

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 12, 2019

H. Chen
China Telecom
Z. Li
China Mobile
F. Xu
Tencent
Y. Gu
Z. Li
Huawei
March 11, 2019

Network-wide Protocol Monitoring (NPM): Use Cases
draft-chen-npm-use-cases-00

Abstract

As networks continue to scale, we need a coordinated effort for diagnosing control plane health issues in heterogeneous environments. Traditionally, operators developed internal solutions to address the identification and remediation of control plane health issues, but as networks increase in size, speed and dynamicity, new methods and techniques will be required.

This document highlights key network health issues, as well as network planning requirements, identified by leading network operators. It also provides an overview of current art and techniques that are used, but highlights key deficiencies and areas for improvement.

This document proposes a unified management framework for coordinating diagnostics of control plane problems and optimization of network design. Furthermore, it outlines requirements for collecting, storing and analyzing control plane data, to minimise or negate control plane problems that may significantly affect overall network performance and to optimize path/peering/policy planning for meeting application-specific demands.

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 September 12, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Role of Telemetry	3
1.2. Role of Control Plane Telemetry	4
2. Terminology	5
3. Problem Statement	5
3.1. Network Troubleshooting Challenges	5
3.2. Network Planning Challenges	7
4. Network-wide Protocol Monitoring (NPM)	7
5. NPM Use Cases	9
5.1. Network Troubleshooting Use Cases	9
5.1.1. IS-IS Route Flapping	10
5.1.2. LSDB Synchronization Failure	11
5.1.3. Route Loop	11
5.1.4. Tunnel Set Up Failure	12

5.2. Network Planning Use Cases	12
5.2.1. Route Policy Validation	12
6. Security Considerations	13
7. Contributors	13
8. Acknowledgments	13
9. References	13
Authors' Addresses	16

1. Introduction

Recently, significant effort has been made to evolve control network resources, using management plane enhancements and control of network state via centralized and distributed control plane methods. There is ongoing effort in the diagnosing of forwarding plane performance degradation, using telemetry-based solutions and in-band data plane OAM. However, less emphasis has been applied on the diagnosing and remediation of health problems related to optimal control of network resources, and diagnosing control plane health issues.

The document outlines the existing set of standards-based tools and highlights the lack of capability for addressing control plane monitoring.

1.1. Role of Telemetry

The concept of network telemetry has been proposed to meet the current and future OAM demands, supporting real-time data collection, process, exportation, and analysis, and an architectural framework of existing Telemetry approaches is introduced in [I-D.song-ntf] [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 Telemetry: The management plane telemetry focuses on network operational state retrieval and configuration management. 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 widely adopted management plane Telemetry approaches. Data consumers can subscribe to specific data stores through SNMP/gRPC/NETCONF.
- o Control Plane Telemetry: The control plane telemetry works on routing protocol monitoring and routing related data retrieval, e.g., topology, route policy, RIB and so on. BGP monitoring

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

- o Data Plane Telemetry: The data plane telemetry works on traffic performance measurement and traffic related data retrieval, e.g., latency, jitter, buffer size and so on. 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 user packet at each network node along the forwarding path. Applications such as path verification, SLA (service-level agreement) assurance can be enabled with iOAM.

1.2. Role of Control Plane Telemetry

The above mentioned telemetry approaches may vary in data type and form, including: encapsulation, serialization, transportation, subscription, and data analysis, thus resulting in various applications. With the network operations and maintenance evolving towards automation and intent-driven, higher requirements are set for each plane. Healthy management plane and control plane are essential for high-quality data service provisioning. The visibility of management and control planes' healthiness provides insights for changes in the data plane.

First of all, the running of control protocols aims to provide and guarantee the network connectivity and reachability, which is the foundation of any data service running above it. The monitoring of the control plane detects the healthiness issue in real time so that immediate troubleshooting actions can be taken, and thus mitigating the affect on data services as much as possible.

Secondly, without route analytics, the dynamic nature of IP networking makes it virtually impossible to know at any time point how traffic is traversing the networks. For example, by collecting real-time BGP routes through BMP and correlating them with traffic data retrieved through data plane telemetry, the operator is able to provide both inter-domain and intra-domain traffic optimization.

Finally, the validation and evaluation of route policies is another common appeal from both carriers and OTTs. The difficulty here majorly lies in the precise definition of the correctness of policies. In other words, the policy validation depends largely on the operator's understanding and manual judgement of the current network status instead of formatted and quantitative command executed

at devices. Thus, it demands visualized presentations of how the policies impact the route changes through control plane telemetry so that operators may have direct judgement of the policy correctness. The conventional separated data collections of route policy and route information is not sufficient for the correctness validation of route policy.

Based on discussions with leading operators, this document identifies the challenges and problems that the current control plane telemetry faces and suggests the data collection requirements. The necessity for a Network-wide Protocol Monitoring (NPM) framework is illustrated and conducted through the discussion of specific use cases.

2. Terminology

IGP: Interior Gateway Protocol

IS-IS: Intermediate System to Intermediate System

BGP: Boarder Gateway Protocol

BGP-LS: Boarder Gateway Protocol-Link State

MPLS: Multi-Protocol Label Switching

RSVP-TE: Resource Reservation Protocol-Traffic Engineering

LDP: Label Distribution Protocol

NPM: Network-wide Protocol Monitoring

NPMS: Network Protocol Monitoring System

BMP: BGP Monitoring Protocol

LSP: Link State Packet

SDN: Software Defined Network

IPFIX: Internet Protocol Flow Information Export

3. Problem Statement

3.1. Network Troubleshooting Challenges

According to Huawei 2016 network issue statistics, about 48% issues of the total amount are routing protocol-related, including protocol adjacency/peer set up failure, adjacency/peer flapping, protocol-

related table error. What's more, the routing protocol issues are not standalone, which simultaneously come with anomaly status in data plane, and are finally reflected on poor service quality and user experience.

Existing methods for protocol troubleshooting include CLI, SNMP, Netconf-YANG/gRPC-YANG and vendor-specific/third party tools.

Using CLI to do per-device check provides adequate per device information, but lacks network-wide vision, thus leading to either massive labor/time consumption checking all devices or fail to localize the source. Besides, complex CLI usage (combination and repeat pattern) requires experience from the NOC person.

Management protocols, like SNMP, Netconf/gRPC, provide information already/to be gathered from the network, which reduces operational complexity, but sacrifices data adequacy compared with CLI. Since the above protocols aren't designed specifically for routing troubleshooting, not all the data source required is currently supported for exportation, and the lack of certain data becomes the troubleshooting bottleneck. For example, in an LSP purge abnormal case caused by continuous corrupted LSP, it's useful to collect the corrupted LSP PDUs for root cause analysis. In addition, for the currently supported, as well as to be supported, data source collection, the data synchronization issue, due to export performance difference of various approaches, can be a concern for data correlation. The data collection requirements depend largely on the use cases, and more details are discussed in Section 5.

Some third party OAM tools provide troubleshooting-customized information collection and analysis. For example, Packet Design uses passive listening to collect IS-IS/OSPF/BGP messages to do route analysis for troubleshooting and path optimization. Such passive listening lacks per-device information collection. For example, to detect the existence of a route loop and analyze the root cause, it not only requires the network-wide RIB/FIB collection, but also requires the route policy information that is responsible for the generation of loop issue.

To summarize here, the currently protocols and tools do not provide sufficient data source for routing troubleshooting. There requires new methods or augmented work to existing methods to enhance the control plane data collection and to support more efficient data correlation.

3.2. Network Planning Challenges

The dynamic nature of IP networks, e.g., peer up/down, prefix advertisement, route change, and so on, has great influence on the service provisioning. With the emerging of new network services, such as automated driving systems, AR (Augmented Reality), and so on, network planning is facing new requirements in order to meet the latency, bandwidth and security demands. The requirements can generally break into two perspectives: 1. sufficient and up-to-date routing data collection as the input for network simulation; 2. accurate what-if simulation to evaluate new network planning actions.

Most existing control plane and data plane simulation tools, e.g., Batfish [Batfish], use device configurations to generate a control/data plane. There exists some concerns w.r.t. such simulation method: 1. in a multi-vendor network understanding and translating the configuration files is a non-trivial task for the simulator; 2. the generated control/control plane is not the 100% mirroring of the actual network, and thus resulting in less accurate simulation results. Thus, it requires real-time routing data collection from the on-going network. Currently, BGP routes and peering states are monitored in real-time by using BMP. However, IS-IS/OSPF/MPLS routing data still lacks legitimate and comprehensive monitoring. Here, not only the data coverage, including RIB/FIB, network topology, peering states and so on, but also the data synchronization of various devices should be considered in order to recover a faithful data/control plane within the simulator.

4. Network-wide Protocol Monitoring (NPM)

With the above mentioned challenges facing the control plane telemetry, it is of great value to identify the requirements from typical use cases, and the gaps between the requirements and existing methods. It is thus necessary to propose a comprehensive control plane telemetry framework, as shown in Figure 1.

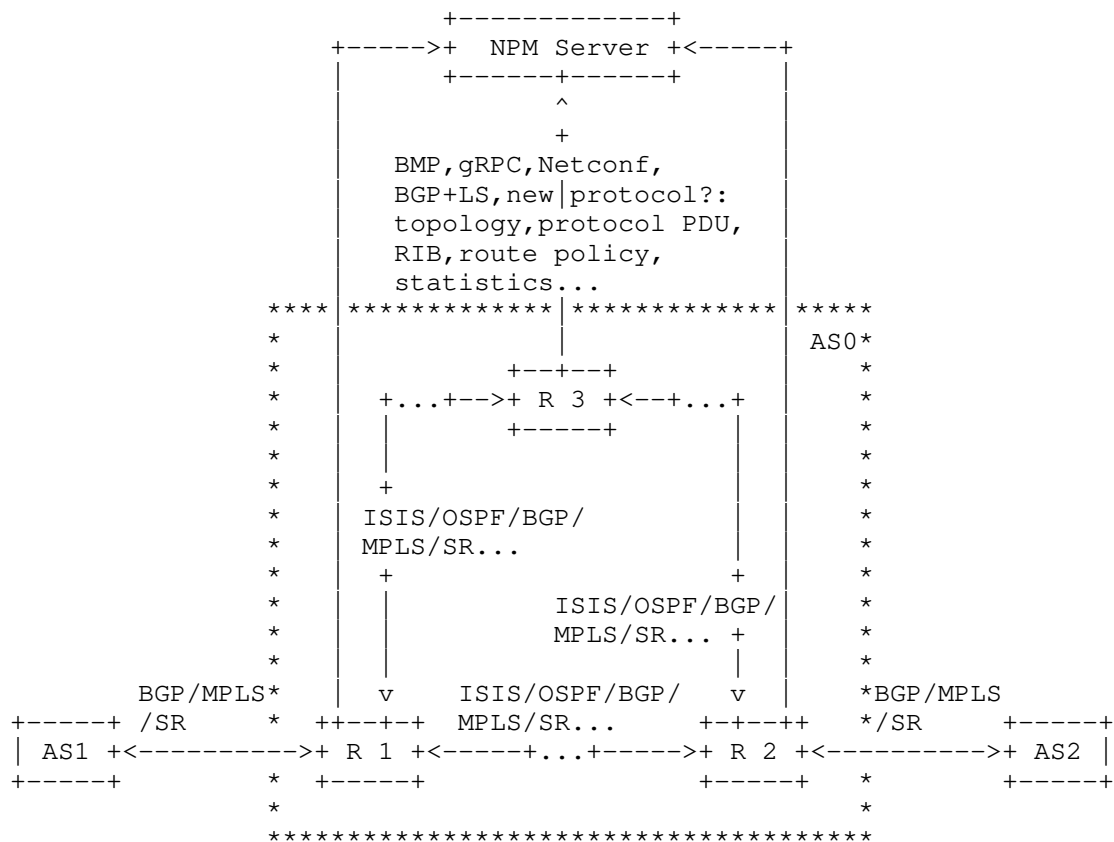


Figure 1: NPM framework

Under the NPM framework, the challenges, use cases, requirements, gaps, and solutions options are to be identified and discussed. The NPM problem space is depicted in Figure 2. Two general requirements are concluded from the challenges discussed above.

- o The requirement of a "tunnel" for the control plane data export: There should be a way (or ways) of exporting the required control plane data, and the export performance (e.g., data modeling, encoding and transmission) should be able to meet per application requirements;
- o The requirement of adequate data collection: In order to support specific troubleshooting and planning use cases, the collected data coverage, including the data type coverage and the network coverage, should be adequate. The data type coverage refers to data such as protocol PDUs, RIBs, policy and so on, and the

network type coverage refers to the devices providing such information.

More specific requirements may vary case by case, but it is a common appeal to guarantee a valid tunnel and adequate data collection.

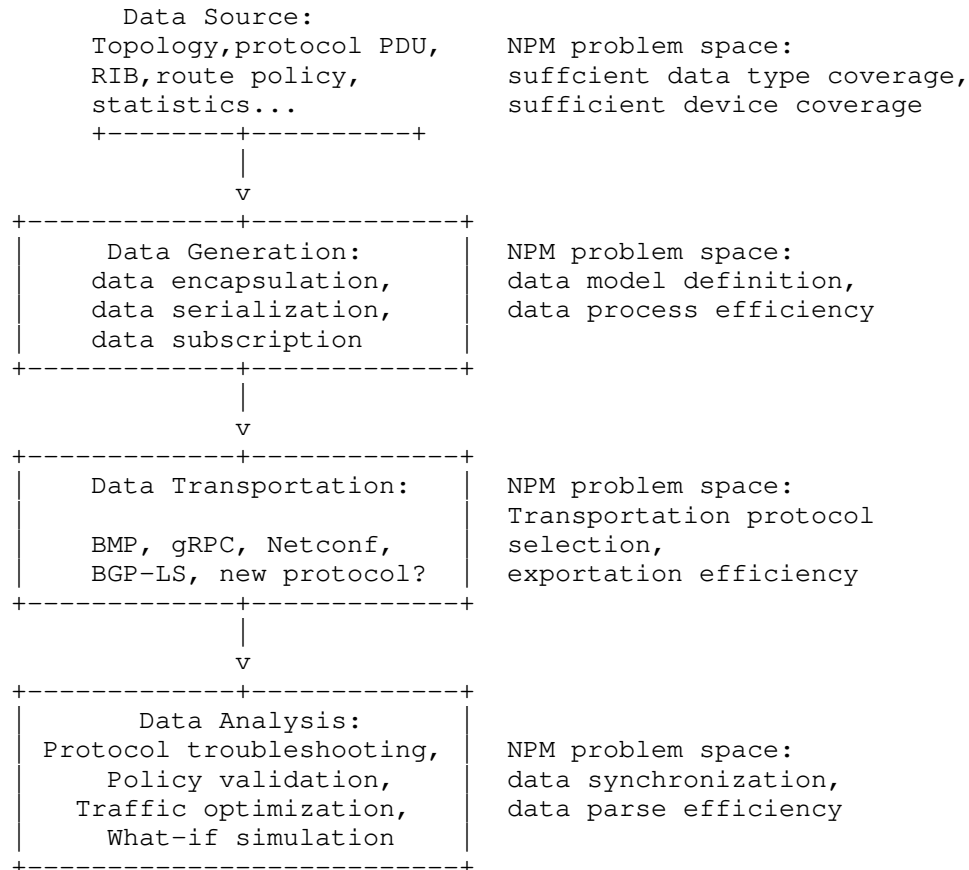


Figure 2: NPM problem space

5. NPM Use Cases

5.1. Network Troubleshooting Use Cases

We have identified several typical routing issues that occur frequently in the network, and are typically hard to localize.

5.1.1. 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 locate 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.
- o 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.
- o 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 Originator 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.

5.1.2. 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:

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

5.1.3. Route Loop

Incorrect import policy, such as incorrect protocol priority (distance) or improper default route configuration, may result in a route loop. TTL anomaly report or packet loss complain triggers loop alarm. However, locating the exact device(s) and more importantly the responsible configuration/policy is definitely non-trivial work. The generation of routing information base/forwarding information base (RIB/FIB) is related to various protocols and massive route

policies, which often makes it hard to locate the loop source in a timely manner.

If the network-wide RIB/FIB data can be collected in real-time, the server is able to run loop detection algorithms to detect and locate the loop. More importantly, with real-time RIB/FIB collected as the input for network simulator, loop can be predicted with what-if simulations of network changes, such as new policy, or link failure.

5.1.4. Tunnel Set Up Failure

The MPLS label switch path set up, either using RSVP-TE or LDP, may fail due to various reasons. Typical troubleshooting procedures are to log in the device, and then check if the failure lies on the configuration, or path computation error, or link failure. Sometimes, it requires the check of multiple devices along the tunnel. Certain reason codes can be carried in the Path-Err/ResvErr messages of RSVP-TE, while other data are currently not supported to be transmitted to the path ingress/egress node, such as the authentication failure. In this case, if the tunnel configurations of devices along the tunnel, as well as the link states, and other reasons diagnosed by each device can be collected centrally, the server is able to do a thorough analysis and find the root cause.

5.2. Network Planning Use Cases

Monitoring and analyzing the network routing events not only help identify the root causes of network issues, but also provide visibility of how routing changes affect network traffic. With the benefit of data plane telemetry, such as iOAM and IPFIX, network traffic matrices can be generated to give a glance of the current network performance. More specifically, traffic matrices visualize the current and historical network changes, such as link utilization, link delay, jitter, and so on. While traffic matrices provide "what" are the network changes, the control plane event monitoring, such as adjacency/peering failure, route flapping, prefix advertize/withdraw, provides "why".

5.2.1. Route Policy Validation

Route policy validation has been a great concern for operators when implementing new policies as well as optimizing existing policies. Validation comes in two perspectives:

- o Firstly, there requires valid monitoring of implemented policy correlated with network changes to understand how one policy impacts routing in both single-device and network-wide views. Conventionally, policy/configuration data collection (e.g.,

through Netconf/YANG) is separate from route information collection (e.g., BMP), which lacks correlation between policy and routes. Thus, even with both information at hand, it is still difficult for the operator to figure out how a policy impacts the route change. If the route change is recorded correlated with policy processing, the server can directly identify the impact through the correlation analysis of such data collected from all devices.

- o Secondly, there requires pre-check of policy impact using simulation tools. Most existing simulation tools use device configurations to generate a control plane/data plane, and then run what-if simulations to evaluate a new policy. However, there exists difference between the on-going network and the generated control/data plane, and thus leading the simulation results less effective. If the control/data plane snapshot (e.g., topology, protocol neighbor state, RIB...) of the on going network is realized and taken as the input of the simulation, the reliability of the evaluation can be greatly improved.

6. Security Considerations

TBD

7. Contributors

TBD

8. Acknowledgments

TBD

9. References

[Batfish] etc., A. F., "A General Approach to Network Configuration Analysis", May 2015.

[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.ietf-grow-bmp-adj-rib-out]
Evens, T., Bayraktar, S., Lucente, P., Mi, K., and S. Zhuang, "Support for Adj-RIB-Out in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-adj-rib-out-03 (work in progress), December 2018.
- [I-D.ietf-grow-bmp-local-rib]
Evens, T., Bayraktar, S., Bhardwaj, M., and P. Lucente, "Support for Local RIB in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-local-rib-02 (work in progress), September 2018.
- [I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-22 (work in progress), February 2019.
- [I-D.openconfig-rtgw-gnmi-spec]
Shakir, R., Shaikh, A., Borman, P., Hines, M., Lebsack, C., and C. Morrow, "gRPC Network Management Interface (gNMI)", draft-openconfig-rtgw-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>>.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <<https://www.rfc-editor.org/info/rfc1191>>.
- [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>>.
- [RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, DOI 10.17487/RFC1213, March 1991, <<https://www.rfc-editor.org/info/rfc1213>>.

- [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>>.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC3719] Parker, J., Ed., "Recommendations for Interoperable Networks using Intermediate System to Intermediate System (IS-IS)", RFC 3719, DOI 10.17487/RFC3719, February 2004, <<https://www.rfc-editor.org/info/rfc3719>>.
- [RFC3988] Black, B. and K. Kompella, "Maximum Transmission Unit Signalling Extensions for the Label Distribution Protocol", RFC 3988, DOI 10.17487/RFC3988, January 2005, <<https://www.rfc-editor.org/info/rfc3988>>.
- [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>>.
- [RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE", RFC 8231, DOI 10.17487/RFC8231, September 2017, <<https://www.rfc-editor.org/info/rfc8231>>.

Authors' Addresses

Huanan Chen
China Telecom
109 West Zhongshan Ave
Guangzhou
China

Email: chenhuanan@gsta.com

Zhenqiang Li
China Mobile
No. 32 Xuanwumenxi Ave., Xicheng District
Beijing
China

Email: lizhenqiang@chinamobile.com

Feng Xu
Tencent
Guangzhou
China

Email: oliverxu@tencent.com

Yunan Gu
Huawei
156 Beiqing Rd
Beijing
China

Email: guyunan@huawei.com

Zhenbin Li
Huawei
156 Beiqing Rd
Beijing
China

Email: lizhenbin@huawei.com

OPSAWG Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 9, 2019

R. Even
B. Wu
Q. Wu
Huawei
Y. Cheng
China Unicom
March 8, 2019

YANG Data Model for Composed VPN Service Delivery
draft-evenwu-opsawg-yang-composed-vpn-03

Abstract

This document defines a YANG data model that can be used by a network operator to configure a VPN service that spans multiple administrative domains and that is constructed from component VPNs in each of those administrative domains. The component VPNs may be L2VPN or L3VPN or a mixture of the two. This model is intended to be instantiated at the management system to deliver the end to end service (i.e., performing service provision and activation functions at different levels through a unified interface).

The model is not a configuration model to be used directly on network elements. This model provides an abstracted common view of VPN service configuration components segmented at different layer and administrative domain. It is up to a management system to take this as an input and generate specific configurations models to configure the different network elements within each administrative domain to deliver the service. How configuration of network elements is done is out of scope of the document.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 9, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
1.1. Terminology	3
1.1.1. Requirements Language	4
1.2. Tree diagram	4
2. Definitions	4
3. Service Model Usage	6
4. The Composed VPN Service Model	7
4.1. VPN Service Types	7
4.2. Composed VPN Physical Network Topology	7
5. Design of the Data Model	9
5.1. VPN Hierarchy	15
5.2. Access Point (AP)	16
5.2.1. AP peering with CE	16
5.2.2. AP peering for inter-domains connection	17
6. Composed VPN YANG Module	19
7. Segment VPN YANG Module	21
8. Service Model Usage Example	53
9. Interaction with other YANG models	56
10. Security Considerations	57
11. IANA Considerations	58
12. References	58
12.1. Normative References	58
12.2. Informative References	60
Appendix A. Acknowledges	60
Authors' Addresses	60

1. Introduction

In some cases, a VPN service needs to span different administrative domains. This will usually arise when there are internal administrative boundaries within a single Service Provider's (SP's)

network. The boundaries may reflect geographic dispersal or functional decomposition, e.g., access, metro, backhaul, core, and data center.

In particular, the different domains could deploy Layer 2 or Layer 3 technologies or both, and could establish layer-dependent connectivity services. For example, some SPs offer a L2VPN service in the metro access network and extend it across the core network as an IP VPN to provide end-to-end BGP IP VPN services to their enterprise customers.

Some SPs integrate Mobile Backhaul Network and Core networks to provide mobile broadband services. These require stitching multiple layer-dependent connectivity services at different administrative domain boundaries.

This document defines a YANG data model that can be used by a network operator to construct an end-to-end service across multiple administrative domains. This service is delivered by provisioning VPN services utilising Layer 2 or Layer 3 technologies in each domain.

This model is intended to be instantiated at the management system to deliver the overall service per [RFC8309]. It is not a configuration model to be used directly on network elements. This model provides an abstracted common view of VPN service configuration components segmented at different layers and administrative domains. It is up to a management system to take this as an input and generate specific configurations models to configure the different network elements within each administrative domain to deliver the service. How configuration of network elements is done is out of scope of the document. END

1.1. Terminology

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

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

The following terms are defined in [RFC7950] and are not redefined here:

- o augment
- o data model
- o data node

The terminology for describing YANG data models is found in [RFC7950].

1.1.1. 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] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Tree diagram

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

2. Definitions

This document uses the following terms:

Service Provider (SP): The organization (usually a commercial undertaking) responsible for operating the network that offers VPN services to clients and customers.

Customer Edge (CE) Device: Equipment that is dedicated to a particular customer and is directly connected to one or more PE devices via attachment circuits. A CE is usually located at the customer premises, and is usually dedicated to a single VPN, although it may support multiple VPNs if each one has separate attachment circuits. The CE devices can be routers, bridges, switches, or hosts.

Provider Edge (PE) Device: Equipment managed by the SP that can support multiple VPNs for different customers, and is directly connected to one or more CE devices via attachment circuits. A PE is usually located at an SP point of presence (PoP) and is managed by the SP.

Administrative Domain: A collection of End Systems, Intermediate Systems, and subnetworks operated by a single organization or administrative authority. The components which make up the domain are assumed to interoperate with a significant degree of mutual

trust among themselves, but interoperate with other Administrative Domains in a mutually suspicious manner [RFC1136].

A group of hosts, routers, and networks operated and managed by a single organization. Routing within an Administrative Domain is based on a consistent technical plan. An Administrative Domain is viewed from the outside, for purposes of routing, as a cohesive entity, of which the internal structure is unimportant. Information passed by other Administrative Domains is trusted less than information from one's own Administrative Domain.

Administrative Domains can be organized into a loose hierarchy that reflects the availability and authoritativeness of routing information. This hierarchy does not imply administrative containment, nor does it imply a strict tree topology.

Routing Domain: A set of End Systems and Intermediate Systems which operate according to the same routing procedures and which is wholly contained within a single Administrative Domain [RFC1136].

A Routing Domain is a set of Intermediate Systems and End Systems bound by a common routing procedure; namely: they are using the same set of routing metrics, they use compatible metric measurement techniques, they use the same information distribution protocol, and they use the same path computation algorithm" An Administrative Domain may contain multiple Routing Domains. A Routing Domain may never span multiple Administrative Domains.

An Administrative Domain may consist of only a single Routing Domain, in which case they are said to be Congruent. A congruent Administrative Domain and Routing Domain is analogous to an Internet Autonomous System.

Access point (AP): Describe an VPN's end point characteristics and its reference to a Termination Point (TP) of the Provider Edge (PE) Node; used as service access point for connectivity service segment in the end-to-end manner and per administrative domain.

Site: Represent a connection of a customer office to one or more VPN services and contain a list of network accesses associated with the site. Each network access can connect to different VPN service.

Segment VPN Describe generic information about a VPN in a single administrative domain, and specific information about APs that connect the Segment VPN to sites or to other Segment VPNs.

Composed VPN Describe generic end-to-end information about a VPN that spans multiple administrative domains, and specific customer-facing information about APs connecting to each site.

3. Service Model Usage

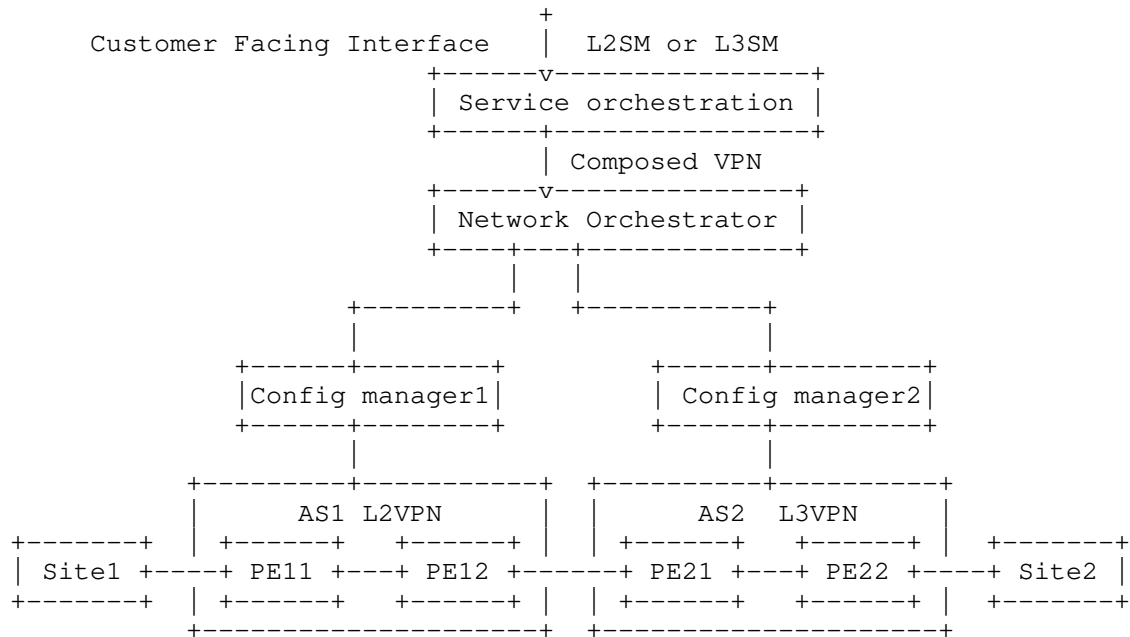


Figure 1: Service Model Usage

In the above use case, the network orchestrator controls and manages the two distinct network domains, each controlled or managed by their own management system or domain controller. There are two typical ways to deploy the composed VPN model:

One typical scenario would be to use the model as an independent model. The orchestration layer could use composed VPN model as an input, and translate it to segmented VPN model for each administrative domain. And the domain management system could further configure network elements based on configuration obtained from the segment VPN.

The other scenario is to use customer facing model such as L3SM service model as an input for the service orchestration layer that will be responsible for translating the parameters of VPN and site in L3SM model to the corresponding parameters of the composed VPN model, then with extra provisioning parameters added ,the composed VPN model

can be further broken down into per domain segmented VPN model and additional Access point configuration.

The usage of this composed VPN model is not limited to this example; it can be used by any component of the management system but not directly by network elements.

4. The Composed VPN Service Model

A composed VPN represents an end-to-end IP or Ethernet connectivity between the access points of PE where the AP can interconnect with the enterprise customer's network or other types of overlay network. The Composed VPN model provides a common understanding of how the corresponding composed VPN service is to be deployed in an end to end manner over the multi-domain infrastructure.

This document presents the Composed VPN Service Delivery Model using the YANG data modeling language [RFC7950] as a formal language that is both human-readable and parsable by software for use with protocols such as NETCONF [RFC6241] and RESTCONF [RFC8040].

4.1. VPN Service Types

From a technology perspective, a Composed VPN can be classified into three categories based on the domain specific VPN types including L2VPN and L3VPN, see Figure 2. And in each category, the interworking option may vary depending on the inter-domain technology, such as IP or MPLS forwarding. In some cases, the number of transit domain can be zero or multiple.

Composed VPN	Domain 1 (source)	Domain 2 (transit)	..	Domain N (dest)	Interworking Option
L3VPN	L2VPN	L2VPN	..	L3VPN	Option A
L3VPN	L3VPN	L3VPN	..	L3VPN	OptionA/B/C
L2VPN	L2VPN	L2VPN	..	L2VPN	OptionA/B/C

Figure 2: Composed VPN classification

4.2. Composed VPN Physical Network Topology

Figure 3 describes a scenario where connectivity in the form of an L3VPN is provided across a Mobile Backhaul Network. The network has two ASes: connectivity across AS A is achieved with an L2VPN, and

across AS B an L3VPN. The ASes are interconnected, and the composed VPN is achieved by interconnecting the L2VPN with the L3VPN.

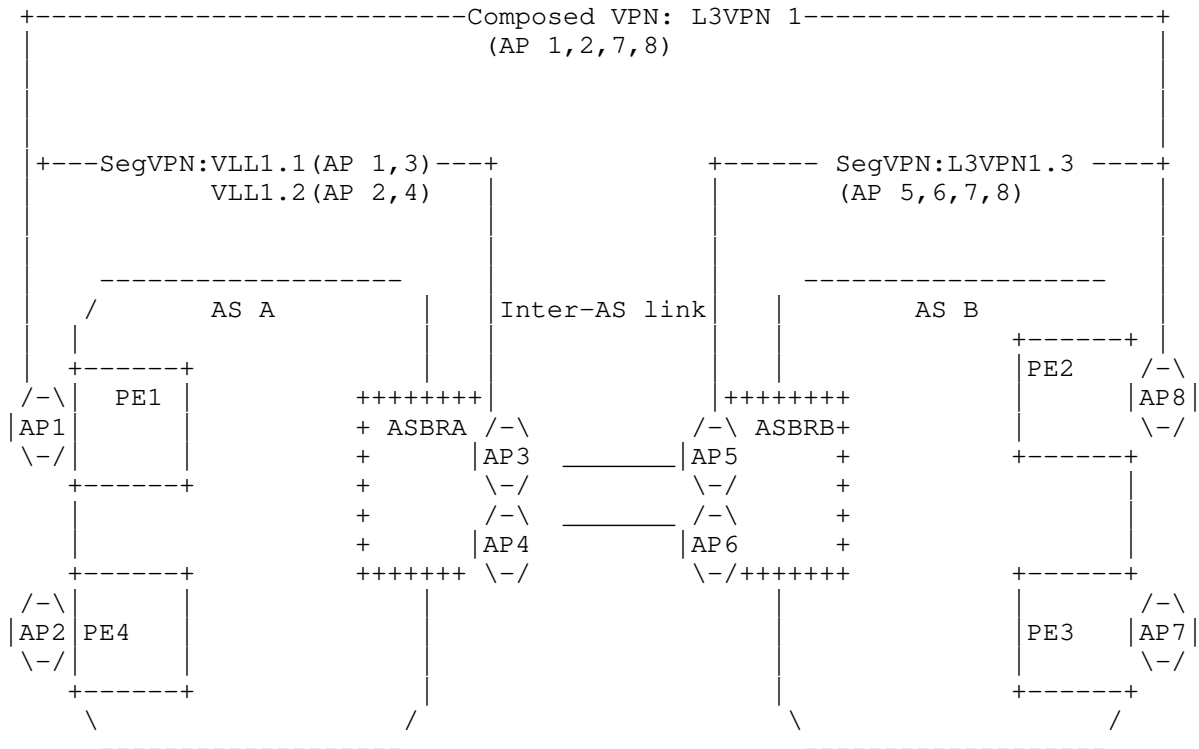


Figure 3: Mobile Backhaul Network Scenario

The Composed VPN is a service that provides connectivity between AP1, AP2, AP7 and AP8. As the APs of the VPN are spanning the two domains, the ASBR A and B and their associated links are required to be identified. Based on the decomposition, two Segment VPN could be constructed to provide per domain connections. Segment VPN 1.1 and Segment VPN 1.2 are connections between AP 1,2,3,4 in the domain A of access metro network, which are L2VPN. Segment VPN 1.3 is the connection between AP 5,6,7,8 in the core network, which is L3VPN. The ASBR A and B at the edge of the access metro network is performing the VPN stitching between Layer 2 VPN and Layer 3 VPN using the technology such as bridging or other interconnection technology.

The operator can predefine several VPN provisioning policies based on the offered business. The policy description may include the naming, path selection, VPN concatenation rules, and resource pools, such as

route target, route distinguisher. How VPN provision policies configuration of network elements is done is out of scope of the document.

5. Design of the Data Model

The idea of the composed VPN model is to decompose an end-to-end L2VPN or L3VPN service across multiple administrative domains into point-to-point VPN segments or multi-point VPN segments in each administrative domain, and to stitch these segments together by using different interworking options. Therefore, a complete composed VPN instance consists of:

- o One composed VPN with corresponding composed VPN set of parameter
- o Two or more APs, each with a corresponding set of AP parameters
- o One or more segment VPN with corresponding segment VPN set of parameter

Similar to the L3SM [RFC8299] and L2SM [RFC8466] modelling structure, the composed VPN model consists of two main components, the VPN component and the AP component.

The figure below describes the overall structure of the YANG module:

```
module: ietf-composed-vpn-svc
  +--rw composed-vpns
    +--rw composed-vpn* [vpn-id]
      +--rw vpn-id          yang:uuid
      +--rw vpn-name?       string
      +--rw customer-name?  yang:uuid
      +--rw topo?           svpn:vpn-topology
      +--rw service-type?   svpn:service-type
      +--rw tunnel-type?    svpn:tunnel-type
      +--rw admin-state?    svpn:admin-state
      +--ro oper-State?     svpn:oper-state
      +--ro sync-state?     svpn:sync-state
      +--rw start-time?     yang:date-and-time
      +--rw segment-vpn* [vpn-id]
        +--rw vpn-id          yang:uuid
        +--rw vpn-name?       string
        +--rw service-type?   service-type
        +--rw topo?           vpn-topology
        +--rw tunnel-type?    tunnel-type
        +--rw admin-state?    admin-state
        +--ro oper-state?     oper-state
        +--ro sync-state?     sync-state
```

```

+--rw access-point* [tp-id]
  +--rw tp-id                                yang:uuid
  +--rw tp-common-attribute
    +--rw tp-id?                            yang:uuid
    +--rw tp-name?                          string
    +--rw node-id?                          yang:uuid
    +--rw access-point-type?                access-point-type
    +--rw inter-as-option?                  enumeration
    +--rw topology-role?                    topology-role
  +--rw peer-remote-node
    +--rw remote-id?                        yang:uuid
    +--rw location?                          string
    +--rw remote-tp-address?                inet:ip-address
    +--rw remote-node-id?                   yang:uuid
    +--rw remote-tp-id?                     yang:uuid
  +--rw tp-connection-specific-attribute
    +--rw connection* [connection-class]
      +--rw connection-class                layer-rate
      +--rw (connection-type)?
        +--:(lr-eth)
          +--rw eth
            +--rw access-type?              eth-encap-type
            +--rw (accessVlanValue)?
              +--:(qinq)
                +--rw qinq
                  +--rw cvlan*              uint64
                  +--rw svlan?              uint64
              +--:(dot1q)
                +--rw dot1q
                  +--rw dot1q*              uint64
            +--rw vlan-action?              ethernet-action
            +--rw action?                    string
          +--:(lr-ip)
            +--rw ip
              +--rw ip-address?             inet:ip-address
              +--rw mtu?                     uint64
          +--:(lr-pw)
            +--rw pw
              +--rw control-word?            boolean
              +--rw vlan-action?             pwttagmode
        +--rw security-attribute
          +--rw security
            +--rw authentication
            +--rw encryption {encryption}?
              +--rw enabled?                  boolean
              +--rw layer?                    enumeration
              +--rw algorithm?                string
              +--rw (key-type)?

```

```

+---:(psk)
+---rw preshared-key?    string
+---rw qos-attribute
+---rw  svc-input-bandwidth    uint64
+---rw  svc-output-bandwidth   uint64
+---rw  svc-mtu                uint16
+---rw  qos {qos}?
+---rw    qos-classification-policy
+---rw      rule* [id]
+---rw        id                string
+---rw        (match-type)?
+---rw          +---:(match-flow)
+---rw            +---rw match-flow
+---rw              +---rw dscp?          inet:dscp
+---rw                +---rw exp?          inet:dscp
+---rw                  +---rw dot1p?       uint8
+---rw                    +---rw ipv4-src-prefix?  inet:ipv4-prefi
+---rw                      +---rw ipv6-src-prefix?  inet:ipv6-prefi
+---rw                        +---rw ipv4-dst-prefix?  inet:ipv4-prefi
+---rw                          +---rw ipv6-dst-prefix?  inet:ipv6-prefi
+---rw                            +---rw l4-src-port?    inet:port-numbe
+---rw                              +---rw peer-remote-node*  string
+---rw                                +---rw l4-src-port-range
+---rw                                  +---rw lower-port?  inet:port-number
+---rw                                    +---rw upper-port?  inet:port-number
+---rw                                      +---rw l4-dst-port?    inet:port-numbe
+---rw                                        +---rw l4-dst-port-range
+---rw                                          +---rw lower-port?  inet:port-number
+---rw                                            +---rw upper-port?  inet:port-number
+---rw                                              +---rw src-mac?    yang:mac-addres
+---rw                                                +---rw dst-mac?    yang:mac-addres
+---rw                                                  +---rw protocol-field?  union
+---rw                                                    +---:(match-application)
+---rw                                                      +---rw match-application?  identityref
+---rw                                                        +---rw target-class-id?    string
+---rw qos-profile
+---rw  (qos-profile)?
+---rw    +---:(standard)
+---rw      +---rw profile?    string
+---rw        +---:(custom)
+---rw          +---rw classes {qos-custom}?
+---rw            +---rw class* [class-id]
+---rw              +---rw class-id    string
+---rw                +---rw direction?  identityref
+---rw                  +---rw rate-limit?  decimal64
+---rw                    +---rw latency
+---rw                      +---rw (flavor)?
+---rw                        +---:(lowest)

```

```

| | | | | +---rw use-lowest-latency?      empty
| | | | | +---:(boundary)
| | | | | +---rw latency-boundary?        uint1
6
| | | | | +---rw jitter
| | | | | | +---rw (flavor)?
| | | | | | +---:(lowest)
| | | | | | | +---rw use-lowest-jitter?    empty
| | | | | | +---:(boundary)
| | | | | |   +---rw latency-boundary?     uint32
| | | | | +---rw bandwidth
| | | | |   +---rw guaranteed-bw-percent    decimal6
4
| | | | | +---rw end-to-end?              empty
| | | | | +---rw protection-attribute
| | | | |   +---rw access-priority?        uint32
+---rw routing-protocol* [type]
+---rw type                                protocol-type
+---rw (para)?
+---:(static)
|   +---rw static* [index]
|   |   +---rw index                      uint32
|   |   +---rw dest-cidr?                 string
|   |   +---rw egress-tp?                 yang:uuid
|   |   +---rw route-preference?          string
|   |   +---rw next-hop?                  inet:ip-address
+---:(bgp)
|   +---rw bgp* [index]
|   |   +---rw index                      uint32
|   |   +---rw autonomous-system          uint32
|   |   +---rw address-family*            address-family
|   |   +---rw max-prefix?                int32
|   |   +---rw peer-address?              inet:ip-address
|   |   +---rw crypto-algorithm            identityref
|   |   +---rw key-string
|   |       +---rw (key-string-style)?
|   |       +---:(keystring)
|   |       |   +---rw keystore?           string
|   |       +---:(hexadecimal) {hex-key-string}?
|   |       +---rw hexadecimal-string?     yang:hex-string
+---rw access-point* [tp-id]
+---rw tp-id                              yang:uuid
+---rw tp-name?                           string
+---rw node-id?                           yang:uuid
+---rw access-point-type?                 access-point-type
+---rw inter-as-option?                   enumeration
+---rw topology-role?                     topology-role
+---rw peer-remote-node
|   +---rw remote-id?                       yang:uuid
|   +---rw location?                         string

```

```

|   +--rw remote-tp-address?  inet:ip-address
|   +--rw remote-node-id?    yang:uuid
|   +--rw remote-tp-id?      yang:uuid
+--rw tp-connection-specific-attribute
|   +--rw connection* [connection-class]
|   |   +--rw connection-class  layer-rate
|   |   +--rw (connection-type)?
|   |   |   +--:(lr-eth)
|   |   |   |   +--rw eth
|   |   |   |   |   +--rw access-type?  eth-encap-type
|   |   |   |   |   +--rw (accessVlanValue)?
|   |   |   |   |   |   +--:(qinq)
|   |   |   |   |   |   |   +--rw qinq
|   |   |   |   |   |   |   |   +--rw cvlan*    uint64
|   |   |   |   |   |   |   |   +--rw svlan?    uint64
|   |   |   |   |   |   |   +--:(dot1q)
|   |   |   |   |   |   |   |   +--rw dot1q
|   |   |   |   |   |   |   |   |   +--rw dot1q*  uint64
|   |   |   |   |   |   |   +--rw vlan-action?  ethernet-action
|   |   |   |   |   |   |   +--rw action?        string
|   |   |   |   |   +--:(lr-ip)
|   |   |   |   |   |   +--rw ip
|   |   |   |   |   |   |   +--rw ip-address?    inet:ip-address
|   |   |   |   |   |   |   +--rw mtu?           uint64
|   |   |   |   |   +--:(lr-pw)
|   |   |   |   |   |   +--rw pw
|   |   |   |   |   |   |   +--rw control-word?  boolean
|   |   |   |   |   |   |   +--rw vlan-action?    pwtagmode
|   +--rw security-attribute
|   |   +--rw security
|   |   |   +--rw authentication
|   |   |   +--rw encryption {encryption}?
|   |   |   |   +--rw enabled?                boolean
|   |   |   |   +--rw layer?                  enumeration
|   |   |   |   +--rw algorithm?              string
|   |   |   |   +--rw (key-type)?
|   |   |   |   |   +--:(psk)
|   |   |   |   |   |   +--rw preshared-key?    string
|   +--rw qos-attribute
|   |   +--rw svc-input-bandwidth    uint64
|   |   +--rw svc-output-bandwidth   uint64
|   |   +--rw svc-mtu                uint16
|   |   +--rw qos {qos}?
|   |   |   +--rw qos-classification-policy
|   |   |   |   +--rw rule* [id]
|   |   |   |   |   +--rw id                                string
|   |   |   |   |   +--rw (match-type)?
|   |   |   |   |   |   +--:(match-flow)

```

```

+--rw match-flow
  +--rw dscp?          inet:dscp
  +--rw exp?           inet:dscp
  +--rw dot1p?         uint8
  +--rw ipv4-src-prefix? inet:ipv4-prefix
  +--rw ipv6-src-prefix? inet:ipv6-prefix
  +--rw ipv4-dst-prefix? inet:ipv4-prefix
  +--rw ipv6-dst-prefix? inet:ipv6-prefix
  +--rw l4-src-port?    inet:port-number
  +--rw peer-remote-node* string
  +--rw l4-src-port-range
    | +--rw lower-port?  inet:port-number
    | +--rw upper-port?  inet:port-number
  +--rw l4-dst-port?    inet:port-number
  +--rw l4-dst-port-range
    | +--rw lower-port?  inet:port-number
    | +--rw upper-port?  inet:port-number
  +--rw src-mac?        yang:mac-address
  +--rw dst-mac?        yang:mac-address
  +--rw protocol-field? union
+--:(match-application)
  +--rw match-application? identityref
+--rw target-class-id?  string
+--rw qos-profile
  +--rw (qos-profile)?
    +--:(standard)
    | +--rw profile?    string
    +--:(custom)
    +--rw classes {qos-custom}?
      +--rw class* [class-id]
        +--rw class-id    string
        +--rw direction?  identityref
        +--rw rate-limit? decimal64
        +--rw latency
          +--rw (flavor)?
            +--:(lowest)
            | +--rw use-lowest-latency?  empty
            +--:(boundary)
            | +--rw latency-boundary?    uint16
        +--rw jitter
          +--rw (flavor)?
            +--:(lowest)
            | +--rw use-lowest-jitter?    empty
            +--:(boundary)
            | +--rw latency-boundary?    uint32
        +--rw bandwidth
          +--rw guaranteed-bw-percent    decimal64
          +--rw end-to-end?              empty

```

```

    |   +--rw protection-attribute
    |       +--rw access-priority?   uint32
+--rw routing-protocol* [type]
    |   +--rw type                     protocol-type
    |   +--rw (para)?
    |       +--:(static)
    |           +--rw static* [index]
    |               +--rw index                uint32
    |               +--rw dest-cidr?           string
    |               +--rw egress-tp?          yang:uuid
    |               +--rw route-preference?   string
    |               +--rw next-hop?          inet:ip-address
    |       +--:(bgp)
    |           +--rw bgp* [index]
    |               +--rw index                uint32
    |               +--rw autonomous-system   uint32
    |               +--rw address-family*    address-family
    |               +--rw max-prefix?        int32
    |               +--rw peer-address?      inet:ip-address
    |               +--rw crypto-algorithm   identityref
    |               +--rw key-string
    |                   +--rw (key-string-style)?
    |                       +--:(keystring)
    |                           | +--rw keystring?          string
    |                           +--:(hexadecimal) {hex-key-string}?
    |                               +--rw hexadecimal-string? yang:hex-string

```

5.1. VPN Hierarchy

The composed VPN and segment VPN contain the following common parameters:

- o **vpn-id:** Refers to an internal reference for this VPN service
- o **vpn-service-type:** Combination of L3VPN service type and L2VPN service type per [RFC8466] and [RFC8299], including VPWS,VPLS,EVPN and L3VPN.
- o **vpn-topology:** Combination of L3VPN topology and L2VPN topology, including hub-spoke, any-to-any and point-to-point.
- o **Tunnel-type:**MPLS,MPLS-TP,SR,SRv6

Suppose a composed VPN is a L3VPN which could initially has sites connected to a single SP domain and may later add more sites to other domains in the SP network. Thus, a composed VPN could has one segment VPN at the beginning, and later has more segment VPNs.

5.2. Access Point (AP)

As the site containers of the L3SM and L2SM represent the connection characteristics that the CE connects to the provider network from the perspective of the customer, AP represents the connection characteristics that the PE connects to VPN from the perspective of the provider. Therefore, there are two main aspects relates to the AP modelling:

- o The AP component under composed VPN container describes the intent parameters mapping from the L3SM and L2SM, and the AP component under the segment VPN container describes the configuration parameters of the specific domain derived from the decomposition of composed VPN model.
- o In a specific segment VPN, the AP component not only describes the CE-PE connection, but also defines inter-domain connection parameters between ASBR peer. The connection between PE and ASBR is related to configuration of network elements and not part of segment VPN model.

5.2.1. AP peering with CE

The AP parameters contains the following group of parameters:

Basic AP parameters: topology role could be hub role, leaf role

Connection: has a knob to accommodate either Layer2 or Layer 3 data plane connection

Control plane peering: has a knob to accommodate either Layer 2 protocol or Layer 3 routing protocol

QoS profile and QoS-classification-policy: has a knob to accommodate either Layer 2 QoS profile and qos-classification-policy or Layer 3 QoS profile and Qos-classification-policy, to describe both per AP bandwidth and per flow QoS.

Security Policy: has a knob to accommodate either Layer 2 QoS profile and qos-classification-policy or Layer 3 QoS profile and qos-classification-policy, to describe both per AP bandwidth and per flow QoS.

Although both the composed VPN and segment VPN use the AP to describe the connection parameters of the CE and the PE, the AP parameter of the composed VPN may not be directly mapped to the AP parameters of the segment VPN. For example, a composed VPN is a L3VPN with one of its AP which specifies the IP connection parameters and per flow QoS

requirement. During decomposition, depending on the capability of the accessed domain which the segment VPN resides, the AP of the segment VPN could only support Ethernet connection and per port bandwidth guarantee. Therefore, the AP could only configure with L2 connection and per AP bandwidth setting.

5.2.2. AP peering for inter-domains connection

The AP which describes the inter-domains connection could only exist in segment VPN. There are three options in connecting segment VPN across inter-domain link. With L3VPN, L2VPN or mixture, the option could be:

Interworking Option	AP type	AP CP remote peer	AP DP
Option A	ASBR LTP	ASBR	Interface
Option B	ASBR	ASBR	LSP label
Option C	PE,ASBR	remote PE	LSP label

The AP parameters contains the following group of parameters:

Basic AP parameters: Inter-AS interworking option could be Option A, Option B or Option C.

Connection: only specifies in Options A, Option B and C use dynamically allocated MPLS labels.

Control plane peering: BGP peering or static routing.

QoS profile and QoS-classification-policy: only applicable in Options A, Option B and C can only use MPLS EXP to differentiate the traffic.

Security policy: Options A use the similar mechanism like CE-PE peering, Option B could use BGP authentication to secure control plane communication and enable mpls label security, and Option C depends on the trust between the inter-domains.

5.2.2.1. Secure inter-domain connection

This model is applied to a single SP. Although there are different domain separation, implicit trust exists between the ASs because they

have the same operational control, for example from orchestrator's perspective.

The model specifies different security parameters depending on the various Inter-AS options:

- o Option A uses interfaces or subinterfaces between autonomous system border routers (ASBRs) to keep the VPNs separate, so there is strict separation between VPNs.
- o Option B can be secured with configuration on the control plane and the data plane. On the control plane, the session can be secured by use of peer authentication of BGP with message digest 5 (MD5) and TCP Authentication Option(TCP-AO), maximum route limits per peer and per VPN, dampening, and so on. In addition, prefix filters can be deployed to control which routes can be received from the other AS. On the data plane, labeled packets are exchanged. The label is derived from the MP-eBGP session; therefore, the ASBR announcing a VPN-IPv4 prefix controls and assigns the label for each prefix it announces. On the data plane, the incoming label is then checked to verify that this label on the data plane has really been assigned on the control plane. Therefore, it is impossible to introduce fake labels from one AS to another. The Authentication parameter could be set under the BGP peering configuration. An MPLS label security could be enabled under the connection node.
- o Option C can also be secured well on the control plane, but the data plane does not provide any mechanism to check and block the packets to be sent into the other AS. On the control plane, model C has two interfaces between autonomous systems: The ASBRs exchange IPv4 routes with labels via eBGP. The purpose is to propagate the PE loopback addresses to the other AS so that LSPs can be established end to end. The other interface is the RRs exchange VPN-IPv4 routes with labels via multihop MP-eBGP. The prefixes exchanged can be controlled through route maps, equally the route targets. On the data plane, the traffic exchanged between the ASBRs contains two labels. One is VPN label set by the ingress PE to identify the VPN. The other is PE label Specifies the LSP to the egress PE. The Authentication and routing policy parameter could be set under the BGP peering configuration.

The security options supported in the model are limited but may be extended via augmentation.

5.2.2.2. Inter-domain QoS decomposition

The APs connected between the domains are aggregation points, and traffic from different CEs of the combined VPN cross-domain will interact through these aggregation points. To provide consistent QoS configuration, when several domains are involved in the provisioning of a VPN, topology, domain functionality and other factors need to be considered.

Option A can achieve most granular QoS implementation since IP traffic passes the inter-domain connection. Thus, Option A can set configuration with per sub-interface and IP DSCP. Option B and Option C only provide MPLS EXP differentiation. QoS mechanisms that are applied only to IP traffic cannot be carried.

In some cases, there is need to re-mark packets at Layer 3 to indicate whether traffic is in agreement. Because MPLS labels include 3 bits that commonly are used for QoS marking, it is possible for "tunnel DiffServ" to preserve Layer 3 DiffServ markings through a service provider's MPLS VPN cloud while still performing re-marking (via MPLS EXP bits) within the cloud to indicate in- or out-of-agreement traffic.

6. Composed VPN YANG Module

```
<CODE BEGINS> file "ietf-composed-vpn-svc.yang"
module ietf-composed-vpn-svc {
  namespace "urn:ietf:params:xml:ns:yang:ietf-composed-vpn-svc" ;
  prefix composed-vpn ;
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-segment-vpn {
    prefix segment-vpn;
  }
  organization "IETF OPSAWG Working Group";
  contact "
    WG Web:    <https://datatracker.ietf.org/wg/opsawg>
    WG List:   <mailto:netmod@ietf.org>

    Editor:    Roni Even
               <mailto:roni.even@huawei.com>
               Bo Wu
               <mailto:lane.wubo@huawei.com>
               Qin Wu
               <mailto:bill.wu@huawei.com>
               Ying Cheng
               <mailto:chengying10@chinaunicom.cn>";
```

```
description "ietf-compsted-vpn";
revision 2018-08-21 {
    reference "draft-evenwu-opsawg-yang-composed-vpn-00";
}

grouping vpn-basic {
    description "VPNBasicInfo Grouping.";
    leaf topo {
        type segment-vpn:vpn-topology;
        description "current support for full-mesh and
            point_to_multipoint(hub-spoke), others is reserved for
            future extensions." ;
    }
    leaf service-type {
        type segment-vpn:service-type;
        description "current support for mpls l3vpn/vxlan/L2VPN/hybrid
            VPN overlay, others is reserved for future extensions." ;
    }
    leaf tunnel-type {
        type segment-vpn:tunnel-type;
        description "mpls|vxlan overlay l3vpn|eth over sdh|nop";
    }
    leaf admin-state {
        type segment-vpn:admin-state;
        description "administrative status." ;
    }
    leaf oper-State {
        type segment-vpn:oper-state;
        config false;
        description "Operational status." ;
    }
    leaf sync-state {
        type segment-vpn:sync-state;
        config false;
        description "Sync status." ;
    }
    leaf start-time {
        type yang:date-and-time;
        description "Service lifecycle: request for service start
            time." ;
    }
}

container composed-vpns{
    description "";
    list composed-vpn {
        key "vpn-id";
        description "List for composed VPNs.";
```

```
        uses composedvpn;
    }
}

grouping composedvpn {
    description "ComposedVPN Grouping.";
    leaf vpn-id {
        type yang:uuid;
        description "Composed VPN identifier." ;
    }
    leaf vpn-name {
        type string {length "0..200";}
        description "Composed VPN Name. Local administration meaning" ;
    }
    leaf customer-name {
        type yang:uuid;
        description
            "Name of the customer that actually uses the VPN service.
            In the case that any intermediary (e.g., Tier-2 provider
            or partner) sells the VPN service to their end user
            on behalf of the original service provider (e.g., Tier-1
            provider), the original service provider may require the
            customer name to provide smooth activation/commissioning
            and operation for the service." ;
    }
    uses vpn-basic;
    list segment-vpn {
        key "vpn-id";
        description "SegVpn list ";
        uses segment-vpn:VPN;
    }
    list access-point {
        key "tp-id";
        description "TP list of the access links which associated
        with CE and PE";
        uses segment-vpn:pe-termination-point;
    }
}
}
<CODE ENDS>
```

7. Segment VPN YANG Module

```
<CODE BEGINS> file "ietf-segment-vpn.yang"
module ietf-segment-vpn {
    namespace "urn:ietf:params:xml:ns:yang:ietf-segment-vpn";
    prefix segment-vpn;
```

```
import ietf-yang-types {
  prefix yang;
}
import ietf-inet-types {
  prefix inet;
}
import ietf-key-chain {
  prefix keychain;
}
import ietf-netconf-acm {
  prefix nacm;
}

organization
  "IETF OPSAWG Working Group";
contact
  "WG Web:   <https://datatracker.ietf.org/wg/opsawg>
  WG List:  <mailto:netmod@ietf.org>

  Editor:
    Roni Even
      <mailto:roni.even@huawei.com>
    Bo Wu
      <mailto:lane.wubo@huawei.com>
    Qin Wu
      <mailto:bill.wu@huawei.com>
    Cheng Ying
      <mailto:chengying10@chinaunicom.cn>";
description
  "This YANG module defines a generic service configuration
  model for segment VPNs.";

revision 2019-01-30 {
  reference
    "draft-opsawg-evenwu-yang-composed-vpn-02";
}

feature encryption {
  description
    "Enables support of encryption.";
}

feature qos {
  description
    "Enables support of classes of services.";
}

feature qos-custom {
```

```
    description
      "Enables support of the custom QoS profile.";
  }

  feature hex-key-string {
    description
      "Support hexadecimal key string.";
  }

  identity protocol-type {
    description
      "Base identity for protocol field type.";
  }

  identity tcp {
    base protocol-type;
    description
      "TCP protocol type.";
  }

  identity udp {
    base protocol-type;
    description
      "UDP protocol type.";
  }

  identity icmp {
    base protocol-type;
    description
      "ICMP protocol type.";
  }

  identity icmp6 {
    base protocol-type;
    description
      "ICMPv6 protocol type.";
  }

  identity gre {
    base protocol-type;
    description
      "GRE protocol type.";
  }

  identity ipip {
    base protocol-type;
    description
      "IP-in-IP protocol type.";
```

```
}

identity hop-by-hop {
  base protocol-type;
  description
    "Hop-by-Hop IPv6 header type.";
}

identity routing {
  base protocol-type;
  description
    "Routing IPv6 header type.";
}

identity esp {
  base protocol-type;
  description
    "ESP header type.";
}

identity ah {
  base protocol-type;
  description
    "AH header type.";
}

identity customer-application {
  description
    "Base identity for customer application.";
}

identity web {
  base customer-application;
  description
    "Identity for Web application (e.g., HTTP, HTTPS).";
}

identity mail {
  base customer-application;
  description
    "Identity for mail application.";
}

identity file-transfer {
  base customer-application;
  description
    "Identity for file transfer application (e.g., FTP, SFTP).";
}
```



```
identity database {
  base customer-application;
  description
    "Identity for database application.";
}

identity social {
  base customer-application;
  description
    "Identity for social-network application.";
}

identity games {
  base customer-application;
  description
    "Identity for gaming application.";
}

identity p2p {
  base customer-application;
  description
    "Identity for peer-to-peer application.";
}

identity network-management {
  base customer-application;
  description
    "Identity for management application
      (e.g., Telnet, syslog, SNMP).";
}

identity voice {
  base customer-application;
  description
    "Identity for voice application.";
}

identity video {
  base customer-application;
  description
    "Identity for video conference application.";
}

identity qos-profile-direction {
  description
    "Base identity for QoS profile direction.";
}
```

```
identity outbound {
  base qos-profile-direction;
  description
    "Identity for outbound direction.";
}

identity inbound {
  base qos-profile-direction;
  description
    "Identity for inbound direction.";
}

identity both {
  base qos-profile-direction;
  description
    "Identity for both inbound direction
    and outbound direction.";
}

typedef access-point-type {
  type enumeration {
    enum ce-peering {
      description
        "indicates access type with connection to CE";
    }
    enum remote-as-peering {
      description
        "indicates access type with connection to ASBR with opion A,B,C ";
    }
  }
  description
    "The access-point-type could be peering with CE or ASBR
    depending on which network that a PE interconnects with.";
}

typedef bgp-password-type {
  type string;
  description
    "Authentication Type (None, Simple Password, Keyed MD5,
    Meticulous Keyed MD5, Keyed SHA1, Meticulous Keyed SHA1";
}

typedef topology-role {
  type enumeration {
    enum hub {
      description
        "hub";
    }
  }
}
```

```
    enum spoke {
        description
            "spoke";
    }
    enum other {
        description
            "other";
    }
}
description
    "Topo Node Role.";
}

typedef qos-config-type {
    type enumeration {
        enum template {
            description
                "standard.";
        }
        enum customer {
            description
                "custom.";
        }
    }
    description
        "Qos Config Type.";
}

typedef address-family {
    type enumeration {
        enum ipv4 {
            description
                "IPv4 address family.";
        }
        enum ipv6 {
            description
                "IPv6 address family.";
        }
    }
    description
        "Defines a type for the address family.";
}

typedef tp-type {
    type enumeration {
        enum phys-tp {
            description
                "Physical termination point";
        }
    }
}
```

```
    }
    enum ctp {
        description
            "CTP";
    }
    enum trunk {
        description
            "TRUNK";
    }
    enum loopback {
        description
            "LoopBack";
    }
    enum tppool {
        description
            "TPPool";
    }
}
description
    "Tp Type.";
}

typedef layer-rate {
    type enumeration {
        enum lr-unknown {
            description
                "Layer Rate UNKNOW.";
        }
        enum lr-ip {
            description
                "Layer Rate IP.";
        }
        enum lr-eth {
            description
                "Layer Rate Ethernet.";
        }
        enum lr_vxlan {
            description
                "Layer Rate VXLAN.";
        }
    }
}
description
    "Layer Rate.";
}

typedef admin-state {
    type enumeration {
        enum active {
```

```
        description
            "Active status";
    }
    enum inactive {
        description
            "Inactive status";
    }
    enum partial {
        description
            "Partial status";
    }
}
description
    "Admin State.";
}

typedef oper-state {
    type enumeration {
        enum up {
            description
                "Up status";
        }
        enum down {
            description
                "Down status";
        }
        enum degrade {
            description
                "Degrade status";
        }
    }
}
description
    "Operational Status.";
}

typedef sync-state {
    type enumeration {
        enum sync {
            description
                "Sync status";
        }
        enum out-sync {
            description
                "Out sync status";
        }
    }
}
description
    "Sync Status";
```

```
}

typedef eth-encap-type {
  type enumeration {
    enum default {
      description
        "DEFAULT";
    }
    enum dot1q {
      description
        "DOT1Q";
    }
    enum qinq {
      description
        "QINQ";
    }
    enum untag {
      description
        "UNTAG";
    }
  }
  description
    "Ethernet Encap Type.";
}

typedef protocol-type {
  type enumeration {
    enum static {
      description
        "Static Routing";
    }
    enum bgp {
      description
        "bgp";
    }
    enum rip {
      description
        "rip";
    }
    enum ospf {
      description
        "ospf";
    }
    enum isis {
      description
        "isis";
    }
  }
}
```

```
    description
      "Routing Protocol Type";
  }

  typedef tunnel-type {
    type enumeration {
      enum MPLS {
        description
          "MPLS";
      }
      enum MPLS-TP {
        description
          "MPLS-TP";
      }
      enum MPLS-SR {
        description
          "MPLS Segment Routing";
      }
      enum SRv6 {
        description
          "SRv6";
      }
    }
    description
      "VPN Tunnel Type.";
  }

  typedef service-type {
    type enumeration {
      enum l3vpn {
        description
          "l3vpn";
      }
      enum l2vpn {
        description
          "l2vpn";
      }
    }
    description
      "VPN Service Type.";
  }

  typedef vpn-topology {
    type enumeration {
      enum point-to-point {
        description
          "point to point";
      }
    }
  }
```

```
    enum any-to-any {
        description
            "any to any";
    }
    enum hub-spoke {
        description
            "hub and spoke VPN topology.";
    }
    enum hub-spoke-disjoint {
        description
            "Hub and spoke VPN topology where
            Hubs cannot communicate with each other ";
    }
}
description
    "Topology.";
}

typedef ethernet-action {
    type enumeration {
        enum nop {
            description
                "nop";
        }
        enum untag {
            description
                "UNTAG";
        }
        enum stacking {
            description
                "STACKING";
        }
    }
}
description
    "Ethernet Action.";
}

typedef color-type {
    type enumeration {
        enum green {
            description
                "green";
        }
        enum yellow {
            description
                "yellow";
        }
        enum red {
```



```
        description
            "red";
    }
    enum all {
        description
            "all";
    }
}
description
    "Color Type.";
}

typedef action-type {
    type enumeration {
        enum nop {
            description
                "nop";
        }
        enum bandwidth {
            description
                "bandwidth";
        }
        enum pass {
            description
                "pass";
        }
        enum discard {
            description
                "discard";
        }
        enum remark {
            description
                "remark";
        }
        enum redirect {
            description
                "redirect";
        }
        enum recolor {
            description
                "recolor";
        }
        enum addRt {
            description
                "addRt";
        }
    }
}
description
```

```
        "Action Type";
    }

    typedef pwttagmode {
        type enumeration {
            enum raw {
                description
                    "RAW";
            }
            enum tagged {
                description
                    "TAGGED";
            }
        }
        description
            "PWTagMode";
    }

    grouping QinQVlan {
        description
            "QinQVlan Grouping.";
        leaf-list cvlan {
            type uint64;
            description
                "cvlan List.";
        }
        leaf svlan {
            type uint64;
            description
                "svlan.";
        }
    }

    grouping Dot1QVlan {
        description
            "Dot1QVlan Grouping.";
        leaf-list dot1q {
            type uint64;
            description
                "dot1q Vlan List";
        }
    }

    grouping tp-connection-type {
        description
            "Tp Type Spec Grouping.";
        choice connection-type {
            description
```

```
        "Spec Value";
    case lr-eth {
        container eth {
            description
                "ethernetSpec";
            uses ethernet-spec;
        }
    }
    case lr-ip {
        container ip {
            description
                "ipSpec";
            uses ipspec;
        }
    }
    case lr-pw {
        container pw {
            description
                "PwSpec";
            uses pwspec;
        }
    }
}

grouping security-authentication {
    container authentication {
        description
            "Authentication parameters.";
    }
    description
        "This grouping defines authentication parameters for a site.";
}

grouping security-encryption {
    container encryption {
        if-feature "encryption";
        leaf enabled {
            type boolean;
            default "false";
            description
                "If true, traffic encryption on the connection is required.";
        }
        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.";
  }
  leaf algorithm {
    type string;
    description
      "Encryption algorithm to be used.";
  }
  choice key-type {
    default "psk";
    case psk {
      leaf preshared-key {
        type string;
        description
          " Pre-Shared Key (PSK) coming from customer.";
      }
    }
    description
      "Type of keys to be used.";
  }
  description
    "Encryption parameters.";
}
description
  "This grouping defines encryption parameters for a site.";

grouping security-attribute {
  container security {
    uses security-authentication;
    uses security-encryption;
    description
      "Site-specific security parameters.";
  }
  description
    "Grouping for security parameters.";
```

```
}

grouping flow-definition {
  container match-flow {
    leaf dscp {
      type inet:dscp;
      description
        "DSCP value.";
    }
    leaf exp {
      type inet:dscp;
      description
        "EXP value.";
    }
    leaf dot1p {
      type uint8 {
        range "0..7";
      }
      description
        "802.1p matching.";
    }
    leaf ipv4-src-prefix {
      type inet:ipv4-prefix;
      description
        "Match on IPv4 src address.";
    }
    leaf ipv6-src-prefix {
      type inet:ipv6-prefix;
      description
        "Match on IPv6 src address.";
    }
    leaf ipv4-dst-prefix {
      type inet:ipv4-prefix;
      description
        "Match on IPv4 dst address.";
    }
    leaf ipv6-dst-prefix {
      type inet:ipv6-prefix;
      description
        "Match on IPv6 dst address.";
    }
    leaf l4-src-port {
      type inet:port-number;
      must 'current() < ../l4-src-port-range/lower-port or current() > ../l4-s
rc-port-range/upper-port' {
        description
          "If l4-src-port and l4-src-port-range/lower-port and
upper-port are set at the same time, l4-src-port
should not overlap with l4-src-port-range.";
      }
    }
  }
}
```

```
    }
    description
      "Match on Layer 4 src port.";
  }
  leaf-list peer-remote-node {
    type string;
    description
      "Identify a peer remote node as traffic destination.";
  }
  container l4-src-port-range {
    leaf lower-port {
      type inet:port-number;
      description
        "Lower boundary for port.";
    }
    leaf upper-port {
      type inet:port-number;
      must ' . >= ../lower-port ' {
        description
          "Upper boundary for port. If it
           exists, the upper boundary must be
           higher than the lower boundary.";
      }
      description
        "Upper boundary for port.";
    }
  }
  description
    "Match on Layer 4 src port range. When
     only the lower-port is present, it represents
     a single port. When both the lower-port and
     upper-port are specified, it implies
     a range inclusive of both values.";
}
leaf l4-dst-port {
  type inet:port-number;
  must 'current() < ../l4-dst-port-range/lower-port or current() > ../l4-d
st-port-range/upper-port' {
    description
      "If l4-dst-port and l4-dst-port-range/lower-port
       and upper-port are set at the same time,
       l4-dst-port should not overlap with
       l4-src-port-range.";
  }
  description
    "Match on Layer 4 dst port.";
}
container l4-dst-port-range {
  leaf lower-port {
    type inet:port-number;
```

```
        description
            "Lower boundary for port.";
    }
    leaf upper-port {
        type inet:port-number;
        must '.. >= ../lower-port' {
            description
                "Upper boundary must be
                 higher than lower boundary.";
        }
        description
            "Upper boundary for port.  If it exists,
             upper boundary must be higher than lower
             boundary.";
    }
    description
        "Match on Layer 4 dst port range.  When only
         lower-port is present, it represents a single
         port.  When both lower-port and upper-port are
         specified, it implies a range inclusive of both
         values.";
}
leaf src-mac {
    type yang:mac-address;
    description
        "Source MAC.";
}
leaf dst-mac {
    type yang:mac-address;
    description
        "Destination MAC.";
}
leaf protocol-field {
    type union {
        type uint8;
        type identityref {
            base protocol-type;
        }
    }
    description
        "Match on IPv4 protocol or IPv6 Next Header field.";
}
description
    "Describes flow-matching criteria.";
}
description
    "Flow definition based on criteria.";
}
```

```
grouping service-qos-profile {
  container qos {
    if-feature "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 {
            uses flow-definition;
          }
          case match-application {
            leaf match-application {
              type identityref {
                base 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 {
    choice qos-profile {
      description
        "Choice for QoS profile.
         Can be standard profile or customized profile.";
      case standard {
```



```
    description
      "Standard QoS profile.";
  leaf profile {
    type string;
    description
      "QoS profile to be used.";
  }
}
case custom {
  description
    "Customized QoS profile.";
  container classes {
    if-feature "qos-custom";
    list class {
      key "class-id";
      leaf class-id {
        type string;
        description
          "Identification of the class of service.
          This identifier is internal to the
          administration.";
      }
      leaf direction {
        type identityref {
          base qos-profile-direction;
        }
        default "both";
        description
          "The direction to which the QoS profile
          is applied.";
      }
      leaf rate-limit {
        type decimal64 {
          fraction-digits 5;
          range "0..100";
        }
        units "percent";
        description
          "To be used if the class must be rate-limited.
          Expressed as percentage of the service
          bandwidth.";
      }
    }
  }
  container latency {
    choice flavor {
      case lowest {
        leaf use-lowest-latency {
          type empty;
          description
```

```
        "The traffic class should use the path with the
          lowest latency.";
      }
    }
    case boundary {
      leaf latency-boundary {
        type uint16;
        units "msec";
        default "400";
        description
          "The traffic class should use a path with a
            defined maximum latency.";
      }
    }
    description
      "Latency constraint on the traffic class.";
  }
  description
    "Latency constraint on the traffic class.";
}
container jitter {
  choice flavor {
    case lowest {
      leaf use-lowest-jitter {
        type empty;
        description
          "The traffic class should use the path with the
            lowest jitter.";
      }
    }
    case boundary {
      leaf latency-boundary {
        type uint32;
        units "usec";
        default "40000";
        description
          "The traffic class should use a path with a
            defined maximum jitter.";
      }
    }
  }
  description
    "Jitter constraint on the traffic class.";
}
description
  "Jitter constraint on the traffic class.";
}
container bandwidth {
  leaf guaranteed-bw-percent {
```

```
        type decimal64 {
            fraction-digits 5;
            range "0..100";
        }
        units "percent";
        mandatory true;
        description
            "To be used to define the guaranteed bandwidth
            as a percentage of the available service bandwidth.";
    }
    leaf end-to-end {
        type empty;
        description
            "Used if the bandwidth reservation
            must be done on the MPLS network too.";
    }
    description
        "Bandwidth constraint on the traffic class.";
}
description
    "List of classes of services.";
}
description
    "Container for list of classes of services.";
}
}
}
description
    "QoS profile configuration.";
}
description
    "QoS configuration.";
}
description
    "This grouping defines QoS parameters for a segment network.";
}

grouping remote-peer-tp {
    description
        "remote-peer-tp Grouping.";
    leaf remote-id {
        type yang:uuid;
        description
            "Router ID of the remote peer";
    }
    leaf location {
        type string {
            length "0..400";
        }
    }
}
```

```
    }
    description
      "CE device location ";
  }
  leaf remote-tp-address {
    type inet:ip-address;
    description
      "TP IP address";
  }
  leaf remote-node-id {
    type yang:uuid;
    description
      "directly connected NE node ID, only valid in
      asbr ";
  }
  leaf remote-tp-id {
    type yang:uuid;
    description
      "Directly connected TP id, only valid in asbr";
  }
}

grouping tp-connection-specific-attribute {
  description
    "tp connectin specific attributes";
  list connection {
    key "connection-class";
    leaf connection-class {
      type layer-rate;
      description
        "connection class and has one to one
        relation with the corresponding layer.";
    }
  }
  uses tp-connection-type;
  description
    "typeSpecList";
}
container security-attribute {
  description
    "tp security Parameters.";
  uses security-attribute;
}
container qos-attribute {
  description
    "tp Qos Parameters.";
  uses segment-service-basic;
  uses service-qos-profile;
}
```

```
    container protection-attribute {
      description
        "tp protection parameters.";
      leaf access-priority {
        type uint32;
        default "100";
        description
          "Defines the priority for the access.
           The higher the access-priority value,
           the higher the preference of the
           access will be.";
      }
    }
  }
}

grouping tp-common-attribute {
  description
    "tp-common-attribute Grouping.";
  leaf tp-id {
    type yang:uuid;
    description
      "An identifier for termination point on a node.";
  }
  leaf tp-name {
    type string {
      length "0..200";
    }
    description
      "The termination point Name on a node. It conforms to
       name rule defined in system. Example FE0/0/1, GE1/2/1.1,
       Eth-Trunk1.1, etc";
  }
  leaf node-id {
    type yang:uuid;
    description
      "Identifier for a node.";
  }
  leaf access-point-type {
    type access-point-type;
    description
      "access-point-type, for example:peering with CE ";
  }
  leaf inter-as-option {
    type enumeration {
      enum optiona {
        description
          "Inter-AS Option A";
      }
    }
  }
}
```

```
        enum optionb {
            description
                "Inter-AS Option B";
        }
        enum optionc {
            description
                "Inter-AS Option C";
        }
    }
    description
        "Foo";
}
leaf topology-role {
    type topology-role;
    description
        "hub/spoke role, etc";
}
}

grouping routing-protocol {
    description
        "Routing Protocol Grouping.";
    leaf type {
        type protocol-type;
        description
            "Protocol type";
    }
    choice para {
        description
            "para";
        case static {
            list static {
                key "index";
                uses static-config;
                description
                    "staticRouteItems";
            }
        }
        case bgp {
            list bgp {
                key "index";
                uses bgp-config;
                description
                    "bgpProtocols";
            }
        }
    }
}
}
```

```
grouping bgp-config {
  description
    "BGP Protocol Grouping.";
  leaf index {
    type uint32;
    description
      "index of BGP protocol item";
  }
  leaf autonomous-system {
    type uint32;
    mandatory true;
    description
      "Peer AS number in case the peer
       requests BGP routing.";
  }
  leaf-list address-family {
    type address-family;
    min-elements 1;
    description
      "If BGP is used on this site, this node
       contains configured value. This node
       contains at least one address family
       to be activated.";
  }
  leaf max-prefix {
    type int32;
    description
      "maximum number limit of prefixes.";
  }
  leaf peer-address {
    type inet:ip-address;
    description
      "peerIp";
  }
  leaf crypto-algorithm {
    type identityref {
      base keychain:crypto-algorithm;
    }
    mandatory true;
    description
      "Cryptographic algorithm associated with key.";
  }
  container key-string {
    description
      "The key string.";
    nacm:default-deny-all;
    choice key-string-style {
      description
```

```
        "Key string styles";
    case keystack {
        leaf keystack {
            type string;
            description
                "Key string in ASCII format.";
        }
    }
    case hexadecimal {
        if-feature "hex-key-string";
        leaf hexadecimal-string {
            type yang:hex-string;
            description
                "Key in hexadecimal string format. When compared
                to ASCII, specification in hexadecimal affords
                greater key entropy with the same number of
                internal key-string octets. Additionally, it
                discourages usage of well-known words or
                numbers.";
        }
    }
}

grouping static-config {
    description
        "StaticRouteItem Grouping.";
    leaf index {
        type uint32;
        description
            "static item index";
    }
    leaf dest-cidr {
        type string;
        description
            "address prefix specifying the set of
            destination addresses for which the route may be
            used. ";
    }
    leaf egress-tp {
        type yang:uuid;
        description
            "egress tp";
    }
    leaf route-preference {
        type string;
        description
```



```
        "route priority. Ordinary, work route have
          higher priority.";
    }
    leaf next-hop {
        type inet:ip-address;
        description
            "Determines the outgoing interface and/or
             next-hop address(es), or a special operation to be
             performed on a packet..";
    }
}

grouping ethernet-spec {
    description
        "Ethernet Spec Grouping.";
    leaf access-type {
        type eth-encap-type;
        description
            "access frame type";
    }
    choice accessVlanValue {
        description
            "accessVlanValue";
        case qinq {
            container qinq {
                description
                    "qinqVlan";
                uses QinqVlan;
            }
        }
        case dot1q {
            container dot1q {
                description
                    "dot1q";
                uses Dot1qVlan;
            }
        }
    }
    leaf vlan-action {
        type ethernet-action;
        description
            "specify the action when the vlan is matched";
    }
    leaf action {
        type string {
            length "0..100";
        }
        description

```

```
        "specify the action value.";
    }
}

grouping pwspec {
    description
        "PwSpec Grouping.";
    leaf control-word {
        type boolean;
        default "false";
        description
            "control Word.";
    }
    leaf vlan-action {
        type pwttagmode;
        description
            "pw Vlan Action.";
    }
}

grouping ipspec {
    description
        "IpSpec Grouping.";
    leaf ip-address {
        type inet:ip-address;
        description
            "master IP address";
    }
    leaf mtu {
        type uint64;
        description
            "mtu for ip layer, scope:46~9600";
    }
}

grouping VPN {
    description
        "VPN Grouping.";
    leaf vpn-id {
        type yang:uuid;
        description
            "VPN Identifier.";
    }
    leaf vpn-name {
        type string {
            length "0..200";
        }
        description

```

```

        "Human-readable name for the VPN service.";
    }
    leaf service-type {
        type service-type;
        description
            "The service type combines service types from
            RFC8299 (L3SM) and RFC8466 (L2SM), for example L3VPN, VPWS etc.
            It could be augmented for future extensions.";
    }
    leaf topo {
        type vpn-topology;
        description
            "The VPN topology could be full-mesh, point-to-point
            and hub-spoke, others is reserved for future extensions.";
    }
    leaf tunnel-type {
        type tunnel-type;
        description
            "Tunnel Type: LDP&#65306; LDP Tunnel, RSVP-TE&#65306; RSVP-TE Tunnel
            SR-TE&#65306; SR-TE Tunnel, MPLS-TP&#65306; MPLS-TP Tunnel, VXLAN&#65306; VX
LAN Tunnel
            ";
    }
    leaf admin-state {
        type admin-state;
        description
            "administrative status.";
    }
    leaf oper-state {
        type oper-state;
        config false;
        description
            "Operational status.";
    }
    leaf sync-state {
        type sync-state;
        config false;
        description
            "Sync status.";
    }
    list access-point {
        key "tp-id";
        description
            "TP list of the access links which associated
            with PE and CE or ASBR";
        uses pe-termination-point;
    }
}

```

```
grouping pe-termination-point {
  description
    "grouping for termination points.";
  uses tp-common-attribute;
  container peer-remote-node {
    description
      "TP Peering Information, including CE
       peering and ASBR peering.";
    uses remote-peer-tp;
  }
  container tp-connection-specific-attribute {
    description
      "Termination point basic info.";
    uses tp-connection-specific-attribute;
  }
  list routing-protocol {
    key "type";
    description
      "route protocol spec.";
    uses routing-protcol;
  }
}

grouping segment-service-basic {
  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,
```

```

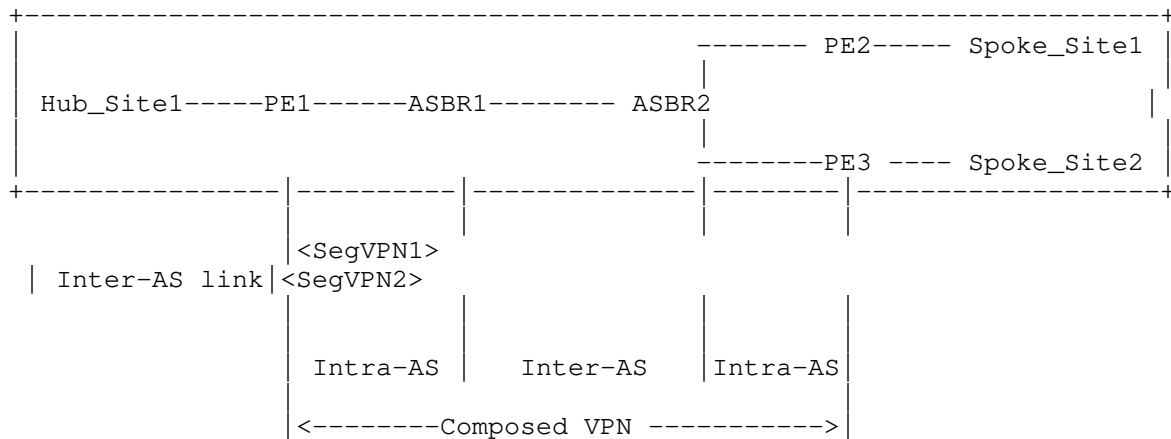
        it refers to the IP MTU.  If CsC is enabled,
        the requested 'svc-mtu' leaf will refer to the
        MPLS MTU and not to the IP MTU.";
    }
    description
        "Defines basic service parameters for a site.";
}

container segment-vpns {
    list segment-vpn {
        key "index";
        description
            "Segment Vpn list.";
        leaf index {
            type uint32;
            description
                "index of segment VPN in a composed VPN.";
        }
        uses VPN;
    }
    description
        "Container for Segment VPN.";
}
}
<CODE ENDS>

```

8. Service Model Usage Example

This section provides an example of how a management system can use this model to configure an IP VPN service on network elements.



Composed VPN Service Model Usage Example

In this example, we want to achieve the provisioning of an end to end VPN service for three sites using a Hub-and-Spoke VPN service topology. The end to end VPN service is stitched by two segmented VPN.

The following XML snippet describes the overall simplified service configuration of this composed VPN.

```
<?xml version="1.0"?>
<composed-vpns xmlns="urn:ietf:params:xml:ns:yang:ietf-composed-vpn-svc">
  <composed-vpn>
    <vpn-id>12456487</vpn-id>
    <topo>hub-spoke</topo>
    <service-type>hybrid</service-type>
    <segment-vpn>
      <index>1</index>
      <vpn-id>111</vpn-id>
      <topo>hub-spoke</topo>
      <service-type>l2vpn</service-type>
      <access-point>
        <tp-id>ap1-tp1</tp-id>
        <node-id>PE1</node-id>
        <topology-role>hub</topology-role>
        <peer-remote-node>
          <remote-node-id>Hub_Site1</remote-node-id>
        </peer-remote-node>
        <tp-connection-specific-attribute>
          <qos-attribute>
            <svc-mtu>1514</svc-mtu>
            <svc-input-bandwidth>10000000</svc-input-bandwidth>
            <svc-output-bandwidth>10000000</svc-output-bandwidth>
          </qos-attribute>
        </tp-connection-specific-attribute>
        <routing-protocol>
          <type>bgp</type>
          <bgp>
            <as-no>AS1</as-no>
          </bgp>
        </routing-protocol>
      </access-point>
      <access-point>
        <tp-id>ap1-tp2</tp-id>
        <node-id>ASBR1</node-id>
        <topo-role>hub</topo-role>
        <peer-remote-node>
          <remote-node-id>ASBR2</remote-node-id>
        </peer-remote-node>
        <inter-AS-option>Option A</inter-AS-option>
      </access-point>
    </segment-vpn>
  </composed-vpn>
</composed-vpns>
```

```

    <tp-connection-specific-attribute>
      <qos-attribute>
        <svc-mtu>1514</svc-mtu>
        <svc-input-bandwidth>10000000</svc-input-bandwidth>
        <svc-output-bandwidth>10000000</svc-output-bandwidth>
      </qos-attribute>
    </tp-connection-specific-attribute>
    <routing-protocol>
      <type>bgp</type>
      <bgp>
        <as-no>AS1</as-no>
      </bgp>
    </routing-protocol>
  </access-point>
</segment-vpn>
<segment-vpn>
  <index>2</index>
  <vpn-id>222</vpn-id>
  <topo>hub-spoke</topo>
  <service-type>l3vpn</service-type>
  <access-point>
    <tp-id>ap2-tp2</tp-id>
    <node-id>PE2</node-id>
    <topo-role>spoke</topo-role>
    <peer-remote-node>
      <remote-node-id>Spoke_Site1</remote-node-id>
    </peer-remote-node>
    <qos-attribute>
      <svc-mtu>1514</svc-mtu>
      <svc-input-bandwidth>10000000</svc-input-bandwidth>
      <svc-output-bandwidth>10000000</svc-output-bandwidth>
    </qos-attribute>
    <routing-protocol>
      <type>bgp</type>
      <bgp>
        <as-no>ASXXX</as-no>
      </bgp>
    </routing-protocol>
  </access-point>
  <access-point>
    <tp-id>ap2-tp1</tp-id>
    <node-id>PE3</node-id>
    <topo-role>spoke</topo-role>
    <peer-remote-node>
      <remote-node-id>Spoke_Site2</remote-node-id>
    </peer-remote-node>
    <qos-attribute>
      <svc-mtu>1514</svc-mtu>

```

```

        <svc-input-bandwidth>10000000</svc-input-bandwidth>
        <svc-output-bandwidth>10000000</svc-output-bandwidth>
        </qos-attribute>
    <routing-protocol>
        <type>bgp</type>
        <bgp>
            <as-no>ASXXX</as-no>
        </bgp>
    </routing-protocol>
</access-point>
<access-point>
    <tp-id>ap2-tp3</tp-id>
    <node-id>ASBR2</node-id>
    <topo-role>hub</topo-role>
    <peer-remote-node>
        <remote-node-id>ASBR1</remote-node-id>
    </peer-remote-node>
    <qos-attribute>
        <svc-mtu>1514</svc-mtu>
        <svc-input-bandwidth>10000000</svc-input-bandwidth>
        <svc-output-bandwidth>10000000</svc-output-bandwidth>
        </qos-attribute>
    <routing-protocol>
        <type>bgp</type>
        <bgp>
            <as-no>interAS-1</as-no>
        </bgp>
    </routing-protocol>
</access-point>
</segment-vpn>
</composed-vpn>
</composed-vpns>

```

9. Interaction with other YANG models

As expressed in Section 4, this composed VPN service model is intended to be instantiated in a management system and not directly on network elements.

The management system's role will be to configure the network elements. The management system may be modular and distinguish the component instantiating the service model (let's call it "service component") from the component responsible for network element configuration (let's call it "configuration component"). The service is built from a combination of network elements and protocols configuration which also include various aspects of the underlying network infrastructure, including functions/devices and their subsystems, and relevant protocols operating at the link and network

layers across multiple device. Therefore there will be a strong relationship between the abstracted view provided by this service model and the detailed configuration view that will be provided by specific configuration models for network elements.

The service component will take input from customer service model such as L3SM service model [RFC8299] or composed VPN service model and translate it into segment VPN in each domain and then further break down the segment VPN into detailed configuration view that will be provided by specific configuration models for network elements.

10. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

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

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 /composed-vpns/composed-vpn

The entries in the list above include the whole composed vpn service configurations which the customer subscribes, and indirectly create or modify the PE,CE and ASBR device configurations. Unexpected changes to these entries could lead to service disruption and/or network misbehavior.

- o /composed-vpns/composed-vpn/segment-vpn

The entries in the list above include the access points configurations. As above, unexpected changes to these entries could lead to service disruption and/or network misbehavior.

- o /composed-vpns/composed-vpn/access-point

The entries in the list above include the access points configurations. As above, unexpected changes to these entries could lead to service disruption and/or network misbehavior.

11. IANA Considerations

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

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

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

This document registers two YANG modules in the YANG Module Names registry [RFC6020].

```
-----
Name: ietf-composite-vpn-svc
Namespace: urn:ietf:params:xml:ns:yang:ietf-composed-vpn-svc
Prefix: composed-svc
Reference: RFC xxxx
Name: ietf-segmented-vpn
Namespace: urn:ietf:params:xml:ns:yang:ietf-segment-vpn
Prefix: segment-vpn
Reference: RFC xxxx
-----
```

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.

- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [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>>.
- [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>>.
- [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>>.

12.2. Informative References

- [RFC1136] Hares, S. and D. Katz, "Administrative Domains and Routing Domains: A model for routing in the Internet", RFC 1136, DOI 10.17487/RFC1136, December 1989, <<https://www.rfc-editor.org/info/rfc1136>>.
- [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>>.

Appendix A. Acknowledges

Geng Liang, Congfeng Xie, Chen Rui, LiYa Zhang, Hui Deng contributed to an earlier version of [I-D.chen-opsawg-composite-vpn-dm]. We would like to thank the authors of that document on the operators' view for the PE-based VPN service configuration for material that assisted in thinking about this document.

Authors' Addresses

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

Email: roni.even@huawei.com

Bo Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: lanawubo@huawei.com

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

YingCheng
China Unicom
No.21 Financial Street, XiCheng District
Beijing 100033
China

Email: chengying10@chinaunicom.cn

Individual
Internet-Draft
Intended status: Informational
Expires: August 5, 2019

S. Homma
H. Nishihara
NTT
T. Miyasaka
KDDI Research
A. Galis
University College London
V. Ram OV
Independent Research Consultant India
D. Lopez
L. Contreras-Murillo
J. Ordonez-Lucena
Telefonica I+D
P. Martinez-Julia
NICT
L. Qiang
Huawei Technologies
R. Rokui
L. Ciavaglia
Nokia
X. de Foy
InterDigital Inc.
February 1, 2019

Network Slice Provision Models
draft-homma-slice-provision-models-00

Abstract

Network slicing is an approach to provide separate virtual network based on service requirements. It's a fundamental concept of the 5G, and the architecture and specification is under standardization in several organizations. However, the definitions and scopes of network slicing vary to some degree from one organization to another. This document provides classification of provisioning models of network slice for clarifying the differences on the definitions and scopes.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 5, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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 and Motivation	3
1.1. Differentiated Roles in Network Slice Provisioning	3
1.2. High-level Problem Statement	4
2. Definition of Terms	4
3. General Requirements for Network Slicing	7
4. Network Slice Structure	7
4.1. Resources for Structuring Network Slices	7
4.2. Basic Network Slice Structure	11
4.3. Stakeholders in the Structuring Network Slices	14
5. Variations of Network Slice Creation	14
5.1. Ready-made Network Slice	14
5.2. Custom-made Network Slice	15
5.3. semi-Custom-made Network Slice	15
6. Network Slice Provision Models	15
6.1. Three Provision Models	15
6.2. Configurable Parameters/Attributes on each Provision Models	18
6.3. Capability of NS Tenant on each Provision Model	18
7. Security Considerations	18
8. IANA Considerations	18
9. Acknowledgement	18
10. Informative References	18
Appendix A. NS Structure in the 3GPP 5GS	20
Authors' Addresses	20

1. Introduction and Motivation

Network slicing is an approach to provide separate virtual networks depending on requirements of each service. Network slicing receives attention due to factors such as diversity of services and devices, and it is also a fundamental concept of the 5G for applying networks to such various types of requirements.

In addition, network slicing is expected to enable a business model to provide dedicated logical networks to 3rd parties or vertical customers on-demand, called NSaaS (Network Slice as a Service). For such usage, in network slicing, provision of networks able to guarantee communication characteristics end to end would be required. However, the definitions are not harmonized over several SDOs (Standards Developing Organizations).

This document clarifies provision patterns of network slice, and provides the definitions and scope of network slicing which are available over several organizations. Furthermore, the deliverables would be help for evaluating applicabilities of existing technologies/solutions to network slicing.

1.1. Differentiated Roles in Network Slice Provisioning

The widespread of system and network virtualization technologies has conducted to new business opportunities, enlarging the offer of IT resources in the form of Network Slices (NS). As a consequence, there is a clear differentiation between the owner of physical resources, the infrastructure operator, and the intermediary that conforms and delivers network services to the final customers, the Virtual Network Operator (VNO).

VNOs aim to exploit the virtualized infrastructures to deliver new and improved services to their customers. However, current NS techniques offer poor support for VNOs to control their resources. It has been considered that the infrastructure operator is responsible of the reliability of the NS elements but several situations advocate the VNO to gain a finer control on its resources. For instance, dynamic events, such as the identification of new requirements or the detection of incidents within the virtual system, might urge a VNO to quickly reform its virtual infrastructure and resource allocation. However, the interfaces offered by current virtualization platforms do not offer the necessary functions for VNOs to perform the elastic adaptations they require to tackle with their dynamic operation environments.

1.2. High-level Problem Statement

Beyond their heterogeneity, which can be resolved by software adapters, NS platforms do not offer common methods and functions, so it is difficult for the virtual network controllers used by the VNOs to actually manage and control virtual resources instantiated on different platforms, not even considering different infrastructure operators. Therefore, it is necessary to reach a common definition of the functions that should be offered by underlying platforms to enable such overlay controllers with the possibility of allocate and deallocate resources dynamically and get monitoring data about them.

Such common methods should be offered by all underlying controllers, regardless of being network-oriented (e.g., ODL, ONOS, Ryu) or computing-oriented (e.g., OpenStack, OpenNebula, Eucalyptus). Furthermore, it is also important for those platforms to offer some "PUSH" function to report resource state, avoiding the need for the VNO's controller to "POLL" for such data. A starting point to get proper notifications within current REST APIs could be to consider the protocol proposed by the [WEBPUSH-WG].

Finally, in order to establish a proper order and allow the coexistence and collaboration of different systems, a common ontology regarding network and system virtualization should be defined and agreed, so different and heterogeneous systems can understand each other without requiring to rely on specific adaptation mechanisms that might break with any update on any side of the relation.

2. Definition of Terms

This section lists definitions and terms related to network slicing. Although this document refers terms and viewpoints on network slicing in 3GPP documents ([TS.28.530-3GPP] and [TS.28.801-3GPP]) and [NGMN-5G-White-Paper], some of definitions in this document may be different from ones of those documents.

Network Slicing: Network slicing indicates a technology, an approach, or a concept to create logical separate networks in support of services, depending on several requirements, on the same physical resources. This is possible by combinations of several network technologies.

Network Slice (NS): An NS is a general name of logical separate networks instantiated on a network infrastructure. It includes Network Slice Instance, Network Slice Subnet Instance, and End-to-End Network Slice Instance.

Network Slice Instance (NSI): An NSI is a logical network instantiated with network(WAN) and computing(NFVI), and some include additional network service functions such as firewall or load-balancer. It is composed of one or more Network Slice Subnet Instances. When it provides connectivity from end to end for end users, it is called End-to-End Network Slice Instance. An NSI is basically an overlay network and is independent of the underlay network's topology.

Network Slice Subnet Instance (NSSI): An NSSI is a partially virtual network instantiated within a single domain, and it basically provides connectivity to other domains or end points. Ways to construct an NSSI depends on the specifications of underlay networks.

End-to-End Network Slice Instance (E2E-NSI): An E2E-NSI is a virtual network connecting among end points. It is composed of one or multiple NSSIs. This term is used in this document when it should be emphasized that the NSI is structured from end to end. As an example, for providing an E2E-NSI on the 3GPP 5G network, combining three types of NSIs: RAN-, TRN-, and CN-NSIs would be required.

Transport (TRN)-NSSI: A set of connections between various network functions (VNF or PNF) with deterministic SLAs. They can be implemented (aka realized) with various technologies (e.g. IP, Optics, FN, Microwave) and various transport (e.g. RSVP, Segment routing, ODU, OCH etc). The overview of NSI composed with TRN-NSSI is shown in Appendix A.

RAN-NSSI: Regardless of RAN deployment (e.g. distributed-RAN, Centralized-RAN or Cloud-RAN, a RAN-NSSI creates a dedicate and logical resource on RAN for each NSI which are completely. The overview of NSI composed with RAN-NSSI is shown in Appendix A.

Core(CN)-NSSI: Regardless of Core deployment, a CN-NSI creates a dedicate and logical resource on Core network for each NSI which are completely. The overview of NSI composed with CN-NSSI is shown in Appendix A.

Network Slice as a Service (NSaaS): An NSaaS is a service delivery model in which a third-party provider or a vertical customer hosts NSIs and makes them available to customers. In this model, there mainly two roles: NS provider and NS tenant.

Network Slice Provider (NS Provider): An NS provider is a person or group that designs and instantiates one or more NSIs/NSSIs, and provides them to NS tenants. In some cases, an NS provider is an

infrastructure operator simultaneously. This includes NSI, NSSI, and E2E-NSI providers.

Network Slice Tenant (NS Tenant): An NS tenant is a person or group that rents and occupies NSIs from NS providers.

Network Slice Stakeholder (NS Stakeholder): An NS stakeholder is an actor in network slicing, and has roles of either NS provider or tenant.

Infrastructure Operator: An infrastructure operator is an organization who manages infrastructure networks or data centers for running NSIs. In the most of cases, infrastructure operators are initial NS providers on NSaaS. Also, some of them may be NS tenants simultaneously.

Vertical Customer: A vertical customer is a organization who provides some communicating services with using NSIs on NSaaS model. In many cases, a vertical customer become the final NS tenant on NSaaS. For example, video gaming companies or vehicle vendors will possibly be vertical customers.

Virtual Network Operator (VNO): A VNO is a person or group that operates virtual networks composed with resources or NSSIs rent from infrastructure operators and provides such virtual networks as NSIs to vertical customers who are final NS tenants. In some cases, infrastructure operators have this role in addition to operating their own infrastructure simultaneously.

Domain: A domain is a group of a network and devices administrated under a policy-based common set of rules and procedures.

Resource: A resource is an element used to create virtual networks. There are several types of resources, i.e., connectivity, computing and storage. The details are described Section 4.1

Virtual Network: A virtual network is a network running a number of virtual network functions.

Virtual Network Function (VNF): A virtual network function (VNF) is a network function whose functional software is decoupled from hardware. One or more VNFs run as different software and processes on top of industry-standard high-volume servers, switches and storage, or cloud computing infrastructure. They are capable of implementing network functions traditionally implemented via custom hardware appliances and middleboxes (e.g., router, NAT, firewall, load balancer, etc.).

Network Operation System: A network operation system is an entity or a group of entities for operating network nodes and functions as compositions of infrastructure network. For example, OSS/BSS, orchestrator, and EMS are considered to be network operation systems.

3. General Requirements for Network Slicing

On network slice operations, capabilities for dynamic instantiation, change, and deletion should be required because an NSI is established based on received orders from tenants in NSaaS. From this aspect, some mechanisms to design a network based on service requirements and to convert those to concrete configurations based on the design would be required.

In addition, each NS has to maintain concrete communication characteristics end to end, and resource reservations on data plane and isolation among NSIs would be required. Isolation is a concept to prevent the reduction of communication quality caused by disturbance from other NSs, and it may have some levels of enforcement, such as hard or soft isolations. In some cases, for providing appropriate communication between client and server, it would be allowed for NS tenants to put their applications as contents server on NSIs by using computing resources.

The required agility of slice operation and granularity of end to end communication quality requested can vary depending on provision model.

4. Network Slice Structure

This section describes resources used for structuring NSs and the basic structure of E2E-NS.

4.1. Resources for Structuring Network Slices

A network slice is structured as combinations of the resources it uses. Such resources are mainly categorized into three classes: network/WAN, computing/NFVI, and functionality resources. Variations of each resources are described below. (Note that the lists are not exhaustive.)

Network (WAN) Resources:

- * Connectivity:

- + (v)Link

- Bandwidth per link/session
- Connected area/end points
- Forwarding route/path (e.g., for traffic engineering, redundancy)
- Communication Priority (e.g., QoS class)
- Range of jitter amount
- + Interface of vNode
 - QoS setting (e.g., Queue size, DSCP remarking, PIR/CIR)
 - Filter setting
- + vRouter/vSwitch (# Treated as a set of (v)links and interfaces of vNodes.)
- * Multicast support
- * Encryption support
- * Authentication support
- * Metadata conveyance (e.g., subscriber ID)
- * Protocols for slice data plane:
 - + VLAN
 - + IPoE (IPv4 or IPv6)
 - + MAP-E
 - + DS-Lite
 - + PPPoE
 - + L2TP
 - + GRE
 - + MPLS
 - + VxLAN

- + Geneve
- + GTP-U
- + Segment Routing MPLS
- + Segment Routing IPv6
- + NSH
- + Other

Computing (NFVI) Resources:

- * (v)CPU core
- * Storage
- * Memory
- * Disk
- * vNIC
- * Connectivity to VNF instances
- * Virtual Deployment Unit:
 - + Virtual Machine (VM)
 - + container
 - + micro kernel
- * Resource Deployment Location (i.e., edge DC, central DC, public cloud, ..., etc.)

Functionality Resources:

- * Image:
 - + Data Plane (DP) NF:
 - GateWay (GW) function:
 - o Access Point Type (e.g., for radio, Wi-Fi, and fixed accesses)

- o Slice Selection Setting
 - o Terminate protocol
 - o Authentication
- Security Appliance:
 - o IPS (Intrusion Prevention System)
 - o IDS (Intrusion Detection System)
 - o WAF (Web Application Firewall)
- DPI
- Load Balancer
- TCP Accelerator
- Video Optimizer
- Parental Control
- Mobile DP functions (Ref. 3GPP 5GS)
 - gNB
 - UPF
 - Uplink Classifier
- + Control Plane(CP) NF:
 - DHCP
 - o Fixed IP address allocation
 - o Dynamic IP address allocation
 - o The number of registered devices
 - DNS
 - VoIP (SBC, SIP server)
 - Mobile CP function (Ref. 3GPP 5GS)

- o AMF (Access and Mobility management Function)
 - o SMF (Session Management Function)
 - o PCF (Policy Control Function)
 - o UDM (Unified Data Management)
 - o NEF (Network Exposure Function)
- * Provided VNF Type (e.g., open source, product of vender#A, ..., etc.)
 - * Function location (e.g., edge DC, central DC, Public cloud, etc.)

In terms of security or usability for NS tenants, some abstraction on resource information would be required, however both setting parameters of underlay infrastructure and abstracted information may coexist in these lists.

For abstraction of parameters of underlay networks, some additional protocols or functions (like [RFC8453]) would be required. Moreover, for providing strict communication qualities, combinations of some technologies may be useful (ref. [I-D.dong-teas-enhanced-vpn]).

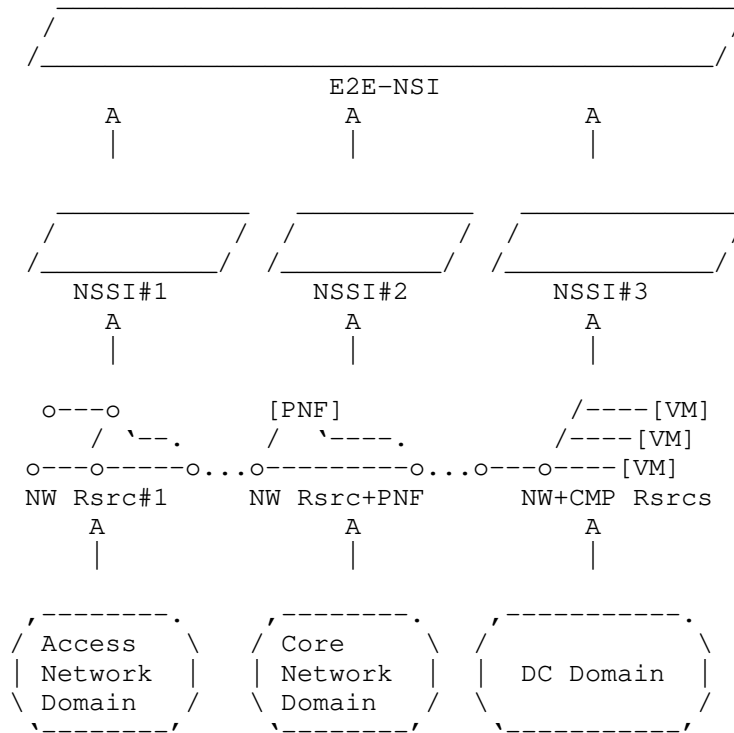
4.2. Basic Network Slice Structure

An E2E-NSI is constructed by stitching NSSIs instantiated on each participating domain. This includes the simplest case of a single NSSI as an E2E NS. Domain types where some NSSIs are established are described below:

- o Fixed access network
 - o Mobile access network
 - o Transport network
 - o Fixed core network
 - o Mobile core network
 - o Data center (DC)
- * Edge DC

- * Central DC
- o Private network
 - * Enterprise
 - * Factory
 - * Utilities
 - * Farming
 - * Home/SOHO
 - * Other

Figure 1 describes the overview of this structure. Resources in each domain (e.g., access, core networks, and DC) are handled by management entities and constitute an NSSI. An E2E-NSI is established by stitching these NSSIs. Ways to stitch NS-subnets are described in [I-D.defoy-coms-subnet-interconnection] and [I-D.homma-nfvrg-slice-gateway].



*Legends

NW Rsrc : Network Resource

CMP Rsrc: Computing Resource

o : virtual/physical node structuring NSI

-- : virtual/physical link structuring NSI

[PNF]: Physical Network Function Appliance on NSI

[VM] : Virtual Machine Instance on NSI

Figure 1: Overview of NS Structure

Although it is shown that an NSSI belongs to just only one E2E-NSI in Figure 1, it may be allowed that multiple E2E-NSIs share an NSSI. Some resources may belong to multiple NSSI as well.

In addition, structure on composition of NSI may be recursive. In other words, even though Figure 1 shows a case where NSSIs compose directly an E2E-NSI, in some cases, NSSIs compose an NSI which is a part of an E2E-NSI. The overview is shown in Figure 2. In this figure, NSI#4 is composed of NSSI#1 and NSSI#2, and it structures E2E-NSI#5 with NSSI#3.

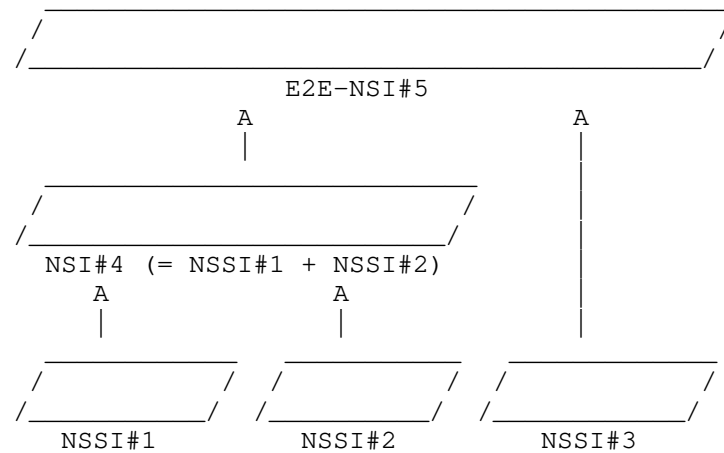


Figure 2: Overview of NS recursive structure

4.3. Stakeholders in the Structuring Network Slices

Potential stakeholders in network slicing are described below:

- o NSSI provider: infrastructure operator
- o Intermediate-NSI provider: infrastructure operator, VNO
- o E2E-NSI provider: infrastructure operator, VNO, service provider
- o NS tenant: infrastructure operator, VNO, service provider, enterprise, mass user
- o End customer: enterprise, mass user, etc.

5. Variations of Network Slice Creation

NSs can be classified according to their creation pattern into two types: ready-made(RM) NS, custom-made(CM), and semi-custom-made(sCM) NS. This section describes the features of these types.

5.1. Ready-made Network Slice

RM-NS is an NS creation pattern in which an infrastructure operator decides service requirements by itself, and established based on the requirements in advance. NS tenants select one of RM-NSs whose features are closer to their requirements.

This model doesn't need immediacy on designing of NSI and enables to mitigate the difficulty of implementation compared with other models.

5.2. Custom-made Network Slice

CM-NS is an NS creation pattern in which an NS is established based on an order from a tenant and is provided to it. As examples of usage of CM-NS, an enterprise builds and operates a virtual private network for connecting several bases, or OTT (Over The Top) or other industrial service providers create NSs based on their own requirements and use them as a part of their own services (e.g., connected vehicles/drones, online video games, or remote surgery).

In this model, network operation system would be required to have incorporate intelligence for designing appropriate NSs on-demand.

5.3. semi-Custom-made Network Slice

sCM-NS is a derivation of a CM-NS. In sCM-NS, an NS provider designs the outline of NSs in advance, and a tenant tunes an NS with deciding some parameters or applications run on resources. For example, an infrastructure operator designs a logical network presenting connectivity, and tenants install their own applications on servers running on the logical network.

6. Network Slice Provision Models

This document classifies NS provision models into three categories defined in the following section. The capabilities which NS tenants can have on management of NSs would vary depending on the selected provision model.

6.1. Three Provision Models

The provision models are categorized into three models: SaaS (Software as a Service), PaaS (Platform as a Service), IaaS (Infrastructure as a Service) like models as below.

SaaS-like Model: In this model, an NS provider designs NS templates in advance, and a tenant selects and uses one which fulfills most its requirement among the templates. The specifications of NSs are abstracted to KPIs as networks and servers and shown to tenants. In short, detailed parameters of infrastructure network are hidden from tenants.

PaaS-like Model: In this model, a tenant makes its request, including connected area, path routes, the KPIs, and included service functions, and a NS provider designs an NS template and

instantiate an NS based on the request dynamically. The configurable values would vary depending on the policy of each NS provider.

IaaS-like Model: In this model, a tenant designs its own NS templates and instantiates NSs by indicating concrete resources to infrastructure operators. In other words, infrastructure operators provide just their resources, and NSs are coordinated by the tenant.

An example of mapping of each NS provision model is shown in Figure 3.

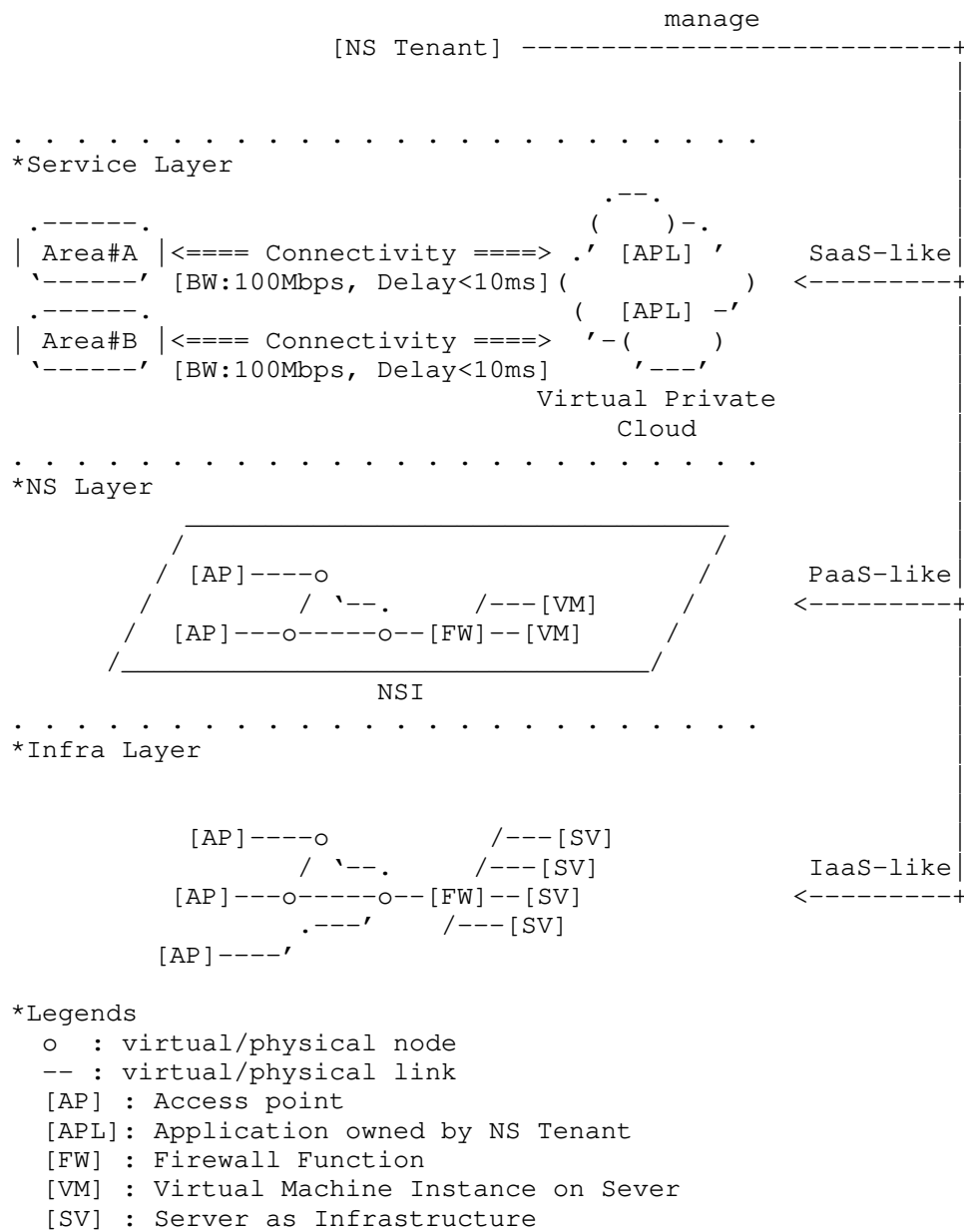


Figure 3: Mapping of NS provision models

In some cases, NSIs provided based on IaaS- or PaaS-like models are coordinated to a form of SaaS-like model by an NS broker , and the NS broker or by the tenant, becoming a NS provider in a recursive manner. For example, a vertical customer sends its high-level requirements to an NS broker create an appropriate NSI with resources provided by infrastructure operators.

6.2. Configurable Parameters/Attributes on each Provision Models

TBD

6.3. Capability of NS Tenant on each Provision Model

TBD

7. Security Considerations

In NSaaS, parts of controls of infrastructures are opened to externals, and thus some mechanisms, such as authentication for APIs, to prevent illegal access would be required.

Other considerations are TBD

8. IANA Considerations

This memo includes no request to IANA.

9. Acknowledgement

The author would like to thank Toru Okugawa for his kind review and valuable feedback.

10. Informative References

[I-D.defoy-coms-subnet-interconnection]

Foy, X., Rahman, A., Galis, A.,
kiran.makhijani@huawei.com, k., and L. Qiang,
"Interconnecting (or Stitching) Network Slice Subnets",
draft-defoy-coms-subnet-interconnection-01 (work in
progress), October 2017.

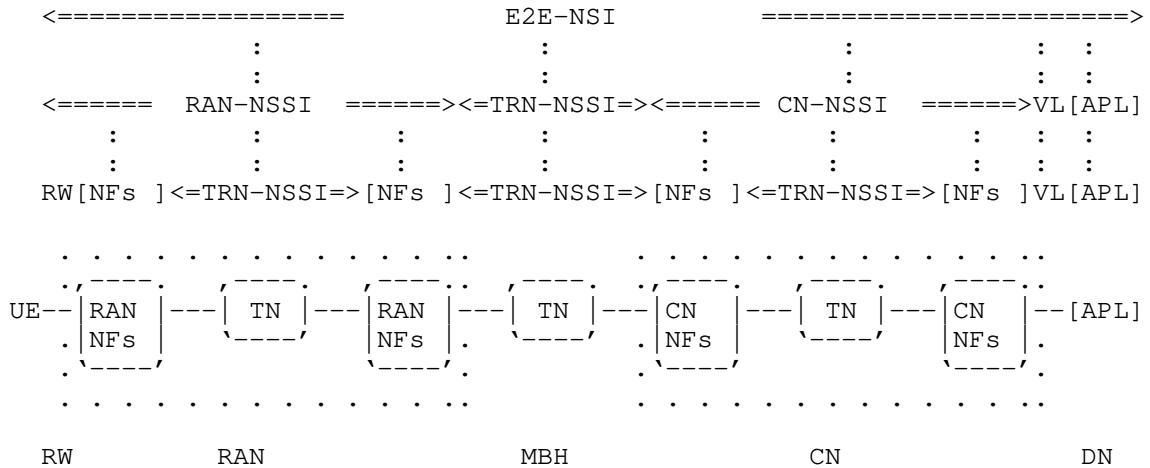
[I-D.dong-teas-enhanced-vpn]

Dong, J., Bryant, S., Li, Z., Miyasaka, T., and Y. Lee, "A
Framework for Enhanced Virtual Private Networks (VPN+)
Service", draft-dong-teas-enhanced-vpn-03 (work in
progress), November 2018.

- [I-D.homma-nfvrg-slice-gateway]
Homma, S., Foy, X., and A. Galis, "Gateway Function for Network Slicing", draft-homma-nfvrg-slice-gateway-00 (work in progress), July 2018.
- [NGMN-5G-White-Paper]
NGMN, "NGMN 5G White Paper", February 2015,
<<https://www.ngmn.org/5g-white-paper/5g-white-paper.html>>.
- [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>>.
- [TS.23.501-3GPP]
3rd Generation Partnership Project (3GPP), "3GPP TS 23.501 (V15.3.0): System Architecture for 5G System; Stage 2", September 2018, <http://www.3gpp.org/ftp//Specs/archive/23_series/23.501/23501-f30.zip>.
- [TS.28.530-3GPP]
3rd Generation Partnership Project (3GPP), "3GPP TS 28.530 (V1.0.0): Management and orchestration of networks and network slicing; Concepts, use cases and requirements (work in progress)", June 2018,
<http://ftp.3gpp.org//Specs/archive/28_series/28.530/28530-100.zip>.
- [TS.28.541-3GPP]
3rd Generation Partnership Project (3GPP), "3GPP TS 28.541 (V15.1.0): 5G Network Resource Model (NRM); Stage 2 and stage 3 (Release 15)", June 2018,
<http://www.3gpp.org/ftp//Specs/archive/28_series/28.541/28541-f01.zip>.
- [TS.28.801-3GPP]
3rd Generation Partnership Project (3GPP), "3GPP TS 28.801 (V15.1.0): Study on Management and Orchestration of Network Slicing for next generation network (Release 15)", June 2018, <http://www.3gpp.org/ftp//Specs/archive/28_series/28.801/28801-f10.zip>.
- [WEBPUSH-WG]
IETF, "Web-Based Push Notifications(webpush)",
<<https://datatracker.ietf.org/wg/webpush/about/>>.

Appendix A. NS Structure in the 3GPP 5GS

The overview of structure of NS in the 3GPP 5GS is shown in Figure 4. The terms are described in the 3GPP documents (e.g., [TS.23.501-3GPP] and [TS.28.530-3GPP]).



*Legends

UE: User Equipment
 RAN: Radio Access Network
 CN: Core Network
 DN: Data Network
 TN: Transport Network
 MBH: Mobile Backhaul
 RW: Radio Wave
 NF: Network Function
 APL: Application Server

Figure 4: Overview of Structure of NS in 3GPP 5GS

Authors' Addresses

Shunsuke Homma
 NTT
 Japan
 Email: shunsuke.homma.fp@hco.ntt.co.jp

Hidetaka Nishihara
NTT
Japan

Email: nishihara.hidetaka@lab.ntt.co.jp

Takuya Miyasaka
KDDI Research
Japan

Email: ta-miyasaka@kddi-research.jp

Alex Galis
University College London
UK

Email: a.galis@ucl.ac.uk

Vishnu Ram OV
Independent Research Consultant India
India

Email: vishnu.u@ieee.org

Diego R. Lopez
Telefonica I+D
Spain

Email: diego.r.lopez@telefonica.com

Luis M. Contreras-Murillo
Telefonica I+D
Spain

Email: luismiguel.contrerasmurillo@telefonica.com

Jose A. Ordonez-Lucena
Telefonica I+D
Spain

Email: joseantonio.ordonezlucena@telefonica.com

Pedro Martinez-Julia
NICT
Japan

Email: pedro@nict.go.jp

Li Qiang
Huawei Technologies
China

Email: qiangli3@huawei.com

Reza Rokui
Nokia
Canada

Email: reza.rokui@nokia.com

Laurent Ciavaglia
Nokia
France

Email: Laurent.ciavaglia@nokia.com

Xavier de Foy
InterDigital Inc.
Canada

Email: Xavier.Defoy@InterDigital.com

OPSAWG
Internet-Draft
Intended status: Informational
Expires: 6 June 2022

H. Song
Futurewei
F. Qin
China Mobile
P. Martinez-Julia
NICT
L. Ciavaglia
Rakuten Mobile
A. Wang
China Telecom
3 December 2021

Network Telemetry Framework
draft-ietf-opsawg-ntf-13

Abstract

Network telemetry is a technology for gaining network insight and facilitating efficient and automated network management. It encompasses various techniques for remote data generation, collection, correlation, and consumption. This document describes an architectural framework for network telemetry, motivated by challenges that are encountered as part of the operation of networks and by the requirements that ensue. This document clarifies the terminologies and classifies the modules and components of a network telemetry system from different perspectives. The framework and taxonomy help to set a common ground for the collection of related work and provide guidance for related technique and standard developments.

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 6 June 2022.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Applicability Statement	4
1.2. Glossary	4
2. Background	6
2.1. Telemetry Data Coverage	7
2.2. Use Cases	8
2.3. Challenges	9
2.4. Network Telemetry	11
2.5. The Necessity of a Network Telemetry Framework	13
3. Network Telemetry Framework	14
3.1. Top Level Modules	15
3.1.1. Management Plane Telemetry	18
3.1.2. Control Plane Telemetry	18
3.1.3. Forwarding Plane Telemetry	19
3.1.4. External Data Telemetry	21
3.2. Second Level Function Components	22
3.3. Data Acquisition Mechanism and Type Abstraction	24
3.4. Mapping Existing Mechanisms into the Framework	26
4. Evolution of Network Telemetry Applications	27
5. Security Considerations	28
6. IANA Considerations	29
7. Contributors	29
8. Acknowledgments	29
9. Informative References	29
Appendix A. A Survey on Existing Network Telemetry Techniques	35
A.1. Management Plane Telemetry	35
A.1.1. Push Extensions for NETCONF	35
A.1.2. gRPC Network Management Interface	36
A.2. Control Plane Telemetry	36
A.2.1. BGP Monitoring Protocol	36
A.3. Data Plane Telemetry	36
A.3.1. The Alternate Marking (AM) technology	36

A.3.2. Dynamic Network Probe	38
A.3.3. IP Flow Information Export (IPFIX) Protocol	38
A.3.4. In-Situ OAM	38
A.3.5. Postcard Based Telemetry	39
A.3.6. Existing OAM for Specific Data Planes	39
A.4. External Data and Event Telemetry	39
A.4.1. Sources of External Events	39
A.4.2. Connectors and Interfaces	41
Authors' Addresses	41

1. Introduction

Network visibility is the ability of management tools to see the state and behavior of a network, which is essential for successful network operation. Network Telemetry revolves around network data that can help provide insights about the current state of the network, including network devices, forwarding, control, and management planes, and that can be generated and obtained through a variety of techniques, including but not limited to network instrumentation and measurements, and that can be processed for purposes ranging from service assurance to network security using a wide variety of data analytical techniques. In this document, Network Telemetry refer to both the data itself (i.e., "Network Telemetry Data"), and the techniques and processes used to generate, export, collect, and consume that data for use by potentially automated management applications. Network telemetry extends beyond the classical network Operations, Administration, and Management (OAM) techniques and expects to support better flexibility, scalability, accuracy, coverage, and performance.

However, the term "network telemetry" lacks an unambiguous definition. The scope and coverage of it cause confusion and misunderstandings. It is beneficial to clarify the concept and provide a clear architectural framework for network telemetry, so we can articulate the technical field, and better align the related techniques and standard works.

To fulfill such an undertaking, we first discuss some key characteristics of network telemetry which set a clear distinction from the conventional network OAM and show that some conventional OAM technologies can be considered a subset of the network telemetry technologies. We then provide an architectural framework for network telemetry which includes four modules, each concerned with a different category of telemetry data and corresponding procedures. All the modules are internally structured in the same way, including components that allow the operator to configure data sources in regard to what data to generate and how to make that available to client applications, components that instrument the underlying data

sources, and components that perform the actual rendering, encoding, and exporting of the generated data. We show how the network telemetry framework can benefit the current and future network operations. Based on the distinction of modules and function components, we can map the existing and emerging techniques and protocols into the framework. The framework can also simplify designing, maintaining, and understanding a network telemetry system. In addition, we outline the evolution stages of the network telemetry system and discuss the potential security concerns.

The purpose of the framework and taxonomy is to set a common ground for the collection of related work and provide guidance for future technique and standard developments. To the best of our knowledge, this document is the first such effort for network telemetry in industry standards organizations. This document does not define specific technologies.

1.1. Applicability Statement

Large-scale network data collection is a major threat to user privacy and may be indistinguishable from pervasive monitoring [RFC7258]. The network telemetry framework presented in this document must not be applied to generating, exporting, collecting, analyzing, or retaining individual user data or any data that can identify end users or characterize their behavior without consent. Based on this principle, the network telemetry framework is not applicable to networks whose endpoints represent individual users, such as general-purpose access networks.

1.2. Glossary

Before further discussion, we list some key terminology and acronyms used in this document. We make an intended differentiation between the terms of network telemetry and OAM. However, it should be understood that there is not a hard-line distinction between the two concepts. Rather, network telemetry is considered as an extension of OAM. It covers all the existing OAM protocols but puts more emphasis on the newer and emerging techniques and protocols concerning all aspects of network data from acquisition to consumption.

AI: Artificial Intelligence. In the network domain, AI refers to the machine-learning based technologies for automated network operation and other tasks.

AM: Alternate Marking, a flow performance measurement method, specified in [RFC8321].

BMP: BGP Monitoring Protocol, specified in [RFC7854].

- DPI: Deep Packet Inspection, referring to the techniques that examines packet beyond packet L3/L4 headers.
- gNMI: gRPC Network Management Interface, a network management protocol from OpenConfig Operator Working Group, mainly contributed by Google. See [gnmi] for details.
- GPB: Google Protocol Buffer, an extensible mechanism for serializing structured data. See [gpb] for details.
- gRPC: gRPC Remote Procedure Call, an open source high performance RPC framework that gNMI is based on. See [grpc] for details.
- IPFIX: IP Flow Information Export Protocol, specified in [RFC7011].
- IOAM: In-situ OAM [I-D.ietf-ippm-ioam-data], a dataplane on-path telemetry technique.
- JSON: An open standard file format and data interchange format that uses human-readable text to store and transmit data objects, specified in [RFC8259].
- MIB: Management Information Base, a database used for managing the entities in a network.
- NETCONF: Network Configuration Protocol, specified in [RFC6241].
- NetFlow: A Cisco protocol for flow record collecting, described in [RFC3954].
- Network Telemetry: The process and instrumentation for acquiring and utilizing network data remotely for network monitoring and operation. A general term for a large set of network visibility techniques and protocols, concerning aspects like data generation, collection, correlation, and consumption. Network telemetry addresses the current network operation issues and enables smooth evolution toward future intent-driven autonomous networks.
- NMS: Network Management System, referring to applications that allow network administrators to manage a network.
- OAM: Operations, Administration, and Maintenance. A group of network management functions that provide network fault indication, fault localization, performance information, and data and diagnosis functions. Most conventional network monitoring techniques and protocols belong to network OAM.
- PBT: Postcard-Based Telemetry, a dataplane on-path telemetry

technique. A representative technique is described in [I-D.ietf-ippm-ioam-direct-export].

RESTCONF: An HTTP-based protocol that provides a programmatic interface for accessing data defined in YANG, using the datastore concepts defined in NETCONF, as specified in [RFC8040].

SMIv2: Structure of Management Information Version 2, defining MIB objects, specified in [RFC2578].

SNMP: Simple Network Management Protocol. Version 1, 2, and 3 are specified in [RFC1157], [RFC3416], and [RFC3411], respectively.

XML: Extensible Markup Language is a markup language for data encoding that is both human-readable and machine-readable, specified by W3C [xml].

YANG: YANG is a data modeling language for the definition of data sent over network management protocols such as the NETCONF and RESTCONF. YANG is defined in [RFC6020] and [RFC7950].

YANG ECA: A YANG model for Event-Condition-Action policies, defined in [I-D.www-netmod-event-yang].

YANG-Push: A mechanism that allows subscriber applications to request a stream of updates from a YANG datastore on a network device. Details are specified in [RFC8641] and [RFC8639].

2. Background

The term "big data" is used to describe the extremely large volume of data sets that can be analyzed computationally to reveal patterns, trends, and associations. Networks are undoubtedly a source of big data because of their scale and the volume of network traffic they forward. When a network's endpoints do not represent individual users (e.g. in industrial, datacenter, and infrastructure contexts), network operations can often benefit from large-scale data collection without breaching user privacy.

Today one can access advanced big data analytics capability through a plethora of commercial and open source platforms (e.g., Apache Hadoop), tools (e.g., Apache Spark), and techniques (e.g., machine learning). Thanks to the advance of computing and storage technologies, network big data analytics gives network operators an opportunity to gain network insights and move towards network autonomy. Some operators start to explore the application of Artificial Intelligence (AI) to make sense of network data. Software tools can use the network data to detect and react on network faults,

anomalies, and policy violations, as well as predicting future events. In turn, the network policy updates for planning, intrusion prevention, optimization, and self-healing may be applied.

It is conceivable that an autonomic network [RFC7575] is the logical next step for network evolution following Software Defined Networking (SDN), aiming to reduce (or even eliminate) human labor, make more efficient use of network resources, and provide better services more aligned with customer requirements. The IETF ANIMA working group is dedicated to developing and maintaining protocols and procedures for automated network management and control of professionally-managed networks. The related technique of Intent-based Networking (IBN) [I-D.irtf-nmrg-ibn-concepts-definitions] requires network visibility and telemetry data in order to ensure that the network is behaving as intended.

However, while the data processing capability is improved and applications require more data to function better, the networks lag behind in extracting and translating network data into useful and actionable information in efficient ways. The system bottleneck is shifting from data consumption to data supply. Both the number of network nodes and the traffic bandwidth keep increasing at a fast pace. The network configuration and policy change at smaller time slots than before. More subtle events and fine-grained data through all network planes need to be captured and exported in real time. In a nutshell, it is a challenge to get enough high-quality data out of the network in a manner that is efficient, timely, and flexible. Therefore, we need to survey the existing technologies and protocols and identify any potential gaps.

In the remainder of this section, first we clarify the scope of network data (i.e., telemetry data) relevant in this document. Then, we discuss several key use cases for today's and future network operations. Next, we show why the current network OAM techniques and protocols are insufficient for these use cases. The discussion underlines the need for new methods, techniques, and protocols, as well as the extensions of existing ones, which we assign under the umbrella term - Network Telemetry.

2.1. Telemetry Data Coverage

Any information that can be extracted from networks (including data plane, control plane, and management plane) and used to gain visibility or as basis for actions is considered telemetry data. It includes statistics, event records and logs, snapshots of state, configuration data, etc. It also covers the outputs of any active and passive measurements [RFC7799]. In some cases, raw data is processed in network before being sent to a data consumer. Such

processed data is also considered telemetry data. The value of telemetry data varies. In some cases, if the cost is acceptable, less but higher quality data are preferred than lots of low quality data. A classification of telemetry data is provided in Section 3. To preserve the privacy of end-users, no user packet content should be collected. Specifically, the data objects generated, exported, and collected by a network telemetry application should not include any packet payload from traffic associated with end-users systems.

2.2. Use Cases

The following set of use cases is essential for network operations. While the list is by no means exhaustive, it is enough to highlight the requirements for data velocity, variety, volume, and veracity, the attributes of big data, in networks.

- * **Security:** Network intrusion detection and prevention systems need to monitor network traffic and activities and act upon anomalies. Given increasingly sophisticated attack vectors coupled with increasingly severe consequences of security breaches, new tools and techniques need to be developed, relying on wider and deeper visibility into networks. The ultimate goal is to achieve security with no, or only minimal, human intervention, and without disrupting legitimate traffic flows.
- * **Policy and Intent Compliance:** Network policies are the rules that constrain the services for network access, provide service differentiation, or enforce specific treatment on the traffic. For example, a service function chain is a policy that requires the selected flows to pass through a set of ordered network functions. Intent, as defined in [I-D.irtf-nmrg-ibn-concepts-definitions], is a set of operational goals that a network should meet and outcomes that a network is supposed to deliver, defined in a declarative manner without specifying how to achieve or implement them. An intent requires a complex translation and mapping process before being applied on networks. While a policy or intent is enforced, the compliance needs to be verified and monitored continuously by relying on visibility that is provided through network telemetry data. Any violation must be reported immediately, potentially resulting in updates to how the policy or intent is applied in the network to ensure that it remains in force, or otherwise alerting the network administrator to the policy or intent violation.
- * **SLA Compliance:** A Service-Level Agreement (SLA) is a service contract between a service provider and a client, which include the metrics for the service measurement and remedy/penalty procedures when the service level misses the agreement. Users

need to check if they get the service as promised and network operators need to evaluate how they can deliver services that can meet the SLA based on realtime network telemetry data, including data from network measurements.

- * **Root Cause Analysis:** Many network failure can be the effect of a sequence of chained events. Troubleshooting and recovery require quick identification of the root cause of any observable issues. However, the root cause is not always straightforward to identify, especially when the failure is sporadic and the number of event messages, both related and unrelated to the same cause, is overwhelming. While technologies such as machine learning can be used for root cause analysis, it is up to the network to sense and provide the relevant diagnostic data which are either actively fed into, or passively retrieved by, the root cause analysis applications.
- * **Network Optimization:** This covers all short-term and long-term network optimization techniques, including load balancing, Traffic Engineering (TE), and network planning. Network operators are motivated to optimize their network utilization and differentiate services for better Return On Investment (ROI) or lower Capital Expenditures (CAPEX). The first step is to know the real-time network conditions before applying policies for traffic manipulation. In some cases, micro-bursts need to be detected in a very short time-frame so that fine-grained traffic control can be applied to avoid network congestion. Long-term planning of network capacity and topology requires analysis of real-world network telemetry data that is obtained over long periods of time.
- * **Event Tracking and Prediction:** The visibility into traffic path and performance is critical for services and applications that rely on healthy network operation. Numerous related network events are of interest to network operators. For example, Network operators want to learn where and why packets are dropped for an application flow. They also want to be warned of issues in advance, so proactive actions can be taken to avoid catastrophic consequences.

2.3. Challenges

For a long time, network operators have relied upon SNMP [RFC3416], Command-Line Interface (CLI), or Syslog [RFC5424] to monitor the network. Some other OAM techniques as described in [RFC7276] are also used to facilitate network troubleshooting. These conventional techniques are not sufficient to support the above use cases for the following reasons:

- * Most use cases need to continuously monitor the network and dynamically refine the data collection in real-time. Poll-based low-frequency data collection is ill-suited for these applications. Subscription-based streaming data directly pushed from the data source (e.g., the forwarding chip) is preferred to provide sufficient data quantity and precision at scale.
- * Comprehensive data is needed, ranging from packet processing engines to traffic manager, from line cards to main control board, from user flows to control protocol packets, from device configurations to operations, and from physical layer to application layer. Conventional OAM only covers a narrow range of data (e.g., SNMP only handles data from the Management Information Base (MIB)). Classical network devices cannot provide all the necessary probes. More open and programmable network devices are therefore needed.
- * Many application scenarios need to correlate network-wide data from multiple sources (i.e., from distributed network devices, different components of a network device, or different network planes). A piecemeal solution is often lacking the capability to consolidate the data from multiple sources. The composition of a complete solution, as partly proposed by Autonomic Resource Control Architecture (ARCA) [I-D.pedro-nmrg-anticipated-adaptation], will be empowered and guided by a comprehensive framework.
- * Some conventional OAM techniques (e.g., CLI and Syslog) lack a formal data model. The unstructured data hinder the tool automation and application extensibility. Standardized data models are essential to support the programmable networks.
- * Although some conventional OAM techniques support data push (e.g., SNMP Trap [RFC2981][RFC3877], Syslog, and sFlow [RFC3176]), the pushed data are limited to only predefined management plane warnings (e.g., SNMP Trap) or sampled user packets (e.g., sFlow). Network operators require the data with arbitrary source, granularity, and precision which are beyond the capability of the existing techniques.
- * The conventional passive measurement techniques can either consume excessive network resources and produce excessive redundant data, or lead to inaccurate results; on the other hand, the conventional active measurement techniques can interfere with the user traffic and their results are indirect. Techniques that can collect direct and on-demand data from user traffic are more favorable.

These challenges were addressed by newer standards and techniques (e.g., IPFIX/Netflow, Packet Sampling (PSAMP), IOAM, and YANG-Push) and more are emerging. These standards and techniques need to be recognized and accommodated in a new framework.

2.4. Network Telemetry

Network telemetry has emerged as a mainstream technical term to refer to the network data collection and consumption techniques. Several network telemetry techniques and protocols (e.g., IPFIX [RFC7011] and gRPC [grpc]) have been widely deployed. Network telemetry allows separate entities to acquire data from network devices so that data can be visualized and analyzed to support network monitoring and operation. Network telemetry covers the conventional network OAM and has a wider scope. For instance, it is expected that network telemetry can provide the necessary network insight for autonomous networks and address the shortcomings of conventional OAM techniques.

Network telemetry usually assumes machines as data consumers rather than human operators. Hence, the network telemetry can directly trigger the automated network operation, while in contrast some conventional OAM tools were designed and used to help human operators to monitor and diagnose the networks and guide manual network operations. Such a proposition leads to very different techniques.

Although new network telemetry techniques are emerging and subject to continuous evolution, several characteristics of network telemetry have been well accepted. Note that network telemetry is intended to be an umbrella term covering a wide spectrum of techniques, so the following characteristics are not expected to be held by every specific technique.

- * Push and Streaming: Instead of polling data from network devices, telemetry collectors subscribe to streaming data pushed from data sources in network devices.
- * Volume and Velocity: The telemetry data is intended to be consumed by machines rather than by human being. Therefore, the data volume can be huge and the processing is optimized for the needs of automation in realtime.
- * Normalization and Unification: Telemetry aims to address the overall network automation needs. Efforts are made to normalize the data representation and unify the protocols, so as to simplify data analysis and provide integrated analysis across heterogeneous devices and data sources across a network.

- * **Model-based:** The telemetry data is modeled in advance which allows applications to configure and consume data with ease.
- * **Data Fusion:** The data for a single application can come from multiple data sources (e.g., cross-domain, cross-device, and cross-layer) based on common naming/ID and needs to be correlated to take effect.
- * **Dynamic and Interactive:** Since the network telemetry means to be used in a closed control loop for network automation, it needs to run continuously and adapt to the dynamic and interactive queries from the network operation controller.

In addition, an ideal network telemetry solution may also have the following features or properties:

- * **In-Network Customization:** The data that is generated can be customized in network at run-time to cater to the specific need of applications. This needs the support of a programmable data plane which allows probes with custom functions to be deployed at flexible locations.
- * **In-Network Data Aggregation and Correlation:** Network devices and aggregation points can work out which events and what data needs to be stored, reported, or discarded thus reducing the load on the central collection and processing points while still ensuring that the right information is ready to be processed in a timely way.
- * **In-Network Processing:** Sometimes it is not necessary or feasible to gather all information to a central point to be processed and acted upon. It is possible for the data processing to be done in network, allowing reactive actions to be taken locally.
- * **Direct Data Plane Export:** The data originated from the data plane forwarding chips can be directly exported to the data consumer for efficiency, especially when the data bandwidth is large and the real-time processing is required.
- * **In-band Data Collection:** In addition to the passive and active data collection approaches, the new hybrid approach allows to directly collect data for any target flow on its entire forwarding path [I-D.song-opsawg-ifit-framework].

It is worth noting that a network telemetry system should not be intrusive to normal network operations by avoiding the pitfall of the "observer effect". That is, it should not change the network behavior and affect the forwarding performance. Moreover, high-volume telemetry traffic may cause network congestion unless proper

isolation or traffic engineering techniques are in place, or congestion control mechanisms ensure that telemetry traffic backs off if it exceeds the network capacity. [RFC8084] and [RFC8085] are relevant Best Current Practices (BCP) in this space.

Although in many cases a system for network telemetry involves a remote data collecting and consuming entity, it is important to understand that there are no inherent assumptions about how a system should be architected. While a network architecture with centralized controller (e.g., SDN) seems a natural fit for network telemetry, network telemetry can work in distributed fashions as well. For example, telemetry data producers and consumers can have a peer-to-peer relationship, in which a network node can be the direct consumer of telemetry data from other nodes.

2.5. The Necessity of a Network Telemetry Framework

Network data analytics (e.g., machine learning) is applied for network operation automation, relying on abundant and coherent data from networks. Data acquisition that is limited to a single source and static in nature will in many cases not be sufficient to meet an application's telemetry data needs. As a result, multiple data sources, involving a variety of techniques and standards, will need to be integrated. It is desirable to have a framework that classifies and organizes different telemetry data source and types, defines different components of a network telemetry system and their interactions, and helps coordinate and integrate multiple telemetry approaches across layers. This allows flexible combinations of data for different applications, while normalizing and simplifying interfaces. In detail, such a framework would benefit the development of network operation applications for the following reasons:

- * Future networks, autonomous or otherwise, depend on holistic and comprehensive network visibility. The use cases and applications are better to be supported uniformly and coherently using an integrated, converged mechanism and common telemetry data representations wherever feasible. Therefore, the protocols and mechanisms should be consolidated into a minimum yet comprehensive set. A telemetry framework can help to normalize the technique developments.
- * Network visibility presents multiple viewpoints. For example, the device viewpoint takes the network infrastructure as the monitoring object from which the network topology and device status can be acquired; the traffic viewpoint takes the flows or packets as the monitoring object from which the traffic quality and path can be acquired. An application may need to switch its

viewpoint during operation. It may also need to correlate a service and its impact on user experience to acquire the comprehensive information.

- * Applications require network telemetry to be elastic in order to make efficient use of network resources and reduce the impact of processing related to network telemetry on network performance. For example, routine network monitoring should cover the entire network with a low data sampling rate. Only when issues arise or critical trends emerge should telemetry data sources be modified and telemetry data rates boosted as needed.
- * Efficient data aggregation is critical for applications to reduce the overall quantity of data and improve the accuracy of analysis.

A telemetry framework collects together all the telemetry-related works from different sources and working groups within IETF. This makes it possible to assemble a comprehensive network telemetry system and to avoid repetitious or redundant work. The framework should cover the concepts and components from the standardization perspective. This document describes the modules which make up a network telemetry framework and decomposes the telemetry system into a set of distinct components that existing and future work can easily map to.

3. Network Telemetry Framework

The top level network telemetry framework partitions the network telemetry into four modules based on the telemetry data object source and represents their relationship. Once the network operation applications acquire the data from these modules, they can apply data analytics and take actions. At the next level, the framework decomposes each module into separate components. Each of the modules follows the same underlying structure, with one component dedicated to the configuration of data subscriptions and data sources, a second component dedicated to encoding and exporting data, and a third component instrumenting the generation of telemetry related to the underlying resources. Throughout the framework, the same set of abstract data acquiring mechanisms and data types (Section 3.3) are applied. The two-level architecture with the uniform data abstraction helps accurately pinpoint a protocol or technique to its position in a network telemetry system or disaggregate a network telemetry system into manageable parts.

3.1. Top Level Modules

Telemetry can be applied on the forwarding plane, the control plane, and the management plane in a network, as well as other sources out of the network, as shown in Figure 1. Therefore, we categorize the network telemetry into four distinct modules (management plane, control plane, forwarding plane, and external data and event telemetry) with each having its own interface to Network Operation Applications.

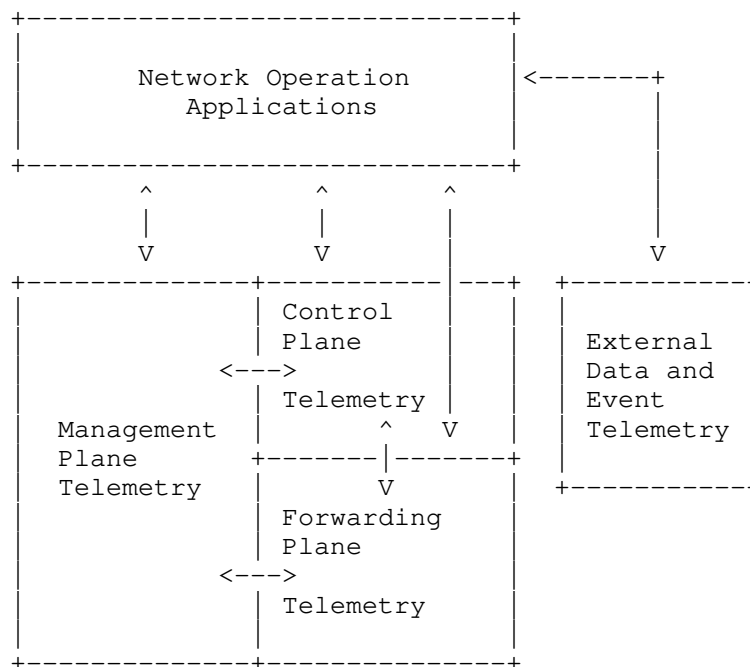


Figure 1: Modules in Layer Category of NTF

The rationale of this partition lies in the different telemetry data objects which result in different data source and export locations. Such differences have profound implications on in-network data programming and processing capability, data encoding and transport protocol, and required data bandwidth and latency. Data can be sent directly, or proxied via the control and management planes. There are advantages/disadvantages to both approaches.

Note that in some cases the network controller itself may be the source of telemetry data that is unique to it or derived from the telemetry data collected from the network elements. Some of the principles and taxonomy specific to the control plane and management

plane telemetry could also be applied to the controller when it is required to provide the telemetry data to Network Operation Applications hosted outside. The scope of the document is focused on the network elements telemetry and further details related to controllers are thus out of scope.

We summarize the major differences of the four modules in the following table. They are compared from six angles:

- * Data Object
- * Data Export Location
- * Data Model
- * Data Encoding
- * Telemetry Application Protocol
- * Data Transport Method

Data Object is the target and source of each module. Because the data source varies, the location where data is mostly conveniently exported also varies. For example, forwarding plane data mainly originates as data exported from the forwarding Application-Specific Integrated Circuits (ASICs), while control plane data mainly originates from the protocol daemons running on the control CPU(s). For convenience and efficiency, it is preferred to export the data off the device from locations near the source. Because the locations that can export data have different capabilities, different choices of data model, encoding, and transport method are made to balance the performance and cost. For example, the forwarding chip has high throughput but limited capacity for processing complex data and maintaining state, while the main control CPU is capable of complex data and state processing, but has limited bandwidth for high throughput data. As a result, the suitable telemetry protocol for each module can be different. Some representative techniques are shown in the corresponding table blocks to highlight the technical diversity of these modules. Note that the selected techniques just reflect the de facto state of the art and are by no means exhaustive (e.g., IPFIX can also be implemented over TCP and SCTP, but that is not recommended for forwarding plane). The key point is that one cannot expect to use a universal protocol to cover all the network telemetry requirements.

Module	Management Plane	Control Plane	Forwarding Plane	External Data
Object	config. & operation state	control protocol & signaling, RIB	flow & packet QoS, traffic stat., buffer & queue stat., ACL, FIB	terminal, social & environmental
Export Location	main control CPU	main control CPU, linecard CPU or forwarding chip	fwding chip or linecard CPU; main control CPU unlikely	various
Data Model	YANG, MIB, syslog	YANG, custom	YANG custom,	YANG, custom
Data Encoding	GPB, JSON, XML	GPB, JSON, XML, plain text	plain text	GPB, JSON XML, plain text
Application Protocol	gRPC, NETCONF, RESTCONF	gRPC, NETCONF, IPFIX, traffic mirroring	IPFIX, traffic mirroring, gRPC, NETFLOW	gRPC
Data Transport	HTTP(S), TCP	HTTP(S), TCP, UDP	UDP	HTTP(S), TCP, UDP

Figure 2: Comparison of the Data Object Modules

Note that the interaction with the applications that consume network telemetry data can be indirect. Some in-device data transfer is possible. For example, in the management plane telemetry, the management plane will need to acquire data from the data plane. Some operational states can only be derived from data plane data sources such as the interface status and statistics. As another example, obtaining control plane telemetry data may require the ability to access the Forwarding Information Base (FIB) of the data plane.

On the other hand, an application may involve more than one plane and interact with multiple planes simultaneously. For example, an SLA compliance application may require both the data plane telemetry and the control plane telemetry.

The requirements and challenges for each module are summarized as follows (note that the requirements may pertain across all telemetry modules; however, we emphasize those that are most pronounced for a particular plane).

3.1.1. Management Plane Telemetry

The management plane of network elements interacts with the Network Management System (NMS), and provides information such as performance data, network logging data, network warning and defects data, and network statistics and state data. The management plane includes many protocols, including the classical SNMP and syslog. Regardless the protocol, management plane telemetry must address the following requirements:

- * **Convenient Data Subscription:** An application should have the freedom to choose which data is exported (see section 4.3) and the means and frequency of how that data is exported (e.g., on-change or periodic subscription).
- * **Structured Data:** For automatic network operation, machines will replace human for network data comprehension. Data modeling languages, such as YANG, can efficiently describe structured data and normalize data encoding and transformation.
- * **High Speed Data Transport:** In order to keep up with the velocity of information, a data source needs to be able to send large amounts of data at high frequency. Compact encoding formats or data compression schemes are needed to reduce the quantity of data and improve the data transport efficiency. The subscription mode, by replacing the query mode, reduces the interactions between clients and servers and helps to improve the data source's efficiency.
- * **Network Congestion Avoidance:** The application must protect the network from congestion by congestion control mechanisms or at least circuit breakers. [RFC8084] and [RFC8085] provide some solutions in this space.

3.1.2. Control Plane Telemetry

The control plane telemetry refers to the health condition monitoring of different network control protocols at all layers of the protocol stack. Keeping track of the operational status of these protocols is beneficial for detecting, localizing, and even predicting various network issues, as well as network optimization, in real-time and with fine granularity. Some particular challenges and issues faced by the control plane telemetry are as follows:

- * One challenging problem for the control plane telemetry is how to correlate the End-to-End (E2E) Key Performance Indicators (KPI) to a specific layer's KPIs. For example, IPTV users may describe their User Experience (UE) by the video smoothness and definition. Then in case of an unusually poor UE KPI or a service disconnection, it is non-trivial to delimit and pinpoint the issue in the responsible protocol layer (e.g., the Transport Layer or the Network Layer), the responsible protocol (e.g., ISIS or BGP at the Network Layer), and finally the responsible device(s) with specific reasons.
- * Conventional OAM-based approaches for control plane KPI measurement include Ping (L3), Traceroute (L3), Y.1731 [y1731] (L2), and so on. One common issue behind these methods is that they only measure the KPIs instead of reflecting the actual running status of these protocols, making them less effective or efficient for control plane troubleshooting and network optimization.
- * An example of the control plane telemetry is the BGP monitoring protocol (BMP). It is currently used for monitoring the BGP routes and enables rich applications, such as BGP peer analysis, AS analysis, prefix analysis, and security analysis. However, the monitoring of other layers, protocols and the cross-layer, cross-protocol KPI correlations are still in their infancy (e.g., IGP monitoring is not as extensive as BMP), which require further research.
- * The requirement and solutions for network congestion avoidance are also applicable to the control plane telemetry.

3.1.3. Forwarding Plane Telemetry

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

- * A data plane device's main function is user traffic processing and forwarding. While supporting network visibility is important, the telemetry is just an auxiliary function, and it should strive to not impede normal traffic processing and forwarding (i.e., the forwarding behavior should not be altered and the trade-off between forwarding performance and telemetry should be well-balanced).

- * Network operation applications require end-to-end visibility across various sources, which can result in a huge volume of data. However, the sheer quantity of data must not exhaust the network bandwidth, regardless of the data delivery approach (i.e., whether through in-band or out-of-band channels).
- * The data plane devices must provide timely data with the minimum possible delay. Long processing, transport, storage, and analysis delay can impact the effectiveness of the control loop and even render the data useless.
- * The data should be structured and labeled, and easy for applications to parse and consume. At the same time, the data types needed by applications can vary significantly. The data plane devices need to provide enough flexibility and programmability to support the precise data provision for applications.
- * The data plane telemetry should support incremental deployment and work even though some devices are unaware of the system.
- * The requirement and solutions for network congestion avoidance are also applicable to the forwarding plane telemetry.

Although not specific to the forwarding plane, these challenges are more difficult to the forwarding plane because of the limited resource and flexibility. Data plane programmability is essential to support network telemetry. Newer data plane forwarding chips are equipped with advanced telemetry features and provide flexibility to support customized telemetry functions.

Technique Taxonomy: concerning about how one instruments the telemetry, there can be multiple possible dimensions to classify the forwarding plane telemetry techniques.

- * **Active, Passive, and Hybrid:** This dimension concerns about the end-to-end measurement. Active and passive methods (as well as the hybrid types) are well documented in [RFC7799]. Passive methods include TCPEUMP, IPFIX [RFC7011], sFlow, and traffic mirroring. These methods usually have low data coverage. The bandwidth cost is very high in order to improve the data coverage. On the other hand, active methods include Ping, OWAMP [RFC4656], TWAMP [RFC5357], STAMP [RFC8762], and Cisco's SLA Protocol [RFC6812]. These methods are intrusive and only provide indirect network measurements. Hybrid methods, including in-situ OAM [I-D.ietf-ippm-ioam-data], Alternate-Marking (AM) [RFC8321], and Multipoint Alternate Marking [RFC8889], provide a well-balanced and more flexible approach. However, these methods are also more complex to implement.
- * **In-Band and Out-of-Band:** Telemetry data carried in user packets before being exported to a data collector is considered in-band (e.g., in-situ OAM [I-D.ietf-ippm-ioam-data]). Telemetry data that is directly exported to a data collector without modifying user packets is considered out-of-band (e.g., the postcard-based approach described in Appendix A.3.5). It is also possible to have hybrid methods, where only the telemetry instruction or partial data is carried by user packets (e.g., AM [RFC8321]).
- * **End-to-End and In-Network:** End-to-End methods start from, and end at, the network end hosts (e.g., Ping). In-Network methods work in networks and are transparent to end hosts. However, if needed, In-Network methods can be easily extended into end hosts.
- * **Data Subject:** Depending on the telemetry objective, the methods can be flow-based (e.g., in-situ OAM [I-D.ietf-ippm-ioam-data]), path-based (e.g., Traceroute), and node-based (e.g., IPFIX [RFC7011]). The various data objects can be packet, flow record, measurement, states, and signal.

3.1.4. External Data Telemetry

Events that occur outside the boundaries of the network system are another important source of network telemetry. Correlating both internal telemetry data and external events with the requirements of network systems, as presented in [I-D.pedro-nmrg-anticipated-adaptation], provides a strategic and functional advantage to management operations.

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

- * The role of the external event detector can be played by multiple elements, including hardware (e.g., physical sensors, such as seismometers) and software (e.g., Big Data sources that can analyze streams of information, such as Twitter messages). Thus, the transmitted data must support different shapes but, at the same time, follow a common but extensible schema.
- * Since the main function of the external event detectors is to perform the notifications, their timeliness is assumed. However, once messages have been dispatched, they must be quickly collected and inserted into the control plane with variable priority, which is higher for important sources and events and lower for secondary ones.
- * The schema used by external detectors must be easily adopted by current and future devices and applications. Therefore, it must be easily mapped to current data models, such as in terms of YANG.
- * As the communication with external entities outside the boundary of a provider network may be realized over the Internet, the risk of congestion is even more relevant in this context and proper counter-measures must be taken. Solutions such as network transport circuit breakers are needed as well.

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

3.2. Second Level Function Components

The telemetry module at each plane can be further partitioned into five distinct conceptual components:

- * Data Query, Analysis, and Storage: This component works at the network operation application block in Figure 1. It is normally a part of the network management system at the receiver side. On the one hand, it is responsible for issuing data requirements. The data of interest can be modeled data through configuration or custom data through programming. The data requirements can be queries for one-shot data or subscriptions for events or streaming

data. On the other hand, it receives, stores, and processes the returned data from network devices. Data analysis can be interactive to initiate further data queries. This component can reside in either network devices or remote controllers. It can be centralized and distributed, and involve one or more instances.

- * **Data Configuration and Subscription:** This component manages data queries on devices. It determines the protocol and channel for applications to acquire desired data. This component is also responsible for configuring the desired data that might not be directly available from data sources. The subscription data can be described by models, templates, or programs.
- * **Data Encoding and Export:** This component determines how telemetry data is delivered to the data analysis and storage component with access control. The data encoding and the transport protocol may vary due to the data export location.
- * **Data Generation and Processing:** The requested data needs to be captured, filtered, processed, and formatted in network devices from raw data sources. This may involve in-network computing and processing on either the fast path or the slow path in network devices.
- * **Data Object and Source:** This component determines the monitoring objects and original data sources provisioned in the device. A data source usually just provides raw data which needs further processing. Each data source can be considered a probe. Some data sources can be dynamically installed, while others will be more static.

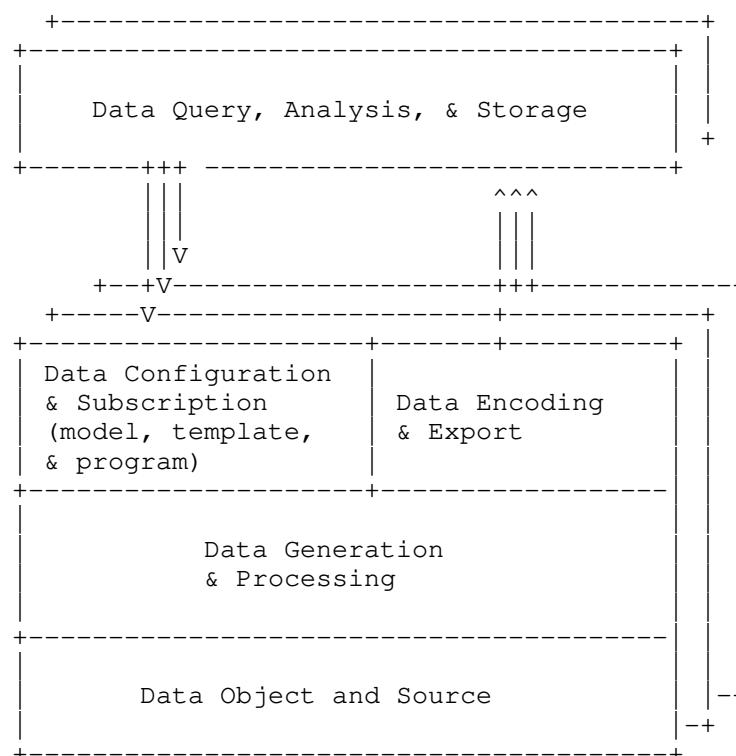


Figure 3: Components in the Network Telemetry Framework

3.3. Data Acquisition Mechanism and Type Abstraction

Broadly speaking, network data can be acquired through subscription (push) and query (poll). A subscription is a contract between publisher and subscriber. After initial setup, the subscribed data is automatically delivered to registered subscribers until the subscription expires. There are two variations of subscription. The subscriptions can be either pre-defined, or the subscribers are allowed to configure and tailor the published data to their specific needs.

In contrast, queries are used when a client expects immediate and one-off feedback from network devices. The queried data may be directly extracted from some specific data source, or synthesized and processed from raw data. Queries work well for interactive network telemetry applications.

In general, data can be pulled (i.e., queried) whenever needed, but in many cases, pushing the data (i.e., subscription) is more efficient, and can reduce the latency of a client detecting a change. From the data consumer point of view, there are four types of data from network devices that a telemetry data consumer can subscribe or query:

- * **Simple Data:** The data that are steadily available from some datastore or static probes in network devices.
- * **Derived Data:** The data need to be synthesized or processed in network from raw data from one or more network devices. The data processing function can be statically or dynamically loaded into network devices.
- * **Event-triggered Data:** The data are conditionally acquired based on the occurrence of some events. An example of event-triggered data could be an interface changing operational state between up and down. Such data can be actively pushed through subscription or passively polled through query. There are many ways to model events, including using Finite State Machine (FSM) or Event Condition Action (ECA) [I-D.www-netmod-event-yang].
- * **Streaming Data:** The data are continuously generated. It can be time series or the dump of databases. For example, an interface packet counter is exported every second. The streaming data reflect realtime network states and metrics and require large bandwidth and processing power. The streaming data are always actively pushed to the subscribers.

The above telemetry data types are not mutually exclusive. Rather, they are often composite. Derived data is composed of simple data; Event-triggered data can be simple or derived; streaming data can be based on some recurring event. The relationships of these data types are illustrated in Figure 4.

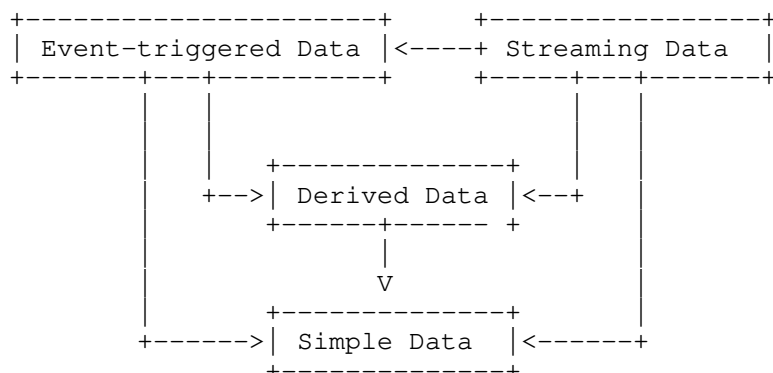


Figure 4: Data Type Relationship

Subscription usually deals with event-triggered data and streaming data, and query usually deals with simple data and derived data. But the other ways are also possible. Advanced network telemetry techniques are designed mainly for event-triggered or streaming data subscription, and derived data query.

3.4. Mapping Existing Mechanisms into the Framework

The following table shows how the existing mechanisms (mainly published in IETF and with the emphasis on the latest new technologies) are positioned in the framework. Given the vast body of existing work, we cannot provide an exhaustive list, so the mechanisms in the tables should be considered as just examples. Also, some comprehensive protocols and techniques may cover multiple aspects or modules of the framework, so a name in a block only emphasizes one particular characteristic of it. More details about some listed mechanisms can be found in Appendix A.

	Management Plane	Control Plane	Forwarding Plane
data config. & subscribe	gNMI, NETCONF, RESTCONF, SNMP, YANG-Push	gNMI, NETCONF, RESTCONF, YANG-Push	NETCONF, RESTCONF, YANG-Push
data gen. & process	MIB, YANG	YANG	IOAM, PSAMP PBT, AM,
data encode. & export	gRPC, HTTP, TCP	BMP, TCP	IPFIX, UDP

Figure 5: Existing Work Mapping

Although the framework is generally suitable for any network environments, the multi-domain telemetry has some unique challenges which deserve further architectural consideration, which is out of the scope of this document.

4. Evolution of Network Telemetry Applications

Network telemetry is an evolving technical area. As the network moves towards the automated operation, network telemetry applications undergo several stages of evolution which add new layer of requirements to the underlying network telemetry techniques. Each stage is built upon the techniques adopted by the previous stages plus some new requirements.

Stage 0 - Static Telemetry: The telemetry data source and type are determined at design time. The network operator can only configure how to use it with limited flexibility.

Stage 1 - Dynamic Telemetry: The custom telemetry data can be dynamically programmed or configured at runtime without interrupting the network operation, allowing a trade-off among resource, performance, flexibility, and coverage.

Stage 2 - Interactive Telemetry: The network operator can continuously customize and fine tune the telemetry data in real time to reflect the network operation's visibility requirements. Compared with Stage 1, the changes are frequent based on the real-time feedback. At this stage, some tasks can be automated, but human operators still need to sit in the middle to make decisions.

Stage 3 - Closed-loop Telemetry: The telemetry is free from the interference of human operators, except for generating the reports. The intelligent network operation engine automatically issues the telemetry data requests, analyzes the data, and updates the network operations in closed control loops.

Existing technologies are ready for stage 0 and stage 1. Individual stage 2 and stage 3 applications are also possible now. However, the future autonomic networks may need a comprehensive operation management system which works at stage 2 and stage 3 to cover all the network operation tasks. A well-defined network telemetry framework is the first step towards this direction.

5. Security Considerations

The complexity of network telemetry raises significant security implications. For example, telemetry data can be manipulated to exhaust various network resources at each plane as well as the data consumer; falsified or tampered data can mislead the decision-making and paralyze networks; wrong configuration and programming for telemetry is equally harmful. The telemetry data is highly sensitive, which exposes a lot of information about the network and its configuration. Some of that information can make designing attacks against the network much easier (e.g., exact details of what software and patches have been installed), and allows an attacker to determine whether a device may be subject to unprotected security vulnerabilities.

Given that this document has proposed a framework for network telemetry and the telemetry mechanisms discussed are more extensive (in both message frequency and traffic amount) than the conventional network OAM concepts, we must also reflect that various new security considerations may also arise. A number of techniques already exist for securing the forwarding plane, the control plane, and the management plane in a network, but it is important to consider if any new threat vectors are now being enabled via the use of network telemetry procedures and mechanisms.

This document proposes a conceptual architectural for collecting, transporting, and analyzing a wide variety of data sources in support of network applications. The protocols, data formats, and configurations chosen to implement this framework will dictate the specific security considerations. These considerations may include:

- * Telemetry framework trust and policy model;
- * Role management and access control for enabling and disabling telemetry capabilities;
- * Protocol transport used for telemetry data and its inherent security capabilities;
- * Telemetry data stores, storage encryption, methods of access, and retention practices;
- * Tracking telemetry events and any abnormalities that might identify malicious attacks using telemetry interfaces.
- * Authentication and integrity protection of telemetry data to make data more trustworthy.

- * Segregating the telemetry data traffic from the data traffic carried over the network (e.g., historically management access and management data may be carried via an independent management network).

Some security considerations highlighted above may be minimized or negated with policy management of network telemetry. In a network telemetry deployment it would be advantageous to separate telemetry capabilities into different classes of policies, i.e., Role Based Access Control and Event-Condition-Action policies. Also, potential conflicts between network telemetry mechanisms must be detected accurately and resolved quickly to avoid unnecessary network telemetry traffic propagation escalating into an unintended or intended denial of service attack.

Further study of the security issues will be required, and it is expected that the security mechanisms and protocols are developed and deployed along with a network telemetry system.

6. IANA Considerations

This document includes no request to IANA.

7. Contributors

The other contributors of this document are Tianran Zhou, Zhenbin Li, Zhenqiang Li, Daniel King, Adrian Farrel, and Alexander Clemm

8. Acknowledgments

We would like to thank Rob Wilton, Greg Mirsky, Randy Presuhn, Joe Clarke, Victor Liu, James Guichard, Uri Blumenthal, Giuseppe Fioccola, Yunan Gu, Parviz Yegani, Young Lee, Qin Wu, Gyan Mishra, Ben Schwartz, Alexey Melnikov, Michael Scharf, Dhruv Dhody, Martin Duke, Roman Danyliw, Warren Kumari, Sheng Jiang, Lars Eggert, Eric Vyncke, Jean-Michel Combes, Erik Kline, Benjamin Kaduk, and many others who have provided helpful comments and suggestions to improve this document.

9. Informative References

- [gnmi] "gNMI - gRPC Network Management Interface",
<<https://github.com/openconfig/reference/tree/master/rpc/gnmi>>.
- [gpb] "Google Protocol Buffers",
<<https://developers.google.com/protocol-buffers>>.

- [grpc] "gPPC, A high performance, open-source universal RPC framework", <<https://grpc.io>>.
- [I-D.ietf-grow-bmp-local-rib]
Evens, T., Bayraktar, S., Bhardwaj, M., and P. Lucente, "Support for Local RIB in BGP Monitoring Protocol (BMP)", Work in Progress, Internet-Draft, draft-ietf-grow-bmp-local-rib-13, 31 August 2021, <<https://www.ietf.org/archive/id/draft-ietf-grow-bmp-local-rib-13.txt>>.
- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-16, 8 November 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-16.txt>>.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-07, 13 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-direct-export-07.txt>>.
- [I-D.ietf-netconf-distributed-notif]
Zhou, T., Zheng, G., Voit, E., Graf, T., and P. Francois, "Subscription to Distributed Notifications", Work in Progress, Internet-Draft, draft-ietf-netconf-distributed-notif-02, 6 May 2021, <<https://www.ietf.org/archive/id/draft-ietf-netconf-distributed-notif-02.txt>>.
- [I-D.ietf-netconf-udp-notif]
Zheng, G., Zhou, T., Graf, T., Francois, P., Feng, A. H., and P. Lucente, "UDP-based Transport for Configured Subscriptions", Work in Progress, Internet-Draft, draft-ietf-netconf-udp-notif-04, 21 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-netconf-udp-notif-04.txt>>.
- [I-D.irtf-nmrg-ibn-concepts-definitions]
Clemm, A., Ciavaglia, L., Granville, L. Z., and J. Tantsura, "Intent-Based Networking - Concepts and Definitions", Work in Progress, Internet-Draft, draft-irtf-nmrg-ibn-concepts-definitions-05, 2 September 2021, <<https://www.ietf.org/archive/id/draft-irtf-nmrg-ibn-concepts-definitions-05.txt>>.

- [I-D.pedro-nmrg-anticipated-adaptation]
Martinez-Julia, P., "Exploiting External Event Detectors to Anticipate Resource Requirements for the Elastic Adaptation of SDN/NFV Systems", Work in Progress, Internet-Draft, draft-pedro-nmrg-anticipated-adaptation-02, 29 June 2018, <<https://www.ietf.org/archive/id/draft-pedro-nmrg-anticipated-adaptation-02.txt>>.
- [I-D.song-ippm-postcard-based-telemetry]
Song, H., Mirsky, G., Filsfils, C., Abdelsalam, A., Zhou, T., Li, Z., Shin, J., and K. Lee, "In-Situ OAM Marking-based Direct Export", Work in Progress, Internet-Draft, draft-song-ippm-postcard-based-telemetry-11, 15 November 2021, <<https://www.ietf.org/archive/id/draft-song-ippm-postcard-based-telemetry-11.txt>>.
- [I-D.song-opsawg-dnp4iq]
Song, H. and J. Gong, "Requirements for Interactive Query with Dynamic Network Probes", Work in Progress, Internet-Draft, draft-song-opsawg-dnp4iq-01, 19 June 2017, <<https://www.ietf.org/archive/id/draft-song-opsawg-dnp4iq-01.txt>>.
- [I-D.song-opsawg-ifit-framework]
Song, H., Qin, F., Chen, H., Jin, J., and J. Shin, "In-situ Flow Information Telemetry", Work in Progress, Internet-Draft, draft-song-opsawg-ifit-framework-16, 21 October 2021, <<https://www.ietf.org/archive/id/draft-song-opsawg-ifit-framework-16.txt>>.
- [I-D.wwx-netmod-event-yang]
Wu, Q., Bryskin, I., Birkholz, H., Liu, X., and B. Claise, "A YANG Data model for ECA Policy Management", Work in Progress, Internet-Draft, draft-wwx-netmod-event-yang-10, 1 November 2020, <<https://www.ietf.org/archive/id/draft-wwx-netmod-event-yang-10.txt>>.
- [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>>.
- [RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, DOI 10.17487/RFC2578, April 1999, <<https://www.rfc-editor.org/info/rfc2578>>.

- [RFC2981] Kavasseri, R., Ed., "Event MIB", RFC 2981, DOI 10.17487/RFC2981, October 2000, <<https://www.rfc-editor.org/info/rfc2981>>.
- [RFC3176] Phaal, P., Panchen, S., and N. McKee, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks", RFC 3176, DOI 10.17487/RFC3176, September 2001, <<https://www.rfc-editor.org/info/rfc3176>>.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, DOI 10.17487/RFC3411, December 2002, <<https://www.rfc-editor.org/info/rfc3411>>.
- [RFC3416] Presuhn, R., Ed., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, DOI 10.17487/RFC3416, December 2002, <<https://www.rfc-editor.org/info/rfc3416>>.
- [RFC3877] Chisholm, S. and D. Romascanu, "Alarm Management Information Base (MIB)", RFC 3877, DOI 10.17487/RFC3877, September 2004, <<https://www.rfc-editor.org/info/rfc3877>>.
- [RFC3954] Claise, B., Ed., "Cisco Systems NetFlow Services Export Version 9", RFC 3954, DOI 10.17487/RFC3954, October 2004, <<https://www.rfc-editor.org/info/rfc3954>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5085] Nadeau, T., Ed. and C. Pignataro, Ed., "Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires", RFC 5085, DOI 10.17487/RFC5085, December 2007, <<https://www.rfc-editor.org/info/rfc5085>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <<https://www.rfc-editor.org/info/rfc5424>>.

- [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>>.
- [RFC6812] Chiba, M., Clemm, A., Medley, S., Salowey, J., Thombare, S., and E. Yedavalli, "Cisco Service-Level Assurance Protocol", RFC 6812, DOI 10.17487/RFC6812, January 2013, <<https://www.rfc-editor.org/info/rfc6812>>.
- [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>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.
- [RFC7575] Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking: Definitions and Design Goals", RFC 7575, DOI 10.17487/RFC7575, June 2015, <<https://www.rfc-editor.org/info/rfc7575>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.

- [RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <<https://www.rfc-editor.org/info/rfc7854>>.
- [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>>.
- [RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <<https://www.rfc-editor.org/info/rfc8084>>.
- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <<https://www.rfc-editor.org/info/rfc8639>>.
- [RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <<https://www.rfc-editor.org/info/rfc8641>>.
- [RFC8671] Evens, T., Bayraktar, S., Lucente, P., Mi, P., and S. Zhuang, "Support for Adj-RIB-Out in the BGP Monitoring Protocol (BMP)", RFC 8671, DOI 10.17487/RFC8671, November 2019, <<https://www.rfc-editor.org/info/rfc8671>>.

- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8889] Fioccola, G., Ed., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8889, DOI 10.17487/RFC8889, August 2020, <<https://www.rfc-editor.org/info/rfc8889>>.
- [RFC8924] Aldrin, S., Pignataro, C., Ed., Kumar, N., Ed., Krishnan, R., and A. Ghanwani, "Service Function Chaining (SFC) Operations, Administration, and Maintenance (OAM) Framework", RFC 8924, DOI 10.17487/RFC8924, October 2020, <<https://www.rfc-editor.org/info/rfc8924>>.
- [xml] "Extensible Markup Language (XML) 1.0 (Fifth Edition)", <<https://www.w3.org/TR/2008/REC-xml-20081126/>>.
- [y1731] "ITU-T Y.1731: OAM Functions and Mechanisms for Ethernet based networks, 2015", <<https://www.itu.int/rec/T-REC-Y.1731/en>>.

Appendix A. A Survey on Existing Network Telemetry Techniques

In this non-normative appendix, we provide an overview of some existing techniques and standard proposals for each network telemetry module.

A.1. Management Plane Telemetry

A.1.1. Push Extensions for NETCONF

NETCONF [RFC6241] is a popular network management protocol recommended by IETF. Its core strength is for managing configuration, but can also be used for data collection. YANG-Push [RFC8641] [RFC8639] extends NETCONF and enables subscriber applications to request a continuous, customized stream of updates from a YANG datastore. Providing such visibility into changes made upon YANG configuration and operational objects enables new capabilities based on the remote mirroring of configuration and operational state. Moreover, distributed data collection mechanism [I-D.ietf-netconf-distributed-notif] via UDP based publication channel [I-D.ietf-netconf-udp-notif] provides enhanced efficiency for the NETCONF based telemetry.

A.1.2. gRPC Network Management Interface

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

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

A.2. Control Plane Telemetry

A.2.1. BGP Monitoring Protocol

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

The BGP routing information is collected from the monitored device(s) to the BMP monitoring station by setting up the BMP TCP session. The BGP peers are monitored by the BMP Peer Up and Peer Down Notifications. The BGP routes (including Adjacency_RIB_In [RFC7854], Adjacency_RIB_out [RFC8671], and Local_Rib [I-D.ietf-grow-bmp-local-rib]) are encapsulated in the BMP Route Monitoring Message and the BMP Route Mirroring Message, providing both an initial table dump and real-time route updates. In addition, BGP statistics are reported through the BMP Stats Report Message, which could be either timer triggered or event-driven. Future BMP extensions could further enrich BGP monitoring applications.

A.3. Data Plane Telemetry

A.3.1. The Alternate Marking (AM) technology

The Alternate Marking method enables efficient measurements of packet loss, delay, and jitter both in IP and Overlay Networks, as presented in [RFC8321] and [RFC8889].

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

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

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

The Multipoint Alternate Marking approach, described in [RFC8889], aims to resolve this issue and make the performance monitoring more flexible in case a detailed analysis is not needed.

An application orchestrates network performance measurements tasks across the network to allow for optimized monitoring. The application can choose how roughly or precisely to configure measurement points depending on the application's requirements.

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

In summary, an application can configure end-to-end network monitoring. If the network does not experience issues, this approximate monitoring is good enough and is very cheap in terms of network resources. However, in case of problems, the application becomes aware of the issues from this approximate monitoring and, in order to localize the portion of the network that has issues, configures the measurement points more extensively, allowing more

detailed monitoring to be performed. After the detection and resolution of the problem, the initial approximate monitoring can be used again.

A.3.2. Dynamic Network Probe

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

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

A.3.3. IP Flow Information Export (IPFIX) Protocol

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

A.3.4. In-Situ OAM

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

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

A.3.5. Postcard Based Telemetry

The postcard-based telemetry, as embodied in IOAM DEX [I-D.ietf-ippm-ioam-direct-export] and IOAM Marking [I-D.song-ippm-postcard-based-telemetry], is a complementary technique to the passport-based IOAM. PBT directly exports data at each node through an independent packet. At the cost of higher bandwidth overhead and the need for data correlation, PBT shows several unique advantages. It can also help to identify packet drop location in case a packet is dropped on its forwarding path.

A.3.6. Existing OAM for Specific Data Planes

Various data planes raise unique OAM requirements. IETF has published OAM technique and framework documents (e.g., [RFC8924] and [RFC5085]) targeting different data planes such as Multi-Protocol Label Switching (MPLS), L2 Virtual Private Network (L2-VPN), Network Virtualization Overlays (NVO3), Virtual Extensible LAN (VXLAN), Bit Indexed Explicit Replication (BIER), Service Function Chaining (SFC), Segment Routing (SR), and Deterministic Networking (DETNET). The aforementioned data plane telemetry techniques can be used to enhance the OAM capability on such data planes.

A.4. External Data and Event Telemetry

A.4.1. Sources of External Events

To ensure that the information provided by external event detectors and used by the network management solutions is meaningful for management purposes, the network telemetry framework must ensure that such detectors (sources) are easily connected to the management solutions (sinks). This requires the specification of a list of potential external data sources that could be of interest in network management and match it to the connectors and/or interfaces required to connect them.

Categories of external event sources that may be of interest to network management include::

- * Smart objects and sensors. With the consolidation of the Internet of Things~(IoT) any network system will have many smart objects attached to its physical surroundings and logical operation environments. Most of these objects will be essentially based on

sensors of many kinds (e.g., temperature, humidity, presence) and the information they provide can be very useful for the management of the network, even when they are not specifically deployed for such purpose. Elements of this source type will usually provide a specific protocol for interaction, especially one of those protocols related to IoT, such as the Constrained Application Protocol (CoAP).

- * Online news reporters. Several online news services have the ability to provide enormous quantity of information about different events occurring in the world. Some of those events can impact on the network system managed by a specific framework and, therefore, such information may be of interest to the management solution. For instance, diverse security reports, such as the Common Vulnerabilities and Exposures (CVE), can be issued by the corresponding authority and used by the management solution to update the managed system if needed. Instead of a specific protocol and data format, the sources of this kind of information usually follow a relaxed but structured format. This format will be part of both the ontology and information model of the telemetry framework.
- * Global event analyzers. The advance of Big Data analyzers provides a huge amount of information and, more interestingly, the identification of events detected by analyzing many data streams from different origins. In contrast with the other types of sources, which are focused on specific events, the detectors of this source type will detect generic events. For example, during a sport event some unexpected movement makes it fascinating and many people connect to sites that are reporting on the event. The underlying networks supporting the services that cover the event can be affected by such situation, so their management solutions should be aware of it. In contrast with the other source types, a new information model, format, and reporting protocol is required to integrate the detectors of this type with the management solution.

Additional types of detector types can be added to the system, but they will be generally the result of composing the properties offered by these main classes.

A.4.2. Connectors and Interfaces

For allowing external event detectors to be properly integrated with other management solutions, both elements must expose interfaces and protocols that are subject to their particular objective. Since external event detectors will be focused on providing their information to their main consumers, which generally will not be limited to the network management solutions, the framework must include the definition of the required connectors for ensuring the interconnection between detectors (sources) and their consumers within the management systems (sinks) are effective.

In some situations, the interconnection between the external event detectors and the management system is via the management plane. For those situations there will be a special connector that provides the typical interfaces found in most other elements connected to the management plane. For instance, the interfaces could accomplish this with a specific data model (YANG) and specific telemetry protocol, such as NETCONF, YANG-Push, or gRPC.

Authors' Addresses

Haoyu Song
Futurewei
United States of America

Email: haoyu.song@futurewei.com

Fengwei Qin
China Mobile
P.R. China

Email: qinfengwei@chinamobile.com

Pedro Martinez-Julia
NICT
Japan

Email: pedro@nict.go.jp

Laurent Ciavaglia
Rakuten Mobile
France

Email: laurent.ciavaglia@rakuten.com

Aijun Wang
China Telecom
P.R. China

Email: wangaj.bri@chinatelecom.cn

OPSAWG
Internet-Draft
Intended status: Informational
Expires: September 7, 2019

H. Song, Ed.
Huawei
ZQ. Li
China Mobile
P. Martinez-Julia
NICT
L. Ciavaglia
Nokia
A. Wang
China Telecom
March 6, 2019

Network Telemetry Framework
draft-song-opsawg-ntf-03

Abstract

This document provides an architectural framework for network telemetry to address the current and future network operation challenges and requirements. As evidenced by the defining characteristics and industry practice, network telemetry covers technologies and protocols beyond the conventional network Operations, Administration, and Management (OAM). Network telemetry promises better flexibility, scalability, accuracy, coverage, and performance and allows automated control loops to suit both today's and tomorrow's network operation requirements. This document clarifies the terminologies and classifies the modules and components of a network telemetry system. The framework and taxonomy help to set a common ground for the collection of related work and provide guidance for future technique and standard developments.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 7, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Requirements Language	3
2. Motivation	4
2.1. Use Cases	4
2.2. Challenges	5
2.3. Glossary	7
2.4. Network Telemetry	8
3. The Necessity of a Network Telemetry Framework	9
4. Network Telemetry Framework	10
4.1. Data Acquiring Mechanisms	11
4.2. Data Objects	12
4.3. Function Components	14
4.4. Existing Works Mapped in the Framework	16
5. Evolution of Network Telemetry	17
6. Security Considerations	18
7. IANA Considerations	19
8. Contributors	19
9. Acknowledgments	19
10. References	19
10.1. Normative References	19
10.2. Informative References	20
Appendix A. A Survey on Existing Network Telemetry Techniques .	23
A.1. Management Plane Telemetry	23
A.1.1. Requirements and Challenges	23
A.1.2. Push Extensions for NETCONF	23
A.1.3. gRPC Network Management Interface	24
A.2. Control Plane Telemetry	24
A.2.1. Requirements and Challenges	24
A.2.2. BGP Monitoring Protocol	25
A.3. Data Plane Telemetry	25
A.3.1. Requirements and Challenges	25

A.3.2.	Technique Taxonomy	26
A.3.3.	The IPFPM technology	27
A.3.4.	Dynamic Network Probe	28
A.3.5.	IP Flow Information Export (IPFIX) protocol	29
A.3.6.	In-Situ OAM	29
A.3.7.	Postcard Based Telemetry	29
A.4.	External Data and Event Telemetry	29
A.4.1.	Requirements and Challenges	30
A.4.2.	Sources of External Events	30
A.4.3.	Connectors and Interfaces	32
Authors' Addresses	32

1. Introduction

Network visibility is essential for network operation. Network telemetry has been widely considered as an ideal mean to gain sufficient network visibility with better flexibility, scalability, accuracy, coverage, and performance than conventional OAM technologies. However, confusion and misunderstandings about the network telemetry remain (e.g., the scope and coverage of the term). We need an unambiguous concept and a clear architectural framework for network telemetry so we can better align the related technology and standard work.

First, we show some key characteristics of network telemetry which set a clear distinction from the conventional network OAM and show that some conventional OAM technologies can be considered a subset of the network telemetry technologies. We then provide an architectural framework for network telemetry to meet the current and future network operation requirements. Following the framework, we classify the components of a network telemetry system so we can easily map the existing and emerging techniques and protocols into the framework. At last, we outline a roadmap for the evolution of the network telemetry system.

The purpose of the framework and taxonomy is to set a common ground for the collection of related work and provide guidance for future technique and standard developments.

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. Motivation

Thanks to the advance of the computing and storage technologies, today's big data analytics and machine learning-based Artificial Intelligence (AI) give network operators an unprecedented opportunity to gain network insights and move towards network autonomy. Software tools can use the network data to detect and react on network faults, anomalies, and policy violations, as well as predicting future events. In turn, the network policy updates for planning, intrusion prevention, optimization, and self-healing may be applied.

It is conceivable that an intent-driven autonomous network is the logical next step for network evolution following Software Defined Network (SDN), aiming to reduce (or even eliminate) human labor, make the most efficient use of network resources, and provide better services more aligned with customer requirements. Although it takes time to reach the ultimate goal, the journey has started nevertheless.

However, the system bottleneck is shifting from data consumption to data supply. Both the number of network nodes and the traffic bandwidth keep increasing at a fast pace; The network configuration and policy change at a much smaller time frame than ever before; More subtle events and fine-grained data through all network planes need to be captured and exported in real time. In a nutshell, it is challenging to get enough high-quality data out of network efficiently, timely, and flexibly. Therefore, we need to examine the existing network technologies and protocols, and identify any potential gaps based on the real network and device architectures.

In the remaining of this section, first we discuss several key use cases for today's and future network operations. Next, we show why the current network OAM techniques and protocols are insufficient for these use cases. The discussion underlines the need for new methods, techniques, and protocols which we may assign under an umbrella term - network telemetry.

2.1. Use Cases

These use cases are essential for network operations. While the list is by no means exhaustive, it is enough to highlight the requirements for data velocity, variety, and volume in networks.

Policy and Intent Compliance: Network policies are the rules that constraint the services for network access, provide service differentiation, or enforce specific treatment on the traffic. For example, a service function chain is a policy that requires the selected flows to pass through a set of ordered network

functions. An intent is a high-level abstract policy which requires a complex translation and mapping process before being applied on networks. While a policy is enforced, the compliance needs to be verified and monitored continuously.

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

Root Cause Analysis: Any network failure can be the cause or effect of a sequence of chained events. Troubleshooting and recovery require quick identification of the root cause of any observable issues. However, the root cause is not always straightforward to identify, especially when the failure is sporadic and the related and unrelated events are overwhelming. While machine learning technologies can be used for root cause analysis, it is up to the network to sense and provide all the relevant data.

Network Optimization: This covers all short-term and long-term network optimization techniques, including load balancing, Traffic Engineering (TE), and network planning. Network operators are motivated to optimize their network utilization and differentiate services for better ROI or lower CAPEX. The first step is to know the real-time network conditions before applying policies for traffic manipulation. In some cases, micro-bursts need to be detected in a very short time-frame so that fine-grained traffic control can be applied to avoid network congestion. The long-term network capacity planning and topology augmentation also rely on the accumulated data of the network operations.

Event Tracking and Prediction: The visibility of user traffic path and performance is critical for healthy network operation. Numerous related network events are of interest to network operators. For example, Network operators always want to learn where and why packets are dropped for an application flow. They also want to be warned of issues in advance so proactive actions can be taken to avoid catastrophic consequences.

2.2. Challenges

For a long time, network operators have relied upon SNMP [RFC3416], Command-Line Interface (CLI), or Syslog to monitor the network. Some other OAM techniques as described in [RFC7276] are also used to facilitate network troubleshooting. These conventional techniques

are not sufficient to support the above use cases for the following reasons:

- o Most use cases need to continuously monitor the network and dynamically refine the data collection in real-time and interactively. The poll-based low-frequency data collection is ill-suited for these applications. Subscription-based streaming data directly pushed from the data source (e.g., the forwarding chip) is preferred to provide enough data quantity and precision at scale.
- o Comprehensive data is needed from packet processing engine to traffic manager, from line cards to main control board, from user flows to control protocol packets, from device configurations to operations, and from physical layer to application layer. Conventional OAM only covers a narrow range of data (e.g., SNMP only handles data from the Management Information Base (MIB)). Traditional network devices cannot provide all the necessary probes. An open and programmable network device is therefore needed.
- o Many application scenarios need to correlate data from multiple sources (i.e., from distributed network devices, different components of a network device, or different network planes). A piecemeal solution is often lacking the capability to consolidate the data from multiple sources. The composition of a complete solution, as partly proposed by Autonomic Resource Control Architecture (ARCA) [I-D.pedro-nmrg-anticipated-adaptation], will be empowered and guided by a comprehensive framework.
- o Some of the conventional OAM techniques (e.g., CLI and Syslog) are lack of formal data model. The unstructured data hinder the tool automation and application extensibility. Standardized data models are essential to support the programmable networks.
- o Although some conventional OAM techniques support data push (e.g., SNMP Trap [RFC2981][RFC3877], Syslog, and sFlow), the pushed data are limited to only predefined management plane warnings (e.g., SNMP Trap) or sampled user packets (e.g., sFlow). We require the data with arbitrary source, granularity, and precision which are beyond the capability of the existing techniques.
- o The conventional passive measurement techniques can either consume too much network resources and render too much redundant data, or lead to inaccurate results; the conventional active measurement techniques can interfere with the user traffic and their results are indirect. We need techniques that can collect direct and on-demand data from user traffic.

2.3. Glossary

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

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

BMP: BGP Monitoring Protocol

DNP: Dynamic Network Probe

DPI: Deep Packet Inspection

gNMI: gRPC Network Management Interface

gRPC: gRPC Remote Procedure Call

IDN: Intent-Driven Network

IPFIX: IP Flow Information Export Protocol

IPFPM: IP Flow Performance Measurement

IOAM: In-situ OAM

NETCONF: Network Configuration Protocol

Network Telemetry: Acquiring network data remotely for network monitoring and operation. A general term for a large set of network visibility techniques and protocols, with the characteristics defined in this document. Network telemetry addresses the current network operation issues and enables smooth evolution toward intent-driven autonomous networks.

NMS: Network Management System

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

SNMP: Simple Network Management Protocol

YANG: A data modeling language for NETCONF

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

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

2.4. Network Telemetry

Network telemetry has emerged as a mainstream technical term to refer to the newer data collection and consumption techniques, distinguishing itself from the convention techniques for network OAM. The representative techniques and protocols include IPFIX [RFC7011] and gPRC [I-D.kumar-rtgwg-grpc-protocol]. Network telemetry allows separate entities to acquire data from network devices so that data can be visualized and analyzed to support network monitoring and operation. Network telemetry overlaps with the conventional network OAM and has a wider scope than it. It is expected that network telemetry can provide the necessary network insight for autonomous networks, address the shortcomings of conventional OAM techniques, and allow for the emergence of new techniques bearing certain characteristics.

One difference between the network telemetry and the network OAM is that the network telemetry assumes machines as data consumer, while the conventional network OAM usually assumes human operators. Hence, the network telemetry can directly trigger the automated network operation, but the conventional OAM tools only help human operators to monitor and diagnose the networks and guide manual network operations. The difference leads to very different techniques.

Although the network telemetry techniques are just emerging and subject to continuous evolution, several characteristics of network telemetry have been well accepted (Note that network telemetry is intended to be an umbrella term covering a wide spectrum of techniques, so the following characteristics are not expected to be held by every specific technique):

- o Push and Streaming: Instead of polling data from network devices, the telemetry collector subscribes to the streaming data pushed from data sources in network devices.
- o Volume and Velocity: The telemetry data is intended to be consumed by machine rather than by a human. Therefore, the data volume is huge and the processing is often in realtime.
- o Normalization and Unification: Telemetry aims to address the overall network automation needs. The piecemeal solutions offered by the conventional OAM approach are no longer suitable. Efforts

need to be made to normalize the data representation and unify the protocols.

- o Model-based: The telemetry data is modeled in advance which allows applications to configure and consume data with ease.
- o Data Fusion: The data for a single application can come from multiple data sources (e.g., cross-domain, cross-device, and cross-layer) and needs to be correlated to take effect.
- o Dynamic and Interactive: Since the network telemetry means to be used in a closed control loop for network automation, it needs to run continuously and adapt to the dynamic and interactive queries from the network operation controller.

Note that a technique does not need to have all the above characteristics to be qualified as telemetry. An ideal network telemetry solution may also have the following features or properties:

- o In-Network Customization: The data can be customized in network at run-time to cater to the specific need of applications. This needs the support of a programmable data plane which allows probes to be deployed at flexible locations.
- o Direct Data Plane Export: The data originated from data plane can be directly exported to the data consumer for efficiency, especially when the data bandwidth is large and the real-time processing is required.
- o In-band Data Collection: In addition to the passive and active data collection approaches, the new hybrid approach allows to directly collect data for any target flow on its entire forwarding path.
- o Non-intrusive: The telemetry system should avoid the pitfall of the "observer effect". That is, it should not change the network behavior and affect the forwarding performance.

3. The Necessity of a Network Telemetry Framework

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

framework would benefit application development for the following reasons:

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

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

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

4. Network Telemetry Framework

Network telemetry techniques can be classified from multiple dimensions. In this document, we provide three unique perspectives: data acquiring mechanisms, data objects, and function components.

4.1. Data Acquiring Mechanisms

Broadly speaking, network data can be acquired through subscription (push) and query (poll). A subscriber may request data when it is ready. It follows a pub-sub mode or a sub-pub mode. In the pub-sub mode, pre-defined data are published and multiple qualified subscribers can subscribe the data. In the sub-pub mode, a subscriber designates what data are of interest and demands the network devices to deliver the data when they are available.

In contrast, a querier expects immediate feedback from network devices. It is usually used in a more interactive environment. The queried data may be directly extracted from some specific data source, or synthesized and processed from raw data.

There are four types of data from network devices:

Simple Data: The data that are steadily available from some data store or static probes in network devices. such data can be specified by YANG model.

Custom Data: The data need to be synthesized or processed from raw data from one or more network devices. The data processing function can be statically or dynamically loaded into network devices.

Event-triggered Data: The data are conditionally acquired based on the occurrence of some event. An event can be modeled as a Finite State Machine (FSM).

Streaming Data: The data are continuously or periodically generated.

The above data types are not mutual exclusive. For example, event-triggered data can be simple or custom, and streaming data can be event triggered. The relationships of these data types are illustrated in Figure 1

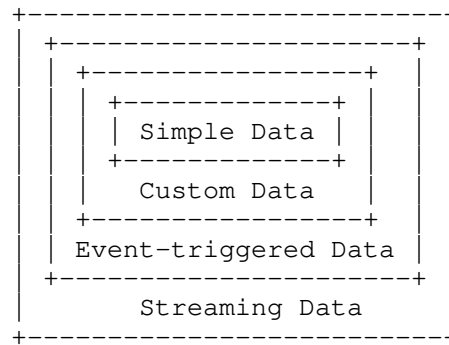


Figure 1: Data Type Relationship

Subscription usually deals with event-triggered data and streaming data, and query usually deals with simple data and custom data. It is easy to see that conventional OAM techniques are mostly about querying simple data only. While these techniques are still useful, advanced network telemetry techniques pay more attention on the other three data types, and prefer subscription and custom data query over simple data query.

4.2. Data Objects

Telemetry can be applied on the forwarding plane, the control plane, and the management plane in a network, as well as other sources out of the network, as shown in Figure 2. Therefore, we categorize the network telemetry into four distinct modules.

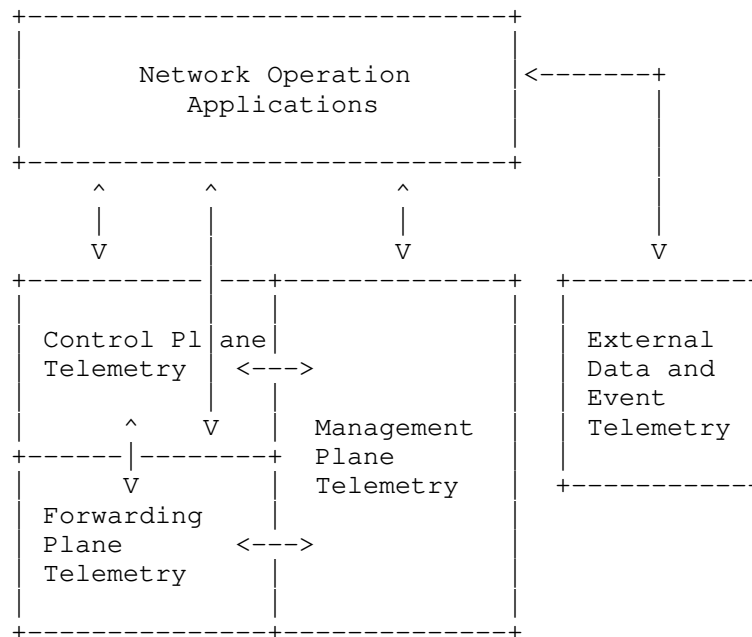


Figure 2: Layer Category of the Network Telemetry Framework

The rationale of this partition lies in the different telemetry data objects which result in different data source and export locations. Such differences have profound implications on in-network data programming and processing capability, data encoding and transport protocol, and data bandwidth and latency.

We summarize the major differences of the four modules in the following table. Some representative techniques are shown in some table blocks to highlight the technical diversity of these modules.

Module	Control Plane	Management Plane	Forwarding Plane	External Data
Object	control protocol & signaling, RIB, ACL	config. & operation state, MIB	flow & packet QoS, traffic stat., buffer & queue stat.	terminal, social & environmental
Export Location	main control CPU, linecard CPU or fwding chip	main control CPU	fwding chip or linecard CPU; main control CPU unlikely	various
Model	YANG, custom	MIB, syslog, YANG, custom	template, YANG, custom	YANG
Encoding	GPB, JSON, XML, plain	GPB, JSON, XML	plain	GPB, JSON, XML, plain
Protocol	gRPC, NETCONF, IPFIX, mirror	gRPC, NETCONF	IPFIX, mirror	gRPC
Transport	HTTP, TCP, UDP	HTTP, TCP	UDP	TCP, UDP

Figure 3: Layer Category of the Network Telemetry Framework

Note that the interaction with the network operation applications can be indirect. For example, in the management plane telemetry, the management plane may need to acquire data from the data plane. Some of the operational states can only be derived from the data plane such as the interface status and statistics. For another example, the control plane telemetry may need to access the FIB in data plane. On the other hand, an application may involve more than one plane simultaneously. For example, an SLA compliance application may require both the data plane telemetry and the control plane telemetry.

4.3. Function Components

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

Data Query, Analysis, and Storage: This component works at the application layer. On the one hand, it is responsible for issuing data queries. The queries can be for modeled data through configuration or custom data through programming. The queries can be one shot or subscriptions for events or streaming data. On the other hand, it receives, stores, and processes the returned data from network devices. Data analysis can be interactive to initiate further data queries.

Data Configuration and Subscription: This component deploys data queries on devices. It determines the protocol and channel for applications to acquire desired data. This component is also responsible for configuring the desired data that might not be directly available from data sources. The subscription data can be described by models, templates, or programs.

Data Encoding and Export: This component determines how telemetry data are delivered to the data analysis and storage component. The data encoding and the transport protocol may vary due to the data exporting location.

Data Generation and Processing: The requested data needs to be captured, processed, and formatted in network devices from raw data sources. This may involve in-network computing and processing on either the fast path or the slow path in network devices.

Data Object and Source: This component determines the monitoring object and original data source. The data source usually just provides raw data which needs further processing. A data source can be considered a probe. A probe can be statically installed or dynamically installed.

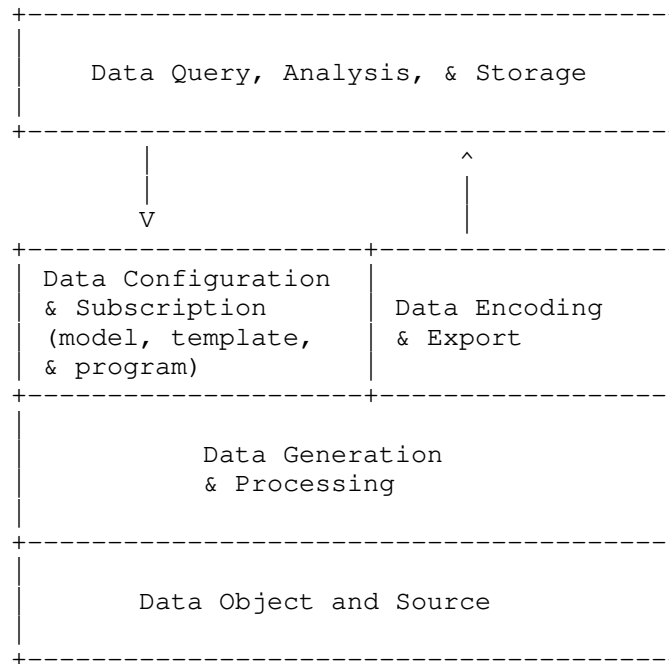


Figure 4: Components in the Network Telemetry Framework

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

4.4. Existing Works Mapped in the Framework

The following two tables provide a non-exhaustive list of existing works (mainly published in IETF and with the emphasis on the latest new technologies) and shows their positions in the framework. The details about the mentioned work can be found in Appendix A.

	Query	Subscription
Simple Data	SNMP, NETCONF, YANG, BMP, IOAM, PBT	
Custom Data	DNP, YANG FSM gRPC, NETCONF	
Event-triggered Data		gRPC, NETCONF, YANG PUSH, DNP IOAM, PBT, YANG FSM
Streaming Data		gRPC, NETCONF, IOAM, PBT, DNP IPFIX, IPFPM

Figure 5: Existing Work Mapping I

	Management Plane	Control Plane	Forwarding Plane
data Config. & subscrib.	gRPC, NETCONF, YANG PUSH	NETCONF/YANG	NETCONF/YANG, YANG FSM
data gen. & processing	DNP, YANG	DNP, YANG	In-situ OAM, PBT, IPFPM, DNP
data export	gRPC, NETCONF YANG PUSH	BMP, NETCONF	IPFIX

Figure 6: Existing Work Mapping II

5. Evolution of Network Telemetry

As the network is evolving towards the automated operation, network telemetry also undergoes several levels of evolution.

Level 0 - Static Telemetry: The telemetry data is determined at design time. The network operator can only configure how to use it with limited flexibility.

Level 1 - Dynamic Telemetry: The telemetry data can be dynamically programmed or configured at runtime, allowing a tradeoff among resource, performance, flexibility, and coverage. DNP is an effort towards this direction.

Level 2 - Interactive Telemetry: The network operator can continuously customize the telemetry data in real time to reflect the network operation's visibility requirements. At this level, some tasks can be automated, although ultimately human operators will still need to sit in the middle to make decisions.

Level 3 - Closed-loop Telemetry: Human operators are completely excluded from the control loop. The intelligent network operation engine automatically issues the telemetry data request, analyzes the data, and updates the network operations in closed control loops.

While most of the existing technologies belong to level 0 and level 1, with the help of a clearly defined network telemetry framework, we can assemble the technologies to support level 2 and make solid steps towards level 3.

6. Security Considerations

Given that this document has proposed a framework for network telemetry and the telemetry mechanisms discussed are distinct (in both message frequency and traffic amount) from the conventional network OAM concepts, we must also reflect that various new security considerations may also arise. A number of techniques already exist for securing the data plane, control plane, and the management plane in a network, but it is important to consider if any new threat vectors are now being enabled via the use of network telemetry procedures and mechanisms.

Security considerations for networks that use telemetry methods may include:

- o Telemetry framework trust and policy model;
- o Role management and access control for enabling and disabling telemetry capabilities;
- o Protocol transport used telemetry data and inherent security capabilities;

- o Telemetry data stores, storage encryption and methods of access;
- o Tracking telemetry events and any abnormalities that might identify malicious attacks using telemetry interfaces.

Some of the security considerations highlighted above may be minimized or negated with policy management of network telemetry. In a network telemetry deployment it would be advantageous to separate telemetry capabilities into different classes of policies, i.e., Role Based Access Control and Event-Condition-Action policies. Also, potential conflicts between network telemetry mechanisms must be detected accurately and resolved quickly to avoid unnecessary network telemetry traffic propagation escalating into an unintended or intended denial of service attack.

Further discussion and development of this section will be required, and it is expected that this security section, and subsequent policy section will be developed further.

7. IANA Considerations

This document includes no request to IANA.

8. Contributors

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

- o Tianran Zhou
- o Zhenbin Li
- o Daniel King

9. Acknowledgments

We would like to thank Adrian Farrel, Randy Presuhn, Victor Liu, James Guichard, Uri Blumenthal, Giuseppe Fioccola, Yunan Gu, Parviz Yegani, Young Lee, Alexander Clemm, Joe Clarke, and many others who have provided helpful comments and suggestions to improve this document.

10. References

10.1. Normative References

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

10.2. Informative References

- [I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., 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.fioccola-ippm-multipoint-alt-mark]
Fioccola, G., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate Marking method for passive and hybrid performance monitoring", draft-fioccola-ippm-multipoint-alt-mark-04 (work in progress), June 2018.
- [I-D.ietf-grow-bmp-adj-rib-out]
Evens, T., Bayraktar, S., Lucente, P., Mi, K., and S. Zhuang, "Support for Adj-RIB-Out in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-adj-rib-out-03 (work in progress), December 2018.
- [I-D.ietf-grow-bmp-local-rib]
Evens, T., Bayraktar, S., Bhardwaj, M., and P. Lucente, "Support for Local RIB in BGP Monitoring Protocol (BMP)", draft-ietf-grow-bmp-local-rib-02 (work in progress), September 2018.
- [I-D.ietf-netconf-udp-pub-channel]
Zheng, G., Zhou, T., and A. Clemm, "UDP based Publication Channel for Streaming Telemetry", draft-ietf-netconf-udp-pub-channel-04 (work in progress), October 2018.
- [I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-22 (work in progress), February 2019.

- [I-D.kumar-rtgwg-grpc-protocol]
Kumar, A., Kolhe, J., Ghemawat, S., and L. Ryan, "gRPC Protocol", draft-kumar-rtgwg-grpc-protocol-00 (work in progress), July 2016.
- [I-D.openconfig-rtgwg-gnmi-spec]
Shakir, R., Shaikh, A., Borman, P., Hines, M., Lebsack, C., and C. Morrow, "gRPC Network Management Interface (gNMI)", draft-openconfig-rtgwg-gnmi-spec-01 (work in progress), March 2018.
- [I-D.pedro-nmrg-anticipated-adaptation]
Martinez-Julia, P., "Exploiting External Event Detectors to Anticipate Resource Requirements for the Elastic Adaptation of SDN/NFV Systems", draft-pedro-nmrg-anticipated-adaptation-02 (work in progress), June 2018.
- [I-D.song-ippm-postcard-based-telemetry]
Song, H., Zhou, T., Li, Z., and J. Shin, "Postcard-based In-band Flow Data Telemetry", draft-song-ippm-postcard-based-telemetry-01 (work in progress), December 2018.
- [I-D.song-opsawg-dnp4iq]
Song, H. and J. Gong, "Requirements for Interactive Query with Dynamic Network Probes", draft-song-opsawg-dnp4iq-01 (work in progress), June 2017.
- [I-D.zhou-netconf-multi-stream-originators]
Zhou, T., Zheng, G., Voit, E., Clemm, A., and A. Bierman, "Subscription to Multiple Stream Originators", draft-zhou-netconf-multi-stream-originators-03 (work in progress), October 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>>.
- [RFC2981] Kavasseri, R., Ed., "Event MIB", RFC 2981, DOI 10.17487/RFC2981, October 2000, <<https://www.rfc-editor.org/info/rfc2981>>.
- [RFC3416] Presuhn, R., Ed., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, DOI 10.17487/RFC3416, December 2002, <<https://www.rfc-editor.org/info/rfc3416>>.

- [RFC3877] Chisholm, S. and D. Romascanu, "Alarm Management Information Base (MIB)", RFC 3877, DOI 10.17487/RFC3877, September 2004, <<https://www.rfc-editor.org/info/rfc3877>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <<https://www.rfc-editor.org/info/rfc7854>>.

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

Appendix A. A Survey on Existing Network Telemetry Techniques

We provide an overview of the challenges and existing solutions for each network telemetry module.

A.1. Management Plane Telemetry

A.1.1. Requirements and Challenges

The management plane of the network element interacts with the Network Management System (NMS), and provides information such as performance data, network logging data, network warning and defects data, and network statistics and state data. Some legacy protocols are widely used for the management plane, such as SNMP and Syslog. However, these protocols are insufficient to meet the requirements of the automatic network operation applications.

New management plane telemetry protocols should consider the following requirements:

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

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

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

A.1.2. Push Extensions for NETCONF

NETCONF [RFC6241] is one popular network management protocol, which is also recommended by IETF. Although it can be used for data collection, NETCONF is good at configurations. YANG Push

[I-D.ietf-netconf-yang-push] extends NETCONF and enables subscriber applications to request a continuous, customized stream of updates from a YANG datastore. Providing such visibility into changes made upon YANG configuration and operational objects enables new capabilities based on the remote mirroring of configuration and operational state. Moreover, distributed data collection mechanism [I-D.zhou-netconf-multi-stream-originators] via UDP based publication channel [I-D.ietf-netconf-udp-pub-channel] provides enhanced efficiency for the NETCONF based telemetry.

A.1.3. gRPC Network Management Interface

gRPC Network Management Interface (gNMI)

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

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

A.2. Control Plane Telemetry

A.2.1. Requirements and Challenges

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

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

protocol layer (e.g., the Transport Layer or the Network Layer), the responsible protocol (e.g., ISIS or BGP at the Network Layer), and finally the responsible device(s) with specific reasons.

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

A.2.2. BGP Monitoring Protocol

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

The BGP routing information is collected from the monitored device(s) to the BMP monitoring station by setting up the BMP TCP session. The BGP peers are monitored by the BMP Peer Up and Peer Down Notifications. The BGP routes (including Adjacency_RIB_In [RFC7854], Adjacency_RIB_out [I-D.ietf-grow-bmp-adj-rib-out], and Local_Rib [I-D.ietf-grow-bmp-local-rib] are encapsulated in the BMP Route Monitoring Message and the BMP Route Mirroring Message, in the form of both initial table dump and real-time route update. In addition, BGP statistics are reported through the BMP Stats Report Message, which could be either timer triggered or event-driven. More BMP extensions can be explored to enrich the applications of BGP monitoring.

A.3. Data Plane Telemetry

A.3.1. Requirements and Challenges

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

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

The industry has agreed that the data plane programmability is essential to support network telemetry. Newer data plane chips are all equipped with advanced telemetry features and provide flexibility to support customized telemetry functions.

A.3.2. Technique Taxonomy

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

Active and Passive: The active and passive methods (as well as the hybrid types) are well documented in [RFC7799]. The passive methods include TCPDUMP, IPFIX [RFC7011], sflow, and traffic mirror. These methods usually have low data coverage. The bandwidth cost is very high in order to improve the data coverage. On the other hand, the active methods include Ping, Traceroute, OWAMP [RFC4656], and TWAMP [RFC5357]. These methods are intrusive and only provide indirect network measurement results. The hybrid methods, including in-situ OAM

[I-D.brockners-inband-oam-requirements], IPFPM [RFC8321], and Multipoint Alternate Marking [I-D.fioccola-ippm-multipoint-alt-mark], provide a well-balanced and more flexible approach. However, these methods are also more complex to implement.

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

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

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

A.3.3. The IPFPM technology

The Alternate Marking method is efficient to perform packet loss, delay, and jitter measurements both in an IP and Overlay Networks, as presented in IPFPM [RFC8321] and [I-D.fioccola-ippm-multipoint-alt-mark].

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

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

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

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

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

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

In summary, an application can configure end-to-end network monitoring. If the network does not experiment issues, this approximate monitoring is good enough and is very cheap in terms of network resources. However, in case of problems, the application becomes aware of the issues from this approximate monitoring and, in order to localize the portion of the network that has issues, configures the measurement points more exhaustively. So a new detailed monitoring is performed. After the detection and resolution of the problem the initial approximate monitoring can be used again.

A.3.4. Dynamic Network Probe

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

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

A.3.5. IP Flow Information Export (IPFIX) protocol

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

A.3.6. In-Situ OAM

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

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

A.3.7. Postcard Based Telemetry

PBT [I-D.song-ippm-postcard-based-telemetry] is an alternative to IOAM. PBT directly exports data at each node through an independent packet. PBT solves several issues of IOAM. It can also help to identify packet drop location in case a packet is dropped on its forwarding path.

A.4. External Data and Event Telemetry

Events that occur outside the boundaries of the network system are another important source of telemetry information. Correlating both internal telemetry data and external events with the requirements of

network systems, as presented in Exploiting External Event Detectors to Anticipate Resource Requirements for the Elastic Adaptation of SDN/NFV Systems [I-D.pedro-nmrg-anticipated-adaptation], provides a strategic and functional advantage to management operations.

A.4.1. Requirements and Challenges

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

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

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

A.4.2. Sources of External Events

To ensure that the information provided by external event detectors and used by the network management solutions is meaningful for the management purposes, the network telemetry framework must ensure that such detectors (sources) are easily connected to the management solutions (sinks). This requires the specification of a simple taxonomy of detectors and match it to the connectors and/or interfaces required to connect them.

Once detectors are classified in such taxonomy, their definitions are enlarged with the qualities and other aspects used to handle them and represented in the ontology and information model (e.g. YANG). Therefore, differentiating several types of detectors as potential sources of external events is essential for the integrity of the management framework. We thus differentiate the following source types of external events:

- o Smart objects and sensors. With the consolidation of the Internet of Things~(IoT) any network system will have many smart objects attached to its physical surroundings and logical operation environments. Most of these objects will be essentially based on sensors of many kinds (e.g. temperature, humidity, presence) and the information they provide can be very useful for the management of the network, even when they are not specifically deployed for such purpose. Elements of this source type will usually provide a specific protocol for interaction, especially one of those protocols related to IoT, such as the Constrained Application Protocol (CoAP). It will be used by the telemetry framework to interact with the relevant objects.
- o Online news reporters. Several online news services have the ability to provide enormous quantity of information about different events occurring in the world. Some of those events can impact on the network system managed by a specific framework and, therefore, it will be interested on getting such information. For instance, diverse security reports, such as the Common Vulnerabilities and Exposures (CVE), can be issued by the corresponding authority and used by the management solution to update the managed system if needed. Instead of a specific protocol and data format, the sources of this kind of information usually follow a relaxed but structured format. This format will be part of both the ontology and information model of the telemetry framework.
- o Global event analyzers. The advance of Big Data analyzers provides a huge amount of information and, more interestingly, the identification of events detected by analyzing many data streams from different origins. In contrast with the other types of sources, which are focused in specific events, the detectors of this source type will detect very generic events. For example, a sports event takes place and some unexpected movement makes it highly interesting and many people connects to sites that are covering such event. The systems supporting the services that cover the event can be affected by such situation so their management solutions should be aware of it. In contrast with the other source types, a new information model, format, and reporting

protocol is required to integrate the detectors of this type with the management solution.

Additional types of detector types can be added to the system but they will be generally the result of composing the properties offered by these main classes. In any case, future revisions of the network telemetry framework will include the required types that cover new circumstances and that cannot be obtained by composition.

A.4.3. Connectors and Interfaces

For allowing external event detectors to be properly integrated with other management solutions, both elements must expose interfaces and protocols that are subject to their particular objective. Since external event detectors will be focused on providing their information to their main consumers, which generally will not be limited to the network management solutions, the framework must include the definition of the required connectors for ensuring the interconnection between detectors (sources) and their consumers within the management systems (sinks) are effective.

In some situations, the interconnection between the external event detectors and the management system is via the management plane. For those situations there will be a special connector that provides the typical interfaces found in most other elements connected to the management plane. For instance, the interfaces will accomplish with a specific information model (YANG) and specific telemetry protocol, such as NETCONF, SNMP, or gRPC.

Authors' Addresses

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

Email: haoyu.song@huawei.com

Zhenqiang Li
China Mobile
No. 32 Xuanwumenxi Ave., Xicheng District
Beijing, 100032
P.R. China

Email: lizhenqiang@chinamobile.com

Pedro Martinez-Julia
NICT
4-2-1, Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: pedro@nict.go.jp

Laurent Ciavaglia
Nokia
Villardeaux 91460
France

Email: laurent.ciavaglia@nokia.com

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, 102209
P.R. China

Email: wangaj.bri@chinatelecom.cn

Operations and Management Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2020

Q. Sun
H. Xu
China Telecom
B. Wu, Ed.
Q. Wu, Ed.
Huawei
C. Eckel, Ed.
Cisco Systems
July 3, 2019

A YANG Data Model for SD-WAN Service Delivery
draft-sun-opsawg-sdwan-service-model-04

Abstract

This document provides a YANG data model for an SD-WAN service. An SD-WAN service is a connectivity service offered by a service provider network to provide connectivity across different locations of a customer network or between a customer network and an external network, such as the Internet or a private/public cloud network. This connectivity is provided as an overlay constructed using one of more underlay networks. The model can be used by a service orchestrator of a service provider to request, configure, and manage the components of an SD-WAN service.

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 4, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
1.1. Terminology	3
1.2. Definitions	3
2. High Level Overview of SD-WAN Service	4
3. Service Data Model Usage	6
4. Design of the Data Model	7
4.1. SD-WAN connectivity service	8
4.1.1. VPNs	8
4.1.2. Sites	9
4.2. Application based Policy Service	10
5. Modules Tree Structure	12
6. YANG Modules	17
7. Security Considerations	43
8. IANA Considerations	43
9. Appendix 1: Terminology Mapping between MEF SD-WAN Service Attributes and IETF SD-WAN model	44
10. Appendix 2: IETF OSE model vs IETF SD-WAN model	44
11. Acknowledgments	45
12. Contributors	45
13. References	45
13.1. Normative References	45
13.2. Informative References	46
Authors' Addresses	47

1. Introduction

An SD-WAN service is a connectivity service offered by a service provider network to provide connectivity across different locations of a customer network or between a customer network and an external network. Compared to a conventional PE-based connectivity service as defined in Layer 3 VPN Service Model [RFC8299] and Layer 2 VPN Service Model [RFC8466], an SD-WAN service is a CE-based connectivity service that uses the Internet or PE-based connectivity services as underlay connectivity services. More specially, an SD-WAN service is an overlay connectivity service that provides the flexibility of

adding, removing, or moving services without needing to change the underlay networks.

Besides being an overlay service, an SD-WAN Service has the following characteristics:

- o Hybrid WAN access: The CE could connect to a variety of Internet access technologies, including fiber, cable, DSL-based, WiFi, or 4G/Long Term Evolution (LTE), which implies wider reachability and shorter provisioning cycles. It can also use private VPN connectivity services defined in [RFC4364] and [RFC4664], or Operator Ethernet Services, as defined in [MEF51.1], to take advantage of better performance.
- o Application based traffic forwarding: There are diverse applications used in enterprises, such as VoIP calling, video conferencing, streaming media, etc. Application traffic across the WAN will be forwarded based on business priorities, SLA requirements, or other enterprise requirements.
- o Centralized service management: Subscribers of the service need to be provided a single point (such as a web portal) from which to dynamically add or modify services, such as configuring application policies, adding new sites, or adding new underlay connectivity services.

This draft specifies the SD-WAN service YANG model which is modelled from a customer perspective. The model parameters can be used as an input to automated control and configuration applications to manage SD-WAN services.

1.1. Terminology

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

1.2. Definitions

CE Device: Customer Edge Device , as per Provider Provisioned VPN Terminology [RFC4026] .

CE-based VPN: Refers to Provider Provisioned VPN Terminology [RFC4026]

PE Device: Provider Edge Device, as per Provider Provisioned VPN Terminology [RFC4026]

PE-Based VPNs: Refers to Provider Provisioned VPN Terminology [RFC4026]

SD-WAN: An automated, programmatic approach to managing enterprise network connectivity and circuit usage. It extends software-defined networking (SDN) into an application that businesses can use to quickly create a hybrid WAN, which comprises business-grade IP VPN, broadband Internet, and wireless services or multiple WANs of the same or different types. SD-WAN is also deemed as extended CE-based VPN.

SD-WAN Controller: Refers to the abstract entity that combines Control Plane (CP) and Management Plane (MP) defined in SDN: Layers and Architecture Terminology [RFC7426], to configure, manage and control the CEs and other corresponding SD-WAN components.

Underlay network: A network that provides connectivity across SD-WAN sites and over which customer network packets are tunnelled. An underlay network does not need to be aware that it is carrying overlay customer network packets. Addresses on an underlay network appear as "outer addresses" in encapsulated overlay packets. In general, an underlay network can use a completely different protocol (and address family) from that of the overlay network.

Overlay network: A virtual network in which the separation of customer networks is hidden from the underlying physical infrastructure. That is, the underlying transport networks do not need to know about customer separation to correctly forward traffic. IPsec tunnels [RFC6071] are an example of an L3 overlay network.

2. High Level Overview of SD-WAN Service

From a customer perspective, an example of SD-WAN service network is shown in figure 1.

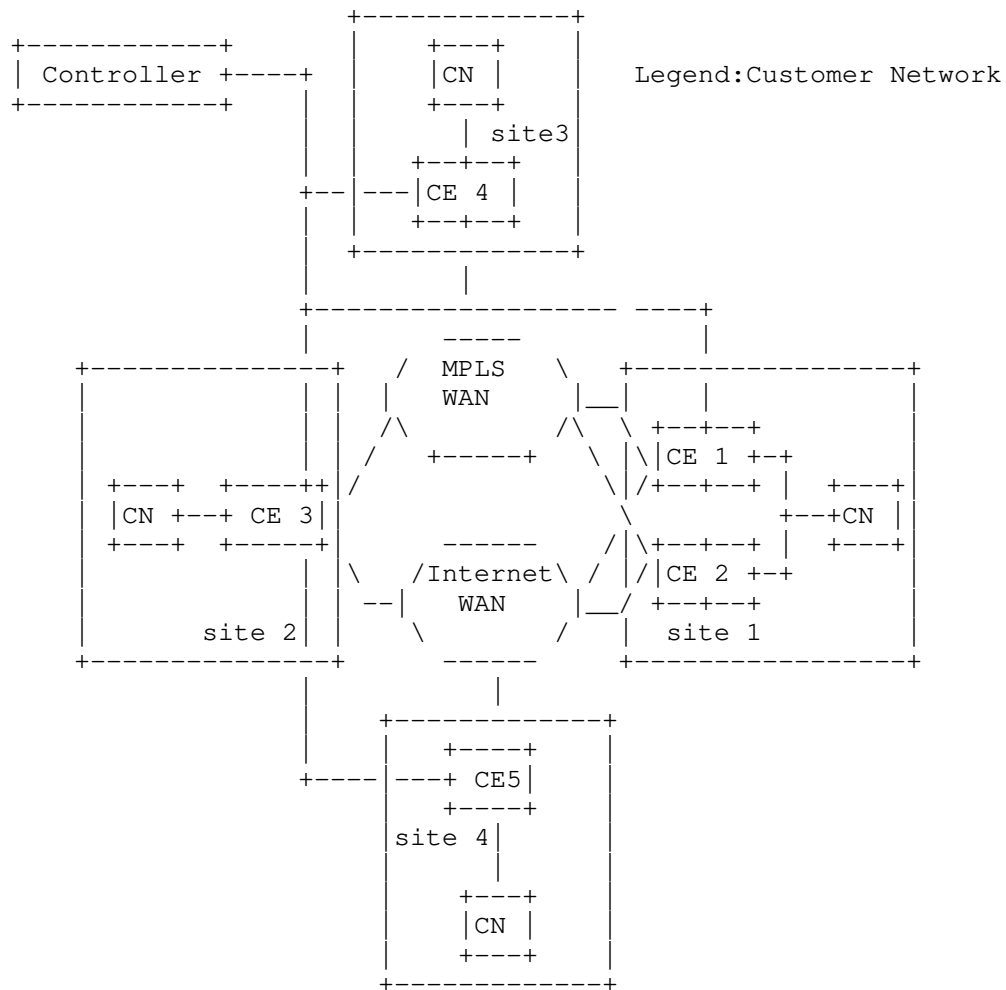


figure 1 SD-WAN network example

As shown in figure 1, the SD-WAN network consists of a number of sites, which are connected through Internet or MPLS VPN.

Within each site, a CE is connected with customer's network on one side, and is also connected to Internet, or to private WAN, or to both on the other side. The customer network could be an L2 or L3 network. For the WAN side, Internet provides ubiquitous IP connectivity via access network like Broadband access or LTE access, while MPLS WAN, like conventional VPN, provides secure and committed connectivity. The boundary between the customer and the service provider is between customer node and the CE device.

Additionally, a site could deploy one or more CEs to improve availability.

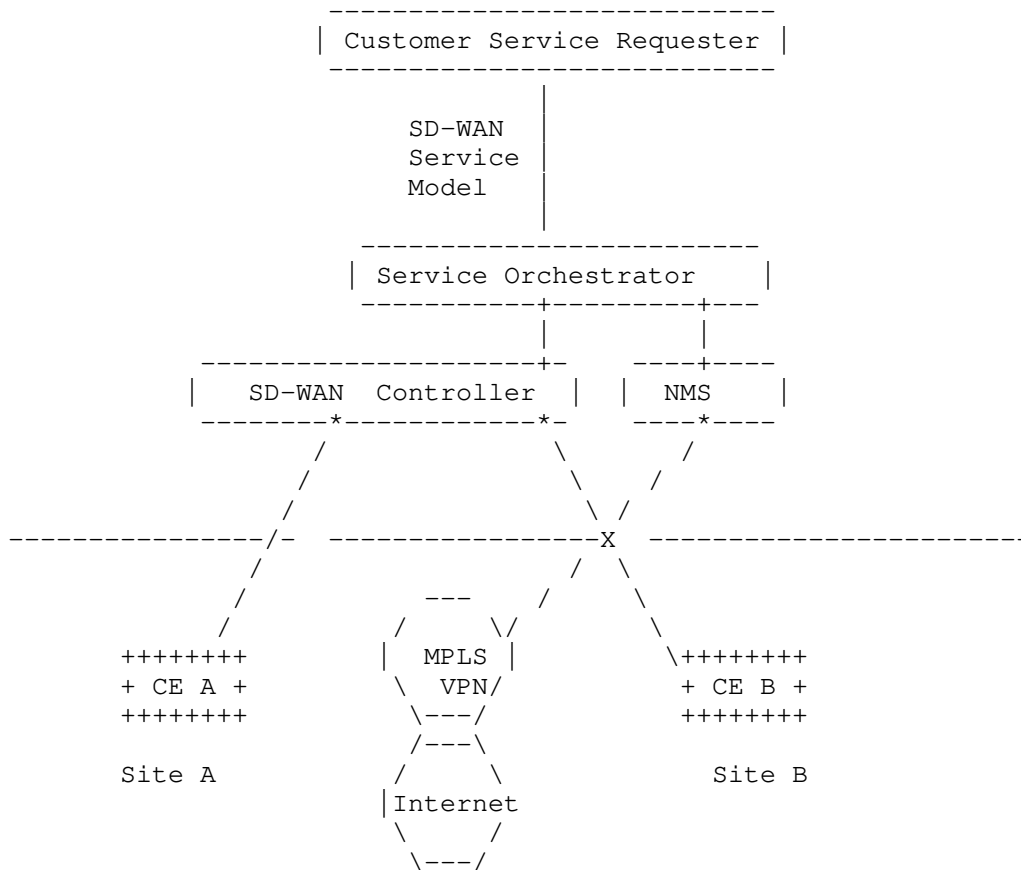
The controller is a centralized entity that manages all the CEs involved in the SD-WAN. The controller could provide bootstrapping of the CEs, ongoing CE configuration, and establishment of secured tunnels between CEs to support the SD-WAN service and application policy enforcement. Various IP tunnelling options (e.g., GRE [RFC2784] and IPSec [RFC6071]), could be used depending on whether traffic from the site is across underlying private VPN or public Internet, and the specific definition is out of scope of this document.

Besides basic connectivity between the sites, the SD-WAN service could be extended by providing direct Internet connectivity, cloud network connectivity, or conventional MPLS VPN interoperability.

3. Service Data Model Usage

The SD-WAN service model provides an abstracted interface to request, configure, and manage the components of an SD-WAN service.

A typical usage for this model is as an input to a service orchestrator that is responsible for service management. Based on the user's service request, the service orchestrator can instruct the SD-WAN controller to add a new site, VPN or application policy in real-time. The orchestrator could orchestrate the other network, such as legacy MPLS VPN network to interconnect with SD-WAN network where Layer 2 VPN Service Mode [RFC8466] or Layer 3 VPN Service Model [RFC8299] could be used.



Reference Architecture for the Use of SD-WAN Service Model Usage

For an SD-WAN to be established under the SP's control, the customer informs the Service Provider of which sites should become part of the requested service and what types of policy will provide. And then the SP configures and updates the service base on the service model and the available resources derived from the SD-WAN controller, and then provisions and manages the customer's service through the SD-WAN controller. How the SD-WAN controller to control and manage the CEs is out of scope of the document.

4. Design of the Data Model

An SD-WAN service consist of two service components:

1. SD-WAN connectivity service

2. SD-WAN application policy service

4.1. SD-WAN connectivity service

SD-WAN connectivity service is the basic component of the SD-WAN service that represents a virtual connection between two or more customer sites. In this model, each virtual connection is defined as a VPN. Each customer can have one or more VPNs, and each VPN can be established between a subset of sites. The association of sites and VPNs is modelled by VPN endpoints.

4.1.1. VPNs

The "sdwan-vpn" list item contains service parameters that apply to an SD-WAN VPN. These parameters are specified as follows:

- o The "vpn-id" leaf is under the vpn-service list, and provides a unique ID for a VPN.
- o The "endpoints" list is under the vpn-service list. Each "endpoint" is a logical point associated with a site. The two main functions of the endpoint are the association of a VPN with a site and per site application based policy enforcement.
- o The "topology" leaf is under the vpn-service list, which refers to a specific topology of the VPN service. Different VPN connection topology can be used. For a VPN with a few sites, simple topologies such as hub-and-spoke or full-mesh can be used. For a large VPN, a hierarchical topology may be taken.
- o The "performance-objectives" container specifies the performance-related properties of an SD-WAN VPN that can be measured. System uptime is the only performance objective defined currently. It indicates the proportion of time, during a given time period that the service is working from the customer perspective. Three parameters are defined, including the start time of the evaluation, the time interval of the evaluation, and the service uptime defined by a percentage.
- o The "reserved-prefixes" container specifies the IP Prefixes that need to be reserved for Service Provider management purposes, such as diagnostics, so as to ensure they are not overlapping with IP Prefixes used by the customer network.

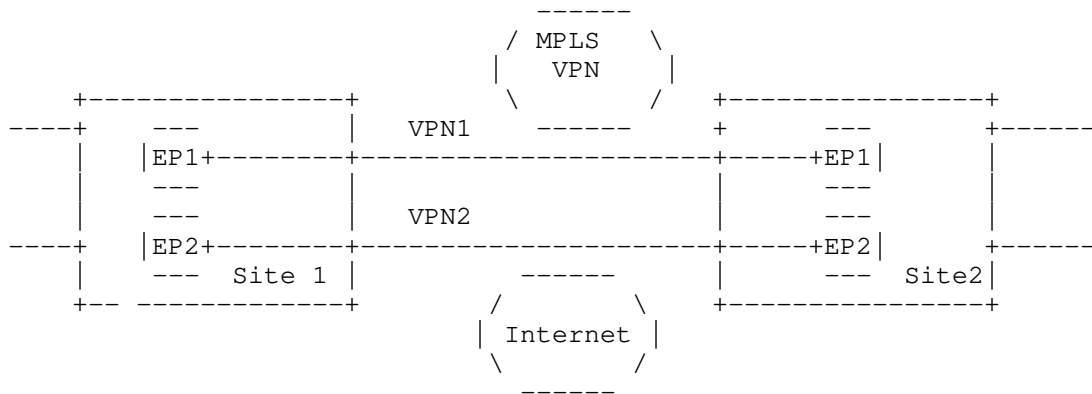


figure 3 SD-WAN VPN example

4.1.2. Sites

A site represents a customer office located at a specific geographic location. The "sites" container specifies the following parameters:

- o "site-id: uniquely identifies the site within the overall network infrastructure.
- o "device" specifies the device type (physical or virtual device) and the number of the devices.
- o "lan-accesses": Specifies the customer network access link parameters. A "site" is composed of at least one "lan-access" where one or more subnets can reside. The "lan-access" consists of the following categories of parameters:
 - * "bearer": defines requirements of the attachment (below Layer 3), bearer type including Ethernet, etc.
 - * IP Connection: defines Layer 3 parameters of the attachment, including IPv4 connection parameters and IPv6 connection parameters.
- o "wan-accesses": Specifies the WAN access link parameters. A "site" is composed of at least one "wan-access". The WAN access can be further specified by access type, service provider name, and bandwidth of the WAN connectivity. The "wan-access" consists of the following categories of parameters:
 - * "access-type": specifies whether the access is Broadband Internet, Wireless Internet or private circuit.

- * "access-provider": specifies the service provider name.
- * bandwidth: specifies the WAN link bandwidth including input and output bandwidth.
- * "bearer": defines requirements of the attachment (below Layer 3), bearer type including Ethernet, etc.
- * IP Connection: defines Layer 3 parameters of the attachment, including IPv4 connection parameters and IPv6 connection parameters.

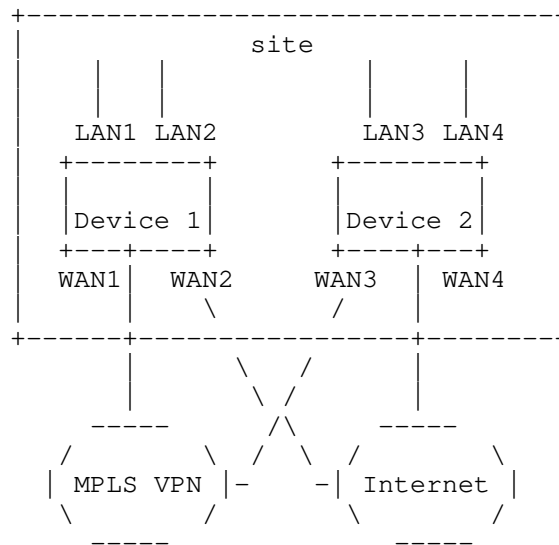


figure 4 Site example

4.2. Application based Policy Service

The connectivity service establishes a virtual connection for the enterprise network, and the Application based Policy Service is designed to ensure business-critical and real-time application experience while also ensuring the security and corporate policies.

Typically, application policies common to each VPN can be defined and then enforced when traffic from a customer's network at a particular site is sent over the WAN.

The application policy assignment is defined under the VPN endpoint container to specify the mapping of application flow name or application group name and their associated policy list names. If an

application flow and the application flow group in which the Application Flow is a member are both assigned a policy at an VPN End Point, the policy assigned to the application flow will supersede the group policy.

The application policy per VPN consists of three lists under the VPN container:

- o application flow list: Describes the characteristics of an enterprise application and is used to identify applications, e.g., based on layer 3 source and destination addresses, layer 4 ports, layer 4 protocol, etc.
- o application group list: Describes application flow aggregation, which is used to deliver aggregation policies, such as bandwidth restrictions for a group of applications.
- o policy list: Defines the application's policy set. Since SD-WAN has more than one WAN connectivity and various encrypted or unencrypted overlay tunnels, there could be multiple tunnel or link selection combination. In this model, different path selection policies are combined to meet different needs based on application SLA, security, cost, and so on. For example, when different applications in a branch need to pass over the WAN, according to the application-aware policy requirements and the IP forwarding table, the Internet application or the SaaS application can be accessed through the Internet, and the data center FTP application can use the Internet encrypted tunnel as the primary path, and the tunnel could only be over broadband Internet instead of wireless internet. This policy combination is not an exhaustive list and could be augmented according to business needs.

An example of a classification of application flows is as follows:

The HTTP traffic from the 192.0.2.0/24 LAN destined for port 80 will be classified in app-id 1.

The FTP traffic from the 192.0.2.0/24 LAN destined for 203.0.113.1/32 will be classified in app-id 2.

An example of a policy list is as follows:

```
"policy": [  
  {  
    "policy-id": "pol-a",  
    "policy-package":  
      {  
        "encryption": "false",  
        "internet-breakout": "true"  
        "public-private": "public",  
        "billing-method": "flat-only"  
        "backup": "false",  
        "bandwidth": "20", "50"  
      }  
    },  
  {  
    "policy-id": "pol-b",  
    "policy-package":  
      {  
        "encryption": "true",  
        "internet-breakout": "false"  
        "public-private": "public",  
        "billing-method": "flat-only"  
        "backup": "false",  
        "bandwidth": "50", "none"  
      }  
    }  
  ]
```

An example of an application policy list is as follows:

```
"app-policy": [  
  {  
    "app-id": "1"  
    "policy-id": "pol-a",  
  },  
  {  
    "app-id": "1"  
    "policy-id": "pol-b",  
  }  
]
```

5. Modules Tree Structure

This document defines an SD-WAN service YANG data model.

```
module: ietf-sdwan-svc  
  +--rw sdwan-svc  
    +--rw vpn-services  
      | +--rw vpn-service* [vpn-id]
```

```

+--rw vpn-id                svc-id
+--rw topology?             identityref
+--rw performance-objective
|   +--rw start-time?        yang:date-and-time
|   +--rw duration?          string
|   +--rw uptime-objective
|       +--rw duration?      decimal64
+--rw reserved-prefixes
|   +--rw prefix*            inet:ip-prefix
+--rw application* [app-id]
|   +--rw app-id             svc-id
|   +--rw ac* [name]
|       +--rw name                                string
|       +--rw (match-type)?
|           +--:(match-flow)
|               +--rw match-flow
|                   +--rw ethertype?                uint16
|                   +--rw cvlan?                    uint8
|                   +--rw ipv4-src-prefix?           inet:ipv4-prefix
|                   +--rw ipv4-dst-prefix?           inet:ipv4-prefix
|                   +--rw l4-src-port?               inet:port-number
|                   +--rw l4-dst-port?               inet:port-number
|                   +--rw ipv6-src-prefix?           inet:ipv6-prefix
|                   +--rw ipv6-dst-prefix?           inet:ipv6-prefix
|                   +--rw protocol-field?            union
|           +--:(match-application)
|               +--rw match-application?            identityref
+--rw application-group* [app-group-id]
|   +--rw app-group-id       svc-id
|   +--rw app-id*            -> ../../application/app-id
+--rw policy* [policy-id]
|   +--rw policy-id          svc-id
|   +--rw policy-package
|       +--rw encryption?    enumeration
|       +--rw public-private? enumeration
|       +--rw local-breakout? boolean
|       +--rw billing-method? enumeration
|       +--rw backup-path?   enumeration
|       +--rw bandwidth
|           +--rw commit?    uint32
|           +--rw max?       uint32
+--rw endpoints* [endpoint-id]
|   +--rw endpoint-id        svc-id
|   +--rw site-role?         identityref
|   +--rw site-attachment
|       |   +--rw site-id?    -> /sdwan-svc/sites/site/site-id
+--rw endpoint-policy-map
|   +--rw app-group-policy* [app-group-id]

```

```

|         | +---rw app-group-id    leafref
|         | +---rw policy-id?     leafref
+---rw app-policy* [app-id]
|         | +---rw app-id        leafref
|         | +---rw policy-id?    leafref
+---rw sites
+---rw site* [site-id]
+---rw site-id      svc-id
+---rw device* [name]
|   +---rw name      string
|   +---rw type?     identityref
+---rw lan-access* [name]
|   +---rw name      string
+---rw l2-technology
|   +---rw l2-type?   identityref
+---rw untagged-interface
|   +---rw speed?    uint32
|   +---rw mode?     neg-mode
+---rw tagged-interface
|   +---rw type?     identityref
|   +---rw dot1q-vlan-tagged
|   |   +---rw tg-type?   identityref
|   |   +---rw cvlan-id   uint16
|   +---rw priority-tagged
|   |   +---rw tag-type?   identityref
+---rw l2-mtu?      uint32
+---rw ip-connection
+---rw ipv4
|   +---rw address-allocation-type? identityref
+---rw dhcp
|   +---rw primary-subnet
|   |   +---rw ip-prefix?
|   |   |   inet:ipv4-prefix
|   +---rw default-router?   inet:ip-address
|   +---rw provider-addresses*
|   |   inet:ipv4-address
|   +---rw subscriber-address? inet:ip-address
|   +---rw reserved-ip-prefix* inet:ip-prefix
+---rw secondary-subnet* [ip-prefix]
+---rw ip-prefix
|   inet:ipv4-prefix
+---rw provider-addresses*
|   inet:ipv4-address
+---rw reserved-ip-prefix*
|   inet:ipv4-prefix
+---rw static
+---rw primary-subnet
|   +---rw ip-prefix?

```

```

|         inet:ipv4-prefix
|         +--rw default-router?          inet:ip-address
|         +--rw provider-addresses*
|         |         inet:ipv4-address
|         +--rw subscriber-address?      inet:ip-address
|         +--rw reserved-ip-prefix*      inet:ip-prefix
+--rw secondary-subnet* [ip-prefix]
|         +--rw ip-prefix
|         |         inet:ipv4-prefix
|         +--rw provider-addresses*
|         |         inet:ipv4-address
|         +--rw reserved-ip-prefix*
|         |         inet:ipv4-prefix
+--rw ipv6
|         +--rw address-allocation-type?  identityref
|         +--rw dhcp
|         |         +--rw subnet* [ip-prefix]
|         |         |         +--rw ip-prefix
|         |         |         |         inet:ipv6-prefix
|         |         +--rw provider-addresses*
|         |         |         inet:ipv6-address
|         |         +--rw reserved-ip-prefix*
|         |         |         inet:ipv6-prefix
|         +--rw slaac
|         |         +--rw subnet* [ip-prefix]
|         |         |         +--rw ip-prefix
|         |         |         |         inet:ipv6-prefix
|         |         +--rw provider-addresses*
|         |         |         inet:ipv6-address
|         |         +--rw reserved-ip-prefix*
|         |         |         inet:ipv6-prefix
|         +--rw static
|         |         +--rw subnet* [ip-prefix]
|         |         |         +--rw ip-prefix
|         |         |         |         inet:ipv6-prefix
|         |         +--rw provider-addresses*
|         |         |         inet:ipv6-address
|         |         +--rw reserved-ip-prefix*
|         |         |         inet:ipv6-prefix
|         +--rw subscriber-address?      inet:ipv6-address
+--rw wan-access* [name]
|         +--rw name                      string
|         +--rw access-type?              identityref
|         +--rw access-provider?          string
|         +--rw bandwidth
|         |         +--rw input-bandwidth?  uint64
|         |         +--rw output-bandwidth? uint64
+--rw l2-technology

```

```

+--rw l2-type?                identityref
+--rw untagged-interface
|   +--rw speed?      uint32
|   +--rw mode?       neg-mode
+--rw tagged-interface
|   +--rw type?                identityref
|   +--rw dot1q-vlan-tagged
|   |   +--rw tg-type?        identityref
|   |   +--rw cvlan-id        uint16
|   +--rw priority-tagged
|   |   +--rw tag-type?       identityref
+--rw l2-mtu?                  uint32
+--rw ip-connection
+--rw ipv4
|   +--rw address-allocation-type?  identityref
+--rw dhcp
|   +--rw primary-subnet
|   |   +--rw ip-prefix?
|   |   |   inet:ipv4-prefix
|   |   +--rw default-router?      inet:ip-address
|   |   +--rw provider-addresses*
|   |   |   inet:ipv4-address
|   |   +--rw subscriber-address?   inet:ip-address
|   |   +--rw reserved-ip-prefix*   inet:ip-prefix
+--rw secondary-subnet* [ip-prefix]
|   +--rw ip-prefix
|   |   inet:ipv4-prefix
+--rw provider-addresses*
|   inet:ipv4-address
+--rw reserved-ip-prefix*
|   inet:ipv4-prefix
+--rw static
+--rw primary-subnet
|   +--rw ip-prefix?
|   |   inet:ipv4-prefix
+--rw default-router?      inet:ip-address
+--rw provider-addresses*
|   inet:ipv4-address
+--rw subscriber-address?   inet:ip-address
+--rw reserved-ip-prefix*   inet:ip-prefix
+--rw secondary-subnet* [ip-prefix]
|   +--rw ip-prefix
|   |   inet:ipv4-prefix
+--rw provider-addresses*
|   inet:ipv4-address
+--rw reserved-ip-prefix*
|   inet:ipv4-prefix
+--rw ipv6

```

```

+--rw address-allocation-type?  identityref
+--rw dhcp
|   +--rw subnet* [ip-prefix]
|       +--rw ip-prefix
|           |   inet:ipv6-prefix
|       +--rw provider-addresses*
|           |   inet:ipv6-address
|       +--rw reserved-ip-prefix*
|           |   inet:ipv6-prefix
+--rw slaac
|   +--rw subnet* [ip-prefix]
|       +--rw ip-prefix
|           |   inet:ipv6-prefix
|       +--rw provider-addresses*
|           |   inet:ipv6-address
|       +--rw reserved-ip-prefix*
|           |   inet:ipv6-prefix
+--rw static
|   +--rw subnet* [ip-prefix]
|       +--rw ip-prefix
|           |   inet:ipv6-prefix
|       +--rw provider-addresses*
|           |   inet:ipv6-address
|       +--rw reserved-ip-prefix*
|           |   inet:ipv6-prefix
+--rw subscriber-address?  inet:ipv6-address

```

6. YANG Modules

<CODE BEGINS> file "ietf-sdwan-svc@2019-06-06.yang"

```

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

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

  organization
    "IETF foo Working Group.";
  contact
    "WG List: foo@ietf.org
     Editor:  ";

```

```
description
  "The YANG module defines a generic service configuration
  model for Managed SD-WAN.";

revision 2019-06-06 {
  description
    "Initial revision";
  reference "A YANG Data Model for SD-WAN service.";
}

typedef svc-id {
  type string;
  description
    "Type definition for service identifier";
}

typedef address-family {
  type enumeration {
    enum ipv4 {
      description
        "IPv4 address family.";
    }
    enum ipv6 {
      description
        "IPv6 address family.";
    }
  }
  description
    "Defines a type for the address family.";
}

typedef neg-mode {
  type enumeration {
    enum full-duplex {
      description
        "Defining Full duplex mode";
    }
    enum auto-neg {
      description
        "Defining Auto negotiation mode";
    }
  }
  description
    "Defining a type of the negotiation mode";
}

typedef device-type {
  type enumeration {
```



```
    enum physical {
        description
            "Physical device";
    }
    enum virtual {
        description
            "Virtual device";
    }
}
description
    "Defines device types.";
}

identity device-type {
    description
        "Base identity for device type.";
}

identity virtual-ce {
    base device-type;
    description
        "Identity for virtual-ce.";
}

identity physical-ce {
    base device-type;
    description
        "Identity for physical-ce.";
}

identity customer-application {
    description
        "Base identity for customer application.";
}

identity web {
    base customer-application;
    description
        "Identity for Web application (e.g., HTTP, HTTPS).";
}

identity mail {
    base customer-application;
    description
        "Identity for mail application.";
}

identity file-transfer {
```

```
    base customer-application;
    description
        "Identity for file transfer application (e.g., FTP, SFTP).";
}

identity database {
    base customer-application;
    description
        "Identity for database application.";
}

identity social {
    base customer-application;
    description
        "Identity for social-network application.";
}

identity games {
    base customer-application;
    description
        "Identity for gaming application.";
}

identity p2p {
    base customer-application;
    description
        "Identity for peer-to-peer application.";
}

identity network-management {
    base customer-application;
    description
        "Identity for management application
        (e.g., Telnet, syslog, SNMP).";
}

identity voice {
    base customer-application;
    description
        "Identity for voice application.";
}

identity video {
    base customer-application;
    description
        "Identity for video conference application.";
}
```

```
identity eth-inf-type {
  description
    "Identity of the Ethernet interface type.";
}

identity tagged {
  base eth-inf-type;
  description
    "Identity of the tagged interface type.";
}

identity untagged {
  base eth-inf-type;
  description
    "Identity of the untagged interface type.";
}

identity lag {
  base eth-inf-type;
  description
    "Identity of the LAG interface type.";
}

identity tag-type {
  description
    "Base identity from which all tag types
    are derived from";
}

identity c-vlan {
  base tag-type;
  description
    "A Customer-VLAN tag, normally using the 0x8100
    Ethertype";
}

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

identity dot1q {
  base tagged-inf-type;
  description
    "Identity for dot1q vlan tagged interface.";
}
```

```
identity priority-tagged {
  base tagged-inf-type;
  description
    "This identity the priority-tagged interface.";
}

identity vpn-topology {
  description
    "Base identity for 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 site-role {
  description
    "Site Role in a VPN topology ";
}

identity any-to-any-role {
  base site-role;
  description
    "Site in an any-to-any IP VPN.";
}

identity hub {
  base site-role;
  description
    "Hub Role in Hub-and-Spoke IP VPN.";
}

identity spoke {
  base site-role;
  description
    "Spoke Role in Hub-and-Spoke IP VPN.";
}

identity access-type {
  description
```

```
    "Access type of a site in a connection to different WAN";
}

identity commodity {
    base access-type;
    description
        "Internet access";
}

identity cellular {
    base access-type;
    description
        "Refers to a subset of 3G/4G/LTE and 5G";
}

identity private {
    base access-type;
    description
        "Refers to private circuits such as Ethernet, T1, etc";
}

identity routing-protocol-type {
    description
        "Base identity for routing protocol type.";
}

identity ospf {
    base routing-protocol-type;
    description
        "Identity for OSPF protocol type.";
}

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

```
identity dhcp {
  base address-allocation-type;
  description
    "Provider network provides DHCP service to customer.";
}

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

identity slaac {
  base address-allocation-type;
  description
    "Use IPv6 SLAAC.";
}

identity ll-only {
  base address-allocation-type;
  description
    "Use IPv6 Link Local.";
}

identity traffic-direction {
  description
    "Base identity for traffic direction";
}

identity inbound {
  base traffic-direction;
  description
    "Identity for inbound";
}

identity outbound {
  base traffic-direction;
  description
    "Identity for outbound";
}

identity both {
  base traffic-direction;
  description
    "Identity for both";
}

identity traffic-action {
```

```
    description
      "Base identity for traffic action";
  }

  identity permit {
    base traffic-action;
    description
      "Identity for permit action";
  }

  identity deny {
    base traffic-action;
    description
      "Identity for deny action";
  }

  identity bd-limit-type {
    description
      "base identity for bd limit type";
  }

  identity percent {
    base bd-limit-type;
    description
      "Identity for percent";
  }

  identity value {
    base bd-limit-type;
    description
      "Identity for value";
  }

  identity protocol-type {
    description
      "Base identity for protocol field type.";
  }

  identity tcp {
    base protocol-type;
    description
      "TCP protocol type.";
  }

  identity udp {
    base protocol-type;
    description
      "UDP protocol type.";
```

```
}

identity icmp {
  base protocol-type;
  description
    "ICMP protocol type.";
}

identity icmp6 {
  base protocol-type;
  description
    "ICMPv6 protocol type.";
}

identity gre {
  base protocol-type;
  description
    "GRE protocol type.";
}

identity ipip {
  base protocol-type;
  description
    "IP-in-IP protocol type.";
}

identity hop-by-hop {
  base protocol-type;
  description
    "Hop-by-Hop IPv6 header type.";
}

identity routing {
  base protocol-type;
  description
    "Routing IPv6 header type.";
}

identity esp {
  base protocol-type;
  description
    "ESP header type.";
}

identity ah {
  base protocol-type;
  description
    "AH header type.";
```



```
}

grouping vpn-endpoint {
  leaf endpoint-id {
    type svc-id;
    description
      "Identity for the vpn endpoint";
  }
  leaf site-role {
    type identityref {
      base site-role;
    }
    default "any-to-any-role";
    description
      "Role of the site in the VPN.";
  }
  container site-attachment {
    leaf site-id {
      type leafref {
        path "/sdwan-svc/sites/site/site-id";
      }
      description
        "Defines site id attached.";
    }
    description
      "Defines site attachment to a vpn endpoint.";
  }
  container endpoint-policy-map {
    list app-group-policy {
      key "app-group-id";
      leaf app-group-id {
        type leafref {
          path "/sdwan-svc/vpn-services/vpn-service"+
            "/application-group/app-group-id";
        }
        description
          "Identity for application";
      }
      leaf policy-id {
        type leafref {
          path "/sdwan-svc/vpn-services/vpn-service/policy/policy-id";
        }
        description
          "Identity for value";
      }
      description
        "list for application group policy";
    }
  }
}
```

```
list app-policy {
  key "app-id";
  leaf app-id {
    type leafref {
      path "/sdwan-svc/vpn-services/vpn-service"+
        "/application/app-id";
    }
    description
      "Identity for application";
  }
  leaf policy-id {
    type leafref {
      path "/sdwan-svc/vpn-services/vpn-service/policy/policy-id";
    }
    description
      "Identity for value";
  }
  description
    "list for application policy";
}
description
  "Identity for policy maps";
}
description
  "grouping for vpn endpoint";
}

grouping flow-definition {
  container match-flow {
    leaf ethertype {
      type uint16;
      description
        "Ethertype value, e.g. 0800 for IPv4.";
    }
    leaf cvlan {
      type uint8 {
        range "0..7";
      }
      description
        "802.1Q matching.";
    }
    leaf ipv4-src-prefix {
      type inet:ipv4-prefix;
      description
        "Match on IPv4 src address.";
    }
    leaf ipv4-dst-prefix {
      type inet:ipv4-prefix;
    }
  }
}
```

```
        description
            "Match on IPv4 dst address.";
    }
    leaf l4-src-port {
        type inet:port-number;
        description
            "Match on Layer 4 src port.";
    }
    leaf l4-dst-port {
        type inet:port-number;
        description
            "Match on Layer 4 dst port.";
    }
    leaf ipv6-src-prefix {
        type inet:ipv6-prefix;
        description
            "Match on IPv6 src address.";
    }
    leaf ipv6-dst-prefix {
        type inet:ipv6-prefix;
        description
            "Match on IPv6 dst address.";
    }
    leaf protocol-field {
        type union {
            type uint8;
            type identityref {
                base protocol-type;
            }
        }
        description
            "Match on IPv4 protocol or IPv6 Next Header field.";
    }
    description
        "Describes flow-matching criteria.";
}
description
    "Grouping for flow definition.";
}

grouping application-criteria {
    list ac {
        key "name";
        ordered-by user;
        leaf name {
            type string;
            description
                "A description identifying application classification
```

```
        criteria.";
    }
    choice match-type {
        default "match-flow";
        case match-flow {
            uses flow-definition;
        }
        case match-application {
            leaf match-application {
                type identityref {
                    base customer-application;
                }
                description
                    "Defines the application to match.";
            }
        }
        description
            "Choice for classification.";
    }
    description
        "List of marking rules.";
}
description
    "This grouping defines QoS parameters for a site.";
}

grouping vpn-service {
    leaf vpn-id {
        type svc-id;
        description
            "Identity for VPN.";
    }
    leaf topology {
        type identityref {
            base vpn-topology;
        }
        description
            "vpn topology: hub-and-spoke or any-to-any";
    }
    container performance-objective {
        leaf start-time {
            type yang:date-and-time;
            description
                "start-time indicates date and time.";
        }
        leaf duration {
            type string;
            description
```

```
        "Time duration.";
    }
    container uptime-objective {
        leaf duration {
            type decimal64 {
                fraction-digits 5;
                range "0..100";
            }
            units "percent";
            description
                "To be used to define the a percentage of the available
                service.";
        }
        description
            "Uptime objective.";
    }
    description
        "The performance objective.";
}
container reserved-prefixes {
    leaf-list prefix {
        type inet:ip-prefix;
        description
            "ip prefix reserved for SP management purpose.";
    }
    description
        "ip prefix list reserved for SP management purpose.";
}
list application {
    key "app-id";
    leaf app-id {
        type svc-id;
        description
            "application name";
    }
    uses application-criteria;
    description
        "list for application";
}
list application-group {
    key "app-group-id";
    leaf app-group-id {
        type svc-id;
        description
            "application name";
    }
    leaf-list app-id {
        type leafref {
```

```
        path "../../application/app-id";
    }
    description
        "application member list in an application group";
}
description
    "list for application group";
}
list policy {
    key "policy-id";
    leaf policy-id {
        type svc-id;
        description
            "Policy names";
    }
    container policy-package {
        leaf encryption {
            type enumeration {
                enum yes {
                    description
                        "Indicates whether or not the application flow requires
                        to send over encrypted overlay tunnel.";
                }
                enum either {
                    description
                        " Either means this policy is not applied";
                }
            }
        }
        description
            "Indicates whether or not the application flow requires
            encryption.";
    }
    leaf public-private {
        type enumeration {
            enum private-only {
                description
                    "The private WAN underlay is specified.";
            }
            enum either {
                description
                    "Both public WAN or private WAN could be used";
            }
        }
        description
            "Indicates whether the Application Flow can traverse
            Public or Private Underlay Connectivity Services
            (or both).Either means this policy is not applied.";
    }
}
```

```
leaf local-breakout {
    type boolean;
    description
        "indicates whether the Application Flow should be
        routed directly to the Internet using Local Internet
        Breakout.It can have values Yes and No.";
}
leaf billing-method {
    type enumeration {
        enum flat-only {
            description
                "Only flat-rate underlay could be used for the
                traffic.";
        }
        enum either {
            description
                "Either flat-rate or usage based underlay could
                be used for the traffic.";
        }
    }
    description
        "billing policy.";
}
leaf backup-path {
    type enumeration {
        enum yes {
            description
                "Only the primary tunnel overlay could be used for
                the traffic.";
        }
        enum no {
            description
                "Either the primary or backup overlay tunnel could be
                used for the traffic.";
        }
    }
    description
        "overlay connection as Primary or both Primary and
        Backup.";
}
container bandwidth {
    leaf commit {
        type uint32;
        description
            "CIR";
    }
    leaf max {
        type uint32;
    }
}
```

```
        description
            "max speed ";
    }
    description
        "Container for the bandwidth policy";
    }
    description
        "Container for policy package";
    }
    description
        "List for policy";
    }
    list endpoints {
        key "endpoint-id";
        uses vpn-endpoint;
        description
            "List of endpoints.";
    }
    description
        "Grouping of vpn service";
    }

    grouping site-l2-technology {
        container l2-technology {
            leaf l2-type {
                type identityref {
                    base eth-inf-type;
                }
                default "untagged";
                description
                    "Defines physical properties of an interface. By default, the
                     Ethernet interface type is set to 'untagged'.";
            }
            container untagged-interface {
                leaf speed {
                    type uint32;
                    units "mbps";
                    default "10";
                    description
                        "Port speed.";
                }
                leaf mode {
                    type neg-mode;
                    default "auto-neg";
                    description
                        "Negotiation mode.";
                }
            }
            description

```



```
        "Container of Untagged Interface Attributes
        configurations.";
    }
    container tagged-interface {
        leaf type {
            type identityref {
                base tagged-inf-type;
            }
            default "dot1q";
            description
                "Tagged interface type. By default,
                the Tagged interface type is dot1q interface. ";
        }
        container dot1q-vlan-tagged {
            leaf tg-type {
                type identityref {
                    base tag-type;
                }
                default "c-vlan";
                description
                    "TAG type.By default, Tag type is Customer-VLAN tag.";
            }
            leaf cvlan-id {
                type uint16;
                mandatory true;
                description
                    "VLAN identifier.";
            }
            description
                "Tagged interface.";
        }
        container priority-tagged {
            leaf tag-type {
                type identityref {
                    base tag-type;
                }
                default "c-vlan";
                description
                    "TAG type.By default, the TAG type is
                    Customer-VLAN tag.";
            }
            description
                "Priority tagged.";
        }
        description
            "Container for tagged Interface.";
    }
    leaf l2-mtu {
```

```
    type uint32;
    units "bytes";
    description
      " L2 Maximum Frame Size MUST be an integer number of bytes
        >= 1522MTU.";
  }
  description
    "Container for l2 technology.";
}
description
  "grouping for l2 technology.";
}

grouping site-ip-connection {
  container ip-connection {
    container ipv4 {
      leaf address-allocation-type {
        type identityref {
          base address-allocation-type;
        }
        description
          "Defines how addresses are allocated.
            If there is no value for address
            allocation type, then the ipv4 is not enabled.";
      }
    }
    container dhcp {
      container primary-subnet {
        leaf ip-prefix {
          type inet:ipv4-prefix;
          description
            "IPv4 address prefix and mask length between 0 and 31,
              in bits.";
        }
      }
      leaf default-router {
        type inet:ip-address;
        description
          "Address of default router.";
      }
    }
    leaf-list provider-addresses {
      type inet:ipv4-address;
      description
        "the Service Provider IPv4 Addresses MUST be within the
          specified IPv4 Prefix.";
    }
    leaf subscriber-address {
      type inet:ip-address;
      description
        "subscriber IPv4 Addresses: Non-empty list

```

```
        of IPv4 addresses";
    }
    leaf-list reserved-ip-prefix {
        type inet:ip-prefix;
        description
            "List of IPv4 Prefixes, possibly empty";
    }
    description
        "Primary Subnet List";
}
list secondary-subnet {
    key "ip-prefix";
    leaf ip-prefix {
        type inet:ipv4-prefix;
        description
            "IPv4 address prefix and mask length between 0 and 31,
            in bits";
    }
    leaf-list provider-addresses {
        type inet:ipv4-address;
        description
            "Service Provider IPv4 Addresses: Non-empty list
            of IPv4 addresses";
    }
    leaf-list reserved-ip-prefix {
        type inet:ipv4-prefix;
        description
            "List of IPv4 Prefixes, possibly empty";
    }
    description
        "Secondary Subnet List";
}
description
    "DHCP allocated addresses related parameters.";
}
container static {
    container primary-subnet {
        leaf ip-prefix {
            type inet:ipv4-prefix;
            description
                "IPv4 address prefix and mask length between 0 and 31,
                in bits.";
        }
    }
    leaf default-router {
        type inet:ip-address;
        description
            "Address of default router.";
    }
}
```

```
    leaf-list provider-addresses {
      type inet:ipv4-address;
      description
        "the Service Provider IPv4 Addresses MUST be within the
        specified IPv4 Prefix.";
    }
    leaf subscriber-address {
      type inet:ip-address;
      description
        "subscriber IPv4 Addresses: Non-empty list
        of IPv4 addresses";
    }
    leaf-list reserved-ip-prefix {
      type inet:ip-prefix;
      description
        "List of IPv4 Prefixes, possibly empty";
    }
    description
      "Primary Subnet List";
  }
  list secondary-subnet {
    key "ip-prefix";
    leaf ip-prefix {
      type inet:ipv4-prefix;
      description
        "IPv4 address prefix and mask length between 0 and 31,
        in bits";
    }
    leaf-list provider-addresses {
      type inet:ipv4-address;
      description
        "Service Provider IPv4 Addresses: Non-empty list
        of IPv4 addresses";
    }
    leaf-list reserved-ip-prefix {
      type inet:ipv4-prefix;
      description
        "List of IPv4 Prefixes, possibly empty";
    }
    description
      "Secondary Subnet List";
  }
  description
    "Static configuration related parameters.";
}
description
  "IPv4-specific parameters.";
}
```

```
container ipv6 {
  leaf address-allocation-type {
    type identityref {
      base address-allocation-type;
    }
    description
      "Defines how addresses are allocated.
      If there is no value for address
      allocation type, then the ipv6 is not enabled.";
  }
  container dhcp {
    list subnet {
      key "ip-prefix";
      leaf ip-prefix {
        type inet:ipv6-prefix;
        description
          "IPv6 address prefix and prefix length between 0 and
          128";
      }
      leaf-list provider-addresses {
        type inet:ipv6-address;
        description
          "Non-empty list of IPv6 addresses";
      }
      leaf-list reserved-ip-prefix {
        type inet:ipv6-prefix;
        description
          "List of IPv6 Prefixes, possibly empty";
      }
      description
        "Subnet List";
    }
    description
      "DHCP allocated addresses related parameters.";
  }
  container slaac {
    list subnet {
      key "ip-prefix";
      leaf ip-prefix {
        type inet:ipv6-prefix;
        description
          "IPv6 address prefix and prefix length of 64 ";
      }
      leaf-list provider-addresses {
        type inet:ipv6-address;
        description
          "Non-empty list of IPv6 addresses";
      }
    }
  }
}
```

```
        leaf-list reserved-ip-prefix {
            type inet:ipv6-prefix;
            description
                "List of IPv6 Prefixes, possibly empty";
        }
        description
            "Subnet List";
    }
    description
        "DHCP allocated addresses related parameters.";
}
container static {
    list subnet {
        key "ip-prefix";
        leaf ip-prefix {
            type inet:ipv6-prefix;
            description
                "IPv6 address prefix and prefix length between 0 and
                128";
        }
        leaf-list provider-addresses {
            type inet:ipv6-address;
            description
                "Non-empty list of IPv6 addresses";
        }
        leaf-list reserved-ip-prefix {
            type inet:ipv6-prefix;
            description
                "List of IPv6 Prefixes, possibly empty";
        }
        description
            "Subnet List";
    }
    leaf subscriber-address {
        type inet:ipv6-address;
        description
            "IPv6 address or Not Specified.";
    }
    description
        "Static configuration related parameters.";
}
description
    "Describes IPv6 addresses used.";
}
description
    "IPv6-specific parameters.";
}
description
```

```
    "This grouping defines IP connection parameters.";
}

container sdwan-svc {
  container vpn-services {
    list vpn-service {
      key "vpn-id";
      uses vpn-service;
      description
        "List for SD-WAN";
    }
    description
      "Container for SD-WAN VPN service";
  }
  container sites {
    list site {
      key "site-id";
      leaf site-id {
        type svc-id;
        description
          "Site Name";
      }
    }
    list device {
      key "name";
      leaf name {
        type string;
        description
          "Device Name";
      }
      leaf type {
        type identityref {
          base device-type;
        }
        description
          "Device Type: virtual or physical CE";
      }
      description
        "List for device";
    }
  }
  list lan-access {
    key "name";
    leaf name {
      type string;
      description
        "lan access link name";
    }
    uses site-l2-technology;
    uses site-ip-connection;
  }
}
```

```
        description
            "container for lan access";
    }
    list wan-access {
        key "name";
        leaf name {
            type string;
            description
                "wan access link name";
        }
        leaf access-type {
            type identityref {
                base access-type;
            }
            description
                "Access type: Internet, private VPN or cellular";
        }
        leaf access-provider {
            type string;
            description
                "Specifies the name of provider";
        }
        container bandwidth {
            leaf input-bandwidth {
                type uint64;
                description
                    "input bandwidth";
            }
            leaf output-bandwidth {
                type uint64;
                description
                    "output bandwidth";
            }
            description
                "Container for bandwidth";
        }
        uses site-l2-technology;
        uses site-ip-connection;
        description
            "container for wan access";
    }
    description
        "List for site";
}
description
    "Container for sites";
}
description
```



```
    "Top-level container for the SD-WAN services.";
  }
}
```

<CODE ENDS>

7. Security Considerations

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

The NETCONF access control model [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.

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.

8. IANA Considerations

IANA has assigned a new URI from the "IETF XML Registry" [RFC3688].

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

IANA has recorded a YANG module name in the "YANG Module Names" registry [RFC6020] as follows:

```
Name: ietf-sdwan-svc
Namespace: urn:ietf:params:xml:ns:yang:ietf-sdwan-svc
Prefix: sdwan-svc
Reference: RFC xxxx
```

9. Appendix 1: Terminology Mapping between MEF SD-WAN Service Attributes and IETF SD-WAN model

SD-WAN Service Attributes and Services [MEF70-Draft-R1], defines the SD-WAN service attributes and services for SD-WAN service delivery. These service attributes can be used for communication between subscribers and services to deliver SD-WAN services while this draft defines a YANG data model for SD-WAN service delivery communicated between customer and service provider. The purpose of both work is very similar.

The below table shows the terminology mapping. The YANG model retains most parameter definition name but adjusts some of the structure to reserve space for future augmentation. For example, the model defines "vpn-service" and "lan-access" as a list, which can accommodate the case where the current MEF service attribute restricts only one VPN per customer and one LAN access and future extension to multiple VPN or LAN accesses per customer.

IETF SD-WAN Service model	MEF70 R1 SD-WAN Services Term
SD-WAN VPN	SD-WAN Virtual Connection (SWVC)
SD-WAN VPN Endpoint	SWVC End Point
Site	User Network Interface (UNI)
lan-access	UNI link Attributes
wan-access	TBD(Underlay connectivity)

10. Appendix 2: IETF OSE model vs IETF SD-WAN model

SD-WAN OSE service delivery model [I-D.wood-rtgwg-sdwan-ose-yang] defines two SD-WAN OSE Open SD-WAN Exchange (OSE) service YANG modules to enable the orchestrator in the enterprise network to implement SD-WAN inter-domain reachability and connectivity services and application aware traffic steering services. Although the OSE YANG model is also a service model instead of being a device model, this model is mainly used for interoperability between multiple SD-WAN domains and service consistency. The differences are shown as follows:

IETF OSE service model	IETF SD-WAN Service model
Domain SD-WAN controller facing	customer-facing
Inter OSE GW connectivity service	unaware of SD-WAN domain in one SP network
Inter SD-WAN domain	Inter-SD-WAN Service Provider TBD
SLA aware dynamic Path selection	static Primary/Backup selection

For the SLA based dynamic path selection policy, the OSE service model uses a similar application classification criteria, but at the same time it will collect the relevant status of the traffic SLA profiles and, based on the measurements calculated from the collected information, the primary or secondary path will be selected.

```

+--primary-backup
  +--rw path-values
    +--rw sla-values
      +--rw latency?          uint32
      +--rw jitter?          uint32
      +--rw packet-loss-rate? uint32

```

11. Acknowledgments

This work has benefited from the discussions of with Jack Pugaczewski, Larry S Samberg, and Pascal Menezes from MEF community.

12. Contributors

The authors would like to thank Zitao Wang for his major contributions to the initial modelling.

13. References

13.1. 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>>.

13.2. Informative References

- [I-D.wood-rtgwg-sdwan-ose-yang]
Wood, S., Bo, W., Wu, Q., and C. Menezes, "YANG Data Model for SD-WAN OSE service delivery", draft-wood-rtgwg-sdwan-ose-yang-00 (work in progress), March 2019.
- [MEF51.1] MEF, Ed., "Operator Ethernet Service Definition", December 2018, <<https://wiki.mef.net/display/CESG/MEF+51.1+-+OVC+Services>>.
- [MEF70-Draft-R1]
MEF, Ed., "SD-WAN Service Attributes and Services", May 2019, <[https://www.mef.net/Assets/Draft-Standards/MEF_70_Draft_\(R1\).pdf](https://www.mef.net/Assets/Draft-Standards/MEF_70_Draft_(R1).pdf)>.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <<https://www.rfc-editor.org/info/rfc2784>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, DOI 10.17487/RFC6071, February 2011, <<https://www.rfc-editor.org/info/rfc6071>>.

- [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>>.
- [RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <<https://www.rfc-editor.org/info/rfc7426>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.

Authors' Addresses

Qiong Sun
China Telecom
Beijing
China

Email: sunqiong.bri@chinatelecom.cn

Honglei Xu
China Telecom
Beijing
China

Email: xuhl.bri@chinatelecom.cn

Bo Wu (editor)
Huawei
Nanjing
China

Email: lana.wubo@huawei.com

Qin Wu (editor)
Huawei
Nanjing
China

Email: bill.wu@huawei.com

Charles Eckel (editor)
Cisco Systems
170 W. Tasman Drive
San Jose, CA
United States

Email: eckelcu@cisco.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 14, 2019

W. Kumari
Google
C. Doyle
Juniper Networks
June 12, 2019

Secure Device Install
draft-wkumari-opsawg-sdi-04

Abstract

Deploying a new network device often requires that an employee physically travel to a datacenter to perform the initial install and configuration, even in shared datacenters with "smart-hands" type support. In many cases, this could be avoided if there were a standard, secure way to initially provision the devices.

This document extends existing auto-install / Zero-Touch Provisioning mechanisms to make the process more secure.

[Ed note: Text inside square brackets ([]) is additional background information, answers to frequently asked questions, general musings, etc. They will be removed before publication. This document is being collaborated on in Github at: <https://github.com/wkumari/draft-wkumari-opsawg-sdi>. The most recent version of the document, open issues, etc should all be available here. The authors (gratefully) accept pull requests.]

[Ed note: This document introduces concepts and serves as the basic for discussion - because of this, it is conversational, and would need to be firmed up before being published]

Status of This Memo

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

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

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

This Internet-Draft will expire on December 14, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Requirements notation	4
2. Overview / Example Scenario	4
3. Vendor Role / Requirements	5
3.1. Device key generation	5
3.2. Certificate Publication Server	5
4. Operator Role / Responsibilities	6
4.1. Administrative	6
4.2. Technical	6
4.3. Initial Customer Boot	7
5. Additional Considerations	9
5.1. Key storage	9
5.2. Key replacement	10
5.3. Device reinstall	10
6. IANA Considerations	10
7. Security Considerations	10
8. Acknowledgements	11
9. References	11
9.1. Normative References	11
9.2. Informative References	11
Appendix A. Changes / Author Notes.	12
Appendix B. Demo / proof of concept	12
B.1. Step 1: Generating the certificate.	13
B.1.1. Step 1.1: Generate the private key.	13
B.1.2. Step 1.2: Generate the certificate signing request.	13
B.1.3. Step 1.3: Generate the (self signed) certificate itself.	13
B.2. Step 2: Generating the encrypted config.	14
B.2.1. Step 2.1: Fetch the certificate.	14

B.2.2. Step 2.2: Encrypt the config file.	14
B.2.3. Step 2.3: Copy config to the config server.	14
B.3. Step 3: Decrypting and using the config.	14
B.3.1. Step 3.1: Fetch encrypted config file from config server.	14
B.3.2. Step 3.2: Decrypt and use the config.	15
Authors' Addresses	15

1. Introduction

In a growing, global network, significant amounts of time and money are spent simply deploying new devices and "forklift" upgrading existing devices. In many cases, these devices are in shared datacenters (for example, Internet Exchange Points (IXP) or "carrier neutral datacenters"), which have staff on hand that can be contracted to perform tasks including physical installs, device reboots, loading initial configurations, etc. There are also a number of (often vendor proprietary) protocols to perform initial device installs and configurations - for example, many network devices will attempt to use DHCP to get an IP address and configuration server, and then fetch and install a configuration when they are first powered on.

Network device configurations contain a significant amount of security related and / or proprietary information (for example, RADIUS or TACACS+ secrets). Exposing these to a third party to load onto a new device (or using an auto-install techniques which fetch an (unencrypted) config file via something like TFTP) is simply not acceptable to many operators, and so they have to send employees to remote locations to perform the initial configuration work. As well as having a significant monetary cost, it also takes significantly longer to install devices and is generally inefficient.

There are some workarounds to this, such as asking the vendor to pre-configure the devices before shipping it; asking the smart-hands to install a terminal server; providing a minimal, unsecured configuration and using that to bootstrap to a complete configuration, etc; but these are often clumsy and have security issues - for example, in the terminal server case, the console port connection could be easily snooped.

This document layers security onto existing auto-install solutions to provide a secure method to initially configure new devices. It is optimized for simplicity, both for the implementor and the operator; it is explicitly not intended to be an "all singing, all dancing" fully featured system for managing installed / deployed devices, nor is it intended to solve all use-cases - rather it is a simple targeted solution to solve a common operational issue. Solutions

such as Secure Zero Touch Provisioning (SZTP)" [RFC8572] are much more fully featured, but also more complex to implement and / or are not widely deployed yet.

1.1. Requirements notation

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

2. Overview / Example Scenario

Sirius Cybernetics Corp needs another peering router, and so they order another router from Acme Network Widgets, to be drop-shipped to the Point of Presence (POP) / datacenter. Acme begins assembling the new device, and tells Sirius what the new device's serial number will be (SN:17894321). When Acme first installs the firmware on the device and boots it, the device generates a public-private keypair, and Acme publishes it on their keyserver (in a certificate, for ease of use).

While the device is being shipped, Sirius generates the initial device configuration, fetches the certificate from Acme keyservers by providing the serial number of the new device. Sirius then encrypts the device configuration and puts this encrypted config on a (local) TFTP server.

When the device arrives at the POP, it gets installed in Sirius' rack, and cabled as instructed. The new device powers up and discovers that it has not yet been configured. It enters its autoboot state, and begins the DHCP process. Sirius' DHCP server provides it with an IP address and the address of the configuration server. The router uses TFTP to fetch its config file (note that all this is existing functionality). The device attempts to load the config file - if the config file is unparsable, (new functionality) the device tries to use its private key to decrypt the file, and, assuming it validates, installs the new configuration.

Only the "correct" device will have the required private key and be able to decrypt and use the config file (See Security Considerations). An attacker would be able to connect to the network and get an IP address. They would also be able to retrieve (encrypted) config files by guessing serial numbers (or perhaps the server would allow directory listing), but without the private keys an attacker will not be able to decrypt the files.

This document uses the serial number of the device as a unique identifier for simplicity; some vendors may not want to implement the

system using the serial number as the identifier for business reasons (a competitor or similar could enumerate the serial numbers and determine how many devices have been manufactured). Implementors are free to choose some other way of generating identifiers (e.g UUID [RFC4122]), but this will likely make it somewhat harder for operators to use (the serial number is usually easy to find on a device, a more complex system is likely harder to track).

[Ed note: This example uses TFTP because that is what many vendors use in their auto-install / ZTP feature. It could easily instead be HTTP, FTP, etc.]

3. Vendor Role / Requirements

This section describes the vendors roles and responsibilities and provides an overview of what the device needs to do.

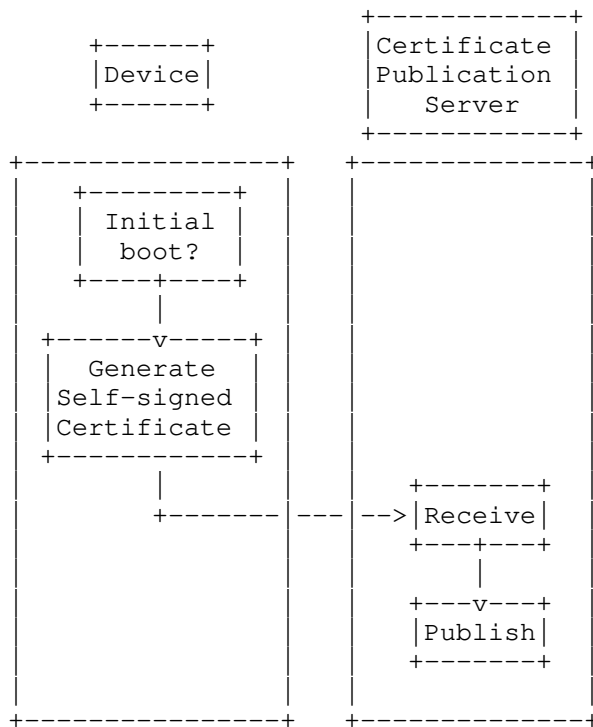
3.1. Device key generation

During the manufacturing stage, when the device is initially powered on, it will generate a public-private keypair. It will send its unique identifier and the public key to the vendor's Certificate Publication Server to be published. The mechanism used to do this is left undefined. Note that some devices may be constrained, and so may send the raw public key and unique identifier to the certificate publication server, while more capable devices may generate and send self-signed certificates.

3.2. Certificate Publication Server

The certificate publication server contains a database of certificates. If newly manufactured devices upload certificates the certificate publication server can simply publish these, if the devices provide raw public keys and unique identifiers the certificate publication server will need to wrap these in a certificate. Note that the certificate publication server MUST only accept certificates or keys from the vendor's manufacturing facilities.

The customers (e.g Sirius Cybernetics Corp) query this server with the serial number (or other provided unique identifier) of a device, and retrieve the associated certificate. It is expected that operators will receive the unique identifier (serial number) of devices when they purchase them, and will download and store / cache the certificate. This means that there is not a hard requirement on the uptime / reachability of the certificate publication server.



Initial certificate generation and publication.

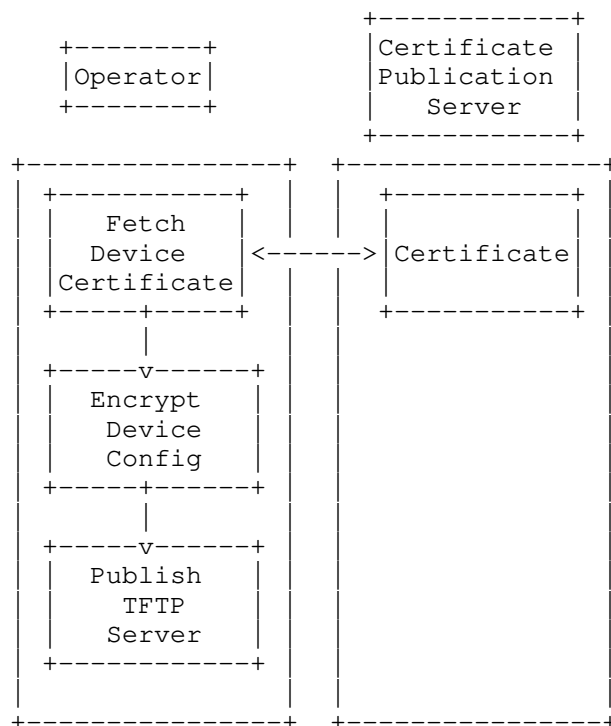
4. Operator Role / Responsibilities

4.1. Administrative

When purchasing a new device, the accounting department will need to get the unique device identifier (likely serial number) of the new device and communicate it to the operations group.

4.2. Technical

The operator will contact the vendor's publication server, and download the certificate (by providing the unique device identifier of the device). The operator **SHOULD** fetch the certificate using a secure transport (e.g HTTPS). The operator will then encrypt the initial configuration to the key in the certificate, and place it on their TFTP server. See Appendix B for examples.



Fetching the certificate, encrypting the configuration, publishing the encrypted configuration.

4.3. Initial Customer Boot

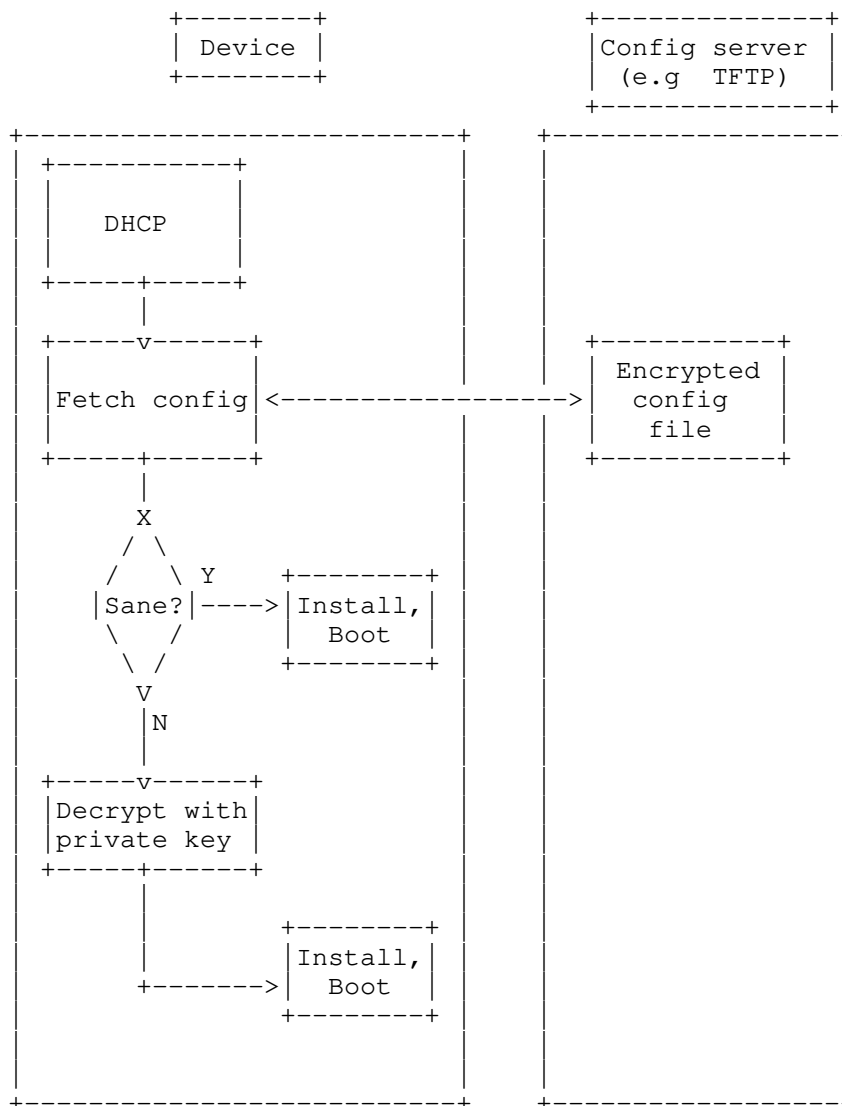
When the device is first booted by the customer (and on subsequent boots), if the device has no valid configuration, it will use existing auto-install type functionality - it performs DHCP Discovery until it gets a DHCP offer including DHCP option 66 or 150, contact the server listed in these DHCP options and download its config file.

After retrieving the config file, the device will examine the file and determine if it seems to be a valid config, and if so, proceeds as it normally would. Note that this is existing functionality (for example, Cisco devices fetch the config file named by the Bootfile-Name DHCP option (67)).

If the file appears be "garbage", the device will attempt to decrypt the configuration file using its private key. If it is able to decrypt and validate the file it will install the configuration, and start using it. The exact method that the device uses to determine

if a config file is "valid" is implementation specific, but a normal config file looks significantly different to an encrypted blob.

Note that the device only needs DHCP and to be able to download the config file; after the initial power-on in the factory it never need to access the Internet or vendor or certificate publication server - it (and only it) has the private key and so has the ability to decrypt the config file.



Device boot, fetch and install config file

5. Additional Considerations

5.1. Key storage

Ideally, the keypair would be stored in a TPM on something which is identified as the "router" - for example, the chassis / backplane. This is so that a keypair is bound to what humans think of as the

"device", and not, for example (redundant) routing engines. Devices which implement IEEE 802.1AR could choose to use the IDevID for this purpose.

5.2. Key replacement

It is anticipated that some operator may want to replace the (vendor provided) keys after installing the device. There are two options when implementing this - a vendor could allow the operator's key to completely replace the initial device generated key (which means that, if the device is ever sold, the new owner couldn't use this technique to install the device), or the device could prefer the operators installed key. This is an implementation decision left to the vendor.

5.3. Device reinstall

Increasingly, operations is moving towards an automated model of device management, whereby portions (or the entire) configuration is programmatically generated. This means that operators may want to generate an entire configuration after the device has been initially installed and ask the device to load and use this new configuration. It is expected (but not defined in this document, as it is vendor specific) that vendors will allow the operator to copy a new, encrypted config (or part of a config) onto a device and then request that the device decrypt and install it (e.g: 'load replace <filename> encrypted)'). The operator could also choose to reset the device to factory defaults, and allow the device to act as though it were the initial boot (see Section 4.3).

6. IANA Considerations

This document makes no requests of the IANA.

7. Security Considerations

This mechanism is intended to replace either expensive (traveling employees) or insecure mechanisms of installing newly deployed devices such as: unencrypted config files which can be downloaded by connecting to unprotected ports in datacenters, mailing initial config files on flash drives, or emailing config files and asking a third-party to copy and paste it over a serial terminal. It does not protect against devices with malicious firmware, nor theft and reuse of devices.

An attacker (e.g a malicious datacenter employee) who has physical access to the device before it is connected to the network the attacker may be able to extract the device private key (especially if

it isn't stored in a TPM), pretend to be the device when connecting to the network, and download and extract the (encrypted) config file.

This mechanism does not protect against a malicious vendor - while the keypair should be generated on the device, and the private key should be securely stored, the mechanism cannot detect or protect against a vendor who claims to do this, but instead generates the keypair off device and keeps a copy of the private key. It is largely understood in the operator community that a malicious vendor or attacker with physical access to the device is largely a "Game Over" situation.

Even when using a secure bootstrapping mechanism, security conscious operators may wish to bootstrapping devices with a minimal / less sensitive config, and then replace this with a more complete one after install.

8. Acknowledgements

The authors wish to thank everyone who contributed, including Benoit Claise, Sam Ribeiro, Michael Richardson, Sean Turner and Kent Watsen. Joe Clarke provided significant comments and review.

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>>.

9.2. Informative References

- [I-D.ietf-sidr-iana-objects] Manderson, T., Vegoda, L., and S. Kent, "RPKI Objects issued by IANA", draft-ietf-sidr-iana-objects-03 (work in progress), May 2011.
- [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <<https://www.rfc-editor.org/info/rfc4122>>.
- [RFC8572] Watsen, K., Farrer, I., and M. Abrahamsson, "Secure Zero Touch Provisioning (SZTP)", RFC 8572, DOI 10.17487/RFC8572, April 2019, <<https://www.rfc-editor.org/info/rfc8572>>.

Appendix A. Changes / Author Notes.

[RFC Editor: Please remove this section before publication]

From -00 to -01

- o Nothing changed in the template!

From -01 to -03:

- o See github commit log (AKA, we forgot to update this!)
- o Added Colin Doyle.

From -03 to -04:

Addressed a number of comments received before / at IETF104 (Prague). These include:

- o Pointer to <https://datatracker.ietf.org/doc/draft-ietf-netconf-zero-touch> -- included reference to (now) RFC8572 (KW)
- o Suggested that 802.1AR IDevID (or similar) could be used. Stress that this is designed for simplicity (MR)
- o Added text to explain that any unique device identifier can be used, not just serial number - serial number is simple and easy, but anything which is unique (and can be communicated to the customer) will work (BF).
- o Lots of clarifications from Joe Clarke.
- o Make it clear it should first try use the config, and if it doesn't work, then try decrypt and use it.
- o The CA part was confusing people - the certificate is simply a wrapper for the key, and the Subject just an index, and so removed that.
- o Added a bunch of ASCII diagrams

Appendix B. Demo / proof of concept

This section contains a rough demo / proof of concept of the system. It is only intended for illustration; presumably things like algorithms, key lengths, format / containers will provide much fodder for discussion.

It uses OpenSSL from the command line, in production something more automated would be used. In this example, the unique identifier is the serial number of the router, SN19842256.

B.1. Step 1: Generating the certificate.

This step is performed by the router. It generates a key, then a csr, and then a self signed certificate.

B.1.1. Step 1.1: Generate the private key.

```
$ openssl genrsa -out key.pem 2048
Generating RSA private key, 2048 bit long modulus
.....
.....+++
.....+++
e is 65537 (0x10001)
```

B.1.2. Step 1.2: Generate the certificate signing request.

```
$ openssl req -new -key key.pem -out SN19842256.csr
Country Name (2 letter code) [AU]:.
State or Province Name (full name) [Some-State]:.
Locality Name (eg, city) []:.
Organization Name (eg, company) [Internet Widgits Pty Ltd]:.
Organizational Unit Name (eg, section) []:.
Common Name (e.g. server FQDN or YOUR name) []:SN19842256
Email Address []:.
```

```
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:.
```

B.1.3. Step 1.3: Generate the (self signed) certificate itself.

```
$ openssl req -x509 -days 36500 -key key.pem -in SN19842256.csr -out
SN19842256.crt
```

The router then sends the key to the vendor's keyserver for publication (not shown).

B.2. Step 2: Generating the encrypted config.

The operator now wants to deploy the new router.

They generate the initial config (using whatever magic tool generates router configs!), fetch the router's certificate and encrypt the config file to that key. This is done by the operator.

B.2.1. Step 2.1: Fetch the certificate.

```
$ wget http://keyserv.example.net/certificates/SN19842256.crt
```

B.2.2. Step 2.2: Encrypt the config file.

I'm using S/MIME because it is simple to demonstrate. This is almost definitely not the best way to do this.

```
$ openssl smime -encrypt -aes-256-cbc -in SN19842256.cfg\  
-out SN19842256.enc -outform PEM SN19842256.crt  
$ more SN19842256.enc  
-----BEGIN PKCS7-----  
MIICigYJKoZIhvcNAQcDoII CezCCAncCAQAxggE+MIIBOgIBADAiMBUxEzARBgNV  
BAMMC1NOMTk4NDIyNTYCCQDJVuBlaTOblDANBgkqhkiG9w0BAQEFAASCAQBABvM3  
...  
LZoq08jqlWhZZWhTKs4XPGHUdmnZRYIP8KXyEtHt  
-----END PKCS7-----
```

B.2.3. Step 2.3: Copy config to the config server.

```
$ scp SN19842256.enc config.example.com:/tftpboot
```

B.3. Step 3: Decrypting and using the config.

When the router connects to the operator's network it will detect that does not have a valid configuration file, and will start the "autoboot" process. This is a well documented process, but the high level overview is that it will use DHCP to obtain an IP address and config server. It will then use TFTP to download a configuration file, based upon its serial number (this document modifies the solution to fetch an encrypted config file (ending in .enc)). It will then then decrypt the config file, and install it.

B.3.1. Step 3.1: Fetch encrypted config file from config server.

```
$ tftp 192.0.2.1 -c get SN19842256.enc
```

B.3.2. Step 3.2: Decrypt and use the config.

```
$ openssl smime -decrypt -in SN19842256.enc -inform pkcs7\  
-out config.cfg -inkey key.pem
```

If an attacker does not have the correct key, they will not be able to decrypt the config:

```
$ openssl smime -decrypt -in SN19842256.enc -inform pkcs7\  
-out config.cfg -inkey wrongkey.pem  
Error decrypting PKCS#7 structure  
140352450692760:error:06065064:digital envelope  
routines:EVP_DecryptFinal_ex:bad decrypt:evp_enc.c:592:  
$ echo $?  
4
```

Authors' Addresses

Warren Kumari
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: warren@kumari.net

Colin Doyle
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
US

Email: cdoyle@juniper.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 22, 2019

G. Zheng
M. Wang
B. Wu
Huawei
June 20, 2019

Yang data model for TACACS+
draft-zheng-opsawg-tacacs-yang-02

Abstract

This document defines a YANG modules that augment the System data model defined in the RFC 7317 with TACACS+ client model. The data model of Terminal Access Controller Access Control System Plus (TACACS+) client allows the configuration of TACACS+ servers for centralized Authentication, Authorization and Accounting.

The YANG modules in this document conforms to the Network Management Datastore Architecture (NMDA) defined in RFC 8342.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 22, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions used in this document	3
2.1. Tree Diagrams	3
3. TACACS+ Client Model	3
4. TACACS+ Client Module	5
5. Security Considerations	11
6. IANA Considerations	11
7. Acknowledgments	12
8. References	12
8.1. Normative References	12
8.2. Informative References	13
Authors' Addresses	13

1. Introduction

This document defines a YANG modules that augment the System data model defined in the [RFC7317] with TACACS+ client model.

TACACS+ provides Device Administration for routers, network access servers and other networked computing devices via one or more centralized servers which is defined in the TACACS+ Protocol.
[I-D.ietf-opsawg-tacacs]

The System Management Model [RFC7317] defines two YANG features to support local or RADIUS authentication:

- o User Authentication Model: Define a list of usernames and passwords and control the order in which local or RADIUS authentication is used.
- o RADIUS Client Model: Defines a list of RADIUS server that a device used.

Since TACACS+ is also used for device management and the feature is not contained in the system model, this document defines a YANG data model that allows users to configure TACACS+ client functions on a device for centralized Authentication, Authorization and Accounting provided by TACACS+ servers.

The YANG models can be used with network management protocols such as NETCONF[RFC6241] to install, manipulate, and delete the configuration of network devices.

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

2. Conventions used in this document

The keywords "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 BCP14, [RFC2119], [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are defined in [RFC6241] and are used in this specification:

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

The following terms are defined in [RFC7950] and are used in this specification:

- o augment
- o data model
- o data node

The terminology for describing YANG data models is found in [RFC7950].

2.1. Tree Diagrams

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

3. TACACS+ Client Model

This model is used to configure TACACS+ client on the device to support deployment scenarios with centralized authentication, authorization, and accounting servers. Authentication is used to

validates a user's name and password, authorization allows the user to access and execute commands at various command levels assigned to the user and accounting keeps track of the activity of a user who has accessed the device.

The `ietf-system-tacacsplus` module is intended to augment the `/sys:system` path defined in the `ietf-system` module with `"tacacsplus"` grouping. Therefore, a device can use local, Remote Authentication Dial In User Service (RADIUS), or Terminal Access Controller Access Control System Plus (TACACS+) to validate users who attempt to access the router by several mechanisms, e.g. a command line interface or a web-based user interface.

The `"server"` list is directly under the `"tacacsplus"` container, which is to hold a list of different TACACS+ server and use `server-type` to distinguish the three protocols. The list of servers is for redundancy purpose.

Most of the parameters in the `"server"` list are taken directly from the TACACS+ protocol [I-D.ietf-opsawg-tacacs], and some are derived from the wide implementation of network equipment manufacturers. For example, when there are multiple interfaces connected to the TACACS+ server, the source address of outgoing TACACS+ packets could be specified, or the source address could be specified through the interface setting. For the TACACS+ server located in a private network, a VRF instance needs to be specified.

The `"statistics"` container under the `"server list"` is to record session statistics and usage information during user access which include the amount of data a user has sent and/or received during a session.

The data model for TACACS+ client has the following structure:

```

module: ietf-system-tacacsplus
augment /sys:system:
  +--rw tacacsplus {tacacsplus}?
    +--rw server* [name]
      +--rw name string
      +--rw server-type? enumeration
      +--rw address inet:host
      +--rw port? inet:port-number
      +--rw shared-secret string
      +--rw (source-type)?
        | +--:(source-ip)
        | | +--rw source-ip? inet:ip-address
        | +--:(source-interface)
        | | +--rw source-interface? if:interface-ref
      +--rw single-connection? boolean
      +--rw timeout? uint16
      +--rw vrf-instance?
        | -> /ni:network-instances/network-instance/name
      +--ro statistics
        +--ro connection-opens? yang:counter64
        +--ro connection-closes? yang:counter64
        +--ro connection-aborts? yang:counter64
        +--ro connection-failures? yang:counter64
        +--ro connection-timeouts? yang:counter64
        +--ro messages-sent? yang:counter64
        +--ro messages-received? yang:counter64
        +--ro errors-received? yang:counter64

```

4. TACACS+ Client Module

<CODE BEGINS> file "ietf-system-tacacsplus@2019-06-20.yang"

```

module ietf-system-tacacsplus {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-system-tacacsplus";
  prefix sys-tacsplus;

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

```

```
    "RFC 8529: YANG Data Model for Network Instances";
}
import ietf-interfaces {
  prefix if;
  reference
    "RFC 8343: A YANG Data Model for Interface Management";
}
import ietf-system {
  prefix sys;
  reference "RFC 7317: A YANG Data Model for System Management";
}
import ietf-netconf-acm {
  prefix nacm;
  reference "RFC 8341: Network Configuration Access Control Model";
}

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

  Editor:   Guangying Zheng
            <mailto:zhengguangying@huawei.com>";
description
  "This module provides configuration of TACACS+ client.

  Copyright (c) 2018 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 2019-06-20 {
  description
    "Initial revision.";
  reference "foo";
}

feature tacacsplus {
  description
```

```
    "Indicates that the device can be configured as a TACACS+
      client.";
    reference "draft-ietf-opsawg-tacacs-11: The TACACS+ Protocol";
  }

  grouping statistics {
    description
      "Grouping for TACACS+ packets statistics attributes";
    container statistics {
      config false;
      description
        "A collection of server-related statistics objects";
      leaf connection-opens {
        type yang:counter64;
        description
          "Number of new connection requests sent to the server, e.g.
            socket open";
      }
      leaf connection-closes {
        type yang:counter64;
        description
          "Number of connection close requests sent to the server, e.g.
            socket close";
      }
      leaf connection-aborts {
        type yang:counter64;
        description
          "Number of aborted connections to the server. These do
            not include connections that are close gracefully.";
      }
      leaf connection-failures {
        type yang:counter64;
        description
          "Number of connection failures to the server";
      }
      leaf connection-timeouts {
        type yang:counter64;
        description
          "Number of connection timeouts to the server";
      }
      leaf messages-sent {
        type yang:counter64;
        description
          "Number of messages sent to the server";
      }
      leaf messages-received {
        type yang:counter64;
        description
```

```
        "Number of messages received by the server";
    }
    leaf errors-received {
        type yang:counter64;
        description
            "Number of error messages received from the server";
    }
}

grouping tacacsplus {
    description
        "Grouping for TACACS+ attributes";
    container tacacsplus {
        if-feature "tacacsplus";
        description
            "Container for TACACS+ configurations and operations.";
        list server {
            key "name";
            ordered-by user;
            description
                "List of TACACS+ servers used by the device

                When the TACACS+ client is invoked by a calling
                application, it sends the query to the first server in
                this list.  If no response has been received within
                'timeout' seconds, the client continues with the next
                server in the list.  If no response is received from any
                server, the client continues with the first server again.
                When the client has traversed the list 'attempts' times
                without receiving any response, it gives up and returns an
                error to the calling application.";

            leaf name {
                type string;
                description
                    "An arbitrary name for the TACACS+ server.";
            }
            leaf server-type {
                type enumeration {
                    enum authentication {
                        description
                            "The server is an authentication server.";
                    }
                    enum authorization {
                        description
                            "The server is an authorization server.";
                    }
                    enum accounting {
```

```
        description
            "The server is an accounting server.";
    }
}
description
    "Server type: authentication/authorization/accounting.";
}
leaf address {
    type inet:host;
    mandatory true;
    description
        "The address of the TACACS+ server.";
}
leaf port {
    type inet:port-number;
    default "49";
    description
        "The port number of TACACS+ Server port.";
}
leaf shared-secret {
    type string;
    mandatory true;
    nacm:default-deny-all;
    description
        "The shared secret, which is known to both the
        TACACS+ client and server. TACACS+ server administrators
        SHOULD configure secret keys of minimum
        16 characters length.";
    reference "TACACS+ protocol:";
}
choice source-type {
    description
        "The source address type for outbound TACACS+ packets.";
    case source-ip {
        leaf source-ip {
            type inet:ip-address;
            description
                "Specifies source IP address for TACACS+ outbound
                packets.";
        }
    }
    case source-interface {
        leaf source-interface {
            type if:interface-ref;
            description
                "Specifies the interface from which the IP address is
                derived for use as the source for the outbound TACACS+
                packet";
        }
    }
}
```

```
    }
  }
}
leaf single-connection {
  type boolean;
  default "false";
  description
    "Whether the single connection mode is enabled for the
    server. By default, the single connection mode is
    disabled.";
}
leaf timeout {
  type uint16 {
    range "1..300";
  }
  units "seconds";
  default "5";
  description
    "The number of seconds the device will wait for a
    response from each TACACS+ server before trying with a
    different server.";
}
leaf vrf-instance {
  type leafref {
    path "/ni:network-instances/ni:network-instance/ni:name";
  }
  description
    "Specifies the VPN Routing and Forwarding (VRF) instance to
    use to communicate with the TACACS+ server.";
}

uses statistics;
}
}
}

augment "/sys:system" {
  description
    "Augment the system model with authorization and accounting
    attributes
    Augment the system model with the tacacsplus model";
  uses tacacsplus;
}
}

<CODE ENDS>
```

5. Security Considerations

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

The NETCONF access control model [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.

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.

This document describes the use of TACACS+ for purposes of authentication, authorization and accounting, it is vulnerable to all of the threats that are present in TACACS+ applications. For a discussion of such threats, see Section 9 of the TACACS+ Protocol [I-D.ietf-opsawg-tacacs].

6. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-system-tacacsplus
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 [RFC7950].

Name: ietf-system-tacacsplus
Namespace: urn:ietf:params:xml:ns:yang:ietf-tacacsplus
Prefix: sys-tacsplus
Reference: RFC XXXX

7. Acknowledgments

The authors wish to thank Alex Campbell and Ebben Aries, Alan DeKok, Joe Clarke, many others for their helpful comments.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [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>>.
- [RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <<https://www.rfc-editor.org/info/rfc7317>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [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>>.

8.2. Informative References

- [I-D.ietf-opsawg-tacacs]
Dahm, T., Ota, A., dcmgash@cisco.com, d., Carrel, D., and L. Grant, "The TACACS+ Protocol", draft-ietf-opsawg-tacacs-13 (work in progress), March 2019.

Authors' Addresses

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

Email: zhengguangying@huawei.com

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

Email: wangzitao@huawei.com

Bo Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: lana.wubo@huawei.com