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/

DT Status

- 3 standards track drafts
  - YANG Logical Network Elements – draft-ietf-rtgwg-lne-model-03
  - YANG Network Instances – draft-ietf-rtgwg-ni-model-03
  - Rtg Area Common YANG Data Types – draft-ietf-rtgwg-routing-types-08

- Other drafts
  - Module Tags – draft-rtgyangdt-netmod-module-tags
  - Device Logical Organization – draft-ietf-rtgwg-device-model

Individual Draft in NetMod

Gated by tags

- NMDA Next Steps

Reminder: NMDA Guidelines

From AD E-Mail

1. All models SHOULD immediately be structured to be NMDA-compatible
   - With no state for config leaves or access to applied/in-use state
2. Models that require immediate support for "in use" and "system created“
   - SHOULD (a) be structured for NMDA and (b) have a companion “-state” non-NMDA module, which may or may not be derived from (a)

- This will be discussed in next presentation

Short Term Impact of NMDA

- Current WG models SHOULD be refactored
  - Modules containing “-state” information should be moved to an appendix
  - E.g., I2RS and TEAS topology models

- Modules that have a “-state” split only need to be updated immediately when they have information in “–state” that is not in “–config” branch
  - This means RFC 8022, YANG Routing Management!
    - Look for “bis” draft in a few weeks, plan is to update current module

Status Summary :draft-ietf-rtgwg-lne-model and draft-ietf-rtgwg-ni-model

- Drafts previously blocked by open issues in Schema Mount
  - Previous blocking issues have been settled
  - Hope to see WG LC soon
- Now using YANG tree representation for Schema Mount
- Both drafts have been updated and are ready for LC

draft-ietf-rtgwg-lne-model-03 Update

- Text aligned with the current (and hopefully final) definition of schema mount
- Cleaned up intro an other editorial issues identified in rtg dir review*
- Cleaned up yang layout per YANG DR review
- Added/filled in missing sections
  - e.g., Terminology and Security Consideration
- Added errors and notifications
- Use new tree representation
- Tried to improve narrative based on comments and questions
- Consolidated and expanded examples in new Appendix B.

LNE: Module Tree

```
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
```

New tree representation

Added to cover cases of asynchronous interface ㄹㅌ NI bind failures

LNE: Module Example

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

Managed=true

Reminder: modules included under root is an implementation time choice

```
+--ro yanglib:modules-state/
|  ...
+--rw 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/
   ...
```

LNE Next Steps

- More feedback
- WG LC?

draft-ietf-rtgwg-ni-model-03 Update

- Text aligned with the current (and hopefully final) definition of schema mount, impacts types & roots
- Resolved open policy question by providing a structure for LxVPN-specific augmentations
- Cleaned up intro an other editorial issues identified in rtg dir review
- Cleaned up yang layout per YANG DR review
- Added/filled in missing sections
  - e.g., Terminology and Security Consideration
- Added errors and notifications
- Use new tree representation
- Tried to improve narrative based on comments and questions
- Consolidated and expanded examples in new Appendix B.

LxVPN Support

Old:

```
module: ietf-network-instance
  +--rw network-instances
     +--rw network-instance* [name]
        +--rw name              string
        +--rw enabled?          boolean
        +--rw description?      string
        +--rw network-instance-policy
        |  ...
        +--mp root
           ...
```

- NI Policy
  - Container for core instance configuration information
  - Place holder, with details pending
- Root
  - Single mount point for use by any NI type

New:

```
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)?
           ...
```

- NI Type
  - For PE/core information
- Root Type
  - For VRF/VSI information in the CE/Vxx context

LxVPN Technology Specific Information

Two type of PE/Core information:

1. Per VRF/VSI instance information
   - May differs based on LxVPN technology
     - L2VPN – VPLS, VxLAN, EVPN, …
     - L3VPN – MPLS, IP tunnels, …
   - Supported via ***ni-types*** choice statement
     - Empty in base model
     - To be augmented with technology specific cases
2. Information shared across NI instances
   - Supported via augmentations to any top top-level module(s)
     - E.g., BGP or even top of NI model

```
module: ietf-network-instance
  +--rw network-instances
     +--rw network-instance* [name]
        +--rw name           string
        +--rw enabled?       boolean
        +--rw description?   string
        +--rw (ni-type)?
Example:|  +--:(l3vpn)  //augmentation
        |     +--rw l3vpn:l3vpn
        |     |  ...    // config data
        |     +--ro l3vpn:l3vpn-state
        |     |  ...    // state data
```

Per VRF/VSI (CE Context) Information

- Supported via standard top level modules under a per-instance root mount point
  - Specific modules included under a mount point is an *implementation* choice
  - Modules are typically based on L2 or L3 type and not (PE) VPN technology
- Three types of Nis have been identified
  1. VRFs for L3VPNs
  2. VSIs for L2VPNs
  3. VSI+VRF for L2+L3VPNs (bridge/routers)
- Schema mount defines the schema (i.e., module list) on a per mount point *name* basis
  - So need named mount point per type

```
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?
             //one root required per NI
```

NI: Module Example

Reminder: modules included under root is an implementation time choice

```
module: ietf-network-instance
  +--rw network-instances
     +--rw network-instance* [name]
        +--rw name            string
        +--rw enabled?        boolean
        +--rw description?    string
        +--rw (ni-type)?
        |  +--:(l3vpn)
        |     +--rw l3vpn:l3vpn
        |     |  ... // config data
        |     +--ro l3vpn:l3vpn-state
        |     |  ... // state data
        +--rw (root-type)?
           +--:(vrf-root)
              +--mp vrf-root
                 ...
```

```
+--ro rt:routing-state/
|  +--ro router-id?                 yang:dotted-
quad
|  +--ro control-plane-protocols
|     +--ro control-plane-protocol* [type name]
|        +--ro ospf:ospf/
|           …
+--rw rt:routing/
|  +--rw router-id?                 yang:dotted-
quad
|  +--rw control-plane-protocols
|     +--rw control-plane-protocol* [type name]
|     +--rw ospf:ospf/
|        +--rw instance* [af]
|           +--rw areas
|              +--rw area* [area-id]
|                 +--rw interfaces
|                    +--rw interface* [name]
|                       +--rw name if:interface-ref
|                       +--rw cost?   uint16
+--ro if:interfaces@
|  ...
+--ro if:interfaces-state@
|  ...
```

Notifications

- Added to cover cases of asynchronous interface ㄹㅌ NI bind failures
- Interface may be bound multiple ways {base, IPv4 & IPv6}
  - Failure can occur on one or more

```
notifications:
  +---n bind-ni-name-failed
     +--ro name              -> /if:interfaces/interface/name
     +--ro interface
     |  +--ro bind-ni-name?   -> /if:interfaces/interface/ni:bind-ni-name
     +--ro ipv4
     |  +--ro bind-ni-name?   -> /if:interfaces/interface/ip:ipv4/ni:bind-ni-name
     +--ro ipv6
     |  +--ro bind-ni-name?   -> /if:interfaces/interface/ip:ipv6/ni:bind-ni-name
     +--ro error-info?   string
```

Open Issues

- Schema mount currently doesn't allow parent-reference filtering on the instance of the mount point, but rather just the schema.
- This means it is not possible to filter based on actual data, e.g., bind-network-instance-name="green".

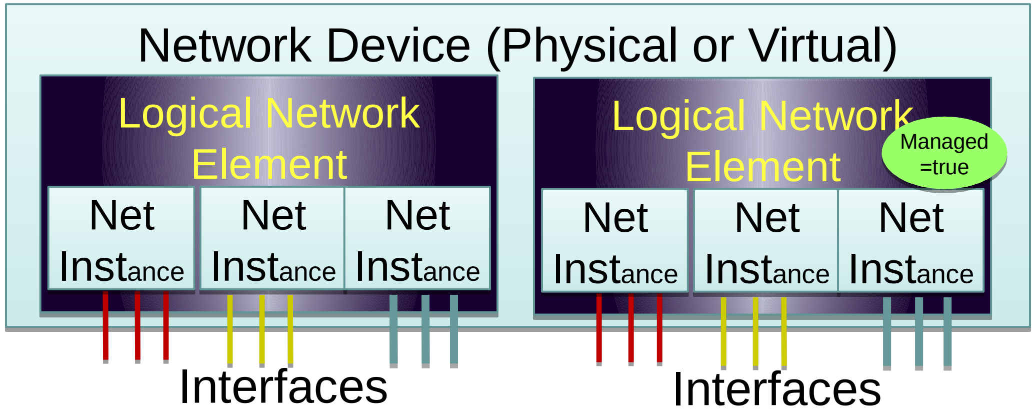
- Recommended resolution:
  - Accept limitation
  - Implementations may choose to impose a limitation on parent references
    - But not required

NI Next Steps

- More feedback
- WG LC?

Some additional details

LNEs and NIs: Reminder



Logical Network Element

- Separate management sub-domains
  - Sub-domains can be managed independently and by a top level manager (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

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.

Schema Mount: Resolved Issues

- Previously gating issues covered identified in draft
  - https://tools.ietf.org/html/draft-ietf-netmod-schema-mount-04#appendix-B
- Interim meeting held, issues largely resolved
  - https://tools.ietf.org/html/draft-ietf-netmod-schema-mount-05

1. Referencing Mount Points Using Schema Node Identifiers
   - Mount point identified by path vs node name – Conclusion: node names
2. Defining the "mount-point" Extension in a Separate Module
   - Rejected, schema mount module always required, even for in-line case
3. Parent References
   - Uses XPath syntax, no filtering based on data, e.g., NI/VRF Name
4. RPC Operations and Notifications in Mounted Modules
   - Limitations/restrictions documented in -05
5. Tree Representation
   - Covered in new NetMod WG draft, see next slide
6. Design-Time Mounts
   - Out of scope / left for future

Only substantive issue,
Can live with limitation

YANG Tree Representation

- Previously
  - All documents copy&paste(&change) the same basic text
  - Readers had to carefully read this text to find the differences between documents
  - No schema mount representation
- Now
  - **draft-ietf-netmod-yang-tree-diagrams-01**
  - Single document defining YANG Tree Representation
  - Includes Schema Mount tree definitions
    - Used in LNE and NI drafts

YANG Tree Representation

- Schema Mount Additions
  - **mp** for schema mount points
  - **/** for a mounted module
  - **@** for a node made available via a schema mount parent reference
  - Module (nodes/leaves/etc) marked **ro** when schema mount config leaf = false

Example

```
+--mp vrf-root?
   +--ro rt:routing-state/
   |  ...
   +--ro rt:routing/
   |  ...
   +--ro if:interfaces@
   |  ...
   +--ro if:interfaces-state@
      ...
```

NI: Full Module Tree

```
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?
augment /if:interfaces/if:interface:
  +--rw bind-ni-name?    -> /network-instances/network-instance/name
augment /if:interfaces/if:interface/ip:ipv4:
  +--rw bind-ni-name?    -> /network-instances/network-instance/name
augment /if:interfaces/if:interface/ip:ipv6:
  +--rw bind-ni-name?    -> /network-instances/network-instance/name
notifications:
  +---n bind-ni-name-failed
     +--ro name          -> /if:interfaces/interface/name
     +--ro interface
     |  +--ro bind-ni-name?  -> /if:interfaces/interface/ni:bind-ni-name
     +--ro ipv4
     |  +--ro bind-ni-name?  -> /if:interfaces/interface/ip:ipv4/ni:bind-ni-name
     +--ro ipv6
     |  +--ro bind-ni-name?  -> /if:interfaces/interface/ip:ipv6/ni:bind-ni-name
     +--ro error-info?   string
```