OPSAWG Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 27, 2021

B. Wu
Q. Wu
Huawei
M. Boucadair
Orange
O. Gonzalez de Dios
Telefonica
B. Wen
Comcast
C. Liu
China Unicom
H. Xu
China Telecom

July 26, 2020

A YANG Model for Network and VPN Service Performance Monitoring
draft-www-opsawg-yang-vpn-service-pm-01

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.

Table of Contents

1. Introduction

[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.

[RFC8299] defines a YANG model for L3VPN Service Delivery. Three types of VPN service topologies are supported in [RFC8299]: "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.

[RFC4176] provides a framework for L3VPN operations and management. Section 2.2.4 of that document describes performance management. 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 [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 Assurance Module

The module defined in this document is a Network and VPN Service assurance module that can be used to monitor and manage the network performance on the topology at higher layer or the service topology

between VPN sites and it is an augmentation to the "ietf-network" and "ietf-network-topology" YANG data model [RFC8345].

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

4. 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.

- 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.

5. Some Model Usage Guidelines

An SP must be able to manage the capabilities and characteristics of the network/VPN services when Network connection is established or VPN sites are setup to communicate with each other.

5.1. Performance Monitoring Data Source

As described in Section 4, once the mapping between the VPN Service topology and the underlying physical network has been setup, 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 information can be fed into data source such as the management system or network devices. The measurement interval and report interval associated with these performance data usually depends on configuration parameters.

5.2. 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.

5.3. 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.

6. Data Model Sructure

This document defines the YANG module "ietf-network-vpn-pm", which has the tree structure described in the following sub-sections.

6.1. Network Level

```
module: ietf-network-vpn-pm
  augment /nw:networks/nw:network/nw:network-types:
    +--rw network-technology-type*   identityref
  augment /nw:networks/nw:network:
    +--rw vpn-attributes
    |  +--rw vpn-topo?                    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 3: Network Level View of the hierarchies

For VPN service performance monitoring, this model defines only the following minimal set of Network level network topology attributes:

o "network-technology-type": Indicates the network technology type such as L3VPN, L2VPN, ISIS, or OSPF. If the "network-technology-type" is "VPN type" (e.g.,L3VPN, L2VPN), the "vpn-topo" MUST be set.

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).

o "vpn-summary-statistics": VPN summary statistics, IPv4 statistics, and IPv6 statistics have been specified separately.

For network performance monitoring, the attributes of "Network Level" that defined in [RFC8345] do not need to be extended.

6.2. Node Level

```
augment /nw:networks/nw:network/nw:node:
    +--rw node-attributes
    |  +--rw node-type?   identityref
    |  +--rw site-id?     string
    |  +--rw site-role?   Identityref
```

Figure 4: Node Level View of the hierarchies

The Network and VPN service performance monitoring model defines only the following minimal set of Node level network topology attributes and constraints:

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.

o "site-role" (Constraint): Defines the role of the site in a particular VPN topology.

6.3. Link and Termination Point Level

```
     augment /nw:networks/nw:network/nt:link:
       +--rw link-type?                        identityref
       +--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:

- "link-type" (Attribute): Indicates the type of the link, such as GRE or IP-in-IP.

- "low-percentile": Indicates low percentile to report. Setting low-percentile into 0.00 indicates the client is not intererested in receiving low percentile.

- "middle-percentile": Indicates middle percentile to report. Setting middle-percentile into 0.00 indicates the client is not intererested in receiving middle percentile.

- "high-percentile": Indicates high percentile to report. Setting low-percentile into 0.00 indicates the client is not intererested in receiving high percentile.

- 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.

- 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..

- 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:

- 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.

- 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.

7. 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-
network-vpn-pm">
                    <node-type>pe</node-type>
                   </node-attribtues>
                   <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-
network-topology">
                    <tp-id>1-0-1</tp-id>
                   <tp-telemetry-attributes
xmlns="urn:ietf:params:xml:ns:yang:ietf-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-
network-vpn-pm">
                    <node-type>pe</node-type>
                   </node-attribtues>
                   <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-
network-topology">
                    <tp-id>2-0-1</tp-id>
                   <tp-telemetry-attributes
xmlns="urn:ietf:params:xml:ns:yang:ietf-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</
period>
      </establish-subscription>
    </rpc>
```

8. 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-
topology">
           <tp-id>1-0-1</tp-id>
           <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-
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-network-vpn-
pm">
          <node-type>pe</node-type>
          </node-attribtues>
          <termination-point xmlns="urn:ietf:params:xml:ns:yang:ietf-network-
topology">
           <tp-id>2-0-1</tp-id>
           <tp-telemetry-attributes xmlns="urn:ietf:params:xml:ns:yang:ietf-
network-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-pm">
           <loss-statistics>
            <packet-loss-count>120</packet-loss-count>
           </loss-statistics>
          </telemetry-attributes>
         </link>
       </network>
     </report>
   </rpc>
```

9. Network and VPN Service Assurance YANG Module

This module uses types defined in [RFC8345], [RFC8299] and [RFC8532].

```
<CODE BEGINS> file "ietf-network-vpn-pm@2020-04-17.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 BESS Working Group";
   contact
     "Editor: Qin Wu
              <bill.wu@huawei.com>
      Editor: Mohamed Boucadair
              <mohamed.boucadair@orange.com>";
   description
     "This module defines a model for the 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 2019-04-17 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Model for Network and VPN Service Performance
                  Monitoring";
```

```
  }

  identity ospf {
    base vpn-common:service-type;
    description
      "Identity for OSPF network type.";
  }

  identity isis {
    base vpn-common:service-type;
    description
      "Identity for ISIS network type.";
  }

  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
      "Augment the network-types with service topologyies types";
    leaf-list network-technology-type {
      type identityref {
        base vpn-common:service-type;
      }
      description
        "Identify the network technology type, e.g., L3VPN,
         L2VPN, ISIS, OSPF.";
    }
  }
  augment "/nw:networks/nw:network" {
    description
      "Augment the network with service topology attributes";
    container vpn-topo-attributes {
      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.";
    }
    uses vpn-summary-statistics;
  }
  augment "/nw:networks/nw:network/nw:node" {
    description
      "Augment the network node with overlay topology attributes";
    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.";
    }
  }
  augment "/nw:networks/nw:network/nt:link" {
    description
      "Augment the network topology link with overlay topology attributes";
     leaf link-type {
        type identityref {
          base link-type;
        }
        description
          "Link type, e.g., GRE,VXLAN, IP in IP.";
      }
     leaf low-percentile {
       type percentile;
       default 10.00;
       description
         "Low percentile to report.Setting low-percentile into 0.00 indicates
          the client is not intererested in receiving low percentile.";
     }
     leaf middle-percentile {
       type percentile;
       default 50.00;
       description
         "Middle percentile to report.Setting middle-percentile into 0.00
indicates
         the client is not intererested 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-
percentile
         into 0.00 indicates the client is not intererested in receiving high
percentile.";
       }
```

```
      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>
```

10. 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

11. 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
```

12. Acknowledgements

Thanks to Adrian Farrel for reviewing this draft and providing important input to this document.

13. Contributors

Michale Wang
Huawei
Email:wangzitao@huawei.com

Roni Even
Huawei
Email: ron.even.tlv@gmail.com

14. References

14.1. 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>.

14.2. 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.

[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