Routing Area Yang Architecture
Design Team Update

Members:  Acee Lindem, Anees Shaikh, Christian Hopps, Dean Bogdanovic, Ebban Aries, Lou Berger, Qin Wu, Rob Shakir, Xufeng Liu, Yingzhen Qu

Wiki:  http://trac.tools.ietf.org/area/rtg/trac/wiki/RtgYangArchDT
Repo:  https://github.com/ietf-rtg-area-yang-arch-dt/
Design Team Status Summary

- Routing WG Standards Track Drafts
- Other Design Team Drafts
- NDMA Status
- Relevant Non-Design Team Drafts
- YANG Network Instance Draft Update
- YANG Tags Draft Update
- Target To Conclude Design Team at IETF 101
Routing WG Standards Track Drafts

- Routing Area Common YANG Data Types  
  – draft-ietf-rtgwg-routing-types-17  
  • On RFC Editor Queue
- YANG Logical Network Elements  
  – draft-ietf-rtgwg-lne-model-04  
  • Dependent on NETMOD YANG Schema Mount  
  • Minimal change to put schema mount point in container  
  • Working Group Last Call complete
- YANG Network Instances  
  – draft-ietf-rtgwg-ni-model-04  
  • Also Dependent on NETMOD YANG Schema Mount  
  • Update Later  
  • Working Group Last Call complete
Other Design Team Drafts

• YANG Module Tags
  • draft-rtgyangdt-netmod-module-tags-02
  • Work proceeding in NETMOD
  • Needs Review

• Network Device YANG Logical Organization
  • draft-ietf-rtgwg-device-model-02
  • Expired – will not be progressed in current form with fixed device hierarchy (we will never reach consensus on one overall YANG model structure for network devices)
  • Could be revived based but based on YANG Module Tags
Network Management Datastore Architecture (NDMA)

• Fully embraced by IETF Routing Area
• RFC 8022 BIS
  • Accepted NETMOD WG document
  • Modules revised rather than renaming (e.g., ietf-routing-2)
  • State Trees Retained (e.g., ietf-routing-state)
  • Status state trees is “obsolete” rather than “deprecated” since there is minimal implementation
• Routing Area YANG models should augment this version of draft.
Non-Design Team Drafts/Issues

- NETMOD Schema Mount
  - Key enabler for Nis and LNEs
  - WG LC completed being discussed in NetMod
- YANG Tree Diagrams
  - Out growth of DT work, but not a DT draft
  - Being discussed in NetMod
- Open issues left from original (private and public list)
  - Handling YANG module revisions
  - Being discussed in NetMod

- Do we (the routing area) need a DT position on this?
Network Instance (NI) Update

- Changes to YANG tree representation of schema mount
- Schema mount points put in containers as required
- Server implementation of schema mount restrictions example
  - Uses parent-reference to limit interface instances to those with bind-network-instance-name equal to current network-instance name.
- L3VPN YANG model aligned with YANG Network Instance model
- Discussions with L2VPN/EVPN YANG model authors initiated
  - Proposed refactoring with instance specific information under ni-type and non-instance core information into new (top-level) modules
- Working Group Last complete
  - Minor changes to examples requested
Logical Network Element (LNE) Update

- Changes to YANG tree representation of schema mount
- Schema mount points put in containers as required
- Server implementation of schema mount restrictions example

- Working Group Last complete
  - Minor changes to examples requested
YANG Tags Draft Update

- Provides user controllable hashtags on modules
- Could be used to support the logical device model
- Being discussed in NetMod
  - Still an individual draft
Future Plans

• Focus on YANG revisions, based on today’s discussion
• Formally disband before IETF 101
• Support documents through publication
  • LNIs, NIs, Types, 8022bis, Schema Mount, Trees
**LNEs and NIs: Modeling Device Partitioning**

**LNE: Logical Network Element**
- Separate management sub-domains
  - Sub-domains can be **managed independently** and by a top level manager \( \text{managed=true} \)
  - Commonly called logical system or router; or virtual switch, chassis, fabric, or device context
- Can be supported via multiple logical devices and VMs
  - Where only limited top level management of subdomains is supported

**NI: Network Instance**
- Separate routing / switching domains
  - Can represent of an RFC 4364 VRF or a Layer 2 Virtual Switch Instance (VSI) or a bridge/router (i.e., both)
  - General virtualized instance implying a separate L2, L3, or L2/L3 context.
    - For L3, this implies a unique IPv4/IPv6 address space.
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
    -> /if:interfaces/interface/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

LNE Root only used when managed=true

Covers cases of asynchronous interface NI bind failures
Implementation example: LNE
Raymond Cheh, Dean Bogdanovic
LNE Architecture Example

Device1

LNE Router 2

LNE Router 1

Device2
Adding Interfaces to LNE

```plaintext
grouping interface-grp {
  list interface {
    key name;
    leaf name {
      type string;
    }
    leaf-list address {
      type inet:ip-prefix;
    }
    leaf iso-address {
      type string;
    }
  }
}
list logical-network-element {
  key name;
  leaf name {
    type string;
  }
  uses interface-grp;
}
```
Since the interfaces of a LNE can reside on different physical devices, potentially, the same interface on the different devices can be assigned to the same LNE.

```plaintext
config {
  ietf-sys:system : {
    hostname: "device1"
  },
  ietf-if:interfaces : {
    interface : [
      name: "eth-0/0/0",
      ietf-lne:bind-lne-name: "Router1"
    ]
  }
}

config {
  ietf-sys:system : {
    hostname: "device2"
  },
  ietf-if:interfaces : {
    interface : [
      name: "eth-0/0/0",
      ietf-lne:bind-lne-name: "Router1"
    ]
  }
}
```
The interface name on an LNE has the device-name and the interface-name on that device together

- Pros: Having the device name in the LNE interface name makes it easy for users to know exactly which interface they are managing.
- Cons: From YANG point-of-view, requires a new construct to be able to take multiple "leaf-ref" to make up the new value.

```
logical-network-elements {
    logical-network-element {
        name : "Router1",
        ietf-if:interfaces : {
            interface : [ 
            { 
                name : "device1:eth-0/0/0",
                description : "Connection to Server 1"
            },
            { 
                name : "device2:eth-0/0/0",
                description : "Connection to Backbone network"
            }
            ]
        }
    }
}
```
One LNE across multiple physical network elements

- Has to consider the capabilities of the different HW when you mix them into a logical network-element.
  - E.g. if a user wants to construct an aggregated-ethernet interface from physical interfaces on different HW, that HW needs to support multi-chassis LAG, and they need to be compatible between each other.
  - When the different HW have different capabilities
    - Should the resulting LNE only use the common features that are present on every one of the component chassis in the LNE or
    - Should the operator be aware of the differences and can make sure of them if the interface resides on the chassis that can do it.
Out-of-band management

• It is easy to identify the management IP address of a physical device (even one with multiple REs but managed as a single device).

• In case of LNE, what is, say, the IP address of the LNE? There needs to be a separate IP address and what about the out-of-band management address?

• This is a problem for multiple LNEs on a single physical device or a LNE spanning multiple physical devices.
Out-of-band management

- The hierarchy is:

  - Public IP Pools (Floating Pool CIDR)
    - nodes (get 1 IP address of the pool)
      - services (map to one container task: 1 internal IP Address, one or many internal ports)
        - ports (The list of ports that are exposed via the public IP, tcp, udp, and the internal port they are mapped to)
Recycled Slides from NETMOD Presentations and Previous IETFs

See:

LxVPN Support

• NI Type
  • For per VRF, PE/core information

• Root Type
  • For VRF/VSI information in the CE/Vxx context

draft-ietf-rtgwg-ni-model-03:
module: ietf-network-instance
  +--rw network-instances
    +--rw network-instance* [name]
      +--rw name string
      +--rw enabled? boolean
      +--rw description? string
      +--rw (ni-type)?
      +--rw (root-type)?
        +=:(vrf-root)
        | +--mp vrf-root?
        +=:(vsi-root)
        | +--mp vsi-root?
        +=:(vv-root)
        +--mp vv-root?
NI Likely Impact on BESS

There are three types of LxVPN information to model:

1. Core/PE + **not** instance specific
   - Goes in augmentation of a module present at top level
     e.g., bgp, interfaces, or even top of network instances

2. Core/PE + **is** associated with a named NI
   - Goes into augmentation of:
     a) ni-types (preferred) or
     b) other module and associated with bind-ni-name (ala interfaces)

3. CE-Context Information, per VRF/VSI
   - Goes in augmentation of a module present under vrf/vsi-root
   - Do any any of these exist?
   - Reminder: Implementations, not models, decide what gets mounted at top level and under each vrf/vsi-root
LNE: Module Example

module: ietf-logical-network-element
  +--rw logical-network-elements
  +--rw logical-network-element* [name]
    +--rw managed?  boolean
    +--rw name  string
    +--mp root
  ...

--ro yanglib:modules-state/  Managed=true
  | ...
  +--ro sys:system/
  | ...
  +--ro sys:system-state/
  | ...
  +--ro rt:routing-state/
  | +--ro router-id? quad
  | +--ro control-plane-protocols
  |  +--ro control-plane-protocol* []
  |  +--ro ospf:ospf/
  |  | +--ro instance* [af]
  |  | ...
  +--rw rt:routing/
  | ...
  +--rw if:interfaces/
  | ...
  +--ro if:interfaces-state/
  ...
Section 2.3: Device View vs Logical Network Element (cont.)

- Each view logical-network-element can have
  *Full* Device View or Logical Network Element *Limited* View

```
+--rw device
  |   +--rw info
  |   +--rw hardware
  |   +--rw interfaces
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |   +--rw qos
  |   +--rw logical-network-elements
  |     +--rw logical-network-element* [network-element-id]
  |       +--rw network-element-id                  uint8
  |       +--rw default-networking-instance-name?   string
  |       +--rw system-management
  |       |   +--rw device-view?             boolean
  |       |   +-- ...                                |
  |   +--rw networking-instances
  |     +--rw networking-instance* [networking-instance-name]
  |       +--rw networking-instance-name      string
  |       +-- ...                                |

device-view=false
```

```
+--rw device
  |   +--rw info
  |   +--rw hardware
  |   +--rw interfaces
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |     |   +--rw interface* [name]
  |     |       +--rw name                       string
  |     |       +--rw bind-network-element-id?   uint8
  |     |       |   ...                                |
  |   +--rw qos
  |   +--rw logical-network-elements
  |     +--rw logical-network-element* [network-element-id]
  |       +--rw network-element-id                  uint8
  |       +--rw default-networking-instance-name?   string
  |       +--rw system-management
  |       |   +--rw device-view?             boolean
  |       |   +-- ...                                |
  |   +--rw networking-instances
  |     +--rw networking-instance* [networking-instance-name]
  |       +--rw networking-instance-name      string
  |       +-- ...                                |

device-view=false
```

Colors not shown for all elements