }
      leaf inbound-nunicast {
        type uint32;
        description
          "The number of non-unicast (i.e., subnetwork-
           broadcast or subnetwork-multicast) packets
           delivered to a higher-layer protocol.";
      }
      leaf inbound-discards {
        type uint32;
        description
          "The number of inbound packets which were chosen
           to be discarded even though no errors had been
           detected to prevent their being deliverable to a
           higher-layer protocol.";
      }
      leaf inbound-errors {
        type uint32;
        description
          "The number of inbound packets that contained
           errors preventing them from being deliverable to a
           higher-layer protocol.";
      }
      leaf outbound-errors {
        type uint32;
        description
          "The number of outbound packets that contained
           errors preventing them from being deliverable to a
           higher-layer protocol.";
      }
      leaf in-unknown-protocol {
        type uint32;
        description
          "The number of packets received via the interface
           which were discarded because of an unknown or
           unsupported protocol.";
      }
      leaf out-octets {
        type uint32;
        description
```

```
          "The total number of octets transmitted out of the
           interface, including framing characters.";
      }
      leaf outbound-unicast {
        type uint32;
        description
          "The total number of packets that higher-level
           protocols requested be transmitted to a
           subnetwork-unicast address, including those that
           were discarded or not sent.";
      }
      leaf outbound-nunicast {
        type uint32;
        description
          "The total number of packets that higher-level
           protocols requested be transmitted to a non-
           unicast (i.e., a subnetwork-broadcast or
           subnetwork-multicast) address, including those
           that were discarded or not sent.";
      }
      leaf outbound-discards {
        type uint32;
        description
          "The number of outbound packets which were chosen
           to be discarded even though no errors had been
           detected to prevent their being transmitted.  One
           possible reason for discarding such a packet could
           be to free up buffer space.";
      }
      leaf outbound-qlen {
        type uint32;
        description
          " Length of the queue of the interface from where
            the packet is forwarded out.  The queue depth could
             be the current number of memory buffers used by the
            queue and a packet can consume one or more memory buffers
            thus constituting device-level information.";
      }
      description
        "Grouping for interface service telemetry.";
    }

    augment "/nw:networks/nw:network/nw:network-types" {
      description
        "Defines the service topologyies types";
      container network-service-type {
        presence "Indicates Network service topology";
        leaf network-service-type {
```

```
        type identityref {
          base vpn-common:service-type;
        }
        description
          "The presence identifies the network service type, e.g., L3VPN,
           L2VPN, etc.";
      }
    }
  }

  augment "/nw:networks/nw:network" {
    description
      "Augment the network with service topology attributes";
    when 'nw:network-types/nvp:network-service-type' {
      description
        "Augment only for VPN Network  topology.";
    }
    container vpn-topo-attributes {
      leaf l3nm-vpn-id {
        type vpn-common:vpn-id;
        description
          "Pointer to the parent L3NM service,
           if any.";
      }
      leaf vpn-topology {
        type identityref {
          base vpn-common:vpn-topology;
        }
        description
          "VPN service topology, e.g., hub-spoke, any-to-any,
           hub-spoke-disjoint";
      }
      description
        "Container for vpn topology attributes.";
    }
  }

  augment "/nw:networks/nw:network/nw:node" {
    description
      "Augment the network node with overlay topology attributes";
    when '../nw:network-types/nvp:network-service-type' {
      description
        "Augment only for VPN Network topology.";
    }
    container node-attributes {
      leaf node-type {
        type identityref {
          base vpn-common:role;
```

```
        }
        description
          "Node type, e.g., PE, P, ASBR.";
      }
      leaf site-id {
        type string;
        description
          "Associated vpn site";
      }
      leaf site-role {
        type identityref {
          base vpn-common:role;
        }
        default "vpn-common:any-to-any-role";
        description
          "Role of the site in the VPN.";
      }
      description
        "Container for overlay topology attributes.";
    }
    uses vpn-summary-statistics;
  }

  augment "/nw:networks/nw:network/nt:link" {
    description
      "Augment the network topology link with overlay topology attributes";
    when '../nw:network-types/nvp:network-service-type' {
      description
        "Augment only for VPN Network topology.";
    }
    leaf link-type {
      type identityref {
        base vpn-common:routing-protocol-type;
      }
      description
        "Underlay-transport type, e.g., GRE, LDP, etc.";
    }
  }

  augment "/nw:networks/nw:network/nt:link" {
    description
      "Augment the network topology link with overlay topology attributes";
    leaf low-percentile {
      type percentile;
      default "10.00";
      description
        "Low percentile to report.Setting low-percentile into 0.00 indicates
         the client is not interested in receiving low percentile.";
```

```
    }
    leaf middle-percentile {
      type percentile;
      default "50.00";
      description
        "Middle percentile to report.Setting middle-percentile into 0.00 indicate
s
         the client is not interested in receiving middle percentile.";
    }
    leaf high-percentile {
      type percentile;
      default "90.00";
      description
        "High percentile to report.";
    }
    leaf reference-time {
      type yang:date-and-time;
      description
        "The time that the current Measurement Interval started.Setting high-perc
entile
         into 0.00 indicates the client is not interested in receiving high perce
ntile.";
    }
    leaf measurement-interval {
      type uint32;
      units "seconds";
      default "60";
      description
        "Interval to calculate performance metric.";
    }
    container link-telemetry-attributes {
      config false;
      uses link-error-statistics;
      uses link-delay-statistics;
      uses link-jitter-statistics;
      description
        "Container for service telemetry attributes.";
    }
  }

  augment "/nw:networks/nw:network/nw:node/nt:termination-point" {
    description
      "Augment the network topology termination point with vpn service attributes
";
    container tp-telemetry-attributes {
      config false;
      uses tp-svc-telemetry;
      description
        "Container for termination point service telemetry attributes.";
    }
  }
}
```

<CODE ENDS>

8. Security Considerations

The YANG modules defined in this document MAY be accessed via the RESTCONF protocol [RFC8040] or NETCONF protocol ([RFC6241]). The lowest RESTCONF or NETCONF layer requires that the transport-layer protocol provides both data integrity and confidentiality, see Section 2 in [RFC8040] and [RFC6241]. 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.

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:

- o /nw:networks/nw:network/svc-topo:svc-telemetry-attributes

- o /nw:networks/nw:network/nw:node/svc-topo:node-attributes

9. IANA Considerations

This document requests IANA to register the following URI in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

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

This document requests IANA to register the following YANG module in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

```
Name:         ietf-network-vpn-pm
Namespace:    urn:ietf:params:xml:ns:yang:ietf-network-vpn-pm
Maintained by IANA: N
Prefix:       nvp
Reference:    RFC XXXX
```

10. Acknowledgements

Thanks to Adrian Farrel for reviewing this draft and providing important input to this document.

11. Contributors

Michale Wang
Huawei
Email:wangzitao@huawei.com

Roni Even
Huawei
Email: ron.even.tlv@gmail.com

12. References

12.1. 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>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <https://www.rfc-editor.org/info/rfc3393>.

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

[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>.

[RFC6370] Bocci, M., Swallow, G., and E. Gray, "MPLS Transport Profile (MPLS-TP) Identifiers", RFC 6370, DOI 10.17487/RFC6370, September 2011, <https://www.rfc-editor.org/info/rfc6370>.

[RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <https://www.rfc-editor.org/info/rfc6374>.

[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>.

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

[RFC7923] Voit, E., Clemm, A., and A. Gonzalez Prieto, "Requirements for Subscription to YANG Datastores", RFC 7923, DOI 10.17487/RFC7923, June 2016, <https://www.rfc-editor.org/info/rfc7923>.

[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>.

[RFC7952] Lhotka, L., "Defining and Using Metadata with YANG", RFC 7952, DOI 10.17487/RFC7952, August 2016, <https://www.rfc-editor.org/info/rfc7952>.

[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>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/info/rfc8345>.

[RFC8532] Kumar, D., Wang, Z., Wu, Q., Ed., Rahman, R., and S. Raghavan, "Generic YANG Data Model for the Management of Operations, Administration, and Maintenance (OAM) Protocols That Use Connectionless Communications", RFC 8532, DOI 10.17487/RFC8532, April 2019, <https://www.rfc-editor.org/info/rfc8532>.

12.2. Informative References

[I-D.ietf-netconf-yang-push] Clemm, A. and E. Voit, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-25 (work in progress), May 2019.

[I-D.ietf-opsawg-model-automation-framework] WU, Q., Boucadair, M., Lopez, D., Xie, C., and L. Geng, "A Framework for Automating Service and Network Management with YANG", draft-ietf-opsawg-model-automation-framework-10 (work in progress), October 2020.

[RFC4176] El Mghazli, Y., Ed., Nadeau, T., Boucadair, M., Chan, K., and A. Gonguet, "Framework for Layer 3 Virtual Private Networks (L3VPN) Operations and Management", RFC 4176, DOI 10.17487/RFC4176, October 2005, <https://www.rfc-editor.org/info/rfc4176>.

[RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <https://www.rfc-editor.org/info/rfc7471>.

[RFC7810] Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 7810, DOI 10.17487/RFC7810, May 2016, <https://www.rfc-editor.org/info/rfc7810>.

[RFC8233] Dhody, D., Wu, Q., Manral, V., Ali, Z., and K. Kumaki, "Extensions to the Path Computation Element Communication Protocol (PCEP) to Compute Service-Aware Label Switched Paths (LSPs)", RFC 8233, DOI 10.17487/RFC8233, September 2017, <https://www.rfc-editor.org/info/rfc8233>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[RFC8571] Ginsberg, L., Ed., Previdi, S., Wu, Q., Tantsura, J., and C. Filsfils, "BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions", RFC 8571, DOI 10.17487/RFC8571, March 2019, <https://www.rfc-editor.org/info/rfc8571>.

Authors' Addresses

Bo Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: lana.wubo@huawei.com

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Oscar Gonzalez de Dios
Telefonica
Madrid
ES

Email: oscar.gonzalezdedios@telefonica.com

Bin Wen
Comcast

Email: bin_wen@comcast.com

Change Liu
China Unicom

Email: liuc131@chinaunicom.cn

Honglei Xu
China Telecom

Email: xuhl.bri@chinatelecom.cn

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 15, 2021

M. Candela
NTT
R. Bush
IIJ & Arrcus
W. Kumari
Google
R. Housley
Vigil Security
October 12, 2020

Finding and Using Geofeed Data
draft-ymbk-opsawg-finding-geofeeds-04

Abstract

This document describes how to find and authenticate geofeed data.

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 15, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

Providers of Internet content and other services may wish to customize those services based on the geographic location of the user of the service. This is often done using the source IP address used to contact the service. Also, infrastructure and other services might wish to publish the locale of their services. [RFC8805] defines geofeed, a syntax to associate geographic locales with IP addresses. But it does not specify how to find the relevant geofeed data given an IP address.

This document specifies how to augment the Routing Policy Specification Language (RPSL) [RFC2622] inetnum: class [INETNUM] to refer to geofeed data, and how to prudently use them. In all places inetnum: is used, inet6num: should also be assumed [INET6NUM].

An optional, but utterly awesome, means for authenticating geofeed data is also defined.

1.1. Requirements Language

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

2. Geofeed Files

Geofeed files are described in [RFC8805]. They provide a facility for an IP address resource 'owner' to associate those IP addresses to geographic locale(s).

Content providers and other parties who wish to locate an IP address to a geographic locale need to find the relevant geofeed data. In Section 3 this document specifies how to find the relevant geofeed file given an IP address.

Geofeed data for large providers with significant horizontal scale and high granularity can be quite large. The size of a file can be even larger if an unsigned geofeed file combines data for many prefixes, as may be likely if the location data are maintained by a different department than address management, dual IPv4/IPv6 spaces are represented, etc.

This document also suggests optional data for geofeed files to provide stronger authenticity to the data.

3. inetnum: Class

RPSL, [RFC2622], as used by the Regional Internet Registries (RIRs), has been augmented with the inetnum: [INETNUM] and the inet6num: [INET6NUM] classes; each of which describes an IP address range and its attributes.

Ideally, RPSL would be augmented to define a new RPSL geofeed: attribute in the inetnum: class. Until such time, this document defines the syntax of a Geofeed remarks: attribute which contains an HTTPS URL of a geofeed file. The format MUST be as in this example, "remarks: Geofeed " followed by a URL which will vary.

```
inetnum: 192.0.2.0/24 # example
remarks: Geofeed https://example.com/geofeed.csv
```

Any particular inetnum: object MAY have, at most, one geofeed reference, whether a remark: or a proper geofeed: attribute when one is defined.

inetnum: objects form a hierarchy, see [INETNUM] Section 4.2.4.1, Hierarchy of INETNUM Objects. Geofeed references SHOULD be at the lowest applicable inetnum: object. When fetching, the most specific inetnum: object with a geofeed reference MUST be used.

When geofeed references are provided by multiple inetnum: objects which have identical address ranges, then the geofeed reference on the inetnum: with the most recent last-modified: attribute SHOULD be preferred.

It is significant that geofeed data may have finer granularity than the inetnum: which refers to them.

Currently, the registry data published by ARIN is not the same RPSL as the other registries; therefore, when fetching from ARIN via FTP [RFC0959], whois [RFC3912], RDAP [RFC7482], or whatever, the "NetRange" attribute/key must be treated as "inetnum" and the "Comment" attribute must be treated as "remarks".

4. Authenticating Geofeed Data

The question arises of whether a particular geofeed data set is valid, i.e. authorized by the 'owner' of the IP address space and is authoritative in some sense. The inetnum: which points to the geofeed file provides some assurance. Unfortunately the RPSL in many repositories is weakly authenticated at best. An approach where RPSL was signed a la [RFC7909] would be good, except it would have to be deployed by all RPSL registries.

An optional authenticator MAY be appended to a geofeed file. It would be essentially a digest of the main body of the file signed by the private key of the relevant RPKI certificate for the covering address range. One needs a format that bundles the relevant RPKI certificate with the signature and the digest of the geofeed text.

Borrowing detached signatures from [RFC5485], after text file canonicalization (Sec 2.2), the Cryptographic Message Syntax (CMS) [RFC3852] would be used to create a detached DER encoded signature which is then BASE64 encoded and line wrapped to 72 or fewer characters.

Both the address ranges of the signing certificate and of the inetnum: MUST cover all prefixes in the geofeed file; and the address range of the signing certificate must cover that of the inetnum:.

An address range A 'covers' address range B if the range of B is identical to or a subset of A. 'Address range' is used here because inetnum: objects and RPKI certificates need not align on CIDR prefix boundaries, while those of geofeed lines must.

As the signer would need to specify the covered RPKI resources relevant to the signature, the RPKI certificate covering the inetnum: object's address range would be included in the [RFC3852] CMS SignedData certificates field.

Identifying the private key associated with the certificate, and getting the department with the HSM to sign the CMS blob is left as an exercise for the implementor. On the other hand, verifying the signature requires no complexity; the certificate, which can be validated in the public RPKI, has the needed public key.

Until [RFC8805] is updated to formally define such an appendix, it MUST be 'hidden' as a series of "#" comments at the end of the geofeed file. This is a cryptographically incorrect, albeit simple example. A correct and full example is in Appendix A.


```
# RPKI Signature: 192.0.2.0/24
# MIIGlwYJKoZIhvcNAQcCoIIGiDCCBoQCAQMxDTALBglghkgBZQMEAgEwDQYLKoZ
# IhvcNAQkQAS+gggSxMIIErTCCA5WgAwIBAgIUJ605QIPX8rW5m4Zwx3WyuW7hZu
...
# imwYkXpiMxw44EZqDjl36MiWsRDLdgoijBBcGbibwyAfGeR46k5raZCGvxG+4xa
# O8PDTxTfIYwAnBjRBKAqAZ7yX5xHfm58jUXsZJ7Ileq1S7G6Kk=
# End Signature: 192.0.2.0/24
```


5. Operational Considerations

To create the needed inetnum: objects, an operator wishing to register the location of their geofeed file needs to coordinate with their RIR/NIR and/or any provider LIR which has assigned prefixes to them. RIRs/NIRs provide means for assignees to create and maintain inetnum: objects. They also provide means of [sub-]assigning IP address resources and allowing the assignee to create whois data, including inetnum: objects, and thereby referring to geofeed files.

The geofeed files SHOULD be published over and fetched using https [RFC2818].

When using data from a geofeed file, one MUST ignore data outside of the referring inetnum: object's inetnum: attribute address range.

Iff the geofeed file is not signed per Section 4, then multiple inetnum: objects MAY refer to the same geofeed file, and the consumer MUST use only geofeed lines where the prefix is covered by the address range of the inetnum: object they have followed.

To minimize the load on RIR whois [RFC3912] services, use of the RIR's FTP [RFC0959] services SHOULD be the preferred access. This also provides bulk access instead of fetching with a tweezers.

Currently, geolocation providers have bulk whois data access at all the RIRs. An anonymized version of such data is openly available for all RIRs except ARIN, which requires an authorization. However, for users without such authorization the same result can be achieved with extra RDAP effort. There is open source code to pass over such data across all RIRs, collect all geofeed references, and process them [geofeed-finder].

An entity fetching geofeed data using these mechanisms MUST NOT do frequent real-time look-ups to prevent load on RPSL and geofeed servers. [RFC8805] Section 3.4 suggests use of the [RFC7234] HTTP Expires Caching Header to signal when geofeed data should be refetched. As the data change very infrequently, in the absence of such an HTTP Header signal, collectors MUST NOT fetch more frequently than weekly. It would be wise not to fetch at magic times such as midnight UTC, the first of the month, etc., because too many others are likely to do the same.

6. Security Considerations

It would be generally prudent for a consumer of geofeed data to also use other sources to cross-validate the data. All of the Security Considerations of [RFC8805] apply here as well.

As mentioned in Section 4, many RPSL repositories have weak if any authentication. This would allow spoofing of inetnum: objects pointing to malicious geofeed files. Section 4 suggests an unfortunately complex method for stronger authentication based on the RPKI.

If an inetnum: for a wide prefix (e.g. a /16) points to an RPKI-signed geofeed file, a customer or attacker could publish a unsigned equal or narrower (e.g. a /24) inetnum: in a whois registry which has weak authorization.

The RPSL providers have had to throttle fetching from their servers due to too-frequent queries. Usually they throttle by the querying IP address or block. Similar defenses will likely need to be deployed by geofeed file servers.

7. IANA Considerations

IANA is asked to register object identifiers for one content type in the "SMI Security for S/MIME CMS Content Type (1.2.840.113549.1.9.16.1)" registry as follows:

Description	OID	Specification
id-ct-geofeedCSVwithCRLF	1.2.840.113549.1.9.16.1.47	[RFC-TBD]

8. Acknowledgements

Thanks to Rob Austein for CMS and detached signature clue. George Michaelson for the first, and a substantial, external review. Erik Kline who was too shy to agree to co-authorship. Additionally, we express our gratitude to early implementors, including Menno Schepers, Flavio Luciani, Eric Dugas, and Kevin Pack. Also to geolocation providers that are consuming geofeeds with this described solution, Jonathan Kosgei (ipdata.co), and Ben Dowling (ipinfo.io).

9. References

9.1. Normative References

[INET6NUM]
RIPE, "Description of the INET6NUM Object", <https://www.ripe.net/manage-ips-and-asns/db/support/documentation/ripe-database-documentation/rpsl-object-types/4-2-descriptions-of-primary-objects/4-2-3-description-of-the-inet6num-object>.

[INETNUM] RIPE, "Description of the INETNUM Object", <https://www.ripe.net/manage-ips-and-asns/db/support/documentation/ripe-database-documentation/rpsl-object-types/4-2-descriptions-of-primary-objects/4-2-4-description-of-the-inetnum-object>.

[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>.

[RFC2622] Alaettinoglu, C., Villamizar, C., Gerich, E., Kessens, D., Meyer, D., Bates, T., Karrenberg, D., and M. Terpstra, "Routing Policy Specification Language (RPSL)", RFC 2622, DOI 10.17487/RFC2622, June 1999, <https://www.rfc-editor.org/info/rfc2622>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, DOI 10.17487/RFC3852, July 2004, <https://www.rfc-editor.org/info/rfc3852>.

[RFC5485] Housley, R., "Digital Signatures on Internet-Draft Documents", RFC 5485, DOI 10.17487/RFC5485, March 2009, <https://www.rfc-editor.org/info/rfc5485>.

[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>.

[RFC8805] Kline, E., Duleba, K., Szamonek, Z., Moser, S., and W. Kumari, "A Format for Self-Published IP Geolocation Feeds", RFC 8805, DOI 10.17487/RFC8805, August 2020, <https://www.rfc-editor.org/info/rfc8805>.

9.2. Informative References

[geofeed-finder] Massimo Candela, "geofeed-finder", <https://github.com/massimocandela/geofeed-finder>.

[RFC0959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, DOI 10.17487/RFC0959, October 1985, <https://www.rfc-editor.org/info/rfc959>.

[RFC3912] Daigle, L., "WHOIS Protocol Specification", RFC 3912, DOI 10.17487/RFC3912, September 2004, <https://www.rfc-editor.org/info/rfc3912>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <https://www.rfc-editor.org/info/rfc7234>.

[RFC7482] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", RFC 7482, DOI 10.17487/RFC7482, March 2015, <https://www.rfc-editor.org/info/rfc7482>.

[RFC7909] Kisteleki, R. and B. Haberman, "Securing Routing Policy Specification Language (RPSL) Objects with Resource Public Key Infrastructure (RPKI) Signatures", RFC 7909, DOI 10.17487/RFC7909, June 2016, <https://www.rfc-editor.org/info/rfc7909>.

Appendix A. Example

This appendix provides an example, including a trust anchor, a CA certificate subordinate to the trust anchor, an end-entity certificate subordinate to the CA for signing the geofeed, and a detached signature.

The trust anchor is represented by a self-signed certificate. As usual in the RPKI, the trust anchor has authority over all IPv4 address blocks, all IPv6 address blocks, and all AS numbers.


```
-----BEGIN CERTIFICATE-----
MIIEPjCCAyagAwIBAgIUPsUFJ4e/7pKZ6E14aBdkbYzms1gwDQYJKoZIhvcNAQEL
BQAwFTETMBEGA1UEAxMKZXhhbXBsZS10YTAeFw0yMDA5MDMxODU0NTRaFw0zMDA5
MDExODU0NTRaMBUxEzARBgNVBAMTCmV4YW1wbGUtdGEwggEiMA0GCSqGSIb3DQEB
AQUAA4IBDwAwggEKAoIBAQCelMmMDCGBhqn/a3VrNAoKMr1HVLKxGoG7VF/13HZJ
0twObUZlh3Jz+XeD+kNAURhELWTrsgdTkQQfqinqOuRemxTl55+x7nLpe5nmwaBH
XqqDOHubmkbAGanGcm6T/rD9KNk1Z46Uc2p7UYu0fwNO0mo0aqFL2FSyvzZwziNe
g7ELYZ4a3LvGn81JfP/JvM6pgtoMNuee5RV6TWaz7LV304ICj8Bhphy/HFpOA1rb
O9gs8CUMgqz+RroAIa8cV8gbF/fPCz9Ofl7Gdmib679JxxFrW4wRJ0nMJgJmsZXq
jaVc0g7ORc+eIAcHw7Uroc6h7Y7lGjOkDZF75j0mLQa3AgMBAAGjggGEMIIBgDAd
BgNVHQ4EFgQU3hNEuwvUGNCHY1TBatcUR03pNdYwHwYDVR0jBBgwFoAU3hNEuwvU
GNCHY1TBatcUR03pNdYwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMCAQYw
GAYDVR0gAQH/BA4wDDAKBggrBgEFBQcOAjCBuQYIKwYBBQUHAQsEgawwgakwPgYI
KwYBBQUHMAqGMnJzeW5jOi8vcnBraS5leGFtcGxlLm5ldC9yZXBvc2l0b3J5L2V4
YW1wbGUtdGEubWZ0MDUGCCsGAQUFBzANhilodHRwczovL3JyZHAuZXhhbXBsZS5u
ZXQvbm90aWZpY2F0aW9uLnhtbDAwBggrBgEFBQcwBYYkcnN5bmM6Ly9ycGtpLmV4
YW1wbGUubmV0L3JlcG9zaXRvcnkvMCcGCCsGAQUFBwEHAQH/BBgwFjAJBAIAATAD
AwEAMAkEAgACMAMDAQAwHgYIKwYBBQUHAQgEEjAQoA4wDDAKAgEAAgUA/////zAN
BgkqhkiG9w0BAQsFAAOCAQEAgZFQ0Sf3CI5Hwev61AUWHYOFniy69PuDTq+WnhDe
xX5rpjSDRrs5L756KSKJcaOJ36lzO45lfOPSY9fH6x30pnipaqRA7t5rApky24jH
cSUA9iRednzxhVyGjWKnfAKyNo2MYfaOAT0db1GjyLKbOADI9FowtHBUu+60ykcM
Quz66XrzxtmxlrRcAnbv/HtV17qOd4my6q5yjTPR1dmYN9oR/2ChlXtGE6uQVguA
rvNZ5CwiJ1TgGGTB7T8ORHwWU6dGTc0jk2rESAaikmLi1roZSNC21fckhapEit1a
x8CyiVxjcVc5e0AmS1rJfL6LIfwmtive/N/eBtIM92HkBA==
-----END CERTIFICATE-----
```


The CA certificate is issued by the trust anchor. This certificate grants authority over one IPv4 address block (192.0.2.0/24) and two AS numbers (64496 and 64497).

```
-----BEGIN CERTIFICATE-----
MIIFBzCCA++gAwIBAgIUcyCzS10hdfG65kbRq7toQAvRDKowDQYJKoZIhvcNAQEL
BQAwFTETMBEGA1UEAxMKZXhhbXBsZS10YTAeFw0yMDA5MDMxOTAyMTlaFw0yMTA5
MDMxOTAyMTlaMDMxMTAvBgNVBAMTKDNBQ0UyQ0VGNEZCMjFCN0QxMUUzRTE4NEVG
QzFFMjk3QjM3Nzg2NDIwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDc
zz1qwTxC2ocw5rqp8ktm2XyYkl8riBVuqlXwfefTxsR2YFpgz9vkYUd5Az9EVEG7
6wGIyZbtmhK63eEeaqbKz2GHub467498BXeVrYysO+YuIGgCEYKznNDZ4j5aaDbo
j5+4/z0Qvv6HEsxQd0f8br6lKJwgeRM6+fm7796HNPB0aqD7Zj9NRCLXjbB0DCgJ
liH6rXMKR86ofgll9V2mRjesvhdKYgkGbOif9rvxVpLJ/6zdru5CE9yeuJZ59l+n
YH/r6PzdJ4Q7yKrJX8qD6A60j4+biaU4MQ72KpsjhQNTTqF/HRwi0N54GDaknEwE
TnJQHgLJDYqww9yKWtjjAgMBAAGjggIvMIICKzAdBgNVHQ4EFgQUOs4s70+yG30R
4+GE78Hil7N3hkIwHwYDVR0jBBgwFoAU3hNEuwvUGNCHY1TBatcUR03pNdYwDwYD
VR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMCAQYwGAYDVR0gAQH/BA4wDDAKBggr
BgEFBQcOAjBhBgNVHR8EWjBYMFagVKBShlByc3luYzovL3Jwa2kuZXhhbXBsZS5u
ZXQvcmVwb3NpdG9yeS8zQUNFMkNFRjRGQjIxQjdEMTFFM0UxODRFRkMxRTI5N0Iz
Nzc4NjQyLmNybDBOBggrBgEFBQcBAQRCMEAwPgYIKwYBBQUHMAKGMnJzeW5jOi8v
cnBraS5leGFtcGxlLm5ldC9yZXBvc2l0b3J5L2V4YW1wbGUtdGEuY2VyMIG5Bggr
BgEFBQcBCwSBrDCBqTA+BggrBgEFBQcwCoYycnN5bmM6Ly9ycGtpLmV4YW1wbGUu
bmV0L3JlcG9zaXRvcnkvZXhhbXBsZS1jYS5tZnQwNQYIKwYBBQUHMA2GKWh0dHBz
Oi8vcnJkcC5leGFtcGxlLm5ldC9ub3RpZmljYXRpb24ueG1sMDAGCCsGAQUFBzAF
hiRyc3luYzovL3Jwa2kuZXhhbXBsZS5uZXQvcmVwb3NpdG9yeS8wHwYIKwYBBQUH
AQcBAf8EEDAOMAwEAgABMAYDBADAAAIwHgYIKwYBBQUHAQgEEjAQoA4wDDAKAgMA
+/ACAwD78TANBgkqhkiG9w0BAQsFAAOCAQEAnLu+d1ZsUTiX3YWGueTHIalW4ad0
Kupi7pYMV2nXbxNGmdJMol9BkzVz9tj55ReMghUU4YLm/ICYe4fz5e0T8o9s/vIm
cGS29+WoGuiznMitpvbS/379gaMezk6KpqjH6Brw6meMqy09phmcmvm3x3WTmx09
mLlQneMptwk8qSYcnMUmGLJs+cVqmkOa3sWRdw8WrGu6QqYtQz3HFZQojF06YzEq
V/dBdCFdEOwTfVl2n2XqhoJl/oEBdC4uu2G0qRk3+WVs+uwVHP0Ttsbt7TzFgZfY
yxqvOg6QoldxZVZmHHncKmETu/BqCDGJot9may31ukrx34Bu+XFMVihm0w==
-----END CERTIFICATE-----
```


The end-entity certificate is issued by the CA. This certificate grants signature authority for one IPv4 address block (192.0.2.0/24). Signature authority for AS numbers is not needed for geofeed data signatures, so no AS numbers are included in the certificate.

```
-----BEGIN CERTIFICATE-----
MIIErTCCA5WgAwIBAgIUJ605QIPX8rW5m4Zwx3WyuW7hZuMwDQYJKoZIhvcNAQEL
BQAwMzExMC8GA1UEAxMoM0FDRTJDRUY0RkIyMUI3RDExRTNFMTg0RUZDMUUyOTdC
Mzc3ODY0MjAeFw0yMDA5MDMxOTA1MTdaFw0yMTA2MzAxOTA1MTdaMDMxMTAvBgNV
BAMTKDkxNDY1MkEzQkQ1MUMxNDQyNjAxOTg4ODlGNUM0NUFCRjA1M0ExODcwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCycTQrOb/qB2W3i3Ki8PhA/DEW
yii2TgGo9pgCwO9lsIRI6Zb/k+aSiWWP9kSczlcQgtPCVwr62hTQZCIowBN0BL0c
K0/5k1imJdi5qdM3nvKswM8CnoR11vB8pQFwruZmr5xphXRvE+mzuJVLgu2V1upm
BXuWloeymudh6WWJ+GDjwPXO3RiXBejBrOFNXhaFLe08y4DPfr/S/tXJOBm7QzQp
tmbPLYtGfprYu45liFFqqP94UeLpISfXd36AKGzqTFCcc3EW9l5UFE1MFLlnoEog
qtoLoKABt0IkOFGKeC/EgeaBdWLe469ddC9rQft5w6g6cmxG+aYDdIEB34zrAgMB
AAGjggG3MIIBszAdBgNVHQ4EFgQUkUZSo71RwUQmAZiIn1xFq/BToYcwHwYDVR0j
BBgwFoAUOs4s70+yG30R4+GE78Hil7N3hkIwDAYDVR0TAQH/BAIwADAOBgNVHQ8B
Af8EBAMCB4AwGAYDVR0gAQH/BA4wDDAKBggrBgEFBQcOAjBhBgNVHR8EWjBYMFag
VKBShlByc3luYzovL3Jwa2kuZXhhbXBsZS5uZXQvcmVwb3NpdG9yeS8zQUNFMkNF
RjRGQjIxQjdEMTFFM0UxODRFRkMxRTI5N0IzNzc4NjQyLmNybDBsBggrBgEFBQcB
AQRgMF4wXAYIKwYBBQUHMAKGUHJzeW5jOi8vcnBraS5leGFtcGxlLm5ldC9yZXBv
c2l0b3J5LzNBQ0UyQ0VGNEZCMjFCN0QxMUUzRTE4NEVGQzFFMjk3QjM3Nzg2NDIu
Y2VyMCEGCCsGAQUFBwEHAQH/BBIwEDAGBAIAAQUAMAYEAgACBQAwRQYIKwYBBQUH
AQsEOTA3MDUGCCsGAQUFBzANhilodHRwczovL3JyZHAuZXhhbXBsZS5uZXQvbm90
aWZpY2F0aW9uLnhtbDANBgkqhkiG9w0BAQsFAAOCAQEABR2T0qT2V1ZlsZjj+yHP
TArIVBECZFSCdP+bJTse85TqYiblMsNS9yEu2SNbaZMNLuSSiAffYooh4nIYq/Rh
6+xGs1n427JZUokoeLtY0UUb2fIsua9JFo8YGTnpqDMGe+xnpbJ0SCSoBlJCIj+b
+YS8WXjEHt2KW6wyA/BcNS8adS2pEUwC2cs/WcwzgbttnkcnG7/wkrQ3oqzpC1ar
Kelyz7PGIIXJGy9nF8C3/aaaEpHd7UgIyvXYuCY/lqWTm97jDxgGIYGC7660mtfO
MkB8YF6kUU+td2dDQsMztcOxbzqiGnicmeJfBwG2li6O0vorW4d5iIOTKpQyqfh4
5Q==
-----END CERTIFICATE-----
```


The end-entity certificate is displayed below in detail. For brevity, the other two certificates are not.

```
   0 1197: SEQUENCE {
   4  917:  SEQUENCE {
   8    3:   [0] {
  10    1:    INTEGER 2
         :     }
  13   20:   INTEGER 27AD394083D7F2B5B99B8670C775B2B96EE166E3
  35   13:   SEQUENCE {
  37    9:    OBJECT IDENTIFIER
         :     sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  48    0:    NULL
         :     }
  50   51:   SEQUENCE {
  52   49:    SET {
  54   47:     SEQUENCE {
  56    3:      OBJECT IDENTIFIER commonName (2 5 4 3)
  61   40:      PrintableString
```

```
          :       '3ACE2CEF4FB21B7D11E3E184EFC1E297B3778642'
          :       }
          :      }
          :     }
 103   30:    SEQUENCE {
 105   13:     UTCTime 03/09/2020 19:05:17 GMT
 120   13:     UTCTime 30/06/2021 19:05:17 GMT
          :     }
 135   51:    SEQUENCE {
 137   49:     SET {
 139   47:      SEQUENCE {
 141    3:       OBJECT IDENTIFIER commonName (2 5 4 3)
 146   40:       PrintableString
          :        '914652A3BD51C144260198889F5C45ABF053A187'
          :       }
          :      }
          :     }
 188  290:    SEQUENCE {
 192   13:     SEQUENCE {
 194    9:      OBJECT IDENTIFIER rsaEncryption
          :       (1 2 840 113549 1 1 1)
 205    0:      NULL
          :      }
 207  271:     BIT STRING, encapsulates {
 212  266:      SEQUENCE {
 216  257:       INTEGER
          :        00 B2 71 34 2B 39 BF EA 07 65 B7 8B 72 A2 F0 F8
          :        40 FC 31 16 CA 28 B6 4E 01 A8 F6 98 02 C0 EF 65
          :        B0 84 48 E9 96 FF 93 E6 92 89 65 8F F6 44 9C CE
          :        57 10 82 D3 C2 57 0A FA DA 14 D0 64 22 28 C0 13
          :        74 04 BD 1C 2B 4F F9 93 58 A6 25 D8 B9 A9 D3 37
          :        9E F2 AC C0 CF 02 9E 84 75 D6 F0 7C A5 01 70 AE
          :        E6 66 AF 9C 69 85 74 6F 13 E9 B3 B8 95 4B 82 ED
          :        95 D6 EA 66 05 7B 96 96 87 B2 9A E7 61 E9 65 89
          :        F8 60 E3 C0 F5 CE DD 18 97 05 E8 C1 AC E1 4D 5E
          :        16 85 2D ED 3C CB 80 CF 7E BF D2 FE D5 C9 38 19
          :        BB 43 34 29 B6 66 CF 2D 8B 46 7E 9A D8 BB 8E 65
          :        88 51 6A A8 FF 78 51 E2 E9 21 27 D7 77 7E 80 28
          :        6C EA 4C 50 9C 73 71 16 F6 5E 54 14 4D 4C 14 B9
          :        67 A0 4A 20 AA DA 0B A0 A0 01 B7 42 24 38 51 8A
          :        78 2F C4 81 E6 81 75 62 DE E3 AF 5D 74 2F 6B 41
          :        FB 79 C3 A8 3A 72 6C 46 F9 A6 03 74 81 01 DF 8C
          :        EB
 477    3:       INTEGER 65537
          :       }
          :      }
          :     }
 482  439:    [3] {
```

```
 486 435:    SEQUENCE {
 490  29:     SEQUENCE {
 492   3:      OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 497  22:      OCTET STRING, encapsulates {
 499  20:       OCTET STRING
        :        91 46 52 A3 BD 51 C1 44 26 01 98 88 9F 5C 45 AB
        :        F0 53 A1 87
        :        }
        :       }
 521  31:     SEQUENCE {
 523   3:      OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 528  24:      OCTET STRING, encapsulates {
 530  22:       SEQUENCE {
 532  20:        [0]
        :         3A CE 2C EF 4F B2 1B 7D 11 E3 E1 84 EF C1 E2 97
        :         B3 77 86 42
        :         }
        :        }
        :       }
 554  12:     SEQUENCE {
 556   3:      OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 561   1:      BOOLEAN TRUE
 564   2:      OCTET STRING, encapsulates {
 566   0:       SEQUENCE {}
        :        }
        :       }
 568  14:     SEQUENCE {
 570   3:      OBJECT IDENTIFIER keyUsage (2 5 29 15)
 575   1:      BOOLEAN TRUE
 578   4:      OCTET STRING, encapsulates {
 580   2:       BIT STRING 7 unused bits
        :        '1'B (bit 0)
        :        }
        :       }
 584  24:     SEQUENCE {
 586   3:      OBJECT IDENTIFIER certificatePolicies (2 5 29 32)
 591   1:      BOOLEAN TRUE
 594  14:      OCTET STRING, encapsulates {
 596  12:       SEQUENCE {
 598  10:        SEQUENCE {
 600   8:         OBJECT IDENTIFIER
        :          resourceCertificatePolicy (1 3 6 1 5 5 7 14 2)
        :          }
        :         }
        :        }
        :       }
 610  97:     SEQUENCE {
 612   3:      OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
```

```
    617   90:       OCTET STRING, encapsulates {
    619   88:        SEQUENCE {
    621   86:         SEQUENCE {
    623   84:          [0] {
    625   82:           [0] {
    627   80:            [6]
            :          'rsync://rpki.example.net/repository/3ACE2CEF4F'
            :          'B21B7D11E3E184EFC1E297B3778642.crl'
            :             }
            :            }
            :           }
            :          }
            :         }
            :        }
    709  108:      SEQUENCE {
    711    8:       OBJECT IDENTIFIER authorityInfoAccess
            :         (1 3 6 1 5 5 7 1 1)
    721   96:       OCTET STRING, encapsulates {
    723   94:        SEQUENCE {
    725   92:         SEQUENCE {
    727    8:          OBJECT IDENTIFIER caIssuers (1 3 6 1 5 5 7 48 2)
    737   80:          [6]
            :           'rsync://rpki.example.net/repository/3ACE2CEF4F'
            :           'B21B7D11E3E184EFC1E297B3778642.cer'
            :           }
            :          }
            :         }
            :        }
    819   33:      SEQUENCE {
    821    8:       OBJECT IDENTIFIER ipAddrBlocks (1 3 6 1 5 5 7 1 7)
    831    1:       BOOLEAN TRUE
    834   18:       OCTET STRING, encapsulates {
    836   16:        SEQUENCE {
    838    6:         SEQUENCE {
    840    2:          OCTET STRING 00 01
    844    0:          NULL
            :           }
    846    6:         SEQUENCE {
    848    2:          OCTET STRING 00 02
    852    0:          NULL
            :           }
            :          }
            :         }
            :        }
    854   69:      SEQUENCE {
    856    8:       OBJECT IDENTIFIER subjectInfoAccess
            :         (1 3 6 1 5 5 7 1 11)
    866   57:       OCTET STRING, encapsulates {
```

```
  868   55:       SEQUENCE {
  870   53:        SEQUENCE {
  872    8:         OBJECT IDENTIFIER '1 3 6 1 5 5 7 48 13'
  882   41:         [6]
          :          'https://rrdp.example.net/notification.xml'
          :          }
          :         }
          :        }
          :       }
          :      }
          :     }
          :    }
  925   13:  SEQUENCE {
  927    9:   OBJECT IDENTIFIER sha256WithRSAEncryption
          :    (1 2 840 113549 1 1 11)
  938    0:   NULL
          :    }
  940  257:  BIT STRING
          :   05 1D 93 D2 A4 F6 57 56 65 B1 98 E3 FB 21 CF 4C
          :   0A C8 54 11 02 64 54 82 74 FF 9B 25 3B 1E F3 94
          :   EA 62 26 E5 32 C3 52 F7 21 2E D9 23 5B 69 93 0D
          :   2E E4 92 88 07 DF 62 8A 21 E2 72 18 AB F4 61 EB
          :   EC 46 B3 59 F8 DB B2 59 52 89 28 78 BB 58 D1 45
          :   1B D9 F2 2C B9 AF 49 16 8F 18 19 39 E9 A8 33 06
          :   7B EC 67 A5 B2 74 48 24 A8 06 52 42 22 3F 9B F9
          :   84 BC 59 78 C4 1E DD 8A 5B AC 32 03 F0 5C 35 2F
          :   1A 75 2D A9 11 4C 02 D9 CB 3F 59 CC 33 81 BB 6D
          :   9E 47 27 1B BF F0 92 B4 37 A2 AC E9 0B 56 AB 29
          :   E9 72 CF B3 C6 20 85 C9 1B 2F 67 17 C0 B7 FD A6
          :   9A 12 91 DD ED 48 08 CA F5 D8 B8 26 3F 96 A5 93
          :   9B DE E3 0F 18 06 21 81 82 EF AE B4 9A D7 CE 32
          :   40 7C 60 5E A4 51 4F AD 77 67 43 42 C3 33 B5 C3
          :   B1 6F 3A A2 1A 78 9C 99 E2 5F 07 01 B6 96 2E 8E
          :   D2 FA 2B 5B 87 79 88 83 93 2A 94 32 A9 F8 78 E5
          :   }
```

To allow reproduction of the signature results, the end-entity private key is provided. For brevity, the other two private keys are not.

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAsnE0Kzm/6gdlt4tyovD4QPwxFsootk4BqPaYAsDvZbCESOmW
/5Pmkollj/ZEnM5XEILTwlcK+toU0GQiKMATdAS9HCtP+ZNYpiXYuanTN57yrMDP
Ap6EddbwfKUBcK7mZq+caYV0bxPps7iVS4LtldbqZgV7lpaHsprnYellifhg48D1
zt0YlwXowazhTV4WhS3tPMuAz36/0v7VyTgZu0M0KbZmzy2LRn6a2LuOZYhRaqj/
eFHi6SEn13d+gChs6kxQnHNxFvZeVBRNTBS5Z6BKIKraC6CgAbdCJDhRingvxIHm
gXVi3uOvXXQva0H7ecOoOnJsRvmmA3SBAd+M6wIDAQABAoIBAQCyB0FeMuKm8bRo
18aKjFGSPEoZi53srIz5bvUgIi92TBLez7ZnzL6Iym26oJ+5th+lCHGO/dqlhXio
pI50C5Yc9TFbblb/ECOsuCuuqKFjZ8CD3GVsHozXKJeMM+/o5YZXQrORj6UnwT0z
ol/JE5pIGUCIgsXX6tz9s5BP3lUAvVQHsv6+vEVKLxQ3wj/1vIL8O/CN036EV0GJ
mpkwmygPjfECT9wbWo0yn3jxJb36+M/QjjUP28oNIVn/IKoPZRXnqchEbuuCJ651
IsaFSqtiThm4WZtvCH/IDq+6/dcMucmTjIRcYwW7fdHfjplllVPve9c/OmpWEQvF
t3ArWUt5AoGBANs4764yHxo4mctLIE7G7l/tf9bP4KKUiYw4R4ByEocuqMC4yhmt
MPCfOFLOQet71OWCkjP2L/7EKUe9yx7G5KmxAHY6jOjvcRkvGsl6lWFOsQ8p126M
Y9hmGzMOjtsdhAiMmOWKzjvm4WqfMgghQe+PnjjSVkgTt+7BxpIuGBAvAoGBANBg
26FF5cDLpixOd3Za1YXsOgguwCaw3Plvi7vUZRpa/zBMELEtyOebfakkIRWNm07l
nE+lAZwxm+29PTD0nqCFE91teyzjnQaLO5kkAdJiFuVV3icLOGo399FrnJbKensm
FGSli+3KxQhCNIJJfgWzq4bE0ioAMjdGbYXzIYQFAoGBAM6tuDJ36KDU+hIS6wu6
O2TPSfZhF/zPo3pCWQ78/QDb+Zdw4IEiqoBA7F4NPVLg9Y/H8UTx9r/veqe7hPOo
Ok7NpIzSmKTHkc5XfZ60Zn9OLFoKbaQ40a1kXoJdWEu2YROaUlAe9F6/Rog6PHYz
vLE5qscRbu0XQhLkN+z7bg5bAoGBAKDsbDEb/dbqbyaAYpmwhH2sdRSkphg7Niwc
DNm9qWa1J6Zw1+M87I6Q8naRREuU1IAVqqWHVLr/ROBQ6NTJ1Uc5/qFeT2XXUgkf
taMKv61tuyjZK3sTmznMh0HfzUpWjEhWnCEuB+ZYVdmO52ZGw2A75RdrILL2+9Dc
PvDXVubRAoGAdqXeSWoLxuzZXzl8rsaKrQsTYaXnOWaZieU1SL5vVe8nK257UDqZ
E3ng2j5XPTUWli+aNGFEJGRoNtcQvO60O/sFZUhu52sqq9mWVYZNh1TB5aP8X+pV
iFcZOLUvQEcN6PA+YQK5FU11rAI1M0Gm5RDnVnUl0L2xfCYxb7FzV6Y=
    -----END RSA PRIVATE KEY-----
```


Signing of "192.0.2.0/24,US,WA,Seattle," (terminated by CR and LF), yields the following detached CMS signature.

```
# RPKI Signature: 192.0.2.0/24
# MIIGlwYJKoZIhvcNAQcCoIIGiDCCBoQCAQMxDTALBglghkgBZQMEAgEwDQYLKoZ
# IhvcNAQkQAS+gggSxMIIErTCCA5WgAwIBAgIUJ605QIPX8rW5m4Zwx3WyuW7hZu
# MwDQYJKoZIhvcNAQELBQAwMzExMC8GA1UEAxMoM0FDRTJDRUY0RkIyMUI3RDExR
# TNFMTg0RUZDMUUyOTdCMzc3ODY0MjAeFw0yMDA5MDMxOTA1MTdaFw0yMTA2MzAx
# OTA1MTdaMDMxMTAvBgNVBAMTKDkxNDY1MkEzQkQ1MUMxNDQyNjAxOTg4ODlGNUM
# 0NUFCRjA1M0ExODcwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCycT
# QrOb/qB2W3i3Ki8PhA/DEWyii2TgGo9pgCwO9lsIRI6Zb/k+aSiWWP9kSczlcQg
# tPCVwr62hTQZCIowBN0BL0cK0/5k1imJdi5qdM3nvKswM8CnoR11vB8pQFwruZm
# r5xphXRvE+mzuJVLgu2V1upmBXuWloeymudh6WWJ+GDjwPXO3RiXBejBrOFNXha
# FLe08y4DPfr/S/tXJOBm7QzQptmbPLYtGfprYu45liFFqqP94UeLpISfXd36AKG
# zqTFCcc3EW9l5UFE1MFLlnoEogqtoLoKABt0IkOFGKeC/EgeaBdWLe469ddC9rQ
# ft5w6g6cmxG+aYDdIEB34zrAgMBAAGjggG3MIIBszAdBgNVHQ4EFgQUkUZSo71R
# wUQmAZiIn1xFq/BToYcwHwYDVR0jBBgwFoAUOs4s70+yG30R4+GE78Hil7N3hkI
# wDAYDVR0TAQH/BAIwADAOBgNVHQ8BAf8EBAMCB4AwGAYDVR0gAQH/BA4wDDAKBg
# grBgEFBQcOAjBhBgNVHR8EWjBYMFagVKBShlByc3luYzovL3Jwa2kuZXhhbXBsZ
# S5uZXQvcmVwb3NpdG9yeS8zQUNFMkNFRjRGQjIxQjdEMTFFM0UxODRFRkMxRTI5
# N0IzNzc4NjQyLmNybDBsBggrBgEFBQcBAQRgMF4wXAYIKwYBBQUHMAKGUHJzeW5
# jOi8vcnBraS5leGFtcGxlLm5ldC9yZXBvc2l0b3J5LzNBQ0UyQ0VGNEZCMjFCN0
# QxMUUzRTE4NEVGQzFFMjk3QjM3Nzg2NDIuY2VyMCEGCCsGAQUFBwEHAQH/BBIwE
# DAGBAIAAQUAMAYEAgACBQAwRQYIKwYBBQUHAQsEOTA3MDUGCCsGAQUFBzANhilo
# dHRwczovL3JyZHAuZXhhbXBsZS5uZXQvbm90aWZpY2F0aW9uLnhtbDANBgkqhki
# G9w0BAQsFAAOCAQEABR2T0qT2V1ZlsZjj+yHPTArIVBECZFSCdP+bJTse85TqYi
# blMsNS9yEu2SNbaZMNLuSSiAffYooh4nIYq/Rh6+xGs1n427JZUokoeLtY0UUb2
# fIsua9JFo8YGTnpqDMGe+xnpbJ0SCSoBlJCIj+b+YS8WXjEHt2KW6wyA/BcNS8a
# dS2pEUwC2cs/WcwzgbttnkcnG7/wkrQ3oqzpC1arKelyz7PGIIXJGy9nF8C3/aa
# aEpHd7UgIyvXYuCY/lqWTm97jDxgGIYGC7660mtfOMkB8YF6kUU+td2dDQsMztc
# OxbzqiGnicmeJfBwG2li6O0vorW4d5iIOTKpQyqfh45TGCAaowggGmAgEDgBSRR
# lKjvVHBRCYBmIifXEWr8FOhhzALBglghkgBZQMEAgGgazAaBgkqhkiG9w0BCQMx
# DQYLKoZIhvcNAQkQAS8wHAYJKoZIhvcNAQkFMQ8XDTIwMDkxMzE4NDUxMFowLwY
# JKoZIhvcNAQkEMSIEICvi8p5S8ckg2wTRhDBQzGijjyqs5T6I+4VtBHypfcEWMA
# 0GCSqGSIb3DQEBAQUABIIBAHUrA4PaJG42BD3hpF8U0usnV3Dg5NQh97SfyKTk7
# YHhhwu/936gkmAew8ODRTCddMvMObWkjj7/XeR+WKffaTF1EAdZ1L6REV+GlV91
# cYnFkT9ldn4wHQnNNncfAehk5PClYUUQ0gqjdJT1hdaolT83b3ttekyYIiwPmHE
# xRaNkSvKenlNqcriaaf3rbQy9dc2d1KxrL2429n134ICqjKeRnHkXXrCWDmyv/3
# imwYkXpiMxw44EZqDjl36MiWsRDLdgoijBBcGbibwyAfGeR46k5raZCGvxG+4xa
# O8PDTxTfIYwAnBjRBKAqAZ7yX5xHfm58jUXsZJ7Ileq1S7G6Kk=
# End Signature: 192.0.2.0/24
```


Authors' Addresses

Massimo Candela
NTT
Siriusdreef 70-72
Hoofddorp 2132 WT
Netherlands

Email: massimo@ntt.net

Randy Bush
IIJ & Arrcus
5147 Crystal Springs
Bainbridge Island, Washington 98110
United States of America

Email: randy@psg.com

Warren Kumari
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: warren@kumari.net

Russ Housley
Vigil Security, LLC
516 Dranesville Road
Herndon, VA 20170
USA

Email: housley@vigilsec.com