Internet Engineering Task Force (IETF)
Request for Comments: 8530
Category: Standards Track
ISSN: 2070-1721

L. Berger
C. Hopps
LabN Consulting, L.L.C.
A. Lindem
Cisco Systems
D. Bogdanovic
X. Liu
Volta Networks

March 2019

YANG Model for Logical Network Elements

Abstract

This document defines a logical network element (LNE) YANG module that is compliant with the Network Management Datastore Architecture (NMDA). This module can be used to manage the logical resource partitioning that may be present on a network device. Examples of common industry terms for logical resource partitioning are logical systems or logical routers. The YANG model in this document conforms with NMDA as defined in RFC 8342.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8530.

Table of Contents

1. Introduction

This document defines an NMDA-compliant YANG module [RFC6020] to support the creation of logical network elements (LNEs) on a network device. An LNE is an independently managed virtual device made up of resources allocated to it from the host or parent network device. An LNE running on a host network device conceptually parallels a virtual machine running on a host system. Using host-virtualization terminology, one could refer to an LNE as a "Guest" and the containing network device as the "Host". While LNEs may be implemented via host-virtualization technologies, this is not a requirement. The YANG model in this document conforms with the Network Management Datastore Architecture defined in [RFC8342].

This document also defines the necessary augmentations for allocating host resources to a given LNE. As the interface management model [RFC8343] is the only module that currently defines host resources, this document currently defines only a single augmentation to cover the assignment of interfaces to an LNE. Future modules that define support for the control of host device resources are expected to, where appropriate, provide parallel support for the assignment of controlled resources to LNEs.

As each LNE is an independently managed device, each will have its own set of YANG-modeled data that is independent of the host device and other LNEs. For example, multiple LNEs may all have their own "Tunnel0" interface defined, which will not conflict with each other and will not exist in the host's interface model. An LNE will have its own management interfaces, possibly including independent instances of NETCONF/RESTCONF/etc servers to support the configuration of their YANG models. As an example of this independence, an implementation may choose to completely rename assigned interfaces; so, on the host, the assigned interface might be called "Ethernet0/1" while within the LNE it might be called "eth1".

In addition to standard management interfaces, a host device implementation may support accessing LNE configuration and operational YANG models directly from the host system. When supported, such access is accomplished through a yang-schema-mount mount point [RFC8528] under which the root-level LNE YANG models may be accessed.

Examples of vendor terminology for an LNE include logical system or logical router and virtual switch, chassis, or fabric.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Readers are expected to be familiar with terms and concepts of YANG [RFC7950] and YANG Schema Mount [RFC8528].

This document uses the graphical representation of data models defined in YANG Tree Diagrams [RFC8340].

2. 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 subdivide their resources into 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 VPN Routing and Forwarding (VRF) and Virtual Switching Instance (VSI) functions, which are referred to below as Network Instances (NIs). This breakdown is represented in Figure 1.

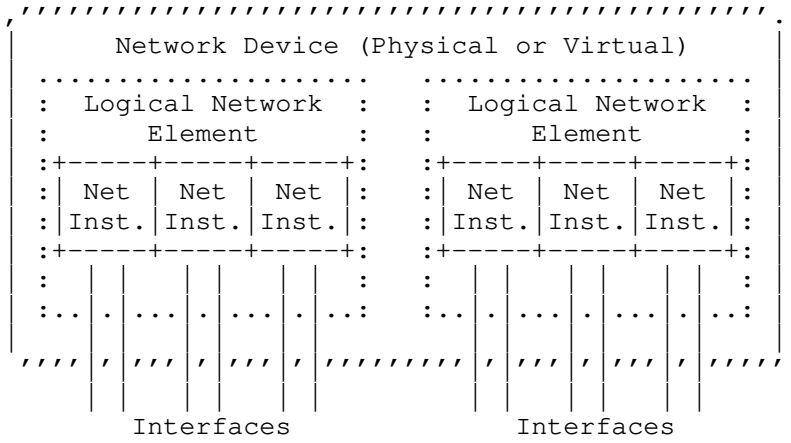

Figure 1: Module Element Relationships

A model for LNEs is described in Section 3, and the model for NIs is covered in [RFC8529].

The interface management model [RFC8343] is an existing model that is impacted by the definition of LNEs and NIs. This document and [RFC8529] define augmentations to the interface model 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.

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 [RFC8343]. The logical-network-element module augments existing interface management models by adding an identifier that is used on interfaces to identify an associated LNE.

The interface-related augmentation is as follows:

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

The interface model defined in [RFC8343] is structured to include all interfaces in a flat list, without regard to logical assignment of resources supported on the device. The bind-lne-name and leaf provides the association between an interface and its associated LNE. Note that as currently defined, to assign an interface to both an LNE and NI, the interface would first be assigned to the LNE and then within that LNE's interface model, the LNE's representation of that interface would be assigned to an NI using the mechanisms defined in [RFC8529].

3. Logical Network Elements

Logical network elements support the ability of some devices to partition resources into independent logical routers and/or switches. Device support for multiple logical network elements is implementation specific. Systems without such capabilities need not include support for the logical-network-element module. In physical devices, some hardware features are shared across partitions, but control-plane (e.g., routing) protocol instances, tables, and

configuration are managed separately. For example, in logical routers or VNFs, this may correspond to establishing multiple logical instances using a single software installation. The model supports configuration of multiple instances on a single device by creating a list of logical network elements, each with their own configuration and operational state related to routing and switching protocols.

The LNE model can be represented as:

```
module: ietf-logical-network-element
  +--rw logical-network-elements
     +--rw logical-network-element* [name]
        +--rw name           string
        +--rw managed?       boolean
        +--rw description?   string
        +--mp root
  augment /if:interfaces/if:interface:
    +--rw bind-lne-name?
          -> /logical-network-elements/logical-network-element/name

  notifications:
    +---n bind-lne-name-failed
       +--ro name             -> /if:interfaces/interface/name
       +--ro bind-lne-name
       |       -> /if:interfaces/interface/lne:bind-lne-name
       +--ro error-info?      string
```

'name' identifies the logical network element. 'managed' indicates if the server providing the host network device will provide the client LNE information via the 'root' structure. The root of an LNE's specific data is the schema mount point 'root'. bind-lne-name is used to associate an interface with an LNE, and bind-lne-name-failed is used in certain failure cases.

An LNE root MUST contain at least the YANG library [RFC7895] and interface module [RFC8343].

3.1. LNE Instantiation and Resource Assignment

Logical network elements may be controlled by clients using existing list operations. When list entries are created, a new LNE is instantiated. The models mounted under an LNE root are expected to be dependent on the server implementation. When a list entry is deleted, an existing LNE is destroyed. For more information, see [RFC7950], Section 7.8.6.

Once instantiated, host network device resources can be associated with the new LNE. As previously mentioned, this document augments

ietf-interfaces with the bind-lne-name leaf to support such associations for interfaces. When a bind-lne-name is set to a valid LNE name, an implementation MUST take whatever steps are internally necessary to assign the interface to the LNE or provide an error message (defined below) with an indication of why the assignment failed. It is possible for the assignment to fail while processing the set, or after asynchronous processing. Error notification in the latter case is supported via a notification.

On a successful interface assignment to an LNE, an implementation MUST also make the resource available to the LNE by providing a system-created interface to the LNE. The name of the system-created interface is a local matter and may be identical or completely different and mapped from and to the name used in the context of the host device. The system-created interface SHOULD be exposed via the LNE-specific instance of the interface model [RFC8343].

3.2. LNE Management -- LNE View

Each LNE instance is expected to support management functions from within the context of the LNE root, via a server that provides information with the LNE's root exposed as the device root. Management functions operating within the context of an LNE are accessed through the LNE's standard management interfaces, e.g., NETCONF and SNMP. Initial configuration, much like the initial configuration of the host device, is a local implementation matter.

When accessing an LNE via the LNE's management interface, a network device representation will be presented, but its scope will be limited to the specific LNE. Normal YANG/NETCONF mechanisms, together with the required YANG library [RFC7895] instance, can be used to identify the available modules. Each supported module will be presented as a top-level module. Only LNE-associated resources will be reflected in resource-related modules, e.g., interfaces, hardware, and perhaps QoS. From the management perspective, there will be no difference between the available LNE view (information) and a physical network device.

3.3. LNE Management -- Host Network Device View

There are multiple implementation approaches possible to enable a network device to support the logical-network-element module and multiple LNEs. Some approaches will allow the management functions operating at the network device level to access LNE configuration and operational information, while others will not. Similarly, even when LNE management from the network device is supported by the implementation, it may be prohibited by user policy.

Independent of the method selected by an implementation, the 'managed' boolean mentioned above is used to indicate when LNE management from the network device context is possible. When the 'managed' boolean is 'false', the LNE cannot be managed by the host system and can only be managed from within the context of the LNE as described in Section 3.2. Attempts to access information below a root node whose associated 'managed' boolean is set to 'false' MUST result in the error message indicated below. In some implementations, it may not be possible to change this value. For example, when an LNE is implemented using virtual machine and traditional hypervisor technologies, it is likely that this value will be restricted to a 'false' value.

It is an implementation choice if the information can be accessed and modified from within the context of the LNE, or even the context of the host device. When the 'managed' boolean is 'true', LNE information SHALL be accessible from the context of the host device. When the associated schema-mount definition has the 'config' leaf set to 'true', then LNE information SHALL also be modifiable from the context of the host device. When LNE information is available from both the host device and from within the context of the LNE, the same information MUST be made available via the 'root' element, with paths modified as described in [RFC8528].

An implementation MAY represent an LNE's schema using either the 'inline' or the 'shared-schema' approaches defined in [RFC8528]. The choice of which to use is completely an implementation choice. The inline approach is anticipated to be generally used in the cases where the 'managed' boolean will always be 'false'. The 'shared-schema' approach is expected to be most useful in the case where all LNEs share the same schema. When 'shared-schema' is used with an LNE mount point, the YANG library rooted in the LNE's mount point MUST match the associated schema defined according to the ietf-yang-schema-mount module.

Beyond the two modules that will always be present for an LNE, as an LNE is a network device itself, all modules that may be present at the top-level network device MAY also be present for the LNE. The list of available modules is expected to be implementation dependent, as is the method used by an implementation to support LNEs. Appendix A provides example uses of LNEs.

4. Security Considerations

The YANG modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

LNE information represents device and network configuration information. As such, the security of this information is important, but it is fundamentally no different than any other interface or device configuration information that has already been covered in other documents such as [RFC8343], [RFC7317], and [RFC8349].

The vulnerable "config true" parameters and subtrees are the following:

/logical-network-elements/logical-network-element: This list specifies the logical network element and the related logical device configuration.

/logical-network-elements/logical-network-element/managed: While this leaf is contained in the previous list, it is worth particular attention as it controls whether information under the LNE mount point is accessible by both the host device and within the LNE context. There may be extra sensitivity to this leaf in environments where an LNE is managed by a different party than the host device, and that party does not wish to share LNE information with the operator of the host device.

/if:interfaces/if:interface/bind-lne-name: This leaf indicates the LNE instance to which an interface is assigned. Implementations should pay particular attention to when changes to this leaf are permitted as removal of an interface from an LNE can have a major impact on the LNE's operation as it is similar to physically removing an interface from the device. Implementations can reject a reassignment using the previously described error message generation.

Unauthorized access to any of these lists can adversely affect the security of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

5. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688].

```
URI: urn:ietf:params:xml:ns:yang:ietf-logical-network-element
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
```

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

```
name:         ietf-logical-network-element
namespace:    urn:ietf:params:xml:ns:yang:ietf-logical-network-element
prefix:       lne
reference:    RFC 8530
```

6. Logical Network Element Model

The structure of the model defined in this document is described by the YANG module below.

```
<CODE BEGINS> file "ietf-logical-network-element@2019-01-25.yang"
module ietf-logical-network-element {
  yang-version 1.1;

  // namespace

  namespace "urn:ietf:params:xml:ns:yang:ietf-logical-network-element";
  prefix lne;

  // import some basic types

  import ietf-interfaces {
    prefix if;
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }
  import ietf-yang-schema-mount {
    prefix yangmnt;
    reference
      "RFC 8528: YANG Schema Mount";
  }
```

```
   organization
     "IETF Routing Area (rtgwg) Working Group";
   contact
     "WG Web:   <https://datatracker.ietf.org/wg/rtgwg/>
      WG List:  <mailto:rtgwg@ietf.org>

      Author:   Lou Berger
                <mailto:lberger@labn.net>

      Author:   Christian Hopps
                <mailto:chopps@chopps.org>

      Author:   Acee Lindem
                <mailto:acee@cisco.com>

      Author:   Dean Bogdanovic
                <mailto:ivandean@gmail.com>";
   description
     "This module is used to support multiple logical network
      elements on a single physical or virtual system.

      Copyright (c) 2019 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 RFC 8530; see
      the RFC itself for full legal notices.";

   revision 2019-01-25 {
     description
       "Initial revision.";
     reference
       "RFC 8530: YANG Model for Logical Network Elements";
   }

   // top level device definition statements

   container logical-network-elements {
     description
       "Allows a network device to support multiple logical
        network element (device) instances.";
     list logical-network-element {
```

```
      key "name";
      description
        "List of logical network elements.";
      leaf name {
        type string;
        description
          "Device-wide unique identifier for the
           logical network element.";
      }
      leaf managed {
        type boolean;
        default "true";
        description
          "True if the host can access LNE information
           using the root mount point.  This value
           may not be modifiable in all implementations.";
      }
      leaf description {
        type string;
        description
          "Description of the logical network element.";
      }
      container root {
        description
          "Container for mount point.";
        yangmnt:mount-point "root" {
          description
            "Root for models supported per logical
             network element.  This mount point may or may not
             be inline based on the server implementation.  It
             SHALL always contain a YANG library and interfaces
             instance.

             When the associated 'managed' leaf is 'false', any
             operation that attempts to access information below
             the root SHALL fail with an error-tag of
             'access-denied' and an error-app-tag of
             'lne-not-managed'.";
        }
      }
    }
  }

  // augment statements

  augment "/if:interfaces/if:interface" {
    description
      "Add a node for the identification of the logical network
```

```
        element associated with an interface.  Applies to
        interfaces that can be assigned per logical network
        element.

        Note that a standard error will be returned if the
        identified leafref isn't present.  If an interface
        cannot be assigned for any other reason, the operation
        SHALL fail with an error-tag of 'operation-failed' and an
        error-app-tag of 'lne-assignment-failed'.  A meaningful
        error-info that indicates the source of the assignment
        failure SHOULD also be provided.";
      leaf bind-lne-name {
        type leafref {
          path "/logical-network-elements/logical-network-element/name";
        }
        description
          "Logical network element ID to which the interface is
           bound.";
      }
    }

    // notification statements

    notification bind-lne-name-failed {
      description
        "Indicates an error in the association of an interface to an
         LNE.  Only generated after success is initially returned
         when bind-lne-name is set.";
      leaf name {
        type leafref {
          path "/if:interfaces/if:interface/if:name";
        }
        mandatory true;
        description
          "Contains the interface name associated with the
           failure.";
      }
      leaf bind-lne-name {
        type leafref {
          path "/if:interfaces/if:interface/lne:bind-lne-name";
        }
        mandatory true;
        description
          "Contains the bind-lne-name associated with the
           failure.";
      }
      leaf error-info {
        type string;
```

```
       description
         "Optionally, indicates the source of the assignment
          failure.";
     }
   }
 }

 <CODE ENDS>
```

7. References

7.1. Normative References

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8528] Bjorklund, M. and L. Lhotka, "YANG Schema Mount", RFC 8528, DOI 10.17487/RFC8528, March 2019, <https://www.rfc-editor.org/info/rfc8528>.

7.2. Informative References

[DEVICE-MODEL] Lindem, A., Berger, L., Bogdanovic, D., and C. Hopps, "Network Device YANG Logical Organization", Work in Progress, draft-ietf-rtgwg-device-model-02, March 2017.

[OSPF-YANG] Yeung, D., Qu, Y., Zhang, Z., Chen, I., and A. Lindem, "YANG Data Model for OSPF Protocol", Work in Progress, draft-ietf-ospf-yang-21, January 2019.

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

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

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8348] Bierman, A., Bjorklund, M., Dong, J., and D. Romascanu, "A YANG Data Model for Hardware Management", RFC 8348, DOI 10.17487/RFC8348, March 2018, <https://www.rfc-editor.org/info/rfc8348>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8529] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Data Model for Network Instances", RFC 8529, DOI 10.17487/RFC8529, March 2019, <https://www.rfc-editor.org/info/rfc8529>.

Appendix A. Examples

The following subsections provide example uses of LNEs.

A.1. Example: Host-Device-Managed LNE

This section describes an example of the LNE model using schema mount to achieve the parent management. An example device supports multiple instances of LNEs (logical routers), each of which supports features of Layer 2 and Layer 3 interfaces [RFC8343], a routing information base [RFC8349], and the OSPF protocol [OSPF-YANG]. Each of these features is specified by a YANG model, and they are combined using the YANG schema mount as shown below. Not all possible mounted modules are shown. For example, implementations could also mount the model defined in [RFC8348].

```
module: ietf-logical-network-element
  +--rw logical-network-elements
     +--rw logical-network-element* [name]
        +--rw name           string
        +--mp root
           +--ro yanglib:modules-state/
           |  +--ro module-set-id    string
           |  +--ro module* [name revision]
           |     +--ro name                yang:yang-identifier
           +--rw sys:system/
           |  +--rw contact?          string
           |  +--rw hostname?         inet:domain-name
           |  +--rw authentication {authentication}?
           |     +--rw user-authentication-order*   identityref
           |     +--rw user* [name] {local-users}?
           |        +--rw name              string
           |        +--rw password?         ianach:crypt-hash
           |        +--rw authorized-key* [name]
           |           +--rw name         string
           |           +--rw algorithm    string
           |           +--rw key-data     binary
           +--ro sys:system-state/
           |     ...
           +--rw rt:routing/
           |  +--rw router-id?                yang:dotted-quad
           |  +--rw control-plane-protocols
           |     +--rw control-plane-protocol* [type name]
           |        +--rw ospf:ospf/
           |          +--rw areas
           |             +--rw area* [area-id]
           |                +--rw interfaces
           |                   +--rw interface* [name]
           |                      +--rw name if:interface-ref
           |                      +--rw cost?   uint16
           |
           +--rw if:interfaces/
              +--rw interface* [name]
                 +--rw name            string
                 +--rw ip:ipv4!/
                 |  +--rw address* [ip]
                 |  ...
```

```
module: ietf-interfaces
  +--rw interfaces
     +--rw interface* [name]
        +--rw name                        string
        +--rw lne:bind-lne-name?          string
        +--ro oper-status        enumeration

module: ietf-yang-library
  +--ro modules-state
     +--ro module-set-id    string
     +--ro module* [name revision]
        +--ro name                yang:yang-identifier

module: ietf-system
  +--rw system
  |  +--rw contact?          string
  |  +--rw hostname?         inet:domain-name
  |  +--rw authentication {authentication}?
  |     +--rw user-authentication-order*   identityref
  |     +--rw user* [name] {local-users}?
  |        +--rw name              string
  |        +--rw password?         ianach:crypt-hash
  |        +--rw authorized-key* [name]
  |           +--rw name         string
  |           +--rw algorithm    string
  |           +--rw key-data     binary
  +--ro system-state
     +--ro platform
     |  +--ro os-name?      string
     |  +--ro os-release?   string
```

To realize the above schema, the example device implements the following schema mount instance:

```
"ietf-yang-schema-mount:schema-mounts": {
  "mount-point": [
    {
      "module": "ietf-logical-network-element",
      "label": "root",
      "shared-schema":  {}
    }
  ]
}
```

By using the implementation of the YANG schema mount, an operator can create instances of logical routers. An interface can be assigned to a logical router, so that the logical router has the permission to access this interface. The OSPF protocol can then be enabled on this assigned interface.

For this implementation, a parent management session has access to the schemas of both the parent hosting system and the child logical routers. In addition, each child logical router can grant its own management sessions, which have the following schema:

```
module: ietf-yang-library
  +--ro modules-state
     +--ro module-set-id    string
     +--ro module* [name revision]
        +--ro name                yang:yang-identifier

module: ietf-system
  +--rw system
  |  +--rw contact?          string
  |  +--rw hostname?         inet:domain-name
  |  +--rw authentication {authentication}?
  |     +--rw user-authentication-order*   identityref
  |     +--rw user* [name] {local-users}?
  |        +--rw name              string
  |        +--rw password?         ianach:crypt-hash
  |        +--rw authorized-key* [name]
  |           +--rw name         string
  |           +--rw algorithm    string
  |           +--rw key-data     binary
  +--ro system-state
     +--ro platform
        +--ro os-name?      string
        +--ro os-release?   string

module: ietf-routing
  rw-- routing
     +--rw router-id?                 yang:dotted-quad
     +--rw control-plane-protocols
        +--rw control-plane-protocol* [type name]
           +--rw ospf:ospf/
             +--rw areas
                +--rw area* [area-id]
                   +--rw interfaces
                      +--rw interface* [name]
                         +--rw name    if:interface-ref
                         +--rw cost?   uint16
```

```
module: ietf-interfaces
  +--rw interfaces
     +--rw interface* [name]
        +--rw name                        string
        +--ro oper-status        enumeration
```

A.1.1. Configuration Data

The following shows an example where two customer-specific LNEs are configured:

```
{
  "ietf-logical-network-element:logical-network-elements": {
    "logical-network-element": [
      {
        "name": "cust1",
        "root": {
          "ietf-system:system": {
            "authentication": {
              "user": [
                {
                  "name": "john",
                  "password": "$0$password"
                }
              ]
            }
          },
          "ietf-routing:routing": {
            "router-id": "192.0.2.1",
            "control-plane-protocols": {
              "control-plane-protocol": [
                {
                  "type": "ietf-routing:ospf",
                  "name": "1",
                  "ietf-ospf:ospf": {
                    "af": "ipv4",
                    "areas": {
                      "area": [
                        {
                          "area-id": "203.0.113.1",
                          "interfaces": {
                            "interface": [
                              {
                                "name": "eth1",
                                "cost": 10
                              }
                            ]
                          }
```

```
                        }
                      ]
                    }
                  }
                }
              ]
            }
          },
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth1",
                "type": "iana-if-type:ethernetCsmacd",
                "ietf-ip:ipv4": {
                  "address": [
                    {
                      "ip": "192.0.2.11",
                      "prefix-length": 24
                    }
                  ]
                },
                "ietf-ip:ipv6": {
                  "address": [
                    {
                      "ip": "2001:db8:0:2::11",
                      "prefix-length": 64
                    }
                  ]
                }
              }
            ]
          }
        }
      },
      {
        "name": "cust2",
        "root": {
          "ietf-system:system": {
            "authentication": {
              "user": [
                {
                  "name": "john",
                  "password": "$0$password"
                }
              ]
            }
          },
          "ietf-routing:routing": {
```

```
            "router-id": "192.0.2.2",
            "control-plane-protocols": {
              "control-plane-protocol": [
                {
                  "type": "ietf-routing:ospf",
                  "name": "1",
                  "ietf-ospf:ospf": {
                    "af": "ipv4",
                    "areas": {
                      "area": [
                        {
                          "area-id": "203.0.113.1",
                          "interfaces": {
                            "interface": [
                              {
                                "name": "eth1",
                                "cost": 10
                              }
                            ]
                          }
                        }
                      ]
                    }
                  }
                }
              ]
            }
          },
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth1",
                "type": "iana-if-type:ethernetCsmacd",
                "ietf-ip:ipv4": {
                  "address": [
                    {
                      "ip": "192.0.2.11",
                      "prefix-length": 24
                    }
                  ]
                }
              }
            ]
          }
        }
      ]
    },
```

```
    "ietf-interfaces:interfaces": {
      "interface": [
        {
          "name": "eth0",
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-ip:ipv4": {
            "address": [
              {
                "ip": "192.0.2.10",
                "prefix-length": 24
              }
            ]
          },
          "ietf-ip:ipv6": {
            "address": [
              {
                "ip": "2001:db8:0:2::10",
                "prefix-length": 64
              }
            ]
          }
        },
        {
          "name": "cust1:eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-logical-network-element:bind-lne-name": "cust1"
        },
        {
          "name": "cust2:eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-logical-network-element:bind-lne-name": "cust2"
        }
      ]
    },

    "ietf-system:system": {
      "authentication": {
        "user": [
          {
            "name": "root",
            "password": "$0$password"
          }
        ]
      }
    }
  }
```

A.1.2. State Data

The following shows possible state data associated with the above configuration data:

```
{
  "ietf-logical-network-element:logical-network-elements": {
    "logical-network-element": [
      {
        "name": "cust1",
        "root": {
          "ietf-yang-library:modules-state": {
            "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
            "module": [
              {
                "name": "iana-if-type",
                "revision": "2014-05-08",
                "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
                "conformance-type": "import"
              },
              {
                "name": "ietf-inet-types",
                "revision": "2013-07-15",
                "namespace":
                  "urn:ietf:params:xml:ns:yang:ietf-inet-types",
                "conformance-type": "import"
              },
              {
                "name": "ietf-interfaces",
                "revision": "2014-05-08",
                "feature": [
                  "arbitrary-names",
                  "pre-provisioning"
                ],
                "namespace":
                  "urn:ietf:params:xml:ns:yang:ietf-interfaces",
                "conformance-type": "implement"
              },
              {
                "name": "ietf-ip",
                "revision": "2014-06-16",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
                "conformance-type": "implement"
              },
              {
                "name": "ietf-ospf",
                "revision": "2018-03-03",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-ospf",
```

```
            "conformance-type": "implement"
          },
          {
            "name": "ietf-routing",
            "revision": "2018-03-13",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
            "conformance-type": "implement"
          },
          {
            "name": "ietf-system",
            "revision": "2014-08-06",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
            "conformance-type": "implement"
          },
          {
            "name": "ietf-yang-library",
            "revision": "2016-06-21",
            "namespace":
              "urn:ietf:params:xml:ns:yang:ietf-yang-library",
            "conformance-type": "implement"
          },
          {
            "name": "ietf-yang-types",
            "revision": "2013-07-15",
            "namespace":
              "urn:ietf:params:xml:ns:yang:ietf-yang-types",
            "conformance-type": "import"
          }
        ]
      },
      "ietf-system:system-state": {
        "platform": {
          "os-name": "NetworkOS"
        }
      },
      "ietf-routing:routing": {
        "router-id": "192.0.2.1",
        "control-plane-protocols": {
          "control-plane-protocol": [
            {
              "type": "ietf-routing:ospf",
              "name": "1",
              "ietf-ospf:ospf": {
                "af": "ipv4",
                "areas": {
                  "area": [
                    {
                      "area-id": "203.0.113.1",
```

```
                        "interfaces": {
                          "interface": [
                            {
                              "name": "eth1",
                              "cost": 10
                            }
                          ]
                        }
                      }
                    ]
                  }
                }
              }
            ]
          }
        },
        "ietf-interfaces:interfaces": {
          "interface": [
            {
              "name": "eth1",
              "type": "iana-if-type:ethernetCsmacd",
              "oper-status": "up",
              "phys-address": "00:01:02:A1:B1:C1",
              "statistics": {
                "discontinuity-time": "2017-06-26T12:34:56-05:00"
              },
              "ietf-ip:ipv4": {
                "address": [
                  {
                    "ip": "192.0.2.11",
                    "prefix-length": 24
                  }
                ]
              },
              "ietf-ip:ipv6": {
                "address": [
                  {
                    "ip": "2001:db8:0:2::11",
                    "prefix-length": 64
                  }
                ]
              }
            }
          ]
        }
      }
    },
    {
```

```
        "name": "cust2",
        "root": {
          "ietf-yang-library:modules-state": {
            "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
            "module": [
              {
                "name": "iana-if-type",
                "revision": "2014-05-08",
                "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
                "conformance-type": "import"
              },
              {
                "name": "ietf-inet-types",
                "revision": "2013-07-15",
                "namespace":
                  "urn:ietf:params:xml:ns:yang:ietf-inet-types",
                "conformance-type": "import"
              },
              {
                "name": "ietf-interfaces",
                "revision": "2014-05-08",
                "feature": [
                  "arbitrary-names",
                  "pre-provisioning"
                ],
                "namespace":
                  "urn:ietf:params:xml:ns:yang:ietf-interfaces",
                "conformance-type": "implement"
              },
              {
                "name": "ietf-ip",
                "revision": "2014-06-16",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
                "conformance-type": "implement"
              },
              {
                "name": "ietf-ospf",
                "revision": "2018-03-03",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-ospf",
                "conformance-type": "implement"
              },
              {
                "name": "ietf-routing",
                "revision": "2018-03-13",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
                "conformance-type": "implement"
              },
              {
```

```
            "name": "ietf-system",
            "revision": "2014-08-06",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
            "conformance-type": "implement"
          },
          {
            "name": "ietf-yang-library",
            "revision": "2016-06-21",
            "namespace":
              "urn:ietf:params:xml:ns:yang:ietf-yang-library",
            "conformance-type": "implement"
          },
          {
            "name": "ietf-yang-types",
            "revision": "2013-07-15",
            "namespace":
              "urn:ietf:params:xml:ns:yang:ietf-yang-types",
            "conformance-type": "import"
          }
        ]
      },
      "ietf-system:system-state": {
        "platform": {
          "os-name": "NetworkOS"
        }
      },
      "ietf-routing:routing": {
        "router-id": "192.0.2.2",
        "control-plane-protocols": {
          "control-plane-protocol": [
            {
              "type": "ietf-routing:ospf",
              "name": "1",
              "ietf-ospf:ospf": {
                "af": "ipv4",
                "areas": {
                  "area": [
                    {
                      "area-id": "203.0.113.1",
                      "interfaces": {
                        "interface": [
                          {
                            "name": "eth1",
                            "cost": 10
                          }
                        ]
                      }
                    }
```

```
                    ]
                  }
                }
              }
            ]
          }
        },
        "ietf-interfaces:interfaces": {
          "interface": [
            {
              "name": "eth1",
              "type": "iana-if-type:ethernetCsmacd",
              "oper-status": "up",
              "phys-address": "00:01:02:A1:B1:C2",
              "statistics": {
                "discontinuity-time": "2017-06-26T12:34:56-05:00"
              },
              "ietf-ip:ipv4": {
                "address": [
                  {
                    "ip": "192.0.2.11",
                    "prefix-length": 24
                  }
                ]
              }
            }
          ]
        }
      }
    ]
  },

  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "phys-address": "00:01:02:A1:B1:C0",
        "statistics": {
          "discontinuity-time": "2017-06-26T12:34:56-05:00"
        },
        "ietf-ip:ipv4": {
          "address": [
            {
              "ip": "192.0.2.10",
              "prefix-length": 24
```

```
            }
          ]
        },
        "ietf-ip:ipv6": {
          "address": [
            {
              "ip": "2001:db8:0:2::10",
              "prefix-length": 64
            }
          ]
        }
      },
      {
        "name": "cust1:eth1",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "phys-address": "00:01:02:A1:B1:C1",
        "statistics": {
          "discontinuity-time": "2017-06-26T12:34:56-05:00"
        },
        "ietf-logical-network-element:bind-lne-name": "cust1"
      },
      {
        "name": "cust2:eth1",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "phys-address": "00:01:02:A1:B1:C2",
        "statistics": {
          "discontinuity-time": "2017-06-26T12:34:56-05:00"
        },
        "ietf-logical-network-element:bind-lne-name": "cust2"
      }
    ]
  },

  "ietf-system:system-state": {
    "platform": {
      "os-name": "NetworkOS"
    }
  },

  "ietf-yang-library:modules-state": {
    "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
    "module": [
      {
        "name": "iana-if-type",
        "revision": "2014-05-08",
        "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
```

```
        "conformance-type": "import"
      },
      {
        "name": "ietf-inet-types",
        "revision": "2013-07-15",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
        "conformance-type": "import"
      },
      {
        "name": "ietf-interfaces",
        "revision": "2014-05-08",
        "feature": [
          "arbitrary-names",
          "pre-provisioning"
        ],
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-ip",
        "revision": "2014-06-16",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-logical-network-element",
        "revision": "2017-03-13",
        "feature": [
          "bind-lne-name"
        ],
        "namespace":
          "urn:ietf:params:xml:ns:yang:ietf-logical-network-element",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-ospf",
        "revision": "2018-03-03",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ospf",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-routing",
        "revision": "2018-03-13",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-system",
```

```
        "revision": "2014-08-06",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-yang-library",
        "revision": "2016-06-21",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-yang-schema-mount",
        "revision": "2017-05-16",
        "namespace":
          "urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-yang-types",
        "revision": "2013-07-15",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
        "conformance-type": "import"
      }
    ]
  },

  "ietf-yang-schema-mount:schema-mounts": {
    "mount-point": [
      {
        "module": "ietf-logical-network-element",
        "label": "root",
        "shared-schema": {}
      }
    ]
  }
}
```

A.2. Example: Self-Managed LNE

This section describes an example of the LNE model using schema mount to achieve child-independent management. An example device supports multiple instances of LNEs (logical routers), each of which has the features of Layer 2 and Layer 3 interfaces [RFC8343], a routing information base [RFC8349], and the OSPF protocol. Each of these features is specified by a YANG model, and they are put together by the YANG schema mount as follows:

```
module: ietf-logical-network-element
  +--rw logical-network-elements
     +--rw logical-network-element* [name]
        +--rw name           string
        +--mp root
           // The internal modules of the LNE are not visible to
           // the parent management.
           // The child manages its modules, including ietf-routing
           // and ietf-interfaces

module: ietf-interfaces
  +--rw interfaces
     +--rw interface* [name]
        +--rw name                          string
        +--rw lne:bind-lne-name?            string
        +--ro oper-status         enumeration

module: ietf-yang-library
  +--ro modules-state
     +--ro module-set-id    string
     +--ro module* [name revision]
        +--ro name                yang:yang-identifier

module: ietf-system
  +--rw system
  |  +--rw contact?          string
  |  +--rw hostname?         inet:domain-name
  |  +--rw authentication {authentication}?
  |     +--rw user-authentication-order*   identityref
  |     +--rw user* [name] {local-users}?
  |        +--rw name              string
  |        +--rw password?         ianach:crypt-hash
  |        +--rw authorized-key* [name]
  |           +--rw name          string
  |           +--rw algorithm     string
  |           +--rw key-data      binary
  +--ro system-state
     +--ro platform
     |  +--ro os-name?      string
     |  +--ro os-release?   string
```

To realize the above schema, the device implements the following schema mount instance:

```
"ietf-yang-schema-mount:schema-mounts": {
  "mount-point": [
    {
      "module": "ietf-logical-network-element",
      "label": "root",
      "inline": {}
    }
  ]
}
```

By using the implementation of the YANG schema mount, an operator can create instances of logical routers, each with their logical router-specific inline modules. An interface can be assigned to a logical router, so that the logical router has the permission to access this interface. The OSPF protocol can then be enabled on this assigned interface. Each logical router independently manages its own set of modules, which may or may not be the same as other logical routers. The following is an example of schema set implemented on one particular logical router:

```
    module: ietf-yang-library
      +--ro modules-state
         +--ro module-set-id    string
         +--ro module* [name revision]
            +--ro name                yang:yang-identifier

    module: ietf-system
      +--rw system
      |  +--rw contact?          string
      |  +--rw hostname?         inet:domain-name
      |  +--rw authentication {authentication}?
      |     +--rw user-authentication-order*   identityref
      |     +--rw user* [name] {local-users}?
      |        +--rw name              string
      |        +--rw password?         ianach:crypt-hash
      |        +--rw authorized-key* [name]
      |           +--rw name         string
      |           +--rw algorithm    string
      |           +--rw key-data     binary
      +--ro system-state
         +--ro platform
         |  +--ro os-name?      string
         |  +--ro os-release?   string

    module: ietf-routing
      +--rw routing
         +--rw router-id?                 yang:dotted-quad
         +--rw control-plane-protocols
            +--rw control-plane-protocol* [type name]
               +--rw ospf:ospf/
                  +--rw areas
                    +--rw area* [area-id]
                       +--rw interfaces
                          +--rw interface* [name]
                             +--rw name    if:interface-ref
                             +--rw cost?   uint16

    module: ietf-interfaces
      +--rw interfaces
         +--rw interface* [name]
            +--rw name                        string
            +--ro oper-status        enumeration
```

A.2.1. Configuration Data

Each of the child virtual routers is managed through its own sessions and configuration data.

A.2.1.1. Logical Network Element 'vnf1'

The following shows an example of configuration data for the LNE name "vnf1":

```
{
  "ietf-system:system": {
    "authentication": {
      "user": [
        {
          "name": "john",
          "password": "$0$password"
        }
      ]
    }
  },
  "ietf-routing:routing": {
    "router-id": "192.0.2.1",
    "control-plane-protocols": {
      "control-plane-protocol": [
        {
          "type": "ietf-routing:ospf",
          "name": "1",
          "ietf-ospf:ospf": {
            "af": "ipv4",
            "areas": {
              "area": [
                {
                  "area-id": "203.0.113.1",
                  "interfaces": {
                    "interface": [
                      {
                        "name": "eth1",
                        "cost": 10
                      }
                    ]
                  }
                }
              ]
            }
          }
        }
      ]
    }
  },
  "ietf-interfaces:interfaces": {
    "interface": [
      {
```

```
            "name": "eth1",
            "type": "iana-if-type:ethernetCsmacd",
            "ietf-ip:ipv4": {
              "address": [
                {
                  "ip": "192.0.2.11",
                  "prefix-length": 24
                }
              ]
            }
          }
        ]
      }
    }
```

A.2.1.2. Logical Network Element 'vnf2'

The following shows an example of configuration data for the LNE name "vnf2":

```
{
  "ietf-system:system": {
    "authentication": {
      "user": [
        {
          "name": "john",
          "password": "$0$password"
        }
      ]
    }
  },
  "ietf-routing:routing": {
    "router-id": "192.0.2.2",
    "control-plane-protocols": {
      "control-plane-protocol": [
        {
          "type": "ietf-routing:ospf",
          "name": "1",
          "ietf-ospf:ospf": {
            "af": "ipv4",
            "areas": {
              "area": [
                {
                  "area-id": "203.0.113.1",
                  "interfaces": {
                    "interface": [
                      {
                        "name": "eth1",
```

```
                          "cost": 10
                        }
                      ]
                    }
                  }
                ]
              }
            }
          }
        ]
      }
    },
    "ietf-interfaces:interfaces": {
      "interface": [
        {
          "name": "eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-ip:ipv4": {
            "address": [
              {
                "ip": "192.0.2.11",
                "prefix-length": 24
              }
            ]
          }
        }
      ]
    }
  }
```

A.2.2. State Data

The following sections show possible state data associated with the above configuration data. Note that there are three views: the host device's view and each of the LNE's views.

A.2.2.1. Host Device

The following shows state data for the device hosting the example LNEs:

```
  {
    "ietf-logical-network-element:logical-network-elements": {
      "logical-network-element": [
        {
          "name": "vnf1",
          "root": {
          }
```

```
        },
        {
          "name": "vnf2",
          "root": {
          }
        }
      ]
    },

    "ietf-interfaces:interfaces": {
      "interface": [
        {
          "name": "eth0",
          "type": "iana-if-type:ethernetCsmacd",
          "oper-status": "up",
          "phys-address": "00:01:02:A1:B1:C0",
          "statistics": {
            "discontinuity-time": "2017-06-26T12:34:56-05:00"
          },
          "ietf-ip:ipv4": {
            "address": [
              {
                "ip": "192.0.2.10",
                "prefix-length": 24
              }
            ]
          }
        },
        {
          "name": "vnf1:eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "oper-status": "up",
          "phys-address": "00:01:02:A1:B1:C1",
          "statistics": {
            "discontinuity-time": "2017-06-26T12:34:56-05:00"
          },
          "ietf-logical-network-element:bind-lne-name": "vnf1"
        },
        {
          "name": "vnf2:eth2",
          "type": "iana-if-type:ethernetCsmacd",
          "oper-status": "up",
          "phys-address": "00:01:02:A1:B1:C2",
          "statistics": {
            "discontinuity-time": "2017-06-26T12:34:56-05:00"
          },
          "ietf-logical-network-element:bind-lne-name": "vnf2"
        }
```

```
      ]
    },

    "ietf-system:system-state": {
      "platform": {
        "os-name": "NetworkOS"
      }
    },

    "ietf-yang-library:modules-state": {
      "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
      "module": [
        {
          "name": "iana-if-type",
          "revision": "2014-05-08",
          "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
          "conformance-type": "import"
        },
        {
          "name": "ietf-inet-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
          "conformance-type": "import"
        },
        {
          "name": "ietf-interfaces",
          "revision": "2014-05-08",
          "feature": [
            "arbitrary-names",
            "pre-provisioning"
          ],
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-ip",
          "revision": "2014-06-16",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-logical-network-element",
          "revision": "2017-03-13",
          "feature": [
            "bind-lne-name"
          ],
          "namespace":
            "urn:ietf:params:xml:ns:yang:ietf-logical-network-element",
```

```
          "conformance-type": "implement"
        },
        {
          "name": "ietf-system",
          "revision": "2014-08-06",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-library",
          "revision": "2016-06-21",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-schema-mount",
          "revision": "2017-05-16",
          "namespace":
            "urn:ietf:params:xml:ns:yang:ietf-yang-schema-mount",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
          "conformance-type": "import"
        }
      ]
    },

    "ietf-yang-schema-mount:schema-mounts": {
      "mount-point": [
        {
          "module": "ietf-logical-network-element",
          "label": "root",
          "inline": {}
        }
      ]
    }
  }
```

A.2.2.2. Logical Network Element 'vnf1'

The following shows state data for the example LNE with the name "vnf1":

```
{
  "ietf-yang-library:modules-state": {
    "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
    "module": [
      {
        "name": "iana-if-type",
        "revision": "2014-05-08",
        "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
        "conformance-type": "import"
      },
      {
        "name": "ietf-inet-types",
        "revision": "2013-07-15",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
        "conformance-type": "import"
      },
      {
        "name": "ietf-interfaces",
        "revision": "2014-05-08",
        "feature": [
          "arbitrary-names",
          "pre-provisioning"
        ],
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-ip",
        "revision": "2014-06-16",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-ospf",
        "revision": "2018-03-03",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ospf",
        "conformance-type": "implement"
      },
      {
        "name": "ietf-routing",
        "revision": "2018-03-13",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
        "conformance-type": "implement"
```

```
        },
        {
          "name": "ietf-system",
          "revision": "2014-08-06",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-library",
          "revision": "2016-06-21",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
          "conformance-type": "import"
        }
      ]
    },

    "ietf-system:system-state": {
      "platform": {
        "os-name": "NetworkOS"
      }
    },

    "ietf-routing:routing": {
      "router-id": "192.0.2.1",
      "control-plane-protocols": {
        "control-plane-protocol": [
          {
            "type": "ietf-routing:ospf",
            "name": "1",
            "ietf-ospf:ospf": {
              "af": "ipv4",
              "areas": {
                "area": [
                  {
                    "area-id": "203.0.113.1",
                    "interfaces": {
                      "interface": [
                        {
                          "name": "eth1",
                          "cost": 10
                        }
                      ]
```

```
                    }
                  }
                ]
              }
            }
          }
        ]
      }
    },

    "ietf-interfaces:interfaces": {
      "interface": [
        {
          "name": "eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "oper-status": "up",
          "phys-address": "00:01:02:A1:B1:C1",
          "statistics": {
            "discontinuity-time": "2017-06-26T12:34:56-05:00"
          },
          "ietf-ip:ipv4": {
            "address": [
              {
                "ip": "192.0.2.11",
                "prefix-length": 24
              }
            ]
          }
        }
      ]
    }
  }
```

A.2.2.3. Logical Network Element 'vnf2'

The following shows state data for the example LNE with the name "vnf2":

```
  {
    "ietf-yang-library:modules-state": {
      "module-set-id": "123e4567-e89b-12d3-a456-426655440000",
      "module": [
        {
          "name": "iana-if-type",
          "revision": "2014-05-08",
          "namespace": "urn:ietf:params:xml:ns:yang:iana-if-type",
          "conformance-type": "import"
        },
```

```
        {
          "name": "ietf-inet-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types",
          "conformance-type": "import"
        },
        {
          "name": "ietf-interfaces",
          "revision": "2014-05-08",
          "feature": [
            "arbitrary-names",
            "pre-provisioning"
          ],
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-ip",
          "revision": "2014-06-16",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-ospf",
          "revision": "2018-03-03",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-ospf",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-routing",
          "revision": "2018-03-13",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-system",
          "revision": "2014-08-06",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
          "conformance-type": "implement"
        },
        {
          "name": "ietf-yang-library",
          "revision": "2016-06-21",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
          "conformance-type": "implement"
        },
```

```
        {
          "name": "ietf-yang-types",
          "revision": "2013-07-15",
          "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types",
          "conformance-type": "import"
        }
      ]
    },

    "ietf-system:system-state": {
      "platform": {
        "os-name": "NetworkOS"
      }
    },

    "ietf-routing:routing": {
      "router-id": "192.0.2.2",
      "control-plane-protocols": {
        "control-plane-protocol": [
          {
            "type": "ietf-routing:ospf",
            "name": "1",
            "ietf-ospf:ospf": {
              "af": "ipv4",
              "areas": {
                "area": [
                  {
                    "area-id": "203.0.113.1",
                    "interfaces": {
                      "interface": [
                        {
                          "name": "eth1",
                          "cost": 10
                        }
                      ]
                    }
                  }
                ]
              }
            }
          }
        ]
      }
    },
```

```
    "ietf-interfaces:interfaces": {
      "interface": [
        {
          "name": "eth1",
          "type": "iana-if-type:ethernetCsmacd",
          "oper-status": "up",
          "phys-address": "00:01:02:A1:B1:C2",
          "statistics": {
            "discontinuity-time": "2017-06-26T12:34:56-05:00"
          },
          "ietf-ip:ipv4": {
            "address": [
              {
                "ip": "192.0.2.11",
                "prefix-length": 24
              }
            ]
          }
        }
      ]
    }
  }
```

Acknowledgments

The Routing Area Yang Architecture design team members included Acee Lindem, Anees Shaikh, Christian Hopps, Dean Bogdanovic, Lou Berger, Qin Wu, Rob Shakir, Stephane Litkowski, and Yan Gang. Useful review comments were also received by Martin Bjorklund, John Scudder, Dan Romascanu, and Taylor Yu.

This document was motivated by, and derived from "Network Device YANG Logical Organization" [DEVICE-MODEL].

Thanks to Alvaro Retana for the IESG review.

Authors' Addresses

Lou Berger
LabN Consulting, L.L.C.

Email: lberger@labn.net

Christian Hopps
LabN Consulting, L.L.C.

Email: chopps@chopps.org

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513
United States of America

Email: acee@cisco.com

Dean Bogdanovic
Volta Networks

Email: ivandean@gmail.com

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com