TEAS Working Group X. Zhang, Ed.
Internet-Draft Huawei Technologies
Intended status: Standards Track J. Ryoo
Expires: May 4, 2017 ETRI
October 31, 2016

A Service YANG Model for Connection-oriented Transport Networks
draft-zhang-teas-transport-service-model-01

Abstract

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic opaquely across the server-layer network resources. A transport network may be constructed from equipments utilizing any of a number of different transport technologies such as the evolving optical transport infrastructure (Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH) and Optical Transport Network (OTN)) or packet transport as epitomized by the MPLS Transport Profile (MPLS-TP).

This draft provides a transport service YANG model that can be used together with the RESTCONF protocol for a northbound client to initiate service requests toward the transport network controllers via the RESTful interface between them so as to enable automated service interations.

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 [RFC2119].

Status of This Memo

This Internet-Draft will expire on May 4, 2017.

Copyright Notice

Table of Contents

1. Introduction

A transport network is a server-layer network designed to provide connectivity services, or more advanced services like Virtual Private Networks (VPN) for a client-layer network to carry the client traffic opaquely across the server-layer network resources. It acts as a pipe provider for upper-layer networks, such as IP network and mobile networks.

Transport networks, such as Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH), Optical Transport Network (OTN), Wavelength Division Multiplexing (WDM), and flexi-grid networks, are often built using equipments from a single vendor and are managed using private interfaces to dedicated Element Management Systems

(EMS) / Network Management Systems (NMS). All transport networks have high benchmarks for reliability and operational simplicity. This suggests a common, technology-independent management/control paradigm that is extended to represent and configure specific technology attributes.

The YANG language [RFC6020] is currently the data modeling language of choice within the IETF and has been adopted by a number of industry-wide open management and control initiatives. YANG may be used to model both configuration and operational states; it is vendor-neutral and supports extensible APIs for control and management of elements.

This document, exploting the YANG language, provides a transport service model that can be used by a northbound client (e.g., an orchestrator, a client network controller etc.) to initiate service requests, as well as retrieving service states, toward the transport network via the RESTful interface provided by the transport network controller(s). The model is currently scoped for connection-oriented transport networks and connection-less transport networks are out of scope. To understand better what interfaces a service model can apply to in the transport SDN architecture (i.e., Abstract Control of Transport Networks (ACTN))and how service model relates to other YANG models defined in IETF in general, please refer to [I-D.zhang-teas-actn-yang] for more detailed information.

2. Terminology

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in the YANG data tree presented later in this draft is defined in [I-D.ietf-netmod-rfc6087bis]. They are provided below for reference.

o Brackets "[" and "]" enclose list keys.

o Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).

o Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

o Ellipsis ("...") stands for contents of subtrees that are not shown.

3. Service Model - YANG Tree

The service model YANG tree is provided below:

```
   module: ietf-transport-service
      +--rw transport_service
         +--rw service* [service-name]
            +--rw service-name    -> ../config/service-name
            +--rw config
            |  +--rw service-name?        string
            |  +--rw service-id?          uint32
            |  +--rw service-endpoints* [node-id tp-id]
            |  |  +--rw type?            enumeration
            |  |  +--rw node-id          union
            |  |  +--rw tp-id            uint32
            |  |  +--rw endpoint-name?   string
            |  +--rw service-type         identityref
            |  +--rw supporting-tunnel
            |  |  +--rw tunnel-name?   string
            |  +--rw bandwidth?           decimal64
            |  +--rw protection-type?     identityref
            |  +--rw constraints
            |     +--rw delay-limit?            uint32
            |     +--rw delayvariation-limit?   uint32
            |     +--rw packetloss-limit?       decimal64
            |     +--rw objective?              identityref
            +--ro state
               +--ro service-name?        string
               +--ro service-id?          uint32
               +--ro service-endpoints* [node-id tp-id]
               |  +--ro type?            enumeration
               |  +--ro node-id          union
               |  +--ro tp-id            uint32
               |  +--ro endpoint-name?   string
               +--ro service-type         identityref
               +--ro supporting-tunnel
               |  +--ro tunnel-name?   string
               +--ro bandwidth?           decimal64
               +--ro protection-type?     identityref
               +--ro creation-time?       ytypes:date-and-time
               +--ro constraints
               |  +--ro delay-limit?            uint32
               |  +--ro delayvariation-limit?   uint32
               |  +--ro packetloss-limit?       decimal64
               |  +--ro objective?              identityref
               +--ro performance-info
               |  +--ro delay?             uint32
               |  +--ro delay-variation?   uint32
               |  +--ro packet-loss?       decimal64
               +--ro oper-status          tet:te-oper-status
```

4. Service Model - YANG Code

```
 <CODE BEGINS>file "ietf-transport-service@2016-10-31.yang"
module ietf-transport-service {
        yang-version 1;
        namespace "urn:ietf:params:xml:ns:yang:ietf-transport-service";
        prefix transservice;

        import ietf-inet-types {
                prefix inet;
        }

        import ietf-yang-types {
          prefix ytypes;
        }

        import ietf-te-types {
                prefix "tet";
        }



        organization "Internet Engineering Task Force";
        contact
                "Editor: Xian ZHANG (zhang.xian@huawei.com)";
        description
                "this module describes a service module that is an essential
                API for a client to ask for a provider network for a path
                without the need to care about underlying technologies.
                Capability to specify constraints/policies are provided as
                optional features.";

        revision 2016-10-31 {
                description
                        "Initial revision.";
                        reference "to add the draft name";
        }

        grouping endpoints-grouping{
                description "end point grouping";
                leaf type {
                        type enumeration {
                                enum root {
                                        description "root";
                                }
                                enum leaf {
```

```
                                        description "leaf";
                                }
                                enum unknown {
                                        description "unknown";
                                }
                        }
                default root;

                description
                        "This attribute indicates whether a endpoint of the
                        service is acting as a root or leaf, and takes on
                        the value either root, leaf or unkown. If the service
                        type is Point-to-Point or Multipoint-to-Multipoint,
                        then the type is equal to root. If the service type
                        is Point-to-Multipoint, then the type can be either
                        root or leaf.";
                }


                leaf node-id {
                        type union {
                                type inet:ip-address; //IPv4 or IPv6
                                type uint32;
                        }
                        description
                                "the node address of a service .";
                }

                leaf tp-id {
                        type uint32;
                        description
                                "the termination point of a service.";
                }

                leaf endpoint-name {
                        type string;
                        description
                                "the name of this service endpoint";
                }
        }

        identity service-type {
                description
                        "base identity from which the specific service
                        type can be derived.";
        }

        identity service-P2P {
```

```
                base service-type;
                description
                        "a point-to-point service type";
        }

        identity service-P2MP {
                base service-type;
                description
                        "a point to multi-point type";
        }

        identity service-MP2MP {
                base service-type;
                description
                        "a multi-point to multi-point type";
        }

//note: service type such as EPL, EVPL, EPLAN
//      EVPLAN, E-Tree can be added when augmenting
//      this model.

        identity service-prot-type {
                description
                        "base identity from which the specific
                         protection type can be derived.";
        }

        identity service-prot-unprotected {
                base service-prot-type;
                description
                        "service protection type: unprotected";
        }

        identity service-prot-1plusR {
                base service-prot-type;
                description
                        "service protection type: dynamic reroute";
                reference
                        "draft-ietf-teas-gmpls-resource-sharing-proc-04";
        }

        identity service-prot-1plus1 {
                base service-prot-type;
                description
                        "service protection type: static 1+1";
        }

        identity service-prot-1plus1plusR {
```

```
                base service-prot-type;
                description
                        "service protection type: 1+1+R";
                reference
                        "draft-ietf-teas-gmpls-resource-sharing-proc-04";
        }

        identity objective-type{
                description
                        "base identity from which the specific obective
                        funciton can be derived.";
        }
        identity objective-mindelay{
                base objective-type;
                description
                        "objective: minimized delay";
        }
        identity objective-mincost {
                base objective-type;
                description
                        "objective: minimized cost";
        }

        identity objective-mindistance {
                base objective-type;
                description
                        "objective: minimized distance";
        }

        grouping service-config-grouping {
                description "service-config-groupoing";

                leaf service-name{
                        type string;
                        description "the name for a service";
                }

                leaf service-id {
                        type uint32;
                        description
                                "an unique identificaiton of a service.";
                }

                list service-endpoints{
                        key "node-id tp-id";
                        description
                                "this provides the service endpoints and it can
be used
```

to support different types of services (including P2P,

```
                                MP2MP, P2MP)";

                        uses endpoints-grouping;
                }

                leaf service-type {
                        type identityref {
                                base service-type;
                        }
                        mandatory true;
                        description
                                "the type of a service request";
                }

                container supporting-tunnel {
                        leaf tunnel-name{
                                type string;
                                description "the name of a tunnel (unique)";
                        }

                        description
                                "the tunnels that can be used to support the
service";
                }

                leaf bandwidth {
                        type decimal64 {
                                fraction-digits 2;
                        }
                        description "the bandwidth requested by a service.";
                }

                leaf protection-type{
                        type identityref {
                                base service-prot-type;
                        }
                        description
                        "the type of protection expected for this
                        service";
                }
        }

        grouping constraints-grouping {
                leaf delay-limit {
                        type uint32;
                        description
                        "Delay variation in micro-seconds.";
                }
```

```
                leaf delayvariation-limit {
                        type uint32;
                        description
                        "Delay variation.";
                }

                leaf packetloss-limit {
                        type decimal64 {
                                fraction-digits 6;
                                range "0 .. 50.331642";
                        }
                        description
                        "Delay variation in %.";
                }
                leaf objective {
                        type identityref
                        {
                                base objective-type;
                        }
                        description
                                "objective function imposed for path
computation

                                from this service.";
                }
                description "constraints gropuing";
        }

        container transport_service {
                description
                        "serves as a top-level container for a list of
services";

                list service {
                        key "service-name";
                        description
                                "an unique identifier of a service";

                        leaf service-name {
                                type leafref {
                                        path "../config/service-name";
                                }
                                description "a unique service identifier.";
                        }
                        container config{
                                description "service config container";

                                uses service-config-grouping;
                                container constraints{
                                        uses constraints-grouping;
                                        description
                                                "specify the constraints
imposed by a
                                                service";
                                }
                        }

                        container state
                        {
                                config false;
                                description "operational state of a service";

                                uses service-config-grouping;

                                leaf creation-time {
                                        type ytypes:date-and-time;
                                        description
                                                "the time when service is
created";
                                }

                                container constraints{
                                        uses constraints-grouping;
                                        description
                                                "specify the constraints
imposed by a
                                                service";
                                }

                                container performance-info{
                                        description "service state-performance
info";
                                        leaf delay{
                                                type uint32;
                                                description "the latency of
this service";
                                        }

                                        leaf delay-variation {
                                                type uint32;
                                                description "the service delay
varation";
                                        }
                                        leaf packet-loss {
                                                type decimal64 {
                                                        fraction-digits 6;
                                                        range "0 .. 50.331642";
                                                }
                                                description "packet loss";
                                        }
                                }
                                leaf oper-status {
                                        type tet:te-oper-status;
                                        mandatory true;
                                        description
                                                "the operational status of this
service";
                                }
                        }//end of state of a service
                }//end of service list (including config and state)
        }//service top container
}


<CODE ENDS>
```

5. General Considerations and Some Open Issues

[I-D.liu-netmod-yang-schedule] defines a YANG data model for configuration scheduling. A service model can also be scheduled, therefore the ietf-schedule model can be used to serve this purpose by specifying the appropriate object using xpath.

Functions that are under considerations for updating the YANG model:

1): Service delivery model augmentation: Information such as PIR/CIR pertaining to ETH service is needed and it is more appropriate to augment the base model; for OTN-based service model, signal-type information needs to be added;

2): To add service-related notification, e.g.: service up; service down, service degraded etc.; Currently, there are two options available, one is using the method provided in [RFC5277], the other one is using the new mechanism proposed in [I-D.ietf-netconf-yang-push].

3): To figure out how to deal with attributes such as bandwidth, schedule where it can be per node for non-P2P, but is per node pair for P2P;

4): to add XRO/IRO constraints;

6. IANA Considerations

TBD.

7. Manageability Considerations

TBD

8. Security Considerations

Since the data model defined in this draft is manipulated via, for example, the interface between an orchestrator and a transport network controller. The YANG module defined in this draft is designed to be accessed via the RESTCONF protocol defined in [I-D.ietf-netconf-restconf], or maybe via the NETCONF protocol [RFC6241]. As mentioned in Security Conserations Section of [I-D.ietf-netconf-restconf] that HTTPS defined in [RFC7230] can be used to provide security while accessing data.

There are a number of data nodes defined in the YANG module which 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., POST) to these data nodes without proper protection can have a negative effect on network operations.

Editor note: to List specific subtrees and data nodes and their sensitivity/vulnerability.

9. Acknowledgements

We would like to thank Baoquan Rao and Gang Xie for their initial discussions that trigger the generation of this draft. We would also like to thank Michael Scharf from Nokia for very useful comments to this draft and the model.

Aspects of the work in this Internet-Draft is funded by the UK Engineering and Physical Sciences Research Council (EPSRC) under the TOUCAN project (EP/L020009/1).

10. Contributors

Zhe Liu
Huawei Technologies
Email: liuzhe123@huawei.com

Sergio Belotti
Nokia
Email:sergio.belotti@nokia.com

Daniel King
Lancaster University
Email:d.kindg@lancaster.ac.uk

11. References

11.1. Normative References

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-13 (work in progress), April 2016.

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-06 (work in progress), March 2016.

[I-D.liu-netmod-yang-schedule]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "A YANG Data Model for Configuration Scheduling", draft-liu-netmod-yang-schedule-02 (work in progress), October 2016.

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

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

11.2. Informative References

[I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", draft-ietf-netconf-yang-push-04 (work in progress), October 2016.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-03 (work in progress), March 2016.

[I-D.zhang-teas-actn-yang]
Lee, Y., Zhang, X., Yoon, B., Dios, O., and J. Shin, "Applicability of YANG models for Abstraction and Control of Traffic Engineered Networks", draft-zhang-teas-actn-yang-02 (work in progress), October 2016.

[RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008, <http://www.rfc-editor.org/info/rfc5277>.

[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, <http://www.rfc-editor.org/info/rfc6241>.

Authors' Addresses

Xian Zhang (editor)
Huawei Technologies
F3-5-B R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong 518129
P.R.China

Email: zhang.xian@huawei.com

Jeong-dong Ryoo
ETRI

Email: ryoo@etri.re.kr