A YANG Model for VPN Service Performance Monitoring

draft-www-bess-yang-vpn-service-pm-01

Abstract

As specified in [RFC8345], 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 VPN Service Performance Monitoring that can be used to monitor and manage network Performance between VPN sites and it is an augmentation to the I2RS network topology YANG data model.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 25, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents...
1. Introduction

[RFC8345] defines an abstract YANG data model for network/service topologies and inventories. Service topology in [RFC8345] includes the a virtual topology for a service layer above the L1, L2, and L3 layers. This virtual 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.

In [RFC8299], the 3 types of VPN service topologies proposed for L3 VPN service data model are any to any, hub and spoke, hub and spoke disjoint. These VPN topology types can be used to describe how VPN sites are communicating with each other.

This document defines a YANG Model for VPN Service Performance Monitoring that can be used to monitor and manage network Performance Monitoring.
between VPN sites and it is an augmentation to the I2RS network topology YANG data model.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying [RFC2119] significance.

The following notations are used within the data tree and carry the meaning as below.

Each node is printed as:

<status> <flags> <name> <opts> <type>

<status> is one of:
+ for current

<flags> is one of:

   rw for configuration data
   ro for non-configuration data
   -x for rpcs
   -n for notifications
   -w for writable

<name> is the name of the node

If the node is augmented into the tree from another module, its name is printed as <prefix>::<name>.

<opts> is one of:

   ? for an optional leaf or choice
   ! for a presence container
   * for a leaf-list or list
   [<keys>] for a list’s keys
   (choice)/:(case) Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":"
   <type> is the name of the type for leafs and leaf-lists
3. VPN Service Topology Overview

As specified in [RFC8345], the data model defined in [RFC8345] can describe vertical layering relationships between networks that can be augmented to cover network/service topologies. The following figure describes relationships between L3VPN Service Topo and Underlying network:

```
    VPN-1           VPN-2
      /               \
 L3VPN-Service-topology 1 L3VPN-Service-topology-2
      /               \
 Site-1A site-1B site1-C site-2A Site-2B Site-2C
      |               |
    CE  CE  CE  CE  CE  CE
      |               |
      PE  PE  PE  PE  PE  PE

```  

layering relationships between L3VPN Service Topo and Underlying network

As shown in figure 1, the Site-1,A,B,C are mapped to node 1,2,3 while Site-2 A,B,C are mapped to node 4,5,6 in the underlying physical network. In this figure, an L3SM has two VPN services topologies with both built on top of one common underlying physical network.

VPN-svc 1: supporting hub-spoke communication for Customer 1 connecting the customers access at 3 sites

VPN-svc 2: supporting hub-spoke disjoint communication for Customer 2 connecting the customers access at 3 sites

L3VPN service topology 1 is hub and spoke topology while L3VPN service topology 2 is hub and spoke disjoint topology. In L3VPN service topology1, Site-1 A plays the role of hub while Site-2 B and C plays the role of spoke. In L3VPN service topoogy2, Site-2 A and B play the role of hub while Site-2 C plays the role of spoke.
4. VPN service assurance model

This module describes VPN Service assurance that can be used to monitor and manage network Performance between VPN sites and it is a augmentation to the I2RS network topology YANG data model. The performance monitoring data is augmented to service topology.

```
+------------+          +---------------------+
|I2RS Network|          |    VPN Service      |
|Topo Model  |<--------|Performance Monitoring|
+------------+ augments|        Model         |
+---------------------+
```

5. Model Usage Guideline

An SP must be able to manage the capabilities and characteristics of their VPN services when VPN sites are setup to communicate with each other. VPN service topology such as hub and spoke describes how these VPN sites are communicating with each other.

5.1. Performance Monitoring Data Source

As described in section 2, once the mapping between VPN Service topology and 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 VPN service such as end to end network performance data between VPN sites can be aggregated or calculated using PCEP solution [RFC5440] or LMAP solution [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 I2RS Pub/Sub

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 VPN service performance data they are interested in, at the data source.

The data source can then use VPN service assurance model and push model [I-D.ietf-netconf-yang-push] to publish 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 the network element, service-assurance applications can also use polling based solution such as RPC model to fetch performance data on demand.

6. Design of the Data Model

This document defines the YANG module "ietf-vpn-svc-pm", which has the following structure

6.1. Network Level

module: ietf-vpn-svc-pm
  augment /nw:networks/nw:network/nw:network-types:
    +--rw svc-topo-type? identityref
  augment /nw:networks/nw:network:
    +--rw svc-topo-attributes
      +--rw vpn-topo? identityref

Network Level View of the hierarchies

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

  o svc-topo-type: Indicate the network type is service topology type such as L3VPN service topology, L2VPN service topology.

  o vpn-topo: The type of VPN service topology. Our proposed model supports any-to-any, Hub and Spoke (where Hubs can exchange traffic), and "Hub and Spoke disjoint" (where Hubs cannot exchange traffic).

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

Node Level View of the hierarchies

The VPN service performance monitoring model defines only the following minimal set of Node level service topology attributes and constraints:
- Node-type (Attribute): Indicate the type of the node, such as PE or ASBR.

- 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.

6.3. Link and Termination Point Level

augment /nw:networks/nw:network/nt:link:
++-ro svc-telemetry-attributes
  +++-ro loss-statistics
    |  +++-ro direction                 identityref
    |  +++-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 min-delay-value?           uint32
    |  +++-ro max-delay-value?           uint32
    |  +++-ro average-delay-value?       uint32
  +++-ro jitter-statistics
    |  +++-ro direction?                identityref
    |  +++-ro min-jitter-value?          uint32
    |  +++-ro max-jitter-value?          uint32
    |  +++-ro average-jitter-value?      uint32

Link and Termination point Level View of the hierarchies

The VPN service performance monitoring model defines only the following minimal set of Link level service topology attributes:

Loss Statistics: A set of loss statistics attributes that are used to measure end to end loss between VPN sites.

Delay Statistics: A set of delay statistics attributes that are used to measure end to end latency between VPN sites.

Jitter Statistics: A set of jitter statistics attributes that are used to measure end to end jitter between VPN sites.
7. Example of I2RS Pub/Sub Retrieval

This example shows the way for a client to subscribe for the performance monitoring information for VPN service between VPN sites. The performance monitoring parameter that the client is interested in is end to end loss attribute.

```xml
<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>
         <network>
           <network-id>vpn1</network-id>
           <node>
             <node-id>A</node-id>
             <node-type xmlns="urn:ietf:params:xml:ns:yang:ietf-svc-topo">pe</node-type>
           </node>
           <node>
             <node-id>B</node-id>
             <node-type xmlns="urn:ietf:params:xml:ns:yang:ietf-svc-topo">pe</node-type>
           </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>
             <svc-telemetry-attributes
                 xmlns="urn:ietf:params:xml:ns:yang:ietf-svc-topo">
               <loss-statistics>
                 <packet-loss-count/>
               </loss-statistics>
             </svc-telemetry-attributes>
           </link>
         </network>
       </networks>
     </stream-subtree-filter>
   </establish-subscription>
</rpc>
```
8. Example of RPC model 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 VPN1.

```xml
<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">
      <network>
        <network-id>vpn1</network-id>
        <node>
          <node-id>A</node-id>
          <node-type xmlns="urn:ietf:params:xml:ns:yang:ietf-svc-topo">pe</node-type>
        </node>
        <node>
          <node-id>B</node-id>
          <node-type xmlns="urn:ietf:params:xml:ns:yang:ietf-svc-topo">pe</node-type>
        </node>
        <link-id>A-B</link-id>
        <source>
          <source-node>A</source-node>
        </source>
        <destination>
          <dest-node>B</dest-node>
        </destination>
          <loss-statistics>
            <packet-loss-count/>
          </loss-statistics>
        </svc-telemetry-attributes>
      </link>
    </networks>
  </report>
</rpc>
```

9. VPN Service Assurance YANG Module

```xml
<CODE BEGINS> file "ietf-vpn-svc-pm.yang"
module ietf-vpn-svc-pm {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-vpn-svc-pm";
  import ietf-network {
    prefix nw;
  }
  import ietf-network-topology {
    prefix nt;
  }
}</CODE BEGINS>
```
import ietf-l3vpn-svc {
  prefix l3vpn-svc;
}

organization
  "IETF xxx Working Group";
contact
  "Zitao Wang: wangzitao@huawei.com
   Qin Wu: bill.wu@huawei.com";
description
  "This module defines a model for the service topology.";

revision 2018-08-29 {
  description
    "Initial revision.";
  reference "foo";
}

identity service-type {
  description
    "Base type for service topology";
}

identity l3vpn-svc {
  base service-type;
  description
    "Identity for layer3 vpn service";
}

identity l2vpn-svc {
  base service-type;
  description
    "Identity for layer2 vpn service";
}

identity node-type {
  description
    "Base identity for node type";
}

identity pe {
  base node-type;
  description
    "Identity for PE type";
}

identity ce {
  base node-type;
description
  "Identity for CE type";
}

identity asbr {
  base node-type;
  description
    "Identity for ASBR type";
}

identity p {
  base node-type;
  description
    "Identity for P type";
}

identity direction {
  description
    "Base Identity for measurement direction including
    one way measurement and two way measurement.";
}

identity oneway {
  base direction;
  description
    "Identity for one way measurement.";
}

identity twoway {
  base direction;
  description
    "Identity for two way measurement.";
}

typedef percentage {
  type decimal64 {
    fraction-digits 5;
    range "0..100";
  }
  description
    "Percentage.";
}

grouping link-error-statistics {
  description
    "Grouping for per link error statistics";
  container loss-statistics {
    description
"Per link loss statistics."
leaf direction {
  type identityref {
    base direction;
  }
  default "oneway";
  description
    "Define measurement direction including one way
    measurement and two way measurement.";
}
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 "oneway";
      description
        "Define measurement direction including one way measurement and two way measurement.";
    }
    leaf min-delay-value {
      type uint32;
      description
        "Minimum delay value observed.";
    }
    leaf max-delay-value {
      type uint32;
      description
        "Maximum delay value observed.";
    }
}
leaf average-delay-value {
    type uint32;
    description
        "Average delay value observed."
}
}

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) as defined in RFC3393.";
        leaf direction {
            type identityref {
                base direction;
            }
            default "oneway";
            description
                "Define measurement direction including one way
                measurement and two way measurement.";
        }
        leaf min-jitter-value {
            type uint32;
            description
                "Minimum jitter value observed."
        }
        leaf max-jitter-value {
            type uint32;
            description
                "Maximum jitter value observed."
        }
        leaf average-jitter-value {
            type uint32;
            description
                "Average jitter value observed."
        }
    }
}

augment "/nw:networks/nw:network/nw:network-types" {
    description
        "Augment the network-types with service topology types";
    leaf svc-topo-type {
        type identityref {
            base svc-topo-type;
            default "internet"
        }
    }
}
base service-type;
}
description
"Identify the topology type to be composited service topology";
}
}
augment "/nw:networks/nw:network" {
description
"Augment the network with service topology attributes";
container svc-topo-attributes {
leaf vpn-topology {
  type identityref {
    base l3vpn-svc:vpn-topology;
  }
description
  "VPN service topology, e.g. hub-spoke, any-to-any, hub-spoke-disjoint, etc";
}description
  "Container for vpn services";
}
}
augment "/nw:networks/nw:network/nw:node" {
description
"Augment the network node with service attributes";
container node-attributes {
  leaf node-type {
    type identityref {
      base node-type;
    }
description
      "Node type, e.g. PE, P, ASBR, etc";
  }
leaf site-id {
  type string;
description
    "Associated vpn site";
}leaf site-role {
  type identityref {
    base l3vpn-svc:site-role;
  }default "l3vpn-svc:any-to-any-role";
description
    "Role of the site in the IP VPN";
}description
    "Container for service topology attributes";
}
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:

- /ni:network-instances/ni:network-instance/svc-topo:svc-telemetry-
  attributes
11. IANA Considerations

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

```
---
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
---
```

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

```
---
Name:         ietf-vpn-svc-pm
Prefix:       vnrsc
Reference:    RFC xxxx
---
```

12. Normative References


[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, 
DOI 10.17487/RFC3688, January 2004, 

DOI 10.17487/RFC5440, March 2009, 

DOI 10.17487/RFC6020, October 2010, 

Internet-Draft  Service Topo YANG  October 2018


Authors’ Addresses

Michael Wang
Huawei Technologies Co., Ltd
101 Software Avenue, Yuhua District
Nanjing 210012
China

Email: wangzitao@huawei.com
Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu  210012
China

Email: bill.wu@huawei.com

Roni Even
Huawei Technologies,Co.,Ltd
Tel Aviv
Israel

Email: roni.even@huawei.com

Bin Wen
Comcast

Email: bin_wen@comcast.com