         A YANG Data Model for Fabric Topology in Data Center Network
              draft-ietf-i2rs-yang-dc-fabric-network-topology-01

Abstract

   This document defines a YANG data model for fabric topology in Data
   Center Network.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   Normally, a data center network is composed of single or multiple
   fabrics which are also known as PODs (a Point Of Delivery). These
   fabrics may be heterogeneous due to implementation of different
   technologies while DC network upgrading or enrolling new techniques
   and features. For example, Fabric A may use VXLAN while Fabric B may
   use VLAN within a DC network. Likewise, a legacy Fabric may use
   VXLAN while a new Fabric B implemented technique discussed in NVO3 WG
   such as GPE[I-D. draft-ietf-nvo3-vxlan-gpe] may be built due to DC
   expansion and upgrading. The configuration and management of such DC
   networks with heterogeneous fabrics will be sophisticated and
   complex.

   Luckily, for a DC network, a fabric can be considered as an atomic
   structure to provide network services and management, as well as
   expand network capacity. From this point of view, the miscellaneous
   DC network management can be decomposed to task of managing each
   fabric respectively along with their connections, which can make the
   entire management much concentrated and flexible, also easy to
   expand.

   With this purpose, this document defines a YANG data model for the
   Fabric-based Data center topology by using YANG [6020][7950]. To do

so, it augments the generic network and network topology data models
defined in [I-D.ietf-i2rs-yang-network-topo] with information
specific to Data Center fabric network.

This model defines the generic configuration and operational state
for a fabric-based network topology, which can be extended by vendors
with specific information. This model can then be used by a network
controller to represent its view of the fabric topology that it
controls and expose it to network administrators or applications for
DC network management.

With the context of topology architecture defined in [I-D.ietf-i2rs-
yang-network-topo] and [I.D. draft-ietf-i2rs-usecase-reqs-summary],
this model can also be treated as an application of I2RS network
topology model [I-D.ietf-i2rs-yang-network-topo] in the scenario of
Data center network management. It can also act as a service
topology when mapping network elements at fabric layer to elements to
other topologies, such as L3 topology defined in [I.D. draft-ietf-
i2rs-yang-l3-topology-01.

By using this fabric topology model, people can treat a fabric as an
entity and focus on characteristics of fabrics (such as encapsulation
type, gateway type, etc.) as well as their interconnections while
putting the underlay topology aside. As such, clients can consume
the topology information at fabric level, while no need to be aware
of entire set of links and nodes in underlay networks. The
configuration of a fabric topology can be made by a network
administractor to the controller by adding physical devices and links
of a fabric into a fabric network. Alternatively, the fabric
topology can also learnt from the underlay network infrastructure.

2. Definitions an Acronyms

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].

2.1. Terminology

DC Fabric: also known as POD, is a module of network, compute,
storage, and application components that work together to deliver
networking services. It is a repeatable design pattern, and its
components maximize the modularity, scalability, and manageability of
data centers.

2.2. Tree diagram

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

  Each node is printed as:
    <status> <flags> <name> <opts> <type>

   <status> is one of:
         + for current
         x for deprecated
         o for obsolete
   <flags> is one of:
        rw for configuration data
        ro for non-configuration data
        -x for rpcs
        -n for notifications
   <name> is the name of the node

   If the node is augmented into the tree from another module, its name
   is printed as <prefix>:<name>.
   <opts> is one of:
         ? for an optional leaf or choice
         ! for a presence container
         * for a leaf-list or list
         [<keys>] for a list's keys
   <type> is the name of the type for leafs and leaf-lists

   In this document, these words will appear with that interpretation
   only when in ALL CAPS. Lower case uses of these words are not to be
   interpreted as carrying RFC-2119 significance.

3. Model Overview

   This section provides an overview of the DC Fabric topology model and
   its relationship with other topology models.

3.1. Topology Model structure

   The relationship of the DC fabric topology model and other topology
   models is shown in the following figure (dotted lines in the figure
   denote augmentations).

```
                  +------------------------+
                  |      network model     |
                  +------------------------+
                              |
                              |
                  +-----------V------------+
                  | network topology model |
                  +------------------------+
                              |
          +-----------+-----+------+-------------+
          |           |            |             |
     +---V----+  +---V----+   +---V----+   +----V---+
     |   L1   |  |   L2   |   |   L3   |   | Fabric |
     |topology|  |topology|   |topology|   |topology|
     | model  |  | model  |   | model  |   | model  |
     +--------+  +--------+   +--------+   +--------+
```
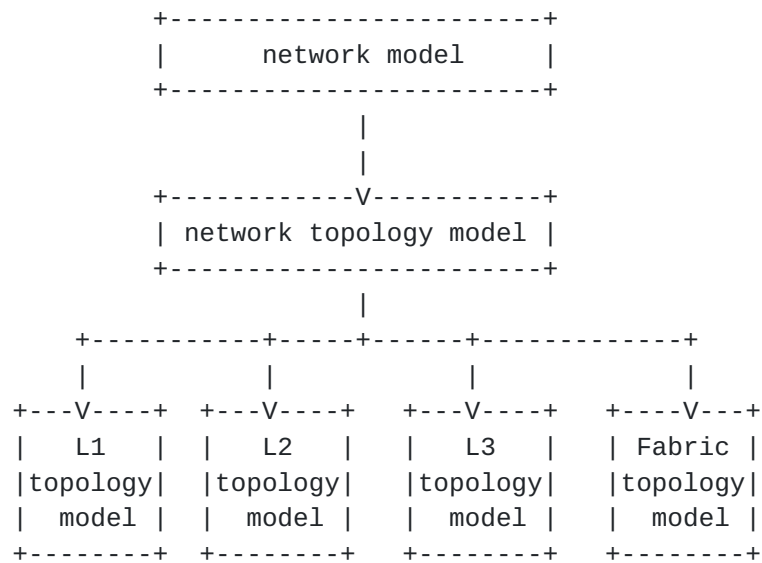
From the perspective of resource management and service provisioning
for a Data Center network, the fabric topology model augments the
basic network topology model with definitions and features specific
to a DC fabric, to provide common configuration and operations for
heterogeneous fabrics.

3.2. Fabric Topology Model

The fabric topology model module is designed to be generic and can be
applied to data center fabrics built with different technologies,
such as VLAN, VXLAN etc al. The main purpose of this module is to
configure and manage fabrics and their connections. provide a fabric-
based topology view for data center network applications.

3.2.1. Fabric Topology

In the fabric topology module, a fabric is modeled as a node in the
network, while the fabric-based Data center network consists of a set
of fabric nodes and their connections known as "fabric port". The
following is the snatch of the definition to show the main structure
of the model:

```
     module: ietf-fabric-topology
     augment /nw:networks/nw:network/nw:network-types:
        +--rw fabric-network!
     augment /nw:networks/nw:network/nw:node:
        +--rw fabric-attribute
              +--rw name?           string
              +--rw type?           fabrictype:underlayer-network-type
              +--rw description?    string
              +--rw options
              +--...
     augment /nw:networks/nw:network/nw:node/nt:termination-point:
        +--ro fport-attribute
              +--ro name?           string
              +--ro role?           fabric-port-role
              +--ro type?           fabric-port-type
```

The fabric topology module augments the generic ietf-network and ietf-network-topology modules as follows:

o A new topology type "ietf-fabric-topology" is introduced and added under the "network-types" container of the ietf-network module.

o Fabric is defined as a node under the network/node container. A new container of "fabric-attribute" is defined to carry attributes for a fabric network such as gateway mode, fabric types, involved device nodes and links etc al.

o Termination points (in network topology module) are augmented with fabric port attributes defined in a container. The "termination-point" here can represent the "port" of a fabric that provides connections to other nodes, such as device internally, another fabric externally and also end hosts.

Details of fabric node and fabric termination point extension will be explained in the following sections.

3.2.2. Fabric node extension

As a network, a fabric itself is composed of set of network elements i.e. devices, and related links. As stated previously, the configuration of a fabric is contained under the "fabric-attribute" container depicted as follows:

```
+--rw fabric-attribute
   +--rw fabric-id?      fabric-id
   +--rw name?           string
   +--rw type?           fabrictype:underlayer-network-type
   +--rw vni-capacity
   |  +--rw min?    int32
   |  +--rw max?    int32
   +--rw description?    string
   +--rw options
   |  +--rw gateway-mode?          enumeration
   |  +--rw traffic-behavior?      enumeration
   |  +--rw capability-supported*   fabrictype:service-capabilities
   +--rw device-nodes* [device-ref]
   |  +--rw device-ref    fabrictype:node-ref
   |  +--rw role?         fabrictype:device-role
   +--rw device-links* [link-ref]
   |  +--rw link-ref    fabrictype:link-ref
   +--rw device-ports* [port-ref]
      +--rw port-ref    fabrictype:tp-ref
      +--rw port-type?   enumeration
      +--rw bandwith?    Enumeration
```

As in the module, additional data objects for nodes are introduced by augmenting the "node" list of the network module. New objects include fabric name, type of the fabric, descriptions of the fabric as well as a set of options defined in an "options" container. The options container includes type of the gateway-mode (centralized or distributed) and traffic-behavior (whether acl needed for the traffic).

Also, it defines a list of device-nodes and related links as supporting-nodes to form a fabric network. These device nodes and links are leaf-ref of existing nodes and links in the physical topology. For the device-node, the "role" object is defined to represents the role of the device within the fabric, such as "SPINE" or "LEAF", which should work together with gateway-mode.

3.2.3. Fabric termination-point extension

Since the fabric is considered as a node, in this concept, "termination-points" can represent "ports" of a fabric that connects to other fabrics or end hosts, besides representing ports that connect devices inside the fabric itself.

As such, the "termination-point" in the fabric topology has three roles, including internal TP that connects to devices within a

   fabric, external TP that connects to outside network, as well as
   access TP to end hosts.

   A set of "termination-point" indicates all connections of a fabric
   including its internal connections, interconnections with other
   fabrics and also connections to end hosts for a DC network.

   The structure of fabric ports is as follows:

```
augment /nw:networks/nw:network/nw:node/nt:termination-point:
   +--ro fport-attribute
      +--ro name?         string
      +--ro role?         fabric-port-role
      +--ro type?         fabric-port-type
      +--ro device-port?  tp-ref
      +--ro (tunnel-option)?
         +--:(gre)
            +--ro src-ip?        inet:ip-prefix
            +--ro dest-ip?       inet:ip-address
```

   It augments the termination points (in network topology module) with
   fabric port attributes defined in a container.

   New nodes are defined for fabric ports which include name, role of
   the port within the fabric (internal port, external port to outside
   network, access port to end hosts), port type (l2 interface, l3
   interface etc al). By using the device-port defined as a tp-ref,
   this fabric port can be mapped to a device node in the underlay
   network.

   Also, a new container for tunnel-options is introduced as well to
   present the tunnel configuration on the port.

   The terminiation points information are all learnt from the underlay
   networks but not configured by the fabric topology layer.

4. Fabric YANG Module

```
<CODE BEGINS> file "ietf-fabric-types@2016-09-29.yang"
module ietf-fabric-types {

   yang-version 1.1;
   namespace "urn:ietf:params:xml:ns:yang:ietf-fabric-types";
   prefix fabrictypes;

   import ietf-inet-types { prefix "inet"; revision-date "2013-07-15"; }
```

```
        import ietf-network-topology { prefix nt;  }

        organization
    "IETF I2RS (Interface to the Routing System) Working Group";

        contact
    "WG Web:    <http://tools.ietf.org/wg/i2rs/ >
     WG List:   <mailto:i2rs@ietf.org>

     WG Chair:  Susan Hares
                <mailto:shares@ndzh.com>

     WG Chair:  Russ White
                <mailto:russ@riw.us>

     Editor:    Yan Zhuang
                <mailto:zhuangyan.zhuang@huawei.com>

     Editor:    Danian Shi
                <mailto:shidanian@huawei.com>";

    description
        "This module contains a collection of YANG definitions for Fabric.";

    revision "2016-09-29" {
        description
            "Initial revision of faas.";
                reference
                        "draft-zhuang-i2rs-yang-dc-fabric-network-topology-02";
    }

    identity fabric-type {
                description
                        "base type for fabric networks";
    }

    identity vxlan-fabric {
        base fabric-type;
                description
                        "vxlan fabric";
    }

    identity vlan-fabric {
        base fabric-type;
                description
                        "vlan fabric";
    }
```

```
    typedef service-capabilities {
        type enumeration {
            enum ip-mapping {
                            description "NAT";
                    }
            enum acl-redirect{
                            description "acl redirect, which can provide
SFC function";
                    }
            enum dynamic-route-exchange{
                            description "dynamic route exchange";
                    }
        }
                description
                        "capability of the device";
    }

    /*
     * Typedefs
     */
    typedef node-ref {
        type instance-identifier;
        description "A reference to a node in topology";
    }

    typedef tp-ref {
        type instance-identifier;
        description "A reference to a termination point in topology";
    }

    typedef link-ref {
                type instance-identifier;
        description "A reference to a link in topology";
    }

    typedef device-role {
        type enumeration {
            enum SPINE {
                description "a spine node";
            }
            enum LEAF {
                description "a leaf node";
            }
            enum BORDER {
                description "a border node";
            }
        }
        default "LEAF";
```

```
      description "device role type";
```

```
     }

     typedef fabric-port-role {
         type enumeration {
             enum internal {
                 description "the port used for devices to access each other.";
             }
             enum external {
                 description "the port used for fabric to access outside
network.";
             }
             enum access {
                 description "the port used for Endpoint to access fabric.";
             }
             enum reserved {
                 description " not decided yet. ";
             }
         }
             description "the role of the physical port ";
     }

     typedef fabric-port-type {
         type enumeration {
             enum layer2interface {
                             description "l2 if";
             }
             enum layer3interface {
                             description "l3 if";
             }
             enum layer2Tunnel {
                             description "l2 tunnel";
             }
             enum layer3Tunnel {
                             description "l3 tunnel";
             }
         }
             description
                 "fabric port type";
     }

     typedef underlayer-network-type {
         type enumeration {
             enum VXLAN {
                 description "vxlan";
             }
             enum TRILL {
                 description "trill";
             }
```

```
enum VLAN {
```

```
                    description "vlan";
            }
        }
                description "";
    }

    typedef layer2-protocol-type-enum {
        type enumeration {
            enum VLAN{
                               description "vlan";
                    }
            enum VXLAN{
                               description "vxlan";
                    }
            enum TRILL{
                               description "trill";
                    }
            enum NvGRE{
                               description "nvgre";
                    }
        }
                description "";
    }

    typedef access-type {
        type enumeration {
            enum exclusive{
                               description "exclusive";
                    }
            enum vlan{
                               description "vlan";
                    }
        }
        description "";
    }

    grouping fabric-port {
                description
                        "attributes of a fabric port";
        leaf name {
            type string;
                        description "name of the port";
        }
        leaf role {
            type fabric-port-role;
                        description "role of the port in a fabric";
        }
        leaf type {
```

```
            type fabric-port-type;
                    description "type of the port";
        }
        leaf device-port {
            type tp-ref;
                    description "the device port it mapped to";
        }
        choice tunnel-option {
                    description "tunnel options";

            case gre {
                leaf src-ip {
                   type inet:ip-prefix;
                                      description "source address";
                }
                leaf dest-ip {
                   type inet:ip-address;
                                      description "destination address";
                }
            }
        }
    }

    grouping route-group {
            description
                    "route attributes";
        list route {
            key "destination-prefix";
                    description "route list";

            leaf description {
                type string;
                description "Textual description of the route.";
            }
            leaf destination-prefix {
                type inet:ipv4-prefix;
                mandatory true;
                description "IPv4 destination prefix.";
            }
            choice next-hop-options {
                            description "choice of next hop options";
                case simple-next-hop {
                    leaf next-hop {
                        type inet:ipv4-address;
                        description "IPv4 address of the next hop.";
                    }
                    leaf outgoing-interface {
                        type nt:tp-id;
```

```
                        description "Name of the outgoing interface.";
                }
            }
        }
    }
}

    grouping port-functions {
              description
                      "port functions";

        container port-function {
                      description "port functions";
            choice function-type {
                              description "type of functions";
                case ip-mapping {
                    list ip-mapping-entry {
                                             key "external-ip";
                                             description "list of NAT
entry";
                        leaf external-ip {
                            type inet:ipv4-address;
                                               description "external
address";
                        }
                        leaf internal-ip {
                            type inet:ipv4-address;
                                               description "internal
address";
                        }
                    }
                }
            }
        }
    }
    grouping acl-list {
              description "acl list";
        list fabric-acl {
                      key fabric-acl-name;
                      description "fabric acl list";
            leaf fabric-acl-name {
                type string;
                              description "acl name";
            }
        }
    }
}
<CODE ENDS>
```

```
<CODE BEGINS> file "ietf-fabric-topology@2017-11-21.yang"
module ietf-fabric-topology {
```

```
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-fabric-topology";
    prefix fabric;

        import ietf-network { prefix nw;  }
        import ietf-network-topology { prefix nt;  }
    import ietf-fabric-types { prefix fabrictype; revision-date "2016-09-29"; }

        organization
    "IETF I2RS (Interface to the Routing System) Working Group";

        contact
    "WG Web:    <http://tools.ietf.org/wg/i2rs/ >
     WG List:   <mailto:i2rs@ietf.org>

      WG Chair:  Susan Hares
                 <mailto:shares@ndzh.com>

      WG Chair:  Russ White
                 <mailto:russ@riw.us>

       Editor:    Yan Zhuang
                  <mailto:zhuangyan.zhuang@huawei.com>

       Editor:    Danian Shi
                  <mailto:shidanian@huawei.com>";


    description
        "This module contains a collection of YANG definitions for Fabric.

                Copyright (c) 2016 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
                draft-zhuang-i2rs-yang-dc-fabric-network-topology;
                see the RFC itself for full legal notices.";

        revision "2017-11-21"{
                description
                        "fix warnings.";
```

```
                    reference
                         "draft-ietf-i2rs-yang-dc-fabric-network-topology-01";
        }

    revision "2017-06-29"{
                description
                        "update to NMDA compliant format";
                reference
                         "draft-zhuang-i2rs-yang-dc-fabric-network-topology-04";
        }

    revision "2017-03-10" {
                description
                        "remove the rpcs and add extra attributes";
                reference
                         "draft-zhuang-i2rs-yang-dc-fabric-network-topology-03";
        }
    revision "2016-09-29" {
        description
            "Initial revision of fabric topology.";
                reference
                         "draft-zhuang-i2rs-yang-dc-fabric-network-topology-02";
    }

    identity fabric-context {
                description
                        "identity of fabric context";
    }

    typedef fabric-id {
        type nw:node-id;
        description
            "An identifier for a fabric in a topology.
            The identifier is generated by compose-fabric RPC.";
    }

      //grouping statements
      grouping fabric-network-type {
    description "Identify the topology type to be fabric.";
    container fabric-network {
      presence "indicates fabric Network";
      description
      "The presence of the container node indicates
       fabric Topology";
    }
    }

    grouping fabric-options {
```

```
                 description "options for a fabric";

        leaf gateway-mode {
            type enumeration {
                enum centralized {
                    description "centerilized gateway";
                }
                enum distributed {
                    description "distributed gateway";
                }
            }
            default "distributed";
                       description "gateway mode";
        }

        leaf traffic-behavior {
            type enumeration {
                enum normal {
                    description "normal";
                }
                enum policy-driven {
                    description "policy driven";
                }
            }
            default "normal";
                       description "traffic behavior of the fabric";
        }

        leaf-list capability-supported {
            type fabrictype:service-capabilities;
                       description
                               "supported services of the fabric";
        }
    }

    grouping device-attributes {
                description "device attributes";
        leaf device-ref {
            type fabrictype:node-ref;
                       description
                               "the device it includes to";
        }
        leaf role {
            type fabrictype:device-role;
            default "LEAF";
                       description
                               "role of the node";
        }
```

```
    }

    grouping link-attributes {
            description "link attributes";
        leaf link-ref {
            type fabrictype:link-ref;
                    description
                            "the link it includes";
        }
    }

        grouping port-attributes {
                description "port attributes";
        leaf port-ref {
            type fabrictype:tp-ref;
                    description
                            "port reference";
        }
        leaf port-type {
            type enumeration {
                enum ETH {
                                    description "ETH";
                            }
                enum SERIAL {
                                    description "Serial";
                            }
            }
                    description
                            "port type: ethernet or serial";
        }
        leaf bandwith {
            type enumeration {
                enum 1G {
                                    description "1G";
                            }
                enum 10G {
                                    description "10G";
                            }
                enum 40G {
                                    description "40G";
                            }
                enum 100G {
                                    description "100G";
                            }
                enum 10M {
                                    description "10M";
                            }
                enum 100M {
```

```
                                     description "100M";
                               }
                 enum 1M {
                                     description "1M";
                               }
           }
                     description
                               "bandwidth on the port";
       }
     }

     grouping fabric-attributes {
             description "attributes of a fabric";

             leaf fabric-id {
             type fabric-id;
                               description
                                       "fabric id";
       }

       leaf name {
           type string;
                     description
                               "name of the fabric";
       }

       leaf type {
           type fabrictype:underlayer-network-type;
           description
                               "The type of physical network that implements
this fabric.Examples are vlan, and trill.";
       }

             container vni-capacity {
           description "number of vnis the fabric has";
           leaf min {
               type int32;
                               description
                                       "vni min capacity";
           }

           leaf max {
               type int32;
                               description
                                       "vni max capacity";
           }
       }
```

```
   leaf description {
```

```
            type string;
                      description
                            "description of the fabric";
      }

      container options {
                      description "options of the fabric";
          uses fabric-options;
      }

      list device-nodes {
                      key device-ref;
                      description "include device nodes in the fabric";
          uses device-attributes;
      }

      list device-links {
                      key link-ref;
                      description "include device links within the fabric";
          uses link-attributes;
      }

            list device-ports {
          key port-ref;
                      description "include device ports within the fabric";
          uses port-attributes;
      }

  }

      // augment statements

      augment "/nw:networks/nw:network/nw:network-types" {
    description
      "Introduce new network type for Fabric-based logical topology";

            uses fabric-network-type;
      }

  augment "/nw:networks/nw:network/nw:node" {
      when "/nw:networks/nw:network/nw:network-types/fabric:fabric-network" {
      description
        "Augmentation parameters apply only for networks
         with fabric topology";
    }
      description "Augmentation for fabric nodes created by faas.";

      container fabric-attribute {
```

```
                        description
                        "attributes for a fabric network";

            uses fabric-attributes;
        }
    }

    augment "/nw:networks/nw:network/nw:node/nt:termination-point" {
        when "/nw:networks/nw:network/nw:network-types/fabric:fabric-network" {
        description
          "Augmentation parameters apply only for networks
           with fabric topology";
      }
        description "Augmentation for port on fabric.";

        container fport-attribute {
            config false;
                        description
                        "attributes for fabric ports";
            uses fabrictype:fabric-port;
        }
    }
}
```
<CODE ENDS>

5. Security Consideration

   The underlay topology is learnt from the physical network, while the
   fabric topology is composed of a collection of the underlay
   nodes. The fabric model doesn't change the connections of underlay
   networks. While there are data nodes for fabric configuration, these
   data nodes may be considered sensitive or vulnerable in some network
   enviroments. Write operations (e.g., edit-config) to these data
   nodes without proper protection can have a negative effect on network
   operations. For example, misconfiguration of underlay nodes to a
   fabric node may lead to inproper activities when management are
   implemented at fabric layer by customers.

6. Acknowledgements

   We wish to acknowledge the helpful contributions, comments, and
   suggestions that were received from Alexander Clemm, Xufeng Liu,
   Susan Hares, Wei Song, Luis M. Contreras and Benoit Claise.

7. References

7.1. Normative References

   [I-D.draft-ietf-i2rs-yang-l3-topology]
            Clemm, A., Medved, J., Tkacik, T., Liu, X., Bryskin, I.,
            Guo, A., Ananthakrishnan, H., Bahadur, N., and V. Beeram,
            "A YANG Data Model for Layer 3 Topologies", I-D draft-
            ietf-i2rs-yang-l3-topology-04, September 2016.

   [I-D.draft-ietf-i2rs-yang-network-topo]
            Clemm, A., Medved, J., Tkacik, T., Varga, R., Bahadur, N.,
            and H. Ananthakrishnan, "A YANG Data Model for Network
            Topologies", I-D draft-ietf-i2rs-yang-network-topo-06,
            September 2016.

   [I-D.draft-ietf-nvo3-vxlan-gpe]
            Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol
            Extension for VXLAN", I-D draft-ietf-i2rs-yang-network-
            topo-02, 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., "YANG - A Data Modeling Language for the
            Network Configuration Protocol (NETCONF)", RFC 6020,
            October 2010.

   [RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991,
            July 2013.

   [RFC7950] Bjorklund, M., "The YANG 1.1 Data Modeling Language",
            RFC 7950, DOI 10.17487/RFC7950, Auguest 2016.

7.2. Informative References

   [I-D.draft-ietf-i2rs-usecase-reqs-summary]
            Hares, S. and M. Chen, "Summary of I2RS Use Case
            Requirements", I-D draft-ietf-i2rs-usecase-reqs-summary-
            01, May 2015.

Appendix A. Non NMDA -state modules

      <CODE BEGINS> file "ietf-fabric-topology-state@2017-11-21.yang"
      module ietf-fabric-topology-state {

```
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-fabric-topology-state";
    prefix sfabric;

        import ietf-network-state { prefix nws;   }
    import ietf-fabric-types { prefix fabrictype; revision-date "2016-09-29"; }
        import ietf-fabric-topology {prefix fabric;}
        organization
    "IETF I2RS (Interface to the Routing System) Working Group";

        contact
        "WG Web:    <http://tools.ietf.org/wg/i2rs/ >
     WG List:   <mailto:i2rs@ietf.org>

     WG Chair:  Susan Hares
                <mailto:shares@ndzh.com>

     WG Chair:  Russ White
                <mailto:russ@riw.us>

      Editor:   Yan Zhuang
                <mailto:zhuangyan.zhuang@huawei.com>

      Editor:   Danian Shi
                <mailto:shidanian@huawei.com>";


    description
        "This module contains a collection of YANG definitions for Fabric
topology state for non NMDA.

                Copyright (c) 2016 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
                draft-zhuang-i2rs-yang-dc-fabric-network-topology;
                see the RFC itself for full legal notices.";

        revision "2017-11-21"{
                description
                        "fix warnings.";
                reference
```

```
                            "draft-ietf-i2rs-yang-dc-fabric-network-topology-01";
        }

    revision "2017-06-29"{
                description
                        "update to NMDA compliant format";
                reference
                        "draft-zhuang-i2rs-yang-dc-fabric-network-topology-04";
        }

      //grouping statements
      grouping fabric-network-type {
    description "Identify the topology type to be fabric.";
    container fabric-network {
      presence "indicates fabric Network";
      description
      "The presence of the container node indicates
       fabric Topology";
    }
  }

    grouping fabric-options {
                description "options for a fabric";

        leaf gateway-mode {
            type enumeration {
                enum centralized {
                    description "centerilized gateway";
                }
                enum distributed {
                    description "distributed gateway";
                }
            }
            default "distributed";
                    description "gateway mode";
        }

        leaf traffic-behavior {
            type enumeration {
                enum normal {
                    description "normal";
                }
                enum policy-driven {
                    description "policy driven";
                }
            }
            default "normal";
                    description "traffic behavior of the fabric";
```

```
        }

        leaf-list capability-supported {
            type fabrictype:service-capabilities;
                        description
                                "supported services of the fabric";
        }
    }

    grouping device-attributes {
                description "device attributes";
        leaf device-ref {
            type fabrictype:node-ref;
                        description
                                "the device it includes to";
        }
        leaf role {
            type fabrictype:device-role;
            default "LEAF";
                        description
                                "role of the node";
        }
    }

    grouping link-attributes {
                description "link attributes";
        leaf link-ref {
            type fabrictype:link-ref;
                        description
                                "the link it includes";
        }
    }

        grouping port-attributes {
                description "port attributes";
        leaf port-ref {
            type fabrictype:tp-ref;
                        description
                                "port reference";
        }
        leaf port-type {
            type enumeration {
                enum ETH {
                                        description "ETH";
                                }
                enum SERIAL {
                                        description "Serial";
                                }
```

```
            }
                    description
                          "port type: ethernet or serial";
        }
        leaf bandwith {
            type enumeration {
                enum 1G {
                                description "1G";
                        }
                enum 10G {
                                description "10G";
                        }
                enum 40G {
                                description "40G";
                        }
                enum 100G {
                                description "100G";
                        }
                enum 10M {
                                description "10M";
                        }
                enum 100M {
                                description "100M";
                        }
                enum 1M {
                                description "1M";
                        }
            }
                    description
                          "bandwidth on the port";
        }
    }

    grouping fabric-attributes {
            description "attributes of a fabric";

            leaf fabric-id {
            type fabric:fabric-id;
                            description
                                    "fabric id";
        }

        leaf name {
            type string;
                    description
                          "name of the fabric";
        }
```

```
        leaf type {
            type fabrictype:underlayer-network-type;
            description
                              "The type of physical network that implements
this fabric.Examples are vlan, and trill.";
        }

            container vni-capacity {
            description "number of vnis the fabric has";
            leaf min {
                type int32;
                              description
                                    "vni min capacity";
            }

            leaf max {
                type int32;
                              description
                                    "vni max capacity";
            }
        }

        leaf description {
            type string;
                    description
                              "description of the fabric";
        }

        container options {
                    description "options of the fabric";
            uses fabric-options;
        }

        list device-nodes {
                    key device-ref;
                    description "include device nodes in the fabric";
            uses device-attributes;
        }

        list device-links {
                    key link-ref;
                    description "include device links within the fabric";
            uses link-attributes;
        }

            list device-ports {
            key port-ref;
                    description "include device ports within the fabric";
```

```
    uses port-attributes;
```

```
            }

     }

         // augment statements

         augment "/nws:networks/nws:network/nws:network-types" {
     description
       "Introduce new network type for Fabric-based logical topology";

                 uses fabric-network-type;
         }

     augment "/nws:networks/nws:network/nws:node" {
         when "/nws:networks/nws:network/nws:network-types/sfabric:fabric-
network" {
         description
           "Augmentation parameters apply only for networks
            with fabric topology.";
       }
         description "Augmentation for fabric nodes.";

         container fabric-attribute-state {
                        config false;
                        description
                        "attributes for a fabric network";

            uses fabric-attributes;
         }
     }
}

                 <CODE ENDS>
```

Authors' Addresses

   Yan Zhuang
   Huawei
   101 Software Avenue, Yuhua District
   Nanjing, Jiangsu 210012
   China

   Email: zhuangyan.zhuang@huawei.com

Danian Shi
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: shidanian@huawei.com

Rong Gu
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing, Beijing 100053
China

Email: gurong_cmcc@outlook.com

Hariharan Ananthakrishnan
Packet Design

Email: hari@packetdesign.com