I2RS Protocol Simple Example

Hares work expansion of
Andy Bierman, Kent Watsen work
Previous Definitions

candidate -> running -> startup

config true;

config false; All operational data exists alongside config=true but there is no datastore defined for config=false data nodes
RFC6244

Yang meta

config  Ephemeral config
State data  Ephemeral state
notification  Ephemeral notification
operation  Ephemeral operations

client  server

Intended Config
Ephemeral config

Applied Config
Derived State
Definitions from ietf-netmod-opstate-req

candidate ➔ running ➔ startup

config true; intended config

applied config

config false; Derived state
Ephemeral Additions

candidate -> running -> startup

intended config
Ephemeral Candidate

applied config (normal + ephemeral)

Derived state
Normal + Ephemeral state

config true;

config false;
module thermostat {
    ...
    leaf desired-temp {
        type int32;
        units “degrees Celsius”;
        description “The desired temperature”;
    }
}

// operational state

leaf actual-temp {
    type int32;
    config false;
    units “degrees Celsius”;
    description “The measured temperature”;
}
Thermostate Model

candidate \rightarrow \text{Desired-temp} \rightarrow \text{startup}

\text{Intended Config Desired-temp}

\text{config true;}

\text{config false;}

\text{applied config Desired-temp}

\text{Derived state Actual-temp}
Thermostat + I2RS

candidate → running → startup

Ephemeral Candidate

intended config

Ephemeral Intended

Scheduler I2RS Client

Hold I2RS Client

config true;

applied config (normal + ephemeral)

Derived state Normal + Ephemeral state

config false;
module thermostat {
    ... 
    leaf desired-temp {
        type int32;
        ephemeral true;
        ephemeral-validation full-check;
        units “degrees Celsius”; 
        description “The desired temperature”; 
    }

    Operational State: 
    leaf actual-temp {
        type int32;
        config false;
        units “degrees Celsius”; 
        description “The measured temperature”; 
    }

    Alternatives: 
    No-check
    No-reference
    Full-check
RESTCONF Example

**RESTCONF Running Datastore Edit**

PUT /restconf/data/thermostat:desired-temp

{ "desired-temp": 18 }

**RESTCONF Ephemeral Datastore Edit of config=true**

PUT /restconf/data/thermostat:desired-temp?context=ephemeral

{ "desired-temp": 18 }
NETCONF

<rpc-message-id=101>
   <xmlns=“urn:ietf:params:xml:ns:base:1.0”>

   <edit-config>
      <target>
         <ephemeral>
            True
            <ephemeral-validation>
               full-check
            </ephemeral-validation>
         </ephemeral>
      </target>
   </edit-config>

   <config>
      <top xmlns=“http://exaple.com/schema/1.0/thermostat/config”>
         <desired-temp> 18 </desired-temp>
      </top>
   </config>

</rpc-message-id=101>
Capability Specification for NETCONF/RESTCONF
NETCONF (1)

- **Capability:** ephemeral-datastore

- **Overview:**
  - Not in intended to survive a reboot, and Never locked
  - Normal Case: Priority of Ephemeral Pane higher than configuration Pane. Error if two clients write same variable (priority scopes error)
  - No Rollback on ephemeral
  - Ephemeral under non-ephemeral; No non-ephemeral under Ephemeral
  - NETCONF <hello> - but no non-ephemeral under ephemeral modules, sub-trees, node

- **Dependencies:**
  - Yang: ephemeral flag, ephemeral-validation
  - Yang modules – must support notification of write conflicts (Config/ephemeral and Priority)
NETCONF (2)

- **New operations:**
  - Link-ephemeral `<target-config>`
  - Bulk-write – [Not sure if need or if rpc better approach]

- **Modifications**
  - `<get-config>` `<get>` - target changes
  - `<edit-config>` - `<merge-priority>` `<replace-priority>`
    - `<default-operations>`: `<merge-priority>` or `<replace-priority>`
    - `<error-option>` - “all-or-nothing” == “rollback-on-error”
  - `<unlock>` `<lock>` - not supported
  - `<confirmed commit>` - not supported
  - `<close-session>` `<kill-session>` - target change
  - `<Writeable-running>` and `<candidate>` – support ephemeral (?)
  - Validate – supports ephemeral data store with three key words:
    Syntax, reduced, full-check
RESTCONF (1)

- **Capability**: ephemeral-context

- **Overview**:
  - Same as netconf except RESTCONF Context

- **Dependencies**:
  - Yang: ephemeral flag, ephemeral-validation
  - Yang modules – must support notification of write conflicts (Config/ephemeral and Priority)
  - I2RS Yang modules support: Yang patch and Yang module library
RESTCONF (20)

- **Data resources**
  - `+restconf/data` – ephemeral data tree with edit collision features of timestamp and Entity Tag
  - Assumption: Entity can be split to client-priority

- **Modifications**
  - Options: provide indication of ephemeral state in data modules, sub-modules [ietf-netconf-yang-library]
  - HEAD – returns ephemeral or config context
  - GET - determines if ephemeral or config
  - POST/PUT/PATCH - context=ephemeral:
    - uses ephemeral rules + validity + priority + no config below ephemeral
  - DELETE – ephemeral context
  - Query – Allows to filter by ephemeral
  - Error/Notifications – must interact with pub/sub push [ietf-netconf-yang-push]
  - Log and traceability -