YANG Library

draft-nmdsdt-netconf-rfc7895bis-01

NETCONF WG
IETF 99 (Prague)
Motivation

- Existing RFCs don’t provide ability to express all that is needed...

- RFC 7950 & RFC 8040 say that all NETCONF and RESTCONF servers MUST support RFC 7895 (YANG Library)
  - Regardless if they support NMDA or not.
  - We want to leverage this requirement

- RFC 7895 (YANG Library) says:
  - There is a mandatory to implement 'modules-state' tree that a server uses to advertise all the modules it supports.
  - But this assumes all modules are in all datastores...
  - Which is not that case with NMDA...
    - some modules MAY only appear in <operational> (e.g., ietf-network-topo)
    - some modules MAY only appear in dynamic datastore (e.g., i2rs-ephemeral-rib).
    - some modules MAY only appear in <running>, when a server hasn’t yet coded support for returning the opstate yet.
    - there may be variations in features/deviations between datastores
Summary of Changes from RFC 7895

• Renames document title:
  – OLD: YANG Module Library
  – NEW: YANG Library

• Deprecates the modules-state tree
  – because it assumes all modules are defined in all datastores.

• Adds new top-level "yang-library" container.
  – new top-level container doesn’t break legacy clients
  – decouples the modules a server supports from which datastores they’re supported in
Updates RFC 7950 & RFC 8040

This draft updates RFC 7950 and RFC 8040
  – both in the same way...

Update to RFC 7950
  – Modifies Section 5.6.4 to say that
    /yang-library/modules/module
    is preferred over
    /modules-state/module

Update to RFC 8040
  – Modifies Section 10.1 to say that
    /yang-library/modules/module
    is preferred over
    /modules-state/module
Proposed Tree Diagram

```text
---ro yang-library
  |  ---ro modules
  |  |  ---ro module* [id]
  |  |  |  ---ro id       string
  |  |  |  ---ro name?    yang:yang-identifier
  |  |  |  ---ro revision? union
  |  |  |  ---ro schema?  inet:uri
  |  |  |  ---ro namespace inet:uri
  |  |  |  ---ro feature*   yang:yang-identifier
  |  |  |  |  ---ro deviation* [name revision]
  |  |  |  |  |  ---ro name    yang:yang-identifier
  |  |  |  |  |  ---ro revision union
  |  |  |  |  ---ro conformance-type enumeration
  |  |  |  ---ro submodule* [name revision]
  |  |  |  |  |  ---ro name    yang:yang-identifier
  |  |  |  |  |  ---ro revision union
  |  |  |  |  ---ro schema?  inet:uri
  |  |  ---ro module-sets
  |  |  |  ---ro module-set*
  |  |  |  |  ---ro id?       string
  |  |  |  |  ---ro module*   -> /yang-library/modules/module/id
  |  |  ---ro datastores
  |  |  |  ---ro datastore* [name]
  |  |  |  |  ---ro name     identityref
  |  |  |  |  ---ro properties
  |  |  |  |  |  ---ro property* identityref
  |  |  |  |  ---ro module-set? -> /yang-library/module-sets/module-set/id
  x---ro modules-state
     ++- <deprecated tree not shown>
```
Flexibility Provided

- Servers can state how they support modules per datastore.

- NMDA-implementations can phase-in <operational> support on a module-by-module basis
  - NMDA-compatible servers may not support the <operational> view for a specific module on Day-1
    - Note: config false nodes don’t count.

- NMDA-implementations may only support <operational> view for a specific module:
  - e.g. server only supports providing topology underlays (no overlays)
More Flexibility

• Enables deviations & features per datastore
  – These MAY vary by datastore...

• Of course, all conventional datastores would point to the same ‘module-set’
  – Hence there would no inconsistency between them...
Questions, comments, concerns?