ConfD NMDA Support

Greg Olin
ConfD Solution Architect
2019-10-22


NMDA IS COMING
imgflip.com

What is NMDA?

- Network Management Datastore Architecture (henceforth NMDA)
- Introduces two new datastores - <intended> and <operational>
  - Also defined to be extensible
- Defined in RFC 8342
- Published in March 2018
  - One of the quickest RFCs I've seen produced
- Extensions to NETCONF and RESTCONF

What is the motivation for NMDA?

- Originally NETCONF/YANG defined 3 datastores
  - Startup
  - Running
  - Candidate
- NETCONF/YANG also had a clean separation between configuration and operational data.
- However devices may have two different groups of values, one that is the one configured and the one actually operational

What is the motivation for NMDA? (cont'd)

- RFC 6244 – An Architecture for Network Management using NETCONF and YANG
- Recognition in section 4.3 "Data distinctions" (taken from 4.3.2)
  - Configuration data is the set of writable data that is required to transform a system from its initial default state into its current state
  - Operational state data is a set of data that has been obtained by the system at runtime and influences the system's behavior similar to configuration data. In contrast to configuration data, operational state is transient and modified by interactions with internal components or other systems via specialized protocols.
  - Statistical data is the set of read-only data created by a system itself. It describes the performance of the system and its components.

What is the motivation for NMDA? (cont'd)

- Section 4.4 - At this time, the only viable solution is to distinctly model the configuration and operational values. The configuration leaf would indicate the desired value, as given by the user, and the operational leaf would indicate the current value, as observed on the device.

What is the motivation for NMDA (cont'd)?

- This caused some YANG models to have multiple parallel lists, one for configuration and one for operational
- You might have a system provided interface in operational list

```
+--rw interfaces
  +--rw interface* [name]
    +--rw name                          string
    +--rw type                          identityref
+--ro interfaces-state
  +--ro interface* [name]
    +--ro name                  string
    +--ro type                  identityref
    +--ro oper-status           enumeration
```

What is the motivation for NMDA (cont'd)?

- This duplication causes redundancies in the YANG model
- Makes it more complicated since not all data might be duplicated
- No real relationship between the parallel lists
  - Can't really enforce a convention to follow
  - Network managers can't depend on the parallelism in all models

What is the motivation for NMDA (cont'd)?

- Some devices allow for pre-configuration or pre-provisioning
  - You might configure an interface on a line card that is not present, but then the interface is available when the line card is hot-inserted.
- So some part of your configuration may be inactive
  - Juniper allows this on their devices
  - ConfD supports the notion of inactive as well using an attribute
- The IETF acquired a lot of feedback from network vendors and network operators
  - In particular gNMI and the OpenConfig community

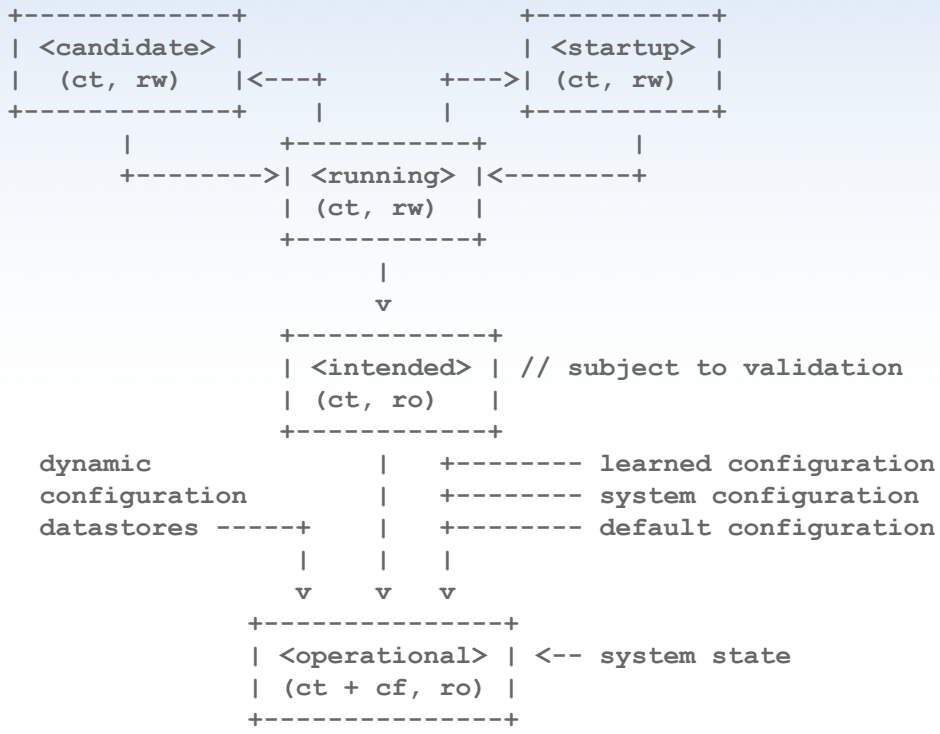
What is the motivation for NMDA (cont'd)?

- In particular, how do you know what is operationally used (applied) and how do you find its related state and statistics

NMDA datastore relationships

- NMDA introduces two new datastores
- One new datastore is <intended>
  - <intended> closing related to <running>
  - Read-only
  - The result once configuration in <running> is expanded, if templates are used, and inactive configuration is removed.
- The other new datastore is <operational>
  - <operational> derives from <intended> plus other sources of configuration.
  - Also read-only
  - State for a configured item can be in the same part of the model but in <operational>

NMDA in beautiful ASCII art



NMDA implications for YANG models

- IETF Interfaces – RFC 8343
- Came out concurrent with NMDA
- Speed with which NMDA came out was to get this new style available for IETF models

```
+--rw interfaces
   +--rw interface* [name]
      +--rw name                          string
      +--rw description?                  string
      +--rw type                          identityref
      +--rw enabled?                      boolean
      +--rw link-up-down-trap-enable?     enumeration {if-mib}?
      +--ro admin-status                  enumeration {if-mib}?
      +--ro oper-status                   enumeration
      +--ro last-change?                  yang:date-and-time
      +--ro if-index                      int32 {if-mib}?
      +--ro phys-address?                 yang:phys-address
```

NETCONF and RESTCONF changes

- NETCONF introduces two new RPCs in RFC 8526
- One new RPC is <get-data>
  - Similar to <get-config> but supports an extensible set of datastores
  - Other new options, such as filter based on origin
- The other new RPC is <edit-data>
  - Only works on writable datastore
    - NMDA is also extensible to allow other datastores
- RESTCONF introduces new datastore resources in RFC 8527
  - {+restconf}/ds/<datastore>
  - New query parameter
  - Requires usage of YANG library

What does this mean for ConfD

- If you use the inactive attribute, ConfD will filter out nodes marked inactive when <get-data> specifies the <intended> datastore
- If you don’t use the inactive attribute, then <get-data> on <running> and <get-data> on <intended> are the same
- ConfD doesn’t have direct support for configuration templates, or services, so the notion of expanding these in <intended> doesn’t apply

What does this mean for ConfD (cont'd)

- For config false nodes in <operational>, your existing data providers will be used
- For config true nodes in <operational>, there are two choices
  - By default, we return config true data from <intended>
    - Correct thing to do in many cases
  - If <operational> data can be different from what is configured, then we have a new annotation on callpoint called `tailf:operational`
  - This means that ConfD will invoke this data provider when these config true nodes are accessed from <operational>
  - In this case, CDB is used for <running> and <intended>, but the data provider is used for <operational>
  - (*Pause for effect* ☺)

What does this mean for ConfD (cont'd)

- You may have models that are purely operational
  - For example IETF Hardware model
- For these, there is new confdc flag, `--datastore operational`
- Model will not be in <running> or <intended>

Thank you for listening

www.tail-f.com