Routing Area Yang Architecture Design Team Update

Members: Acee Lindem, Anees Shaikh, Christian Hopps, Dean Bogdanovic, Lou Berger, Qin Wu, Rob Shakir, Stephane Litkowski, Yan Gang

Repo: [https://github.com/ietf-rtg-area-yang-arch-dt/](https://github.com/ietf-rtg-area-yang-arch-dt/)
DT current “work” topics

1. Meta-Model:
   YANG Device Model Structure

2. OpState:
   YANG Relationship of Config and Operational State (and intended)

3. Conventions
Status: Meta-Model

- Significant progress from last meeting!
- Identified need for “schema mount”
  - To simplify organization (more on this later)
    - And remove instance/LNE from all models!
  - NETMOD interim held, NETMOD took action to provide solution
- Published draft-rtgyangdt-rtgwg-device-model-03
  - Assuming schema mount
- Next steps
  - Track schema mount solution development
  - Socialize solution
    - Mostly sync’ed with draft-ietf-netmod-routing-cfg
Status: OpState

- Requirements documented and generally understood
  - Telemetry/streaming use case not called out in netmod req document, but understood by those working solution
- NETMOD solutions authors working on consolidated / unified approach
- Lack of standard OpState solution is blocking other work
  - Options available today:
    1. Ignore OpState – assume solution won’t require model changes
    2. Manually add to every model – current OpenConfig approach
- In holding pattern – next DT steps:
  - Track solution discussion in netmod
  - Once there is a solution, sanity check, update drafts as needed
Status: Conventions

Objectives (from AD):

- Provide YANG structure conventions for area
  - E.g., containers within groupings, lists within containers, etc.

- Provide guidance to routing area protocol WG's on:
  - Process for modifying existing models
  - What to do, i.e., not forget about, WRT YANG when defining new protocol extensions
    - TBD, e.g., a new “YANG considerations section”…
Summary

• Meta-Model looks to be in good shape
  • Presuming schema mount solution will be delivered by NETMOD
  • Time for RTG WGs to review & comment
• OpState solution remains a critical impediment
  • Will track/support NETMOD’s work in this area
• Conventions
  • To be looked at post BA
Topics

• Brief Review of Models, LNEs, and NIs
• Challenges
• Use of Schema Mount
• Draft Changes since 01
• Model Disposition
• Open issues
• Next steps
Defined Models

1. module: network-device
   • Overall structure for any network device type
     • From small router to Carrier Class
     • Covers relations amongst models – **Not to be implemented directly**

2. module: logical-network-element
   • Separates management/resource domains
     • Commonly called logical system or router, and virtual switch, chassis, or fabric, virtual device contexts, contexts

3. module: network-instance
   • Separates routing or switching domain
     • e.g., VRF or VSI
   • Will eventually be broken into three documents
Logical Network Elements

- Separate management sub-domains
  - Sub-domains can be managed independently and by a top level manager \((\text{managed}=\text{true})\)

- Differs from multiple logical devices and VMs
  - Where top level management of subdomains not supported
Network Instances

- 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.
Challenge Context

- There are many “top-level” modules out there
  - Some RFCs
  - Many drafts
  - Many private/proprietary/consortia
  - Some from other SDOs (e.g., from IEEE)
- None are LNE aware
- draft-ietf-netmod-routing-cfg changed to remove routing instances
- One example: RFC7223 – A “top-level” module

```
Namespace "urn:ietf:params:xml:ns:yang:ietf-interfaces";
+-rw interfaces
 |  +-rw interface* [name]
 |     +-rw name            string
 |     +-rw description?    string
 |     +-rw type            identityref
 |     +-rw enabled?        boolean
 |     +-rw link-up-down-trap-enable? enumeration
```

Top-level is sometimes referred to as root-level
Original (draft -01) Approach

• An explicit structure with LNEs and NIs

```
  ├── rw device       (Real or virtual)
  │   └── rw info
  │       ├── rw hardware
  │       └── rw interfaces (RFC7223, RFC7277, drafts)
  │           └── rw qos
  │               └── rw logical-network-elements (logical partition)
  │                   └── rw network-instances (rtg-cfg draft, e.g., VRF/VSI)
  └── rw qos

• Pro:
  • Can support any type of device
  • No YANG modification required

• Cons:
  • Every model and device would see at least 1 LNE and NI
  • Would impact every module
    • Each module would need to pick path based on model type
      ▪ Physical at the top
      ▪ Per management domain, under LNE
      ▪ Per VRF/VSI, under NI
Current (draft -03) Approach

- Rely on “schema” mount
  - Works for any module – \textit{without modification}
- Adds two tables
  - LNE: logical-network-inventory
  - NI: network-instance
- Each table defines a per \{LNE, NI\} instance root
  - Under which any top-level model may be \textit{instantiated}
    - Note this is defined in the schema
  - Choice of available model is up to the implementation
    - Some type of device profile definition is expected
  - ietf-yang-library is used to enumerate available models

The term \textit{schema mount} is used to be solution neutral.
Example: A Top-Level Device

Namespace "urn:ietf:params:xml:ns:yang:...";
   +-rw ietf-yang-library
   |                  
   +-rw interfaces
   +-rw hardware
   +-rw qos
   |
   +-rw system-management
   +-rw network-services
   +-rw oam-protocols
   |
   +-rw routing
   +-rw mpls
   +-rw ieee-dot1Q
   |
   +-rw ietf-acl
   +-rw ietf-key-chain
   |
   +-rw logical-network-element
   +-rw network-instance

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

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

module: network-device
   +-rw routing
      +-rw control-plane-protocols
         | +-rw control-plane-protocol* [type]
         | ... 
         +-rw policy 
         +-rw ribs
            +-rw rib* [name]
               | +-rw name string
               | +-rw type=static
               | +-rw type=constrained-paths
               | +-rw type=igp-congruent

module: network-device
   +-rw mpls
      +-rw global
      +-rw mpls* [type]
         | +-rw type=static
         | ...
Example: LNE Model

```
//network-device state
module: logical-network-element
  +--rw logical-network-inventory
    +--rw logical-network-element* [name]
      +--rw name="one" string
      +--rw manged=true boolean
      +--rw root
       schema-mount
    //Example LNE state when exposed to network-device
    +--rw ietf-yang-library
    +--rw interfaces
    +--rw hardware
    +--rw qos
    +--rw system-management
    +--rw network-services
    +--rw oam-protocols
    +--rw routing
    +--rw mpls
    +--rw ieee-dot1Q
    +--rw network-instance
```
Key Requirements of This Use Case

1. That any data model can be instantiated within another module
   • Instantiated means that information is maintained only within the ‘mounted’ context
   • This use case only requires mounting of top-level models

2. That no additional model is needed to support 1
   • The schema defines what other modules can be mounted

3. That a server can control which models are mounted

4. That all capabilities that exist with the mounted module are available e.g. RPC operations, notifications, and augmentations
Changes: Schema Mount

- Allows device hierarchy to vary for different classes of devices.
  - All modules present in the top level may also be mounted within an LNE.
  - Modules supported within an LNE is implementation dependent.
  - Network Instances can be mounted at top or within LNE.
  - All modules can also be mounted within LNE though for many it doesn’t make sense.
  - Modules supported by a device learned through ietf-yang-library.
Changes: LNE Model

- Logical Network Element is now in a separate model
- Unlike meta model, it is to be directly implemented by a device

module: logical-network-element
  +-rw logical-network-inventory
    +-rw logical-network-element* [name]
      +-rw name?    string
      +-rw description? string
      +-rw managed?  boolean
      +-rw root?    schema-mount
    augment /if:interfaces/if:interface:
      +-rw bind-lne-name? string
LNE Management

- Different devices will support different management models
- The “managed” leaf indicates whether or not the LNE can be managed at the device level.
- LNE management at the level are provided using the conventional facilities (e.g., NETCONF/RESTCONF, SNMP)
  - Only LNE associated resources (e.g., interfaces will be accessible at LNE level)
Changes: Network Instance Model - Separate Model

module: 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
Changes: ietf-routing

Relationship

- ietf-routing no longer includes routing-instance list
- ietf-routing is now a module that would be mounted at the top, LNE, or NI level.
- ietf-routing includes its own list of routing protocols since this is needed for static routing definition.
  - Should this list be elsewhere?
- ietf-routing includes a list of interface – this would not be needed with LNE and NI bindings.
Open Issues/Plans

- Relying on Standardized Schema Mount Solution from NETMOD
  - Instantiation of LNEs and NIs triggered simply by list addition?
- Alignment with OpsState Requirements
- Clarification of relationship with different policy containers
- Hardware/QoS structuring
- System management, network services, and OAM protocol base models
Model Disposition Issues

- Plan to move LNE model and NI model to separate standards track RTGWG drafts
- Question is what to do with the device model?
  - Keep it informational and it will not necessary dictate model hierarchy or inter-module relationships?
    - Risk is that the work will not have impact
  - Make it standards track and move to NETMOD WG?
    - Would dictate where other models fit in the hierarchy
    - Hard to get consensus on overall device layout – “Haters gonna hate!”