Abstract

Currently new services create new opportunities for both network providers and service providers. Simplified Use of Policy Abstractions (SUPA) was proposed to develop a model that abstracts network resources and services and a methodology by which the management and monitoring of network services can be done using standardized policy rules. This document defines a VPN service management YANG data model and gives an example for DDC use case.
1. Introduction

Currently new services bring new challenges and opportunities for both network and service providers. Meanwhile, legacy services such as VPN [RFC4110] also need specialized management and controlling capability from the network management systems to improve the experiences for fast deployment and dynamic configuration.
Simplified Use of Policy Abstractions (SUPA) [SUPA-problem-statement] [SUPA-framework] was proposed to introduce the concepts of multi-level and multi-technology network abstractions to address the current separation between development and deployment operations. The first example that SUPA will focus on will be VPN management.

This document introduces YANG [RFC6020] [RFC6021] data models for SUPA configuration. Such models can facilitate the standardization for the interface of SUPA, as they are compatible to a variety of protocols such as NETCONF [RFC6241] and [RESTCONF]. Please note that in the context of SUPA, the term "application" refers to an operational and management applications employed, and possibly implemented, by an operator. The first configuration model is based on the first example - VPN management.

2. Conventions used in this document

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

3. Network Service Modules

In this section, several specific network service models are described based on a set of specific network services and the framework of SUPA [SUPA-framework].

3.1. Generic VPN Service Module

A virtual private network (VPN) extends a private network across a public network, such as the Internet. It enables a computer or network-enabled device to send and receive data across shared or public networks as if it were directly connected to the private network, while benefiting from the functionality, security and management policies of the public network. [VPN]

VPN systems may be classified by multiple ways, e.g., tunneling protocols, tunnel’s termination point location, etc. A typical one
among these is by the OSI layer they present to the connecting network, such as Layer 2 circuits or Layer 3 network connectivity.

In use cases of [SUPA-DDC], the links between DCs are VPNs, including L2VPN, L3VPN, etc. In this draft, before going deep into specific VPN services, a generic VPN model is firstly proposed below. It could be used by other specific VPN service models, such as L3VPN service models in [l3vpn-service-yang] and section 3.2 of this draft.

module: ietf-supa-vpn
    +-rw vpn-instance
        +-rw vpn-instance* [instance-name]
            +-rw instance-name       string
            +-rw connection-type?    enumeration
            +-rw service-type?       enumeration
            +-rw access-management
                |  +-rw user-name        string
                |  +-rw user-password    string
            +-rw instance-state
                |  +-rw admin-status?    vpn-instance-admin-status
            +-rw access-interface* [access-interface]
                +-rw access-interface    if:interface-ref
                +-rw role                enumeration

Below attributes are under discussion and will be added after that:
3.1.1. VPN YANG Model

```yang
module ietf-sup-a-vpn {
  namespace "urn:ietf:params:xml:ns:yang:ietf-sup-a-vpn";
  // replace with IANA namespace when assigned
  prefix vpn;

  import ietf-inet-types {
    prefix inet;
    //RFC6991
  }

  import ietf-interfaces {
    prefix "if";
  }

  organization "IETF";
  contact
    "Editor: Dacheng Zhang
dacheng.zdc@alibaba-inc.com

    Ying Cheng
    chengying10@chinaunicom.cn"
  ;

  description
    "This YANG module defines a generic VPN
    (Virtual Private Network service)
    configuration model common across all of the
    vendor implementations.";

  revision 2015-03-24 {
    description
      "Initial revision.";
    reference "RFC 4664, RFC4364, RFC7277";
  }
}
```
typedef vpn-instance-admin-status {
    description "Administrative status of a vpn instance.";
    type enumeration {
        enum "unknown";
        enum "up";
        enum "down";
    }
}

container vpn-instance {
    list vpn-instance {
        key "instance-name";
        description "Indicates the name of the VPN instance created.";

        leaf instance-name {
            type string;
            description "instance name.";
        }

        leaf connection-type {
            type enumeration {
                enum L2VPN {
                    value 0;
                    description "L2VPN";
                }
                enum L3VPN {
                    value 1;
                    description "L3VPN";
                }
            }
            description "Indicates the type of VPN, may be L2VPN or L3VPN";
        }

        leaf service-type {
            type enumeration {
                enum full-mesh {
                    value "0";
                    description "full-mesh";
                }
                enum hub-spoke {
                    value "1";
                    description "hub-spoke";
                }
            }
        }
    }
}
default "full-mesh";
description "Topology type."

container access-management
{
leaf user-name {
    type string;
    mandatory true;
    description "User name for this access interface.";
}

leaf user-password {
    type string;
    mandatory true;
    description "User password for the access interface. User name and password are listed here because VPN need the authentication, one typical way is to use user name and password, so that in CE dynamic migration case, CEs are able to access with authentication regardless location.";
}
}

container instance-state {
    description "The operational parameters describing the vpn-instance state.";
    leaf admin-status {
        type vpn-instance-admin-status;
    }
}

list access-interface {
    key "access-interface";
    description "Access interface ID.";

    leaf access-interface {
        type if:interface-ref;
        description "Access interface ID.";
    }

    leaf role {
        type enumeration {
            enum edge-if {
                value "0";
            }
        }
    }
}
3.2. L3VPN Service Module

A Layer 3 Virtual Private Network (L3VPN) interconnects sets of hosts and routers based on Layer 3 addresses and forwarding. L3VPN can be based on MPLS or IP technologies. L3VPN is a PE-based VPN managed by operators. L3VPN is widely used in carrier metro networks to provide VPN service for enterprise users.

A L3VPN model is a collection of L3VPN instances. A L3VPN instance contains a set of access interfaces to network devices as well as other attributes, such as routing protocol, address family, topology, and so on.

To configure a L3VPN instance, the administrator needs to specify which port(s) of a network device belongs to a L3VPN instance. Those ports and network device information can be derived from a network topology model in a network management system. The administrator also needs to specify what routing protocol needs to be configured for a L3VPN instance.

The following describes the information model for L3VPN, based on which users can develop applications to configure L3VPN instances.
module: ietf-sup-a-l3vpn
  +++rw l3vpn-instance* [instance-name]
  +++rw instance-name          string
  +++rw service-type?          enumeration
  +++rw address-family-type?   enumeration
  +++rw access-interface*      [access-interface]
  +++rw access-interface-ref  interface-ref
  +++rw access-interface-address ip-prefix
  +++rw role                  enumeration
  +++rw user-name             string
  +++rw user-password         string
  +++rw physical-node-id      string
  +++rw physical-access-interface string
  +++rw protocol
  +++rw protocol-type?        enumeration
  +++rw bgp-attribute
  +++rw remote-as-number?     string
  +++rw remote-peer-address   string
  +++rw igp-attribute
  +++rw protocol-id?          uint32

3.2.1. L3VPN YANG Model

<CODE BEGINS>
module ietf-sup-a-l3vpn {
  namespace "urn:ietf:params:xml:ns:yang:ietf-sup-a-l3vpn";
  // replace with IANA namespace when assigned
  prefix l3vpn;
  organization "IETF";
  contact
    "Editor: Dacheng Zhang
dacheng.zdc@alibaba-inc.com

Adel Zaalouk
adel.ietf@gmail.com

Kostas Pentikousis
k.pentikousis@eict.de
"

description
  "This YANG module defines a generic service
configuration model for L3VPN (Layer 3 Virtual
Private Network), based on which programmers can
develop applications to configure L3VPN instances.";

revision 2015-02-04 {
    description
      "Initial revision.";
    reference "RFC4364, RFC7277";
}

list l3vpn-instance {
    key "instance-name";
    description "Indicates the name of the L3VPN instance";

    leaf instance-name {
        type string;
        description "L3VPN instance name";
    }

    leaf service-type {
        type enumeration {
            enum full-mesh {
                value "0";
                description "full-mesh";
            }
            enum hub-spoke {
                value "1";
                description "hub-spoke";
            }
        }
        default "full-mesh";
        description "Topology type";
    }

    leaf address-family-type{
        type enumeration {
            enum ipv4uni {
                value "0";
                description "ipv4 unicast";
            }
            enum ipv6uni {
                value "1";
                description "ipv6 unicast";
            }
        }
        default "ipv4uni";
        description "Address family type: IPv4 or IPv6";
    }

    list access-interface {
        key "access-interface";
    }
}
description "Access interface ID";

leaf access-interface {
  type if:interface-ref;
  description "Access interface ID";
}

leaf access-interface-address {
  type inet:ip-prefix;
  description "Access interface address, IPv4 or IPv6";
}

leaf role {
  type enumeration {
    enum edge-if {
      value "0";
      description "edge interface";
    }
    enum center-if {
      when "../../service-type = 'hub-spoke'";
      value "1";
      description "center interface";
    }
  }
  mandatory true;
  description "center-if is only available in hub-spoke mode; there are two scenarios: in full mesh mode, the role of all the access-interface is edge-if, while in hub-spoke mode, the role of the interface of hub node is center-if, the one of spoke node is edge-if";
}

leaf user-name {
  type string;
  mandatory true;
  description "User name for this access interface";
}

leaf user-password {
  type string;
  mandatory true;
  description "User password for the access interface. User name and password are listed here because VPN need the authentication, one typical way is to use user name and password, so that in CE dynamic
migration case, CEs are able to access with authentication regardless location.

leaf physical-node {
    type string;
    mandatory true;
    description "This is the physical node ID of access PE";
}

leaf physical-access-interface {
    type if:interface-ref;
    mandatory true;
    description "This is the Physical access interface of access PE."
}

container protocol-id {
    description "The routing protocol between PE and CE."
    leaf protocol-type {
        type enumeration {
            enum bgp {
                value "0";
                description "bgp";
            }
            enum ospf {
                value "1";
                description "ospf";
            }
            enum isis {
                value "2";
                description "isis";
            }
            enum rip {
                value "3";
                description "rip";
            }
            enum static {
                value "4";
                description "static";
            }
            enum eigrp {
                value "5";
                description "eigrp";
            }
        }
    }
}
3.3. L2VPN Service Module

This section describes service model for Ethernet L2VPN.
There are different ways of classifying L2VPNs. According to the Ethernet services defined by Metro Ethernet Forum, there are mainly three types of Ethernet L2VPN service that can be provided by service providers: E-line, E-tree and E-lan.

- E-line is point to point service.
- E-lan is multipoint to multipoint service.
- E-tree is multipoint to multipoint service, but the communications between some consumer sites are not allowed.

Meanwhile according to [RFC4664], there are two fundamentally different kinds of Layer 2 VPN service that a service provider could offer to a customer: Virtual Private Wire Service (VPWS) and Virtual Private LAN Service (VPLS).

- VPWS is a L2 VPN service that provides L2 point-to-point service.
- VPLS is a L2 VPN service that emulates LAN service across a Wide Area Network (WAN).

Based on different degrees of abstraction, the interfaces can be categorized into two kinds: a) The user can provide service type and site information to the controller and controller create VPN automatically, based on the network, maybe by pseudo wire or other method. This is the more abstracted Ethernet L2VPN service interface. b) If the provider is MPLS or MPLS-TP, controller can provide less abstracted interfaces to the user. The user can use such interfaces to control the network more agility. This document describes the former one, a more abstracted model for L2VPN.

The following describes the information model for L2VPN, based on which users can develop applications to configure L2VPN instances.
module: ietf-supa-l2vpn

module ietf-supa-l2vpn {
  // replace with IANA namespace when assigned
  prefix l2vpn;
  organization "IETF";
  contact
    "Editor: Ying Cheng
     chengying10@chinaunicom.cn

    Vikram Choudhary
    vikschw@gmail.com"

description
  "This YANG module defines a generic service
  configuration model for L2VPN (Layer 2 Virtual
  Private Network), based on which programmers can
  develop applications to configure L2VPN instances.";

revision 2015-04-09 {
  description
    "Initial revision";
  reference "RFC4664";
}

container l2vpn-instances {
  list l2vpn-instance {
    key "instance-name";
    }
description
"Indicates the name of the L2VPN instance";

leaf instance-name {
  type string;
  description "L2VPN instance name";
}

leaf instance-admin-status {
  type enumeration {
    enum up {
      value "0";
      description "the instance is up";
    }
    enum down {
      value "1";
      description "the instance is down";
    }
  }
  default "up";
  description
  "Administrate status of this instance, user may use this value to manage the instance";
}

leaf instance-operate-status {
  type enumeration {
    enum up {
      value "0";
      description "the instance is up";
    }
    enum down {
      value "1";
      description "the instance is down";
    }
  }
  description
  "Status of this instance, user may use this value to inquire the status of this instance";
}

leaf instance-service-type {
  type enumeration {
    enum e-line {
      value "0";
      description "the service type is e-line";
    }
  }
}

enum e-tree {
  value "1";
  description "the service type is e-tree";
}
enum e-lan {
  value "2";
  description "the service type is e-lan";
}

mandatory true;

description
  "The service type of this instance,
   user may choose from e-line, e-tree, e-lan";

list "access-interface" {
  key "interface-id";
  description "Access interface ID";

  leaf interface-id {
    type if:interface-ref;
    description "Access interface ID";
  }

  leaf role {
    when "../../../instance-service-type = 'e-tree'";
    type enumeration {
      enum root {
        value "0";
        description "root interface";
      }
      enum leaf {
        value "1";
        description "leaf interface";
      }
    }
    mandatory true;
    description
      "the role of interface, root or leaf,
       only available in e-tree mode";
  }

  leaf node-id {
    type string;
    mandatory true;
    description "physical node ID of access PE";
  }
}
3.4. Module for DDC services

The following describes SUPA VPN management model designed for DDC services use case [SUPA-DDC]. [SUPA-DDC] took a large-scale Internet Data Center (IDC) operator as an example to describe what
SUPA needs to do including DDC service initiation, VPN-based connectivity initiation, optimize traffic route, traffic adjustment and monitor.

Module "ietf-supa-ddc" defines generic VPN management aspects which are common to all DDC services use case regardless of their type of vendor. In effect, the module can be viewed as providing a generic VPN management for DDC services.

This model is designed based on vDC Connectivity use case in section 6.2 of [SUPA-DDC].
module: ietf-supadcdc
  +--rw ddc-services
    +--rw ddc-service* [name]
      +--rw name                string
      +--rw tenant-name         string
      +--rw dc-name*            string
      +--rw interface-name*     string
      +--rw connection-type?    enumeration
      +--rw connection-name     string
      +--rw bandwidth           uint32
      +--rw latency             uint32

3.4.1. Model for DDC services

<CODE BEGINS>
module ietf-supadcdc {
  namespace "urn:ietf:params:xml:ns:yang:ietf-supadcdc";
  // replace with IANA namespace when assigned
  prefix ddc;

  organization "IETF";
  contact
    "Editor: Ying Cheng
     chengying10@chinaunicom.cn";

  description
    "This YANG module defines a component that describing
    the ddc service model for creating and optimizing
    tenant’s DC (data center) services that are deployed
    in multiple data centers."

  Terms and Acronyms
  DDC: Distributed Data Center
  L2VPN: Layer 2 Virtual Private Network
  L3VPN: Layer 3 Virtual Private Network"

  revision 2014-12-25 {
    description
      "Initial revision.";
    reference "RFC4364, RFC7277";
  }

  container ddc-services {
    description
      "To create service for tenant’s network that are deployed
      in multiple data centers. The following data are needed:
      name of data centers that the tenant’s service are

deployed in, connected method between data centers for
the tenant (e.g. L2VPN, L3VPN, Native IP, etc.), name
of tenant, ID of networks that belong to the tenant";

list ddc-service {
  key "name";
  description
    "Overall ddc operational data, including the names of
data center, the connection method between data centers,
name of service, etc.";

  leaf name {
    type string;
    description
      "Indicates the name of the service";
  }

  leaf tenant-name {
    type string;
    description
      "Indicates the name of the tenant for whom
the ddc service is being created.";
  }

  leaf-list dc-name {
    type string;
    description
      "List of the names of data center on which
tenant’s service is deployed in.";
  }

  leaf-list interface-name {
    type string;
    description
      "Indicates a set of access interface names of the
network device that the data centers (deployment
tenant’s service) are connected to.";
  }

  leaf connection-type {
    type enumeration {
      enum L2VPN {
        value 0;
        description "L2VPN";
      }
      enum L3VPN {
        value 1;
      }
    }
  }
}

description "L3VPN";
}
enum native-ipv4 {
  value 2;
  description "native IPv4";
}
enum native-ipv6 {
  value 3;
  description "native IPv6";
}

description
  "Indicates the connection type between the Data centers on which tenant service is being deployed. The connection type may be VPN (L2VPN or L3VPN) or Native IP (IPv4 or IPv6)";
}
leaf connection-name {
  type leafref { path "/l2vpn:l2vpn-instance/
    instance-name"; }
  mandatory true;
  description
    "Indicates the name of the connection e.g., VPN instance";
}
leaf bandwidth {
  type uint32;
  description
    "Indicates the bandwidth of the network connection instance that is created for tenant.";
}
leaf latency {
  type uint32;
  description
    "Indicates the latency of the network connection instance that is created for tenant.";
}

4. Security Considerations
5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgments

This document has benefited from reviews, suggestions, comments and proposed text provided by the following members, listed in alphabetical order: Vikram Choudhary, Feng Dong, Jing Huang, Junru Lin, Jan Lindblad, Felix Lu, Wu Nan, Juergen Schoenwaelder, Yiyong Zha, and Cathy Zhou.

Will Liu contributed to an early version of this draft.

7. References

7.1. Normative References


7.2. Informative References


[SUPA-DDC] Y. Cheng, and JF. Tremblay, "Use Cases for Distributed Data Center Applications in SUPA", IETF Internet draft, draft-cheng-supa-ddc-use-cases, January 2015


Authors’ Addresses

Dacheng Zhang (Editor)
Chaoyang Dist
Beijing 100000
P.R. China
Dacheng.zhang@gmail.com

Adel Zaalouk (Editor)
EICT GmbH
Torgauer Strasse 12-15
Berlin 10829
Germany
Email: adel.ietf@gmail.com

Kostas Pentikousis
EICT GmbH
Torgauer Strasse 12-15
Berlin 10829
Germany
Email: k.pentikousis@eict.de

Ying Cheng
China Unicom
P.R. China
Email: chengying10@chinaunicom.cn