Refined YANG Datastores
draft-wilton-netmod-refined-datastores-01

IETF 96 – Berlin, NETMOD WG
Rob Wilton – Cisco
rwilton@cisco.com
Problem Description

• One of two possible datastore solution drafts
• Primarily aims to solve two problems:
  – Opstate split between intended vs applied configuration
  – Remove the need for a config/state split between “feature” and
    “feature-state” sub-trees
Proposed Solution

1) Formally defines an “Operational State Datastore”, that contains:
   - Applied configuration
   - System controlled configuration
   - All config false nodes (inc ephemeral opstate & statistics)

2) Split the Running Configuration Datastore into:
   - Abstract “Intended Configuration Ds” and “Applied Configuration Ds”

3) Split the abstract Intended Configuration Datastore into:
   - Persistent configuration datastore (recovered after reboot)
   - Ephemeral configuration datastore (for I2RS, lost on reboot)

4) Introduces abstract datastores
Existing NETCONF Datastores
(as per RFC 6241)

Candidate Configuration DS

Startup Configuration DS

Running Configuration DS (r/w, ct)
+ Config false nodes

Writable via <edit-config>
Readable by <get-config>

Content returned in NETCONF <get> request
Define “Operational State Datastore”

- **Candidate Configuration DS**
- **Startup Configuration DS**
- **Running Configuration DS (r/w, ct)**
- **Running Configuration DS (ro copy, ct) + Config false nodes**

- **Operational State Datastore (ro, ct + cf)**

Writable via `<edit-config>`
Readable by `<get-config>`

Content returned in NETCONF `<get>` request
Add System Controlled Configuration

Candidate Configuration DS

Startup Configuration DS

Running Configuration DS (r/w, ct)

Runnable via <edit-config>
Readable by <get-config>

Operational State Datastore (ro, ct + cf)

Content returned in NETCONF <get> request

+ System created config nodes (ct)
+ Config false nodes
Split intended vs applied configuration

Candidate Configuration DS

Startup Configuration DS

Intended Configuration DS (r/w, ct)

Writable via <edit-config>
Readable by <get-config>

Applied Configuration DS (ro, ct)

+ System created config nodes (ct)
+ Config false nodes

Operational State Datastore (ro, ct + cf)

Content returned in NETCONF <get> request
Add support for Ephemeral & Statistics

Candidate Configuration DS

Startup Configuration DS

Writable via <edit-config>
Readable by <get-config>
Read/written by I2RS agent

Ephemeral Configuration DS (r/w, ct)

Includes applied ephemeral config (ct)

Operational State Datastore (ro, ct + cf)

Config false nodes include ephemeral (cf) and statistics

Persistent Configuration DS (r/w, ct)

Applied Configuration abstract DS (ro, ct)
+ System created config nodes (ct)
+ Config false nodes

Content returned in NETCONF <get> request

Intended Configuration abstract Datastore (ro, ct + cf)
Summary

• A “partly backwards compatible” refinement of datastores
• Solves:
  – Opstate split between intended vs applied configuration
  – Removes the need for a config/state split between “feature” and “feature-state” sub-trees
• Also potentially solves:
  – Pre-configuration (present in intended but not applied)
  – System created configuration (e.g. interfaces)
  – Device deviations (when not possible to express in a YANG model)

Thanks for listening. Any questions?