YANG Data Model for Fabric Service delivery in Data Center Network
draft-zhuang-i2rs-dc-fabric-service-model-00

Abstract

This document defines a YANG data model that can be used to deliver fabric service for data center networks. This model is intended to be instantiated by management system to deploy the overall fabric service for a data center network. It provides an abstraction of fabric services, including configurations and functions, however is not a configuration model be used directly onto network infrastructures. It should be used combined with such as fabric topology data model defined in [I-D.zhuang-i2rs-yang-dc-fabric-network-topology] with specific topology information to configure the network elements and deliver the service to data center infrastructures.

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 http://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 May 4, 2017.

Copyright Notice

Copyright (c) 2016 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 (http://trustee.ietf.org/license-info) in effect on the date of
1. Introduction

Currently, network service provisioning is coupled with specific network topology and technologies applied, which is technology oriented and more device oriented, that makes the network management and application deployment for users complex and lack of flexibility.

In the area of data center networks, the situation is even worse due to massive network devices involved and various applications developed and deployed by users also known as tenants in a more dynamical way. As such, lack of network abstraction and tightly coupled networks in Data center network makes it hard to deploy high level application centric service requests to physical network infrastructures agilely and flexibly.

With existing network management concept, to deliver services from the upper layer users dynamically over the date center network, the network administrator has to configure and operate all involved
devices for each tenant with knowledge of the entire network topology, resources and implemented technologies.

To make the network service provisioning more flexible and easy for network administrators, a bottom-up abstraction of network topology in an application centric perspective is introduced, so as to simplify the network service deployment and management of Data Center Network for tenants.

With this purpose, this document defines a YANG data model for the fabric service by using YANG [RFC6020] [RFC7950] to provide a logical fabric network to define and deploy upper layer network services regardless of the specific topology information and underlay/overlay technologies and involved devices. The configuration and operation onto physical infrastructure will be further processed by the topology manager with topology information and technology information.

This model defines the generic configuration for fabric network services regardless of specific fabric topology information. Besides, this model also provides a set of rpc command for operation. In general, this model can be used to expose the logic network to a network orchestrator to define and deploy requested services for user applications regardless of details of topology and technology information within the DC network, while leaving the topology and technology information and management to other components such as a controller.

The data model includes two main modules:

(a) Module "ietf-fabric-service" defines a module for a logical fabric network from the application centric view for user network service. To do so, it augments general network topology model defined in [I-D.ietf-i2rs-yang-network-topo] with logical components such as logical switches, logical routers as well as logical ports to carry network services requested by upper layer users.

(b) Module "ietf-fabric-endpoint" defines a module for endpoints, which represents hosts that runs applications and generating traffics. The major point of this module is to indicate the attachment points of the endpoint in a logical network as well as in a physical fabric network, so as to build bindings between layers.

Besides, the model "ietf-fabric-topology" defined in [I-D.zhuang-i2rs-yang-dc-fabric-network-topology] with topology and resource as well as technology information is used to work together to implement configurations and operations from the logical fabric network onto the specific fabric infrastructure.
2. 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].

3. Application centric abstraction for DC fabric networks

To make the network service more application centric and easy for network administrator to deploy and manage services for tenants applications, this section describes an application centric network abstraction for DC fabric networks, according to which the network can be abstracted to a logical network for upper layer services.

3.1. Logical fabric network architecture

From the application centric point of view, hosts of tenants can be considered to connect with other hosts through a switch if they are L2 reachable, alternatively, connect through a router if they are L3 reachable.

As such, a tenant network can be abstracted to a logical network in which logical switches connecting hosts that are L2 reachable and a logical router connecting switches which represents L3 reachable regardless of the detailed topology of involved devices and technology information. The diagram for an abstracted network for a tenant is shown below.
As a very initial effort to abstract services for fabric-based DC networks, this draft will focus on the abstraction of network connections which is the basis function for a network, while leaving the other network appliance aside.

3.2. Logical Network components

Within a logical fabric network, there are four major components as depicted in figure 1:

Logical Switch:

Works as a switch within a logical fabric network to provide L2 connections between endpoints or to a router or to external networks. It can be bounded to one or several physical switches.

Logical Router:

Works as a router within a logical fabric network to provide L3 connections between switches or to external networks. It can be bounded to one or several physical routers in the physical networks.

Logical Port:
Provides port function on logical switches or logical routers to connect to other components or external networks.

Endpoint:

Represents tenant hosts which can be a VM for example.

3.3. Connections between the components

There are 5 connections for a logical network. Functions of these connections are as follows:

C1: Endpoint attachment. It is used by an endpoint to connect to a logical switch.

C2: L2 to L3 attachment. Interface between a logical switch and a logical router.

C3: L3 interconnection which connects to a logical router.

C4: L2 interconnection which connects to a logical switch.

C5: Route population.

4. Fabric service model usage

4.1. Usage architecture

According to the idea of abstracted fabric network introduced in section 3, a fabric service model is provided to define tenant/user network services in a more concentrated and intuitive way but leaving details of specific network topology, infrastructures and implemented technologies to the topology layer.

With the logical network information from the orchestrator as well topology information, a topology manager will further parse network services requested by logical networks onto configuration/operations of network elements in the physical layer and distribute these configuration and operations onto involved devices to complete the whole process as shown in Fig.2.
According to the topology architecture stated in [I-D.draft-ietf-i2rs-usecase-reqs-summary], the idea of fabric service model is used to provide network services for applications over a data center network, maybe by an orchestration manager for example. By mapping between elements from layers, the topology manager further deploys requested configurations and operations for applications over the physical fabric infrastructures. This mapping is dynamically generated by hosts of tenants defined as endpoints when being online.

4.2. Multi-Layer relationship

There are three layers in this usage as shown in Fig.3.

The fabric service model is abstracted from fabric topology used as an application-centric interface at the service layer to define tenant networks. It is more focused on the network services.
regardless of specific network types, topologies, technologies etc al.

While the fabric layer collects and maintains the fabric topology information (including single fabric and multi-fabric connections) and specific technologies for each fabric. According to logical network services and fabric topology information, this layer will configure and operate involved network devices in the physical layer to implement the tenant network services.

The physical network layer is at the bottom and consists of all network devices.
Logical Network Topology

+--------+
|   LR   |
+--------+

+-+Endpoint 1 +-              +-----------+
|   /      \               |Endpoint 2 ++
| +----\----+
|    +-----+
|     +----+
|      +---+
|       +--+
|        ++-
|         | |  LSW1 |  LSW2
|         +----+
|               +-----+
|               |     |
|               |     |  Service Layer
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
|               |     |
+-----------------+  +-----------------+

Figure 3: Layer Relationship
The mapping of nodes within these layers is realized by endpoints. An endpoint is instantiated by the orchestrator to indicate the locations of a host both in the logical layer as well as in the physical layer, so as to connect network elements of these three layers and deliver services requested from the logical layer onto the physical infrastructure in a dynamic manner.

5. Design of the data model

5.1. Fabric service module

As explained in previous, network service for tenant network can be abstracted to sets of logical switches, logical routers and logical ports. Upon these logical elements, acl policies and gateway functions can be attached.

The fabric service module is defined by YANG module "ietf-fabric-service". The module is depicted in the following diagram.

```yang
module: ietf-fabric-service
  augment /nw:networks/nw:network/nw:node:
    +--ro lsw-attribute
        +--ro lsw-uuid? yang:uuid
        +--ro name? string
        +--ro segment-id? uint32
        +--ro network? inet:ip-prefix
        +--ro external? boolean
        +--ro fabric-acl* [fabric-acl-name]
            +--ro fabric-acl-name string
    augment /nw:networks/nw:network/nw:node:
      +--ro lr-attribute
        +--ro lr-uuid? yang:uuid
        +--ro name? string
        +--ro vrf-ctx? uint32
        +--ro fabric-acl* [fabric-acl-name]
            +--ro fabric-acl-name string
        +--ro routes
            +--ro route* [destination-prefix]
                +--ro description? string
                +--ro destination-prefix inet:ipv4-prefix
                +--ro (next-hop-options)?
                    +--:(simple-next-hop)
                        +--ro next-hop? inet:ipv4-address
                        +--ro outgoing-interface? nt:tp-id
    augment /nw:networks/nw:network/nw:node/nt:termination-point:
      +--ro lport-attribute
        +--ro lport-uuid? yang:uuid
        +--ro name? string
```
To provide a logical network topology for DC fabric network, the module augments the original ietf-network and ietf-network-topology modules:

- New nodes for logical switch and logical router with additional data objects are introduced by augmenting the "node" list of the network module.

- Termination points for logical ports are augmented with logical port information and its reference to termination ports in the underlay topologies. As stated in section 3, the logical port may act as an access port which will be bounded to some physical port, or else it may be as a service point which connects to internal gateway or external gateway. Besides, it can also be attached with ACL rules.

In addition to the configuration part, the module defines several rpc commands to create and remove logical elements of a logical network, as well as to add and delete functions and policies such as gateway and acl on these logical network elements as shown below.

rpcs:
  +++-x create-logical-switch
Internet-Draft   YANG for Fabric Service delivery in DC     October 2016

| +---w input  
|    | +---w name?        string               
|    | +---w vni?         int32                
|    | +---w external?    boolean              
| +--ro output 
|    | +---ro node-id?   nw:node-id           
|    | +---ro lsw-uuid?   yang:uuid            
|    | +---ro name?       string               
|    | +---ro external?   boolean              
|    | +---ro fabric-acl* [fabric-acl-name]   
|         |    | +---ro fabric-acl-name   string        
| +---x rm-logical-switch 
| +---w input 
|    | +---w node-id?     nw:node-id          
| +---x create-logical-router 
| +---w input 
|    | +---w name?        string               
| +--ro output 
|    | +---ro node-id?    nw:node-id          
|    | +---ro lr-uuid?    yang:uuid            
|    | +---ro name?       string               
|    | +---ro vrf-ctx?    uint32               
|    | +---ro fabric-acl* [fabric-acl-name]   
|    | | +---ro fabric-acl-name    string       
|    | +---ro routes 
|    |    | +---ro route* [destination-prefix]    
|    |    |    | +---ro description?   string          
|    |    |    | +---ro destination-prefix inet-ipv4-prefix 
|    |    |    | +---ro (next-hop-options)?          
|    |    |    | | +---ro simple-next-hop         
|    |    |    | |    | +---ro next-hop? inet-ipv4-address 
|    |    |    | |    | +---ro outgoing-interface? nt:tp-id 
| +---x rm-logical-router 
| +---w input 
|    | +---w node-id?     nw:node-id          
| +---x add-static-route 
| +---w input 
|    | +---w node-id?     nw:node-id          
|    | +---w route* [destination-prefix]    
|    |    | +---w description?   string          
|    |    | +---w destination-prefix inet-ipv4-prefix 

Zhuang & Shi               Expires May 4, 2017                 [Page 12]
|        +---w (next-hop-options)?         
|           +--:(simple-next-hop)         
|              +---w next-hop?             inet:ipv4-address
|              +---w outgoing-interface?   nt:tp-id
|          +---x rm-static-route         
|              +---w input               
|                  +---w node-id?          nw:node-id
|                  +---w destination-prefix* inet:ipv4-prefix
|          +---x clear-static-route       
|              +---w input               
|                  +---w node-id?          nw:node-id
|          +---x create-logical-port      
|              +---w input               
|                  +---w name?             string
|                  +---w logical-device?    nw:node-id
|                  +---w attribute        
|                      +---w lport-uuid?         yang:uuid
|                      +---w name?             string
|                      +---w port-layer     
|                          +---w layer-1-info
|                              |  +---w location?   nt:tp-id
|                          +---w layer-2-info
|                              |  +---w access-type?      access-type
|                          +---w layer-3-info
|                              |  +---w ip?               inet:ip-address
|                              |  +---w network?          inet:ip-prefix
|                              |  +---w mac?              yang:mac-address
|                              |  +---w forward-enable?    boolean
|                              |  +---w logical-switch?    nw:node-id
|                      +---w fabric-acl* [fabric-acl-name]
|                          +---w fabric-acl-name string
|                      +---w port-function
|                          +---w (function-type)? 
|                              +--:(ip-mapping)
|                              |  +---w ip-mapping-entry* [external-ip]
|                              |      +---w external-ip inet:ipv4-address
|                              |      +---w internal-ip? inet:ipv4-address
|                              +---w underlayer-ports* [port-ref]
|                          +---w port-ref instance-identifier
|          +--ro output                 
|              +--ro tp-id?             nt:tp-id
|              +--ro lport-uuid?         yang:uuid
|              +--ro name?             string
|              +--ro port-layer
|     |     +--ro logical-switch? nw:node-id
|     +--ro fabric-acl* [fabric-acl-name]
|     |  +--ro fabric-acl-name string
|     +--ro port-function
|     |  +--ro (function-type)?
|     |  |  +--:(ip-mapping)
|     |  |  |  +--ro ip-mapping-entry* [external-ip]
|     |  |  |  |  +--ro external-ip inet:ipv4-address
|     |  |  +--ro internal-ip? inet:ipv4-address
|     +--ro underlayer-ports* [port-ref]
|     |  +--ro port-ref instance-identifier

++++x rm-gateway
|     +---w input
|     |  +---w ip-address? inet:ipv4-address
|     |  +---w logical-router? nw:node-id

++++x port-binding-logical-to-fabric
|     +---w input
|     |  +---w logical-device? nw:node-id
|     |  +---w logical-port? nt:tp-id
|     +---w fabric-port? nt:tp-id

++++x port-binding-logical-to-device
|     +---w input
|     |  +---w logical-device? nw:node-id
|     |  +---w logical-port? nt:tp-id
|     +---w physical-port? fabrictype:tp-ref

++++x add-port-function
|     +---w input
|     |  +---w logical-device? nw:node-id
|     |  +---w logical-port? nt:tp-id
|     +---w port-function
|     |  |  +---w (function-type)?
|     |  |  |  +--:(ip-mapping)
|     |  |  |  |  +--w ip-mapping-entry* [external-ip]
|     |  |  |  |  |  +--w external-ip inet:ipv4-address
|     |  |  |  |  +--w internal-ip? inet:ipv4-address

++++x add-acl
|     +---w input
|     |  +---w logical-device? nw:node-id
|     |  +---w logical-port? nt:tp-id
|     |  +---w acl-name? string

++++x del-acl
|     +---w input
5.2. Endpoint module

To represent user attachment points and map logical fabric configurations and operations of applications onto the physical fabric infrastructure, an endpoint is instantiated to represent a host of a user that runs applications.

The fabric endpoint module is defined by YANG module "ietf-fabric-endpoint". The module is depicted as follows:

```
module: ietf-fabric-endpoint
  +--ro endpoints
    +--ro endpoint* [endpoint-uuid]
      +--ro endpoint-uuid       yang:uuid
      +--ro mac-address?        yang:mac-address
      +--ro ip-address?         inet:ip-address
      +--ro gateway?            inet:ip-address
      +--ro public-ip?          inet:ip-address
      +--ro location
        |   +--ro node-ref?       fabrictype:node-ref
        |   +--ro tp-ref?         fabrictype:tp-ref
        |   +--ro access-type?    fabrictype:access-type
        |   +--ro access-segment? uint32
      +--ro logical-location
        +--ro node-id?   nw:node-id
        +--ro tp-id?     nt:tp-id
```

By indicating locations of an endpoint in "location" container, the logical network elements such as logical nodes and logical termination points are bounded to the network elements in a specific fabric. Then the network configurations and operations from the logical network together with its belonged fabric topology information will further be distributed onto the bounding/related physical elements by the network topology manager.

Besides, the module defines three rpc commands to register, unregister and locate the endpoint onto both logical network and physical network shown as follows.
rpcs:
  +---x register-endpoint
     +---w input
       |  +---w endpoint-uuid?  yang:uuid
       |  +---w mac-address?  yang:mac-address
       |  +---w ip-address?   inet: ip-address
       |  +---w gateway?      inet: ip-address
       |  +---w public-ip?    inet: ip-address
       +---w location
         |  +---w node-ref?     fabrictype: node-ref
         |  +---w tp-ref?       fabrictype: tp-ref
         |  +---w access-type?  fabrictype: access-type
         |  +---w access-segment?  uint32
         +---ro output
           +---ro endpoint-id? yang:uuid
  +---x unregister-endpoint
     +---w input
       |  +---w ids*         yang:uuid
  +---x locate-endpoint
     +---w input
       |  +---w endpoint-id?  yang:uuid
       +---w location
         |  +---w node-ref?     fabrictype: node-ref
         |  +---w tp-ref?       fabrictype: tp-ref
         |  +---w access-type?  fabrictype: access-type
         |  +---w access-segment?  uint32

Figure 7: Fabric endpoint module RPC

6. Fabric Service YANG Modules

<CODE BEGINS> file "ietf-fabric-types@2016-10-13.yang"
module ietf-fabric-types {
  yang-version 1.1;
  prefix fabrictypes;

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

Zhuang & Shi Expires May 4, 2017 [Page 17]
import ietf-network-topology { prefix nt; }  
import ietf-network { prefix nw; }  
import ietf-yang-types { prefix "yang"; revision-date "2013-07-15"; }

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:sshares@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 "2016-10-13" {
  description
    "Initial revision of faas."
}

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
    "
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";
}

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";
        }
    }
}
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";
    }
  }
  description "underlayer network type";
}
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";
        }
    }
}

///groupings for logical element

grouping logical-switch {
    description "grouping attributes for a logical switch.";
}
leaf lsw-uuid {
    type yang:uuid;
    description "logical switch id";
}
leaf name {
    type string;
    description "logical switch name";
}
leaf segment-id {
    type uint32;
    description "segment id";
}
leaf network {
    type inet:ip-prefix;
    description "subnet";
}
leaf external {
    type boolean;
    description "whether its a lsw to external network";
}
uses acl-list;
}

grouping logical-router {
    description "grouping attributes for a logical router";
    leaf lr-uuid {
        type yang:uuid;
        description "logical router id";
    }
    leaf name {
        type string;
        description "logical router name";
    }
    leaf vrf-ctx {
        type uint32;
        description "logical router vrf id";
    }
}
uses acl-list;

container routes {
    description "routes";
    uses route-group;
}

}
leaf lport-uuid {
    type yang:uuid;
    description "logical port id";
}
leaf name {
    type string;
    description "logical port name";
}
container port-layer {
    description "layer information of the lport";
    container layer-1-info {
        description "layer 1 information of the lport";
        leaf location {
            type nt:tp-id;
            description "L1 tp id";
        }
    }
    container layer-2-info {
        description "layer 2 information of the lport";
        leaf access-type {
            type access-type;
            description "l2 access type";
        }
        leaf access-segment {
            type uint32;
            description "access segment";
        }
    }
    container layer-3-info {
        description "layer 3 information of the lport";
        leaf ip {
            type inet:ip-address;
            description "ip address";
        }
        leaf network {
            type inet:ip-prefix;
            description "ip prefix";
        }
        leaf mac {
            type yang:mac-address;
            description "mac address";
        }
        leaf forward-enable {
            type boolean;
            description "whether enable forward";
        }
        leaf logical-switch {
            description "logical switch to";
type nw:node-id;  
  description "lsw id";
}
}

uses acl-list;
uses port-functions;

list underlayer-ports {
  key port-ref;
  description "list of the corresponding underlay ports";
  leaf port-ref {
    type instance-identifier;
    description "port reference";
  }
}

<CODE ENDS>

<CODE BEGINS> file "ietf-fabric-service@2016-10-12.yang"
module ietf-fabric-service {

  yang-version 1.1;
  prefix fabric-services;

  import ietf-network { prefix nw;  }
  import ietf-network-topology { prefix nt;  }
    import ietf-fabric-types { prefix fabric-type; revision-date "2016-10-13"; }
    import ietf-inet-types { prefix "inet"; revision-date "2013-07-15"; }
  //import yang-ext { prefix "ext"; revision-date "2013-07-09"; }
  import ietf-fabric-topology { prefix fabric; revision-date "2016-09-29"; }
  //import ietf-access-control-list { prefix acl; revision-date "2016-02-18"; }

  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

Zhuang & Shi Expires May 4, 2017 [Page 26]
This module contains a collection of YANG definitions for Fabric services. 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-fabric-services; see the RFC itself for full legal notices.

revision "2016-10-12" {
  description "Initial revision of fabric service.";
  reference "draft-zhuang-i2rs-yang-fabric-service-00";
}

augment "nw:networks/nw:network/nw:node" {
  description "Augmentation for logic switch nodes provided by fabrices."

  container lsw-attribute {
    config false;
    description "attributes for logical switches";
    uses fabrictype:logical-switch;
  }
}

augment "nw:networks/nw:network/nw:node" {
  description "Augmentation for logical router nodes provided by fabric services."

  container lr-attribute {
    config false;
    description "attributes for logical routers";
    uses fabrictype:logical-router;
  }
}
augment "/nw:networks/nw:network/nw:node/nt:termination-point" {
    description "Augmentation for logical port provided by fabric services."
    container lport-attribute {
        config false;
        description "attributes for logical ports"
        uses fabrictype:logical-port;
    }
}

fortunately service api
rpc create-logical-switch {
    description "create a logical switch"
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "its belonged fabric id";
        }
        leaf name {
            type string;
            description "lsw name";
        }
        leaf vni {
            type int32;
            description "virtual network identifier.usually it is allocated by fabric itself. ";
        }
        leaf external {
            type boolean;
            default false;
            description "whether it is connected to external network";
        }
    }
    output {
        leaf node-id {
            type nw:node-id;
            description "node id";
        }
        uses fabrictype:logical-switch;
    }
}

rpc rm-logical-switch {
    description "remove a logical switch"
    input {
        leaf fabric-id {

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

rpc create-logical-router {
    description "create a logical router";
    input {
        leaf fabric-id {
            type fabric:id;
            description "fabric id";
        }
        leaf name {
            type string;
            description "name";
        }
    }
    output {
        leaf node-id {
            type nw:node-id;
            description "node id";
        }
        uses fabrictype:logical-router;
    }
}

rpc rm-logical-router {
    description "remove a logical router";
    input {
        leaf fabric-id {
            type fabric:id;
            description "fabric id";
        }
        leaf node-id {
            type nw:node-id;
            description "node id";
        }
    }
}

rpc add-static-route {
    description "add static routes";
    input {

leaf fabric-id {
    type fabric:fabric-id;
    description "fabric id";
}
leaf node-id {
    type nw:node-id;
    description "node id";
}
uses fabric-type:route-group;
}

rpc rm-static-route {
    description "remove static route";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf node-id {
            type nw:node-id;
            description "node id";
        }
        leaf-list destination-prefix {
            type inet:ipv4-prefix;
            description "DA";
        }
    }
}

rpc clear-static-route {
    description "clear all static route on a node";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf node-id {
            type nw:node-id;
            description "node id";
        }
    }
}

rpc create-logical-port {
    description "create a logical port";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
    }
}

rpc create-logical-port {
    description "create a logical port";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
    }
}
type fabric:fabric-id;
    description "fabric id";
}
leaf name {
    type string;
    description "name";
}
leaf logical-device {
    type nw:node-id;
    description "node id";
}
container attribute {
    description "logical port attributes";
    uses fabrictype:logical-port;
}
}
output {
    leaf tp-id {
        type nt:tp-id;
        description "tp id";
    }
    uses fabrictype:logical-port;
}
}
rpc rm-logical-port {
    description "remove a logical port";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf logical-device {
            type nw:node-id;
            description "node id";
        }
        leaf tp-id {
            type nt:tp-id;
            description "tp id";
        }
    }
}
rpc create-gateway {
    description "create a gateway";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
        }
    }
}
description "fabric id";
}
leaf ip-address {
    type inet:ip-address;
    description "ip addr";
}
leaf network {
    type inet:ip-prefix;
    description "ip prefix";
}
leaf logical-router {
    type nw:node-id;
    description "lr id";
}
leaf logical-switch {
    type nw:node-id;
    description "lsw id";
}
output {
    leaf tp-id {
        type nt:tp-id;
        description "port id";
        uses fabric-type:logical-port;
    }
}
}
rpc rm-gateway {
    description "remove gateway";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf ip-address {
            type inet:ip-address;
            description "ip addr";
        }
        leaf logical-router {
            type nw:node-id;
            description "lr id";
        }
    }
}
}
}
}
rpc port-binding-logical-to-fabric {
    description "binding logical device to a fabric";
input {
    leaf fabric-id {
        type fabric:fabric-id;
        description "fabric id";
    }
    leaf logical-device {
        type nw:node-id;
        description "node id";
    }
    leaf logical-port {
        type nt:tp-id;
        description "port id";
    }
    leaf fabric-port {
        type nt:tp-id;
        description "port id";
    }
}

rpc port-binding-logical-to-device {
    description "binding a logical port on a logical device to a physical device and port";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf logical-device {
            type nw:node-id;
            description "node id";
        }
        leaf logical-port {
            type nt:tp-id;
            description "logical port id";
        }
        leaf physical-port {
            type fabric-type:tp-ref;
            description "physical port ref";
        }
    }
}

rpc add-port-function {
    description "add port function to a port";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
    }
}
leaf logical-device {
    type nw:node-id;
    description "node id";
}
leaf logical-port {
    type nt:tp-id;
    description "logical port id";
}
uses fabric-type:port-functions;
}
}

rpc add-acl {
    description "add acl";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf logical-device {
            type nw:node-id;
            description "logical device id";
        }
        leaf logical-port {
            type nt:tp-id;
            description "logical port id";
        }
        leaf acl-name {
            type string;
            description "acl name";
        }
    }
}

rpc del-acl {
    description "delete acl";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf logical-device {
            type nw:node-id;
            description "logical device id";
        }
        leaf logical-port {
            type nt:tp-id;
            description "logical port id";
        }
    }
}
description "logical port id";
}
leaf acl-name {
    type string;
    description "acl name";
}
}
}

<CODE ENDS>

<CODE BEGINS> file "ietf-fabric-endpoint@2016-10-12.yang"
module ietf-fabric-endpoint {
    yang-version 1.1;
    prefix fabric-endpoints;
    import ietf-inet-types { prefix "inet"; revision-date "2013-07-15"; }
    import ietf-yang-types { prefix "yang"; revision-date "2013-07-15"; }
    import ietf-network { prefix nw; }
    import ietf-network-topology { prefix nt; }
    import ietf-fabric-types { prefix fabric-type; revision-date "2016-10-13"; }
    import ietf-fabric-topology { prefix fabric; 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 endpoints in Fabric service."
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 "2016-10-12" {
  description "Initial revision of faas.";
  reference "draft-zhuang-i2rs-yang-fabric-service-00";
}

grouping device-location {
  description "the location for this endpoint in the physical network.";

  leaf node-ref {
    type fabric:node-ref;
    description "node reference";
  }

  leaf tp-ref {
    type fabric:tp-ref;
    description "port reference";
  }

  leaf access-type {
    type fabric:access-type;
    default "exclusive";
    description "access type";
  }

  leaf access-segment {
    type uint32;
    default 0;
    description "access segment";
  }
}

grouping endpoint-attributes {

description "endpoint attributes";

leaf endpoint-uuid {
    type yang:uuid;
    description "endpoint id";
}

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

leaf mac-address {
    type yang:mac-address;
    description "mac addr";
}

leaf ip-address {
    type inet:ip-address;
    description "ip addr";
}

leaf gateway {
    type inet:ip-address;
    description "gateway ip";
}

leaf public-ip {
    type inet:ip-address;
    description "public ip addr";
}

container location {
    description "physical location of the endpoint";
    uses device-location;
}

container logical-location {
    description "The location for this endpoint in the logical network.";
    leaf node-id {
        type nw:node-id;
        description "node id";
    }
    leaf tp-id {
        type nt:tp-id;
        description "port id";
    }
}
container endpoints {
    config false;
    description "endpoints registry for faas.";

    list endpoint {
        key "endpoint-uuid";
        description "endpoint list";
        uses endpoint-attributes;
    }
}

/***********************RPC**************************************/
rpc register-endpoint {
    description "Register a new endpoint into the registry.";

    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        uses endpoint-attributes;
    }

    output {
        leaf endpoint-id {
            type yang:uuid;
            description "endpoint id";
        }
    }
}

rpc unregister-endpoint {
    description "Unregister an endpoint or endpoints from the registry.";

    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf-list ids {
            type yang:uuid;
            description "a list of ids";
        }
    }
}
rpc locate-endpoint {
    description "Set the physical location of the endpoint.";
    input {
        leaf fabric-id {
            type fabric:fabric-id;
            description "fabric id";
        }
        leaf endpoint-id {
            type yang:uuid;
            description "endpoint id";
        }
        container location {
            description "locations";
            uses device-location;
        }
    }
}

<CODE ENDS>

7. Security Considerations

None.

8. IANA Considerations

None.

9. References

9.1. Normative References

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

[I-D.zhuang-i2rs-yang-dc-fabric-network-topology]
Zhuangyan, Z., Shi, D., and R. Gu, "A YANG Data Model for
Fabric Topology in Data Center Network", draft-zhuang-
i2rs-yang-dc-fabric-network-topology-01 (work in
progress), September 2016.
9.2. Informative References

[I-D.ietf-i2rs-usecase-reqs-summary]

Authors’ Addresses

Yan Zhuang (editor)
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