Abstract

This document defines a network instance module. This module along with the logical network element module can be used to manage the logical and virtual resource representations that may be present on a network device. Examples of common industry terms for logical resource representations are Logical Systems or Logical Routers. Examples of common industry terms for virtual resource representations are Virtual Routing and Forwarding (VRF) instances and Virtual Switch Instances (VSIs).

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 November 18, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
1. Introduction

This document defines the second of two new modules that 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. Both make use of emerging YANG functionality supported by YANG Schema Mount [I-D.ietf-netmod-schema-mount]. This document is expected to use whatever Schema Mount solution is agreed upon by the Netmod Working Group.

Two forms of resource partitioning are supported:

The first form, which is defined in [LNE-MODEL], 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 [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, which is defined in this document, 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 below. 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, [RTG-DEVICE-MODEL].

1.1. Status of Work and Open Issues

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. 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.
A model for LNEs is described in [LNE-MODEL] and the model for network instances is covered in Section 3. For more information on how these models may be used within an overall device model structure, see [RTG-DEVICE-MODEL].

The interface management model [RFC7223] is an existing model that is impacted by the definition of LNEs and network instances. This document and [LNE-MODEL] define 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.

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 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 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).
3. Network Instances

The network instance container is used to represent virtual routing and forwarding instances (VRFs) and virtual switching instances (VSIs), [RFC4026]. VRFs and VSIs are commonly used to isolate routing and switching domains, for example to create virtual private networks, each with their own active protocols and routing/switching policies. The model represents both core/provider and virtual instances. Network instances reuse and build on [I-D.ietf-netmod-routing-cfg] and are shown below:

```yaml
module: ietf-network-instance
  +--rw network-instances
    +--rw network-instance* [name]
      +--rw name string
      +--rw type? identityref
      +--rw enabled? boolean
      +--rw description? string
      +--rw network-instance-policy
        | ...  
      +--rw root? schema-mount
        | ...
    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
```

A network instance is identified by a ‘name’ string. This string is used both as an index within the network-instance module and to associate resources with a network instance as shown above in the interface augmentation. Type is used to indicate the type NI, such as L3-VRF, VPLS, L2-VSI, etc. Network instance policy and root are discussed in greater detail below.

3.1. Network Instance Policy

Network instance policies are used to control how NI information is represented at the device level, VRF routing policies, and VRF/VSI identifiers. Examples include BGP route targets (RTs) and route distinguishers (RDs), virtual network identifiers (VN-IDs), VPLS neighbors, etc. The structure is expected to be:
3.2. Network Instance Management

Modules that may be used to represent network instance specific information will be available under 'root'. As with LNEs, actual module availability is expected to be implementation dependent. The yang library module [I-D.ietf-netconf-yang-library] is expected to be the primary method used to identify supported modules. Resource related control and assignment is expected to be managed at the network-device level, not the network instance level, based on the 'bind-network-instance-name' augmentation mentioned above.

As an example, consider the case where a network instance with a 'name' of "green" is defined on a network device. In this case the following structure might be made available:

```yaml
---rw yanglib:modules-state           [I-D.ietf-netconf-yang-library]
  ---rw if:interfaces                   [RFC7223]
    | ---rw bind-network-instance-name="green" string
  ---rw network-instances
    ---rw network-instance* [name]
      ---rw name="green" string
      ---rw type? identityref
      ---rw enabled=true boolean
      ---rw description="The Green VRF" string
      ---rw network-instance-policy
        | ... (RT=1000:1, RD=1.2.3.4)
      ---rw root? schema-mount
        ---rw yanglib:modules-state [I-D.ietf-netconf-yang-library]
        ---rw if:interfaces [RFC7223]
        ---rw mm:network-services
        ---rw nn:oam-protocols
        ---rw oo:routing
        ---rw pp:mpls
```

All modules that represent control-plane and data-plane information may be present at the 'root', and be accessible via paths modified per [I-D.ietf-netmod-schema-mount]. The list of available modules is expected to be implementation dependent. As is the method used by an implementation to support NIs.
3.3. Network Instance Instantiation

TBD -- need to resolve if instantiation is based on new list entry creation per the pending Schema Mount solution definition.

4. Security Considerations

LNE portion is TBD

NI portion is TBD

5. 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].

6. Network Instance Model

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

<CODE BEGINS> file "ietf-network-instance@2016-05-01.yang"
module ietf-network-instance {  
  yang-version "1";
  // namespace
  prefix "ni";
  // import some basic types
  import ietf-interfaces {
    prefix if;
  }
  import ietf-ip {
    prefix ip;
  }
  // meta
  organization "IETF RTG YANG Design Team Collaboration with OpenConfig";

  contact

This module is used to support multiple network instances within a single physical or virtual device. Network instances are commonly known as VRFs (virtual routing and forwarding) and VSIs (virtual switching instances).
"interface-specific configuration for IP interfaces, IPv4 and IPv6";

} grouping ipv4-interface-protocols {
  container ipv4-interface-protocols {
    list ipv4-interface-protocol {
      key "type";
      leaf type {
        type identityref {
          base ipv4-interface-protocol-type;
        }
        mandatory true;
        description
          "ARP, ICMP, VRRP, DHCP Client, etc.";
      }
      description
        "List of IPv4 protocols configured on an interface";
    }
    description
      "Container for list of IPv4 protocols configured on an interface";
  }
  description
    "Grouping for IPv4 protocols configured on an interface";
}

grouping ipv6-interface-protocols {
  description
    "Grouping for IPv6 protocols configured on an interface.";
  container ipv6-interface-protocols {
    description
      "Container for list of IPv6 protocols configured on an interface.";
    list ipv6-interface-protocol {
      key "type";
      description
        "List of IPv6 protocols configured on an interface";
      leaf type {
        type identityref {
          base ipv6-interface-protocol-type;
        }
        mandatory true;
        description
          "List of IPv6 protocols configured on an interface";
      }
      description
        "Container for list of IPv6 protocols configured on an interface";
    }
  }
  description
    "Grouping for IPv6 protocols configured on an interface";
}
"ND, ICMPv6, VRRP, DHCPv6 Client, etc.";
}
}
}


grouping network-instance-policy {
  description
    "Network instance policies such as route
distinguisher, route targets, VPLS ID and neighbor,
Ethernet ID, etc. ";
  reference
    "RFC 4364 - BGP/MPLS Virtual Private Networks (VPNs)
RFC 6074 - Provisioning, Auto-Discovery, and Signaling
in Layer 2 Virtual Private Networks (L2VPNs)
RFC 7432 - BGP MPLS-Based Ethernet VPN";
  container network-instance-policy {
    description "Network Instance Policy -- details TBD";
  }
}

// top level device definition statements
container network-instances {
  description "Network instances each of which have
  an independent IP/IPv6 addressing space
  and protocol instantiations. For layer 3,
  this consistent with the routing-instance
  definition in ietf-routing";
  reference "draft-ietf-netmod-routing-cfg";
  list network-instance {
    key name;
    description "List of network-instances";
    leaf name {
        type string;
        description "device scoped
            identifier for the network
            instance";
    }
    leaf type {
        type identityref {
            base network-instance-type;
        }
        description
            "The network instance type -- details TBD
            Likely types include core, L3-VRF, VPLS,
            L2-cross-connect, L2-VSI, etc.";
    }
    leaf enabled {
        }
leaf description {
  type string;
  description "Description of the network instance and its intended purpose";
}

uses network-instance-policy;
leaf root {
  type schema-mount;
  description "Root for models supported per network instance";
}

// augment statements

augment "/if:interfaces/if:interface" {
  description "Add a node for the identification of the logical network instance (which is within the interface’s identified logical network element) associated with the IP information configured on an interface";

  leaf bind-network-instance-name {
    type string;
    description "Network Instance to which an interface is bound";
  }
}

augment "/if:interfaces/if:interface/ip:ipv4" {
  description "Add a node for the identification of the logical network instance (which is within the interface’s identified physical or virtual device) associated with the IP information configured on an interface";

  leaf bind-network-instance-name {
    type string;
    description "Network Instance to which IPv4 interface is bound";
  }
}
augment "/if:interfaces/if:interface/ip:ipv6" {
  description
  "Add a node for the identification of the logical
  network instance (which is within the interface’s
  identified physical or virtual device) associated with
  the IP information configured on an interface";

  leaf bind-network-instance-name {
    type string;
    description
    "Network Instance to which IPv6 interface is bound";
  }
}

// rpc statements

// notification statements

7. References

7.1. Normative References

[I-D.ietf-netmod-schema-mount]

[LNE-MODEL]


7.2. Informative References

[I-D.ietf-netconf-yang-library]

[I-D.ietf-netmod-opstate-reqs]

[I-D.ietf-netmod-routing-cfg]

[I-D.openconfig-netmod-opstate]

Appendix A. 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.

The RFC text was produced using Marshall Rose’s xml2rfc tool.
Appendix B. Contributors

Contributors’ Addresses

TBD

Authors’ Addresses

Lou Berger
LabN Consulting, L.L.C.

Email: lberger@labn.net

Christian Hopps
Deutsche Telekom

Email: chopps@chopps.org

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC  27513
USA

Email: acee@cisco.com

Dean Bogdanovic

Email: ivandean@gmail.com