

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: September 14, 2017

A. Lindem, Ed.  
Cisco Systems  
L. Berger, Ed.  
LabN Consulting, L.L.C.  
D. Bogdanovic

C. Hopps  
Deutsche Telekom  
March 13, 2017

Network Device YANG Logical Organization  
draft-ietf-rtgwg-device-model-02

## Abstract

This document presents an approach for organizing YANG models in a comprehensive logical structure that may be used to configure and operate network devices. The structure is itself represented as an example YANG model, with all of the related component models logically organized in a way that is operationally intuitive, but this model is not expected to be implemented. The identified component modules are expected to be defined and implemented on common network devices.

This document is derived from work submitted to the IETF by members of the informal OpenConfig working group of network operators and is a product of the Routing Area YANG Architecture design team.

## 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 September 14, 2017.

Internet-Draft

RTG YANG Device Model

March 2017

## Copyright Notice

Copyright (c) 2017 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 publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">1.1.</a>	Status of Work and Open Issues . . . . .	<a href="#">4</a>
<a href="#">2.</a>	Module Overview . . . . .	<a href="#">5</a>
<a href="#">2.1.</a>	Interface Model Components . . . . .	<a href="#">7</a>
<a href="#">2.2.</a>	System Management . . . . .	<a href="#">9</a>
<a href="#">2.3.</a>	Network Services . . . . .	<a href="#">10</a>
<a href="#">2.4.</a>	OAM Protocols . . . . .	<a href="#">11</a>
<a href="#">2.5.</a>	Routing . . . . .	<a href="#">11</a>
<a href="#">2.6.</a>	MPLS . . . . .	<a href="#">12</a>
<a href="#">3.</a>	Security Considerations . . . . .	<a href="#">12</a>
<a href="#">4.</a>	IANA Considerations . . . . .	<a href="#">13</a>
<a href="#">5.</a>	Network Device Model Structure . . . . .	<a href="#">13</a>
<a href="#">6.</a>	References . . . . .	<a href="#">19</a>
<a href="#">6.1.</a>	Normative References . . . . .	<a href="#">19</a>
<a href="#">6.2.</a>	Informative References . . . . .	<a href="#">20</a>
<a href="#">Appendix A.</a>	Acknowledgments . . . . .	<a href="#">21</a>
	Authors' Addresses . . . . .	<a href="#">22</a>

[1.](#) Introduction

"Operational Structure and Organization of YANG Models"  
[[I-D.openconfig-netmod-model-structure](#)], highlights the value of organizing individual, self-standing YANG [[RFC6020](#)] models into a more comprehensive structure. This document builds on that work and presents a derivative logical structure for use in representing the networking infrastructure aspects of physical and virtual devices.

[\[I-D.openconfig-netmod-model-structure\]](#) and earlier versions of this document presented a single device-centric model root, this document no longer contains this element. Such an element would have translated to a single device management model that would be the root

of all other models and was judged to be overly restrictive in terms of definition, implementation, and operation.

The document presents a logical network device YANG organizational structure that provides a conceptual framework for the models that may be used to configure and operate network devices. The structure is itself presented as an example YANG module, with all of the related component modules logically organized in a way that is operationally intuitive. This network device model is not expected to be implemented, but rather provide as context for the identified representative component modules with are expected to be defined, and supported on typical network devices.

This document refers to two new modules that are expected to be implemented. These models are defined to support the configuration and operation of network-devices that allow for the partitioning of resources from both, or either, management and networking perspectives. Two forms of resource partitioning are referenced:

The first form provides a logical partitioning of a network device where each partition is separately managed as essentially an independent network element which is 'hosted' by the base network device. These hosted network elements are referred to as logical network elements, or LNEs, and are supported by the logical-network-element module defined in [\[I-D.ietf-rtgwg-lne-model\]](#). The module is used to identify LNEs and associate resources from the network-device with each LNE. LNEs themselves are represented in YANG as independent network devices; each accessed independently. Optionally, and when supported by the implementation, they may also be accessed from the host system. Examples of vendor terminology for an LNE include logical system or logical router, and virtual switch, chassis, or fabric.

The second form provides support what is commonly referred to as Virtual Routing and Forwarding (VRF) instances as well as Virtual Switch Instances (VSI), see [\[RFC4026\]](#). In this form of resource

partitioning multiple control plane and forwarding/bridging instances are provided by and managed via a single (physical or logical) network device. This form of resource partitioning is referred to as Network Instances and are supported by the network-instance module defined in [[I-D.ietf-rtgwg-ni-model](#)]. Configuration and operation of each network-instance is always via the network device and the network-instance module.

This document was motivated by, and derived from, [[I-D.openconfig-netmod-model-structure](#)]. The requirements from that document have been combined with the requirements from "Consistent Modeling of Operational State Data in YANG",

[[I-D.openconfig-netmod-opstate](#)], into "NETMOD Operational State Requirements", [[I-D.ietf-netmod-opstate-reqs](#)]. This document is aimed at the requirement related to a common model-structure, currently Requirement 7, and also aims to provide a modeling base for Operational State representation.

The approach taken in this (and the original) document is to organize the models describing various aspects of network infrastructure, focusing on devices, their subsystems, and relevant protocols operating at the link and network layers. The proposal does not consider a common model for higher level network services. We focus on the set of models that are commonly used by network operators, and suggest a corresponding organization.

A significant portion of the text and model contained in this document was taken from the -00 of [[I-D.openconfig-netmod-model-structure](#)].

### 1.1. Status of Work and Open Issues

This version of the document and structure are a product of the Routing Area YANG Architecture design team and is very much a work in progress rather than a final proposal. This version is a major change from the prior version and this change was enabled by the work on the previously mentioned Schema Mount.

Schema Mount enables a dramatic simplification of the presented device model, particularly for "lower-end" devices which are unlikely to support multiple network instances or logical network elements.

Should structural-mount/YSDL not be available, the more explicit tree structure presented in earlier versions of this document will need to be utilized.

The top open issues are:

1. This document will need to match the evolution and standardization of [[I-D.openconfig-netmod-opstate](#)] or [[I-D.ietf-netmod-opstate-reqs](#)] by the Netmod WG.
2. Interpretation of different policy containers requires clarification.
3. It may make sense to use the identityref structuring with hardware and QoS model.
4. Which document(s) define the base System management, network services, and oam protocols modules is TBD. This includes the

possibility of simply using [RFC7317](#) in place of the presented System management module.

5. The model will be updated once the "opstate" requirements are addressed.

## [2.](#) Module Overview

In this document, we consider network devices that support protocols and functions defined within the IETF Routing Area, e.g, routers, firewalls and hosts. Such devices may be physical or virtual, e.g., a classic router with custom hardware or one residing within a server-based virtual machine implementing a virtual network function (VNF). Each device may sub-divide their resources into logical network elements (LNEs) each of which provides a managed logical device. Examples of vendor terminology for an LNE include logical system or logical router, and virtual switch, chassis, or fabric. Each LNE may also support virtual routing and forwarding (VRF) and virtual switching instance (VSI) functions, which are referred to below as a network instances (NIs). This breakdown is represented in Figure 1.

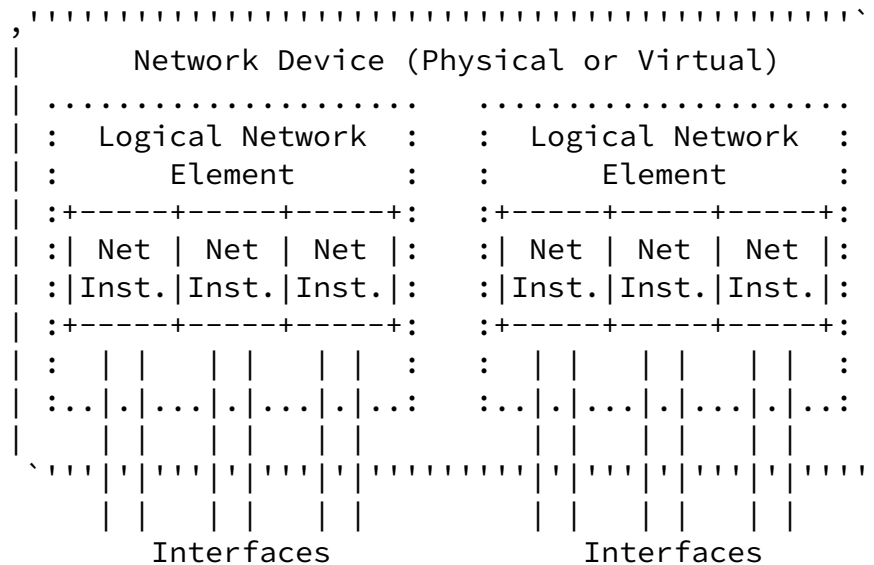


Figure 1: Module Element Relationships

A model for LNEs is described in [[I-D.ietf-rtgwg-lne-model](#)] and the model for network instances is covered in [[I-D.ietf-rtgwg-ni-model](#)].

The presented notional network device module can itself be thought of as a "meta-model" as it describes the relationships between individual models. We choose to represent it also as a simple YANG module consisting of other models, which are in fact independent top

level individual models. Although it is never expected to be implemented.

The presented modules do not follow the hierarchy of any Particular implementation, and hence is vendor-neutral. Nevertheless, the structure should be familiar to network operators and also readily mapped to vendor implementations.

The overall structure is:

```

module: example-network-device
  +--rw modules-state [RFC7895]
  |
  +--rw interfaces [RFC7223]
  +--rw hardware

```

```

+--rw qos
|
+--rw system-management          [RFC7317 or derived]
+--rw network-services
+--rw oam-protocols
|
+--rw routing                    [I-D.ietf-netmod-routing-cfg]
+--rw mpls
+--rw ieee-dot1Q
|
+--rw acls                      [I-D.ietf-netmod-acl-model]
+--rw key-chains                 [I-D.ietf-rtgwg-yang-key-chain]
|
+--rw logical-network-elements  [I-D.ietf-rtgwg-lne-model]
+--rw network-instances         [I-D.ietf-rtgwg-ni-model]

```

The network device is composed of top level modules that can be used to configure and operate a network device. (This is a significant difference from earlier versions of this document where there was a strict model hierarchy.) Importantly the network device structure is the same for a physical network device or a logical network device, such as those instantiated via the logical-network-element model. Extra spacing is included to denote different types of modules included.

YANG library [[RFC7895](#)] is included as it used to identify details of the top level modules supported by the (physical or logical) network device. The ability to identify supported modules is particularly important for LNEs which may have a set of supported modules which differs from the set supported by the host network device.

The interface management model [[RFC7223](#)] is included at the top level. The hardware module is a placeholder for a future device-

specific configuration and operational state data model. For example, a common structure for the hardware model might include chassis, line cards, and ports, but we leave this unspecified. The quality of service (QoS) section is also a placeholder module for device configuration and operational state data which relates to the treatment of traffic across the device. This document references augmentations to the interface module to support LNEs and NIs. Similar elements, although perhaps only for LNEs, may also need to be

included as part of the definition of the future hardware and QoS modules.

System management, network services, and oam protocols represent new top level modules that are used to organize data models of similar functions. Additional information on each is provided below.

The routing and MPLS modules provide core support for the configuration and operation of a devices control plane and data plane functions. IEEE dot1Q [[IEEE-8021Q](#)] is an example of another module that provides similar functions for VLAN bridging, and other similar modules are also possible. Each of these modules is expected to be LNE and NI unaware, and to be instantiated as needed as part of the LNE and NI configuration and operation supported by the logical-network-element and network-instance modules. (Note that this is a change from [[I-D.ietf-netmod-routing-cfg](#)] which is currently defined with VRF/NI semantics.)

The access control list (ACL) and key chain modules are included as examples of other top level modules that may be supported by a network device.

The logical network element and network instance modules enable LNEs and NIs respectively and are defined below.

## [2.1.](#) Interface Model Components

Interfaces are a crucial part of any network device's configuration and operational state. They generally include a combination of raw physical interfaces, link-layer interfaces, addressing configuration, and logical interfaces that may not be tied to any physical interface. Several system services, and layer 2 and layer 3 protocols may also associate configuration or operational state data with different types of interfaces (these relationships are not shown for simplicity). The interface management model is defined by [[RFC7223](#)].

The logical-network-element and network-instance modules defined in [[I-D.ietf-rtgwg-lne-model](#)] and [[I-D.ietf-rtgwg-ni-model](#)] augment the existing interface management model in two ways: The first, by the

logical-network-element module, adds an identifier which is used on



physical interface types to identify an associated LNE. The second, by the network-instance module, adds a name which is used on interface or sub-interface types to identify an associated network instance. Similarly, this name is also added for IPv4 and IPv6 types, as defined in [\[RFC7277\]](#).

The interface related augmentations are as follows:

```
module: ietf-logical-network-element
augment /if:interfaces/if:interface:
  +--rw bind-lne-name?   string

module: ietf-network-instance
augment /if:interfaces/if:interface:
  +--rw bind-network-instance-name?   string
augment /if:interfaces/if:interface/ip:ipv4:
  +--rw bind-network-instance-name?   string
augment /if:interfaces/if:interface/ip:ipv6:
  +--rw bind-network-instance-name?   string
```

The following is an example of envisioned combined usage. The interfaces container includes a number of commonly used components as examples:

```
+--rw if:interfaces
|   +--rw interface* [name]
|       +--rw name                               string
|       +--rw lne:bind-lne-name?                 string
|       +--rw ethernet
|           |   +--rw ni:bind-network-instance-name? string
|           |   +--rw aggregates
|           |   +--rw rstp
|           |   +--rw lldp
|           |   +--rw ptp
|       +--rw vlans
|       +--rw tunnels
|       +--rw ipv4
|           |   +--rw ni:bind-network-instance-name? string
|           |   +--rw arp
|           |   +--rw icmp
|           |   +--rw vrrp
|           |   +--rw dhcp-client
|       +--rw ipv6
|           |   +--rw ni:bind-network-instance-name? string
|           |   +--rw vrrp
|           |   +--rw icmpv6
|           |   +--rw nd
|           |   +--rw dhcpv6-client
```

The [\[RFC7223\]](#) defined interface model is structured to include all interfaces in a flat list, without regard to logical or virtual instances (e.g., VRFs) supported on the device. The bind-lne-name and bind-network-instance-name leaves provide the association between an interface and its associated LNE and NI (e.g., VRF or VSI).

## 2.2. System Management

[Editor's note: need to discuss and resolve relationship between this structure and [RFC7317](#) and determine if 7317 is close enough to simply use as is.]

System management is expected to reuse definitions contained in [\[RFC7317\]](#). It is expected to be instantiated per device and LNE. Its structure is shown below:

```
module: example-network-device
+--rw system-management
|   +--rw system-management-global
|   +--rw system-management-protocol* [type]
|       +--rw type          identityref
```

System-management-global is used for configuration information and state that is independent of a particular management protocol. System-management-protocol is a list of management protocol specific elements. The type-specific sub-modules are expected to be defined.

The following is an example of envisioned usage:

```
module: example-network-device
  +--rw system-management
    +--rw system-management-global
      | +--rw statistics-collection
      | ...
    +--rw system-management-protocol* [type]
      | +--rw type=syslog
      | +--rw type=dns
      | +--rw type=ntp
      | +--rw type=ssh
      | +--rw type=tacacs
      | +--rw type=snmp
      | +--rw type=netconf
```

### [2.3.](#) Network Services

A device may provide different network services to other devices, for example a device may act as a DHCP server. The model may be instantiated per device, LNE, and NI. An identityref is used to identify the type of specific service being provided and its associated configuration and state information. The defined structure is as follows:

```
module: example-network-device
  +--rw network-services
    | +--rw network-service* [type]
    |   +--rw type identityref
```

The following is an example of envisioned usage: Examples shown below include a device-based Network Time Protocol (NTP) server, a Domain Name System (DNS) server, and a Dynamic Host Configuration Protocol

(DHCP) server:

```
module: example-network-device
  +--rw network-services
    +--rw network-service* [type]
      +--rw type=ntp-server
      +--rw type=dns-server
      +--rw type=dhcp-server
```

Lindem, et al.

Expires September 14, 2017

[Page 10]

---

Internet-Draft

RTG YANG Device Model

March 2017

#### [2.4.](#) OAM Protocols

OAM protocols that may run within the context of a device are grouped within the oam-protocols model. The model may be instantiated per device, LNE, and NI. An identifyref is used to identify the information and state that may relate to a specific OAM protocol. The defined structure is as follows:

```
module: example-network-device
  +--rw oam-protocols
    +--rw oam-protocol* [type]
      +--rw type      identityref
```

The following is an example of envisioned usage. Examples shown below include Bi-directional Forwarding Detection (BFD), Ethernet Connectivity Fault Management (CFM), and Two-Way Active Measurement Protocol (TWAMP):

```
module: example-network-device
  +--rw oam-protocols
    +--rw oam-protocol* [type]
      +--rw type=bfd
      +--rw type=cfm
      +--rw type=twamp
```

#### [2.5.](#) Routing

Routing protocol and IP forwarding configuration and operation information is modeled via a routing model, such as the one defined in [[I-D.ietf-netmod-routing-cfg](#)].

The routing module is expected to include all IETF defined control plane protocols, such as BGP, OSPF, LDP and RSVP-TE. It is also expected to support configuration and operation of or more routing information bases (RIB). A RIB is a list of routes complemented with administrative data. Finally, policy is expected to be represented within each control plane protocol and RIB.

The anticipated structure is as follows:

```
module: example-network-device
  +--rw rt:routing [I-D.ietf-netmod-routing-cfg]
    +--rw control-plane-protocol* [type]
      | +--rw type identityref
      | +--rw policy
    +--rw rib* [name]
      +--rw name string
      +--rw description? string
      +--rw policy
```

## [2.6.](#) MPLS

MPLS data plane related information is grouped together, as with the previously discussed modules, is unaware of VRFs/NIs. The model may be instantiated per device, LNE, and NI. MPLS control plane protocols are expected to be included in [Section 2.5](#). MPLS may reuse and build on [\[I-D.openconfig-mpls-consolidated-model\]](#) or other emerging models and has an anticipated structure as follows:

```
module: example-network-device
  +--rw mpls
    +--rw global
    +--rw lsps* [type]
      +--rw type identityref
```

Type refers to LSP type, such as static, traffic engineered or routing congruent. The following is an example of such usage:

```
module: example-network-device
  +--rw mpls
    +--rw global
      +--rw lsp* [type]
        +--rw type=static
        +--rw type=constrained-paths
        +--rw type=igp-congruent
```

### [3.](#) Security Considerations

The network-device model structure described in this document does not define actual configuration and state data, hence it is not directly responsible for security risks.

Each of the component models that provide the corresponding configuration and state data should be considered sensitive from a security standpoint since they generally manipulate aspects of network configurations. Each component model should be carefully evaluated to determine its security risks, along with mitigations to reduce such risks.

LNE portion is TBD

NI portion is TBD

### [4.](#) IANA Considerations

This YANG model currently uses a temporary ad-hoc namespace. If it is placed or redirected for the standards track, an appropriate namespace URI will be registered in the "IETF XML Registry" [[RFC3688](#)]. The YANG structure modules will be registered in the "YANG Module Names" registry [[RFC6020](#)].

### [5.](#) Network Device Model Structure

```
<CODE BEGINS> file "example-network-device@2017-03-13.yang"
module example-network-device {

  yang-version "1";
```

```

// namespace
namespace "urn:example:network-device";

prefix "nd";

// import some basic types

// meta
organization "IETF RTG YANG Design Team Collaboration
              with OpenConfig";

contact
  "Routing Area YANG Architecture Design Team -
   <rtg-dt-yang-arch@ietf.org>";

description
  "This module describes a model structure for YANG
   configuration and operational state data models. Its intent is
   to describe how individual device protocol and feature models
   fit together and interact.";

revision "2017-03-13" {
  description
    "IETF Routing YANG Design Team Meta-Model";
  reference "TBD";
}

// extension statements

```

```

// identity statements

identity oam-protocol-type {
  description
    "Base identity for derivation of OAM protocols";
}

identity network-service-type {
  description
    "Base identity for derivation of network services";
}

```

```

identity system-management-protocol-type {
    description
        "Base identity for derivation of system management
        protocols";
}

identity oam-service-type {
    description
        "Base identity for derivation of Operations,
        Administration, and Maintenance (OAM) services.";
}

identity control-plane-protocol-type {
    description
        "Base identity for derivation of control-plane protocols";
}

identity mpls-lsp-type {
    description
        "Base identity for derivation of MPLS LSP types";
}

// typedef statements

// grouping statements

grouping ribs {
    description
        "Routing Information Bases (RIBs) supported by a
        network-instance";
    container ribs {
        description
            "RIBs supported by a network-instance";
        list rib {
            key "name";
            min-elements "1";

```

```

description
    "Each entry represents a RIB identified by the
    'name' key. All routes in a RIB must belong to the
    same address family.

```



```

        For each routing instance, an implementation should
        provide one system-controlled default RIB for each
        supported address family.";
    leaf name {
        type string;
        description
            "The name of the RIB.";
    }
    reference "draft-ietf-netmod-routing-cfg";
    leaf description {
        type string;
        description
            "Description of the RIB";
    }
    // Note that there is no list of interfaces within
    container policy {
        description "Policy specific to RIB";
    }
}
}

// top level device definition statements
container ietf-yang-library {
    description
        "YANG Module Library as defined in
        draft-ietf-netconf-yang-library";
}

container interfaces {
    description
        "Interface list as defined by RFC7223/RFC7224";
}

container hardware {
    description
        "Hardware / vendor-specific data relevant to the platform.
        This container is an anchor point for platform-specific
        configuration and operational state data. It may be further
        organized into chassis, line cards, ports, etc. It is
        expected that vendor or platform-specific augmentations
        would be used to populate this part of the device model";
}

```

```
container qos {
  description "QoS features, for example policing, shaping, etc.";
}

container system-management {
  description
    "System management for physical or virtual device.";
  container system-management-global {
    description "System management - with reuse of RFC 7317";
  }
  list system-management-protocol {
    key "type";
    leaf type {
      type identityref {
        base system-management-protocol-type;
      }
      mandatory true;
      description
        "Syslog, ssh, TACAC+, SNMP, NETCONF, etc.";
    }
    description "List of system management protocol
      configured for a logical network
      element.";
  }
}

container network-services {
  description
    "Container for list of configured network
    services.";
  list network-service {
    key "type";
    description
      "List of network services configured for a
      network instance.";
    leaf type {
      type identityref {
        base network-service-type;
      }
      mandatory true;
      description
        "The network service type supported within
        a network instance, e.g., NTP server, DNS
        server, DHCP server, etc.";
    }
  }
}
```

```
container oam-protocols {
  description
    "Container for configured OAM protocols.";
  list oam-protocol {
    key "type";
    leaf type {
      type identityref {
        base oam-protocol-type;
      }
      mandatory true;
      description
        "The Operations, Administration, and
        Maintenance (OAM) protocol type, e.g., BFD,
        TWAMP, CFM, etc.";
    }
    description
      "List of configured OAM protocols.";
  }
}

container routing {
  description
    "The YANG Data Model for Routing Management revised to be
    Network Instance / VRF independent. ";
  // Note that there is no routing or network instance
  list control-plane-protocol {
    key "type";
    description
      "List of control plane protocols configured for
      a network instance.";
    leaf type {
      type identityref {
        base control-plane-protocol-type;
      }
      mandatory true;
      description
        "The control plane protocol type, e.g., BGP,
        OSPF IS-IS, etc.";
    }
    container policy {
```

```

        description
            "Protocol specific policy,
            reusing [RTG-POLICY]";
    }
}
list rib {
    key "name";
    description

```

"Each entry represents a RIB identified by the 'name' key. All routes in a RIB must belong to the same address family.

For each routing instance, an implementation should provide one system-controlled default RIB for each supported address family."

```

leaf name {
    type string;
    mandatory true;
    description
        "The name of the RIB.";
}
reference "draft-ietf-netmod-routing-cfg";
leaf description {
    type string;
    description
        "Description of the RIB";
}
// Note that there is no list of interfaces within
container policy {
    description "Policy specific to RIB";
}
}
}

container mpls {
    description "MPLS and TE configuration";
    container global {
        description "Global MPLS configuration";
    }
    list lsps {
        key "type";

```

```

        description
            "List of LSP types.";
        leaf type {
            type identityref {
                base mpls-lsp-type;
            }
            mandatory true;
            description
                "MPLS and Traffic Engineering protocol LSP types,
                static, LDP/SR (igp-congruent),
                RSVP TE (constrained-paths) , etc.";
        }
    }
}

```

Lindem, et al.

Expires September 14, 2017

[Page 18]

---

Internet-Draft

RTG YANG Device Model

March 2017

```

container ieee-dot1Q {
    description
        "The YANG Data Model for VLAN bridges as defined by the IEEE";
}

container ietf-acl {
    description "Packet Access Control Lists (ACLs) as specified
                in draft-ietf-netmod-acl-model";
}

container ietf-key-chain {
    description "Key chains as specified in
                draft-ietf-rtgwg-yang-key-chain";
}

container logical-network-element {
    description
        "This module is used to support multiple logical network
        elements on a single physical or virtual system.";
}

container network-instance {
    description
        "This module is used to support multiple network instances
        within a single physical or virtual device. Network
        instances are commonly know as VRFs (virtual routing

```

```

        and forwarding) and VSIs (virtual switching instances).";
    }
    // rpc statements

    // notification statements

}
<CODE ENDS>

```

## 6. References

### 6.1. Normative References

- [I-D.ietf-rtgwg-lne-model]  
 Berger, L., Hopps, C., Lindem, A., and D. Bogdanovic,  
 "YANG Logical Network Elements", [draft-ietf-rtgwg-lne-model-01](#) (work in progress), October 2016.
- [I-D.ietf-rtgwg-ni-model]  
 Berger, L., Hopps, C., Lindem, A., and D. Bogdanovic,  
 "YANG Network Instances", [draft-ietf-rtgwg-ni-model-01](#)  
 (work in progress), October 2016.

Lindem, et al. Expires September 14, 2017 [Page 19]

---

Internet-Draft RTG YANG Device Model March 2017

- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<http://www.rfc-editor.org/info/rfc3688>>.
- [RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", [RFC 4026](#), DOI 10.17487/RFC4026, March 2005, <<http://www.rfc-editor.org/info/rfc4026>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#), DOI 10.17487/RFC6020, October 2010, <<http://www.rfc-editor.org/info/rfc6020>>.
- [RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", [RFC 7223](#), DOI 10.17487/RFC7223, May 2014, <<http://www.rfc-editor.org/info/rfc7223>>.
- [RFC7277] Bjorklund, M., "A YANG Data Model for IP Management",

[RFC 7277](#), DOI 10.17487/RFC7277, June 2014,  
<<http://www.rfc-editor.org/info/rfc7277>>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", [RFC 7317](#), DOI 10.17487/RFC7317, August 2014, <<http://www.rfc-editor.org/info/rfc7317>>.

## 6.2. Informative References

- [I-D.ietf-netmod-acl-model]  
Bogdanovic, D., Koushik, K., Huang, L., and D. Blair,  
"Network Access Control List (ACL) YANG Data Model",  
[draft-ietf-netmod-acl-model-09](#) (work in progress), October 2016.
- [I-D.ietf-netmod-opstate-reqs]  
Watsen, K. and T. Nadeau, "Terminology and Requirements for Enhanced Handling of Operational State", [draft-ietf-netmod-opstate-reqs-04](#) (work in progress), January 2016.
- [I-D.ietf-netmod-routing-cfg]  
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", [draft-ietf-netmod-routing-cfg-25](#) (work in progress), November 2016.
- [I-D.ietf-rtgwg-yang-key-chain]  
Lindem, A., Qu, Y., Yeung, D., Chen, I., Zhang, Z., and Y. Yang, "Routing Key Chain YANG Data Model", [draft-ietf-rtgwg-yang-key-chain-15](#) (work in progress), February 2017.

Lindem, et al. Expires September 14, 2017 [Page 20]

---

Internet-Draft RTG YANG Device Model March 2017

- [I-D.openconfig-mpls-consolidated-model]  
George, J., Fang, L., eric.osborne@level3.com, e., and R. Shakir, "MPLS / TE Model for Service Provider Networks", [draft-openconfig-mpls-consolidated-model-02](#) (work in progress), October 2015.
- [I-D.openconfig-netmod-model-structure]  
Shaikh, A., Shakir, R., D'Souza, K., and L. Fang, "Operational Structure and Organization of YANG Models", [draft-openconfig-netmod-model-structure-00](#) (work in progress), March 2015.

[I-D.openconfig-netmod-opstate]

Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", [draft-openconfig-netmod-opstate-01](#) (work in progress), July 2015.

[IEEE-8021Q]

Holness, M., "IEEE 802.1Q YANG Module Specifications", IEEE-Draft <http://www.ieee802.org/1/files/public/docs2015/new-mholness-yang-8021Q-0515-v04.pdf>, May 2015.

[RFC7895] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Module Library", [RFC 7895](#), DOI 10.17487/RFC7895, June 2016, <<http://www.rfc-editor.org/info/rfc7895>>.

## [Appendix A](#). Acknowledgments

This document is derived from [draft-openconfig-netmod-model-structure-00](#). We thank the Authors of that document and acknowledge their indirect contribution to this work. The authors include: Anees Shaikh, Rob Shakir, Kevin D'Souza, Luyuan Fang, Qin Wu, Stephane Litkowski and Gang Yan.

This work was discussed in and produced by the Routing Area Yang Architecture design team. Members at the time of writing included Acee Lindem, Anees Shaikh, Christian Hopps, Dean Bogdanovic, Lou Berger, Qin Wu, Rob Shakir, Stephane Litkowski, and Gang Yan.

The identityref approach was proposed by Mahesh Jethanandani.

The RFC text was produced using Marshall Rose's xml2rfc tool.

### Authors' Addresses

Acee Lindem (editor)  
Cisco Systems  
301 Midenhall Way



Cary, NC 27513  
USA

Email: [acee@cisco.com](mailto:acee@cisco.com)

Lou Berger (editor)  
LabN Consulting, L.L.C.

Email: [lberger@labn.net](mailto:lberger@labn.net)

Dean Bogdanovic

Email: [ivandean@gmail.com](mailto:ivandean@gmail.com)

Christan Hopps  
Deutsche Telekom

Email: [chopps@chopps.org](mailto:chopps@chopps.org)