I2RS Requirements

Draft-ietf-i2rs-ephemeral-state-14.txt

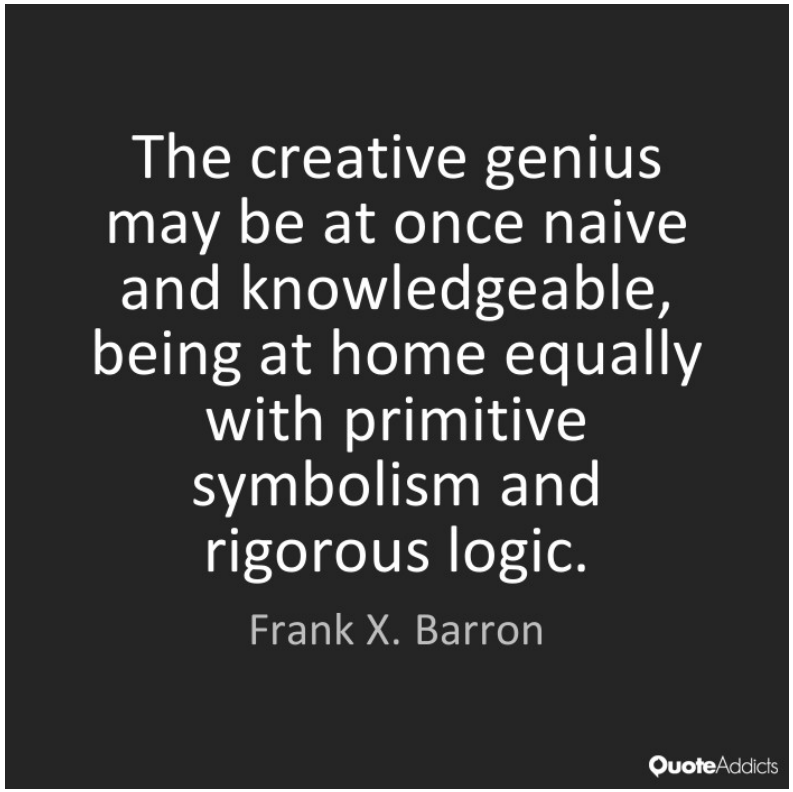
Presenter: Susan Hares

Co-authors: Jeff Haas + Susan Hares

I2RS Ephemeral State Requirements

- Being Naïve
- Data Store + ephemeral
  - My thoughts: draft-hares-i2rs-protcol-strawman-03.txt (see section)
- Walking through Ephemeral State Requirements for Yang (1MT)

Being Naïve about Ephemeral

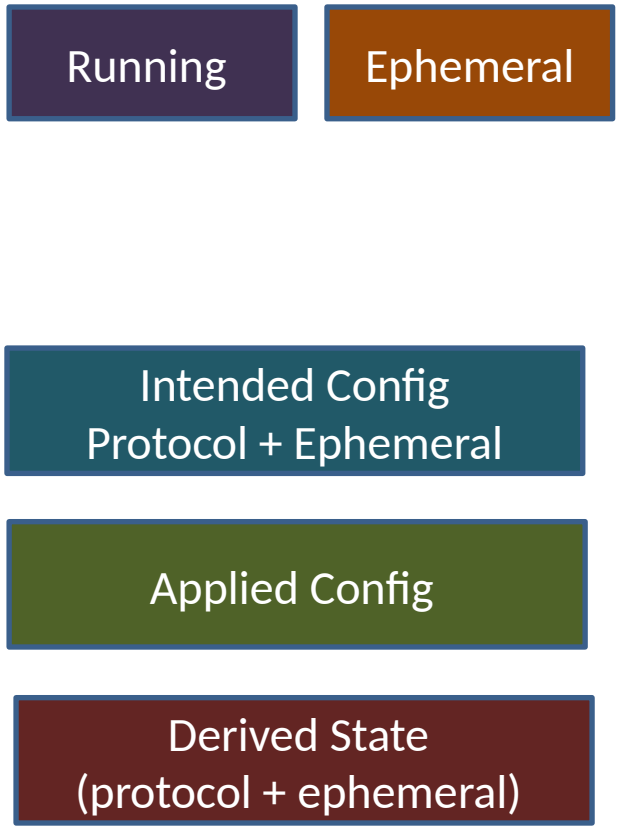


We've talked About Ephemeral For 4 years … so we understand it

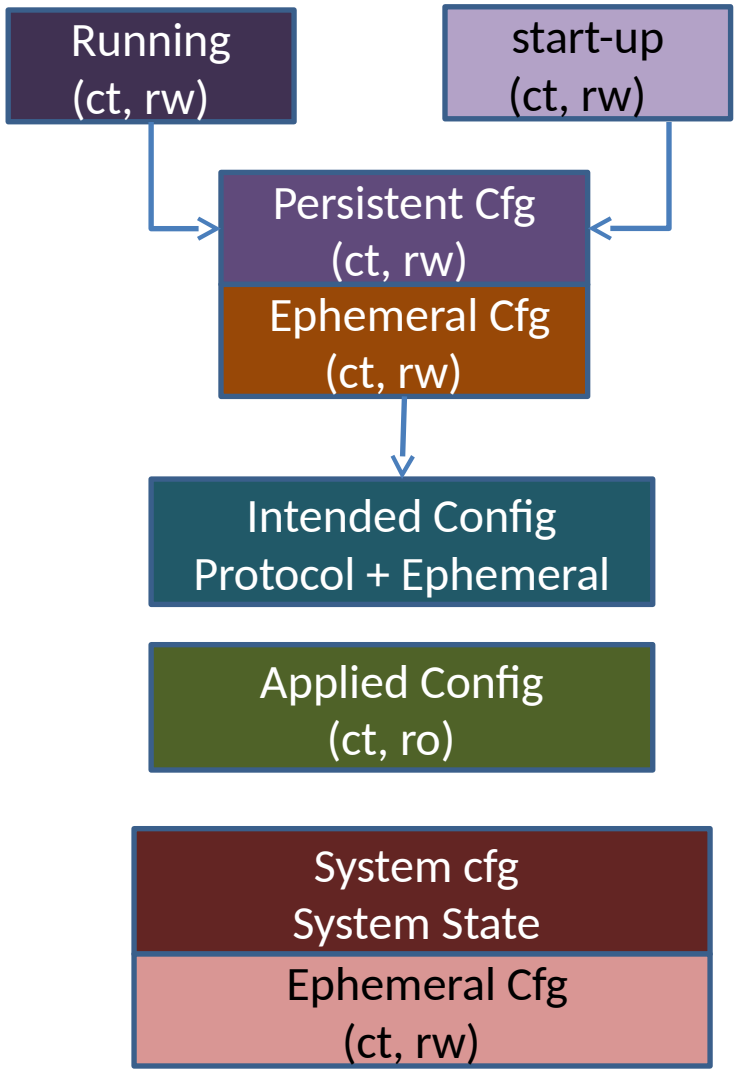
Ephemeral Data Store

- Operational State Models discussed in
  - netmod (Mon 15:40-17:40 +Tues. 16:20-18:20)
  - netconf (Wed. 10:00 – 12:30)
- I2RS protocol-strawman reviews 4 models and implementation issues
  - Draft-hares-i2rs-protocol-strawman

4 ephemeral Models

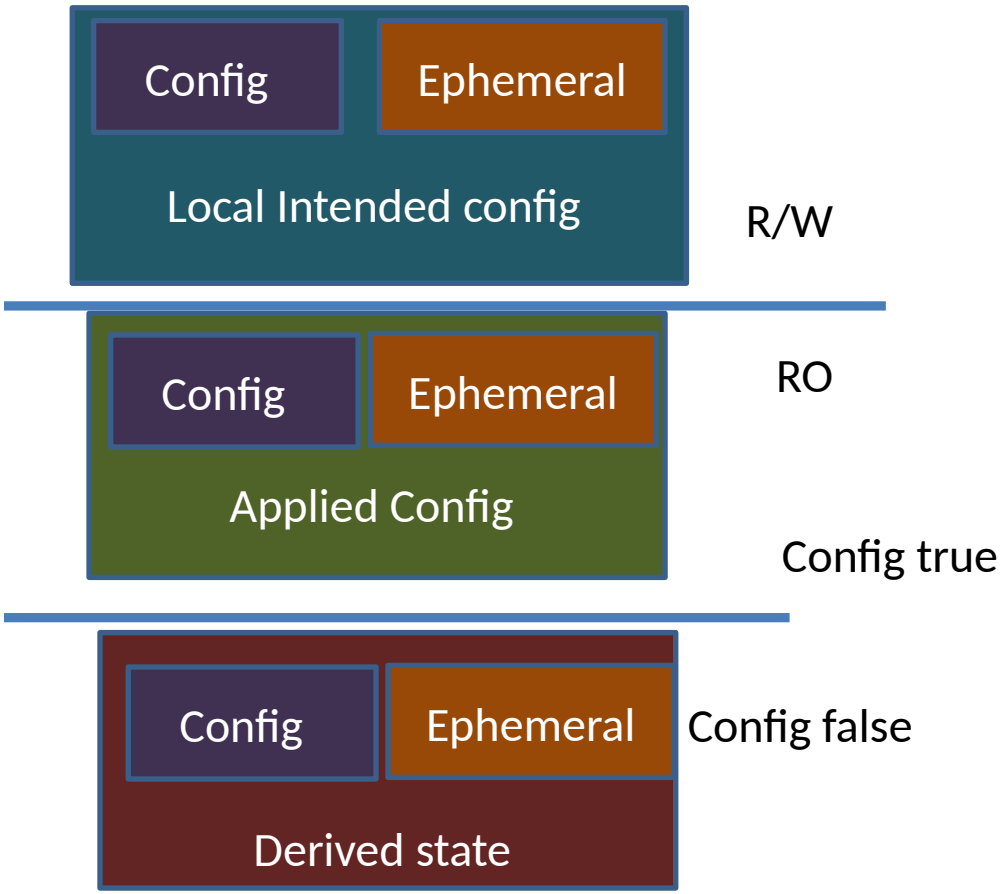


draft-wilton

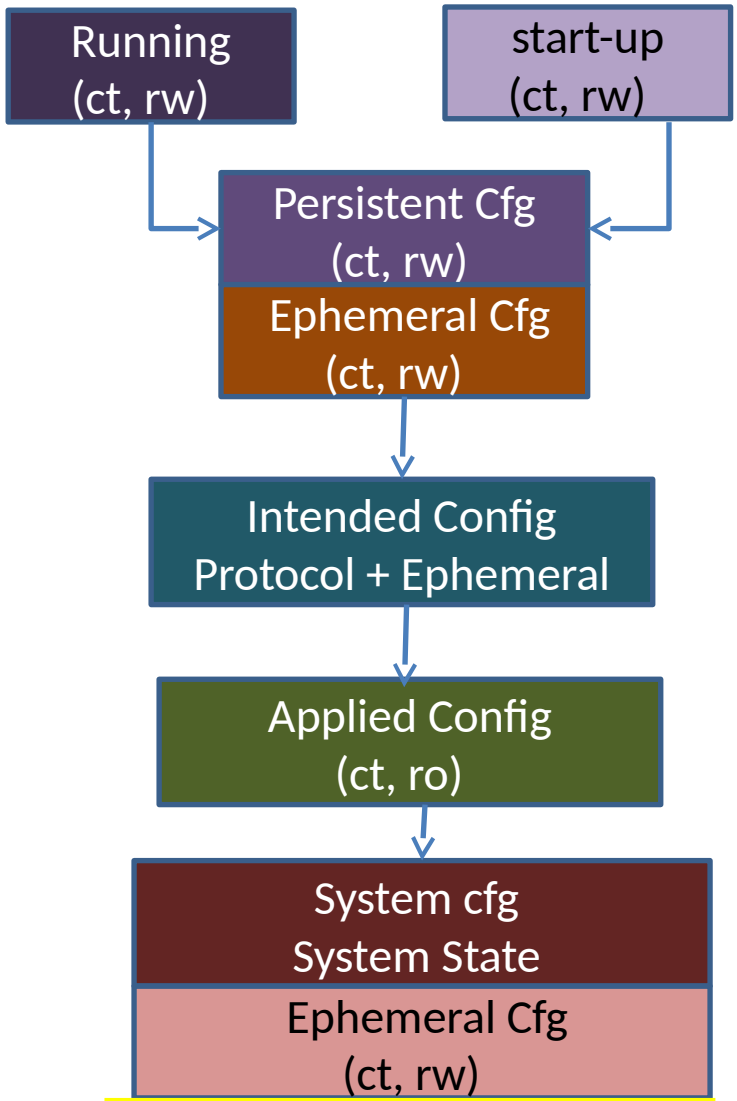


draft-wilton

4 ephemeral Models

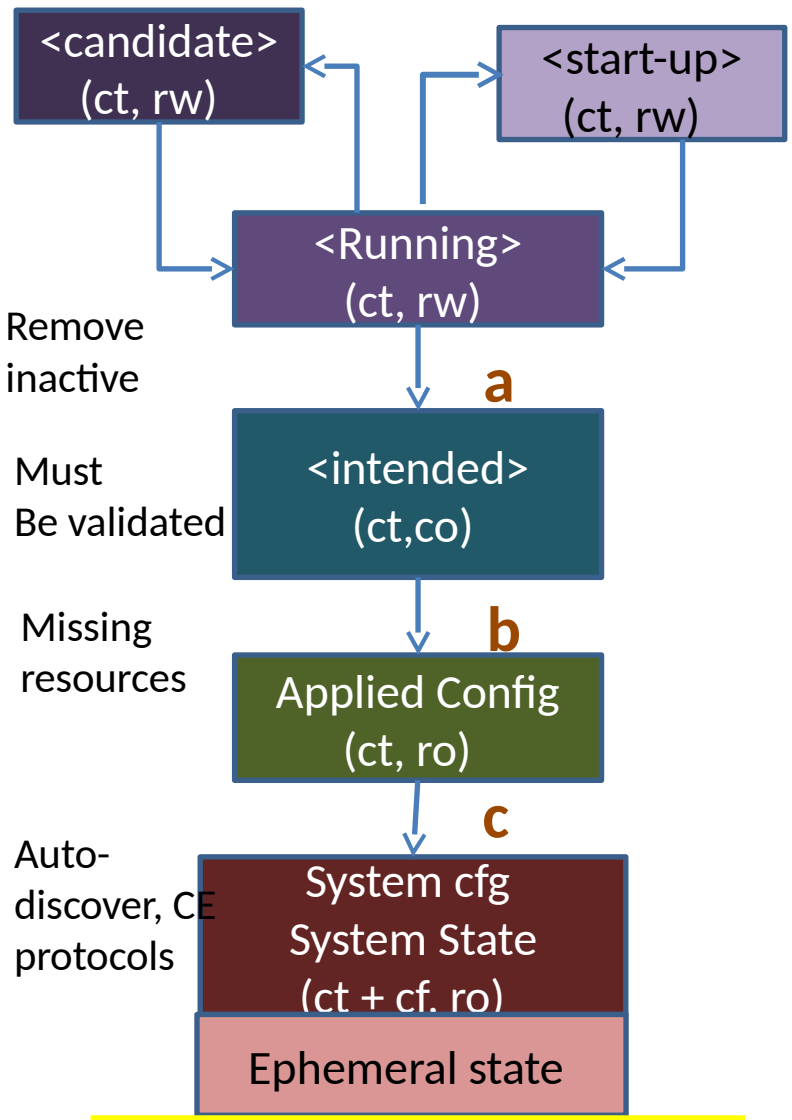


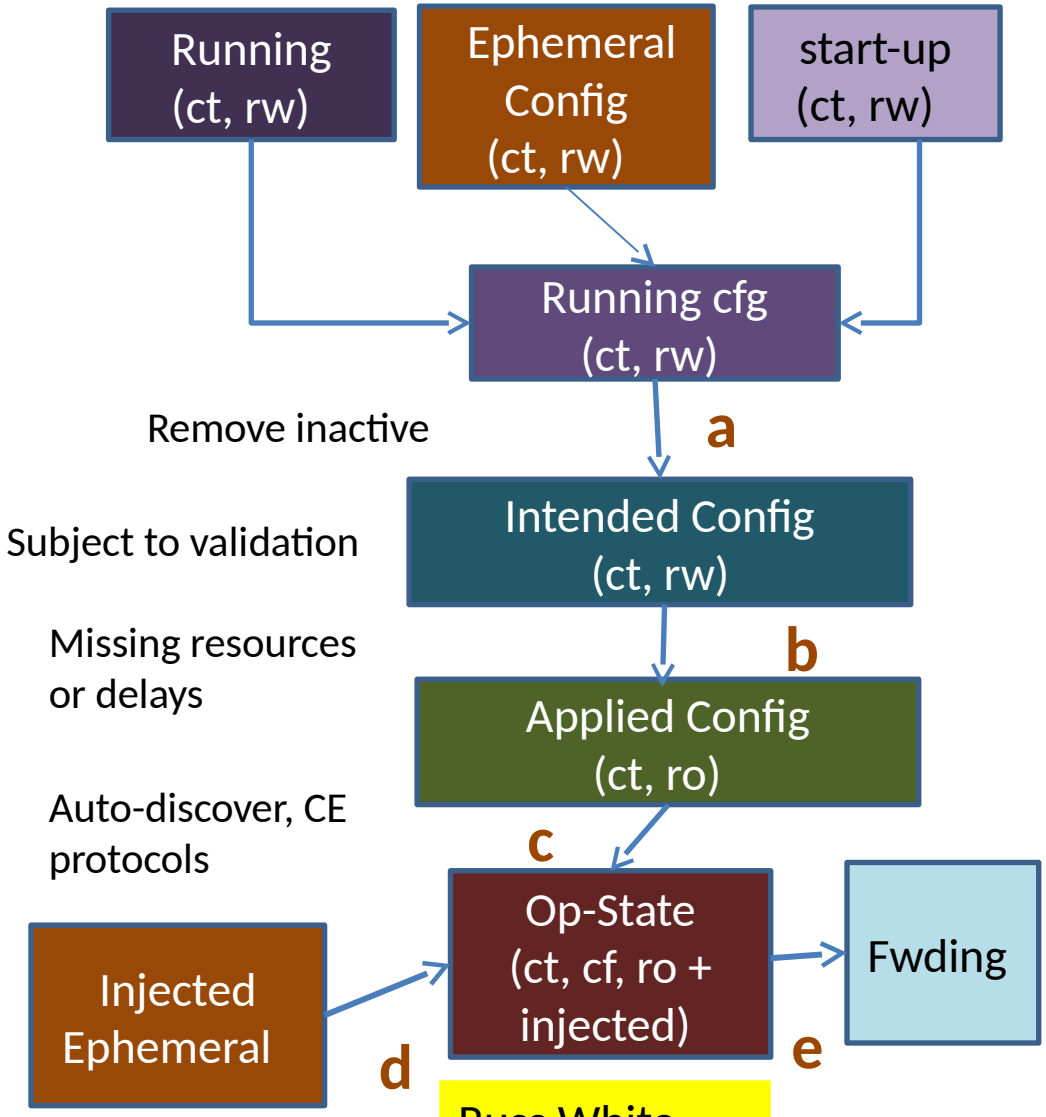
Original discussion (draft-kwatsen-netmod-opstate)



draft-wilton-netmod-refined-datastores

4 ephemeral Models





Draft-schoenw-netmod-revised-datastore

Russ White
I2rs discussion

Pro	Watsen	Wilton	Schoenw	White/Hares
Ephemeral checking message checking (Y 8.3.1)	Y	Y		Y
NETCONF or RESTCONF (Y8.3.2, 8.3.3)	Y	Y		Y
Operation state in ephemeral only models	Y	Y		Y
Ephemeral natural augment	Y	Y		Y
Event /notify	Y	Y		Y
Query Ephemeral + local separate or together	Y	Y		Y
Aligns with Ephemeral Requirements	P	Y		Y
Ephemeral tailors is own validation			Y	P
Ephemeral like other Control plane traffic			Y	Y

Con	Watsen	Wilton	Schoenw	White /Hares
Ephemeral cannot elect to just Yang 1.1 8.3.1 (speed up) – because of data store validity	Y	Y		
Ephemeral must create its own validation checking		Y	Y	
No easy way to see overlay of ephemeral configuration and local configuration			Y	
Not clear how Event/Notify works with Data store				

Feedback needed

I2RS Requirements in WG LC

- 15 Ephemeral State
  - 1 Persistence (REQ-01)
  - 6 Constraints (REQ-02 to REQ-07)
  - 1 Yang feature (REQ-08)
  - 2 Protocol (NETCONF/RESTCONF) (REQ-09/10)
  - 4 Multi-headed control (REQ-11 to REQ-14)
  - 1 Multiple Transactions
- 3 Pub/sub + ephemeral

Ephemeral Requirements

Backup slides only

Ephemeral Persists

- Ephemeral-REQ-01: I2RS requires ephemeral state; i.e. state that does not persist across reboots. If state must be restored, it should be done solely by replay actions from the I2RS client via the I2RS agent.
  - While at first glance this may seem equivalent to the writable-running data store in NETCONF, running-config can be copied to a persistent data store, like startup config. I2RS ephemeral state MUST NOT be persisted.

Ephemeral Constraints

- Ephemeral-REQ-02: Non-ephemeral state MUST NOT refer to ephemeral state for constraint purposes; it SHALL be considered a validation error if it does.

- Ephemeral-REQ-03: Ephemeral state MUST be able to have constraints that refer to operational state, this includes potentially fast changing or short lived operational state nodes, such as MPLS LSP-ID or a BGP IN-RIB. Ephemeral state constraints should be assessed when the ephemeral state is written, and if any of the constraints change to make the constraints invalid after that time the I2RS agent should notify the I2RS Client.

Ephemeral Constraints

- Ephemeral-REQ-04: Ephemeral state MUST be able to refer to non- ephemeral state as a constraint. Non-ephemeral state can be configuration state or operational state.

- Ephemeral-REQ-05: I2RS pub-sub, logging, RPC or other mechanisms may lead to undesirable or unsustainable resource consumption on a system implementing an I2RS Agent. It is RECOMMENDED that mechanisms be made available to permit prioritization of I2RS operations, when appropriate, to permit implementations to shed work load when operating under constrained resources. An example of such a work shedding mechanism is rate-limiting.

Ephemeral Constraints

- Ephemeral-REQ-06: YANG MUST have the ability to:
  - 1. to define a YANG module or submodule schema that only contains data nodes with the property of being ephemeral, and
  - 2. to augment a YANG data model with additional YANG schema nodes that have the property of being ephemeral.

Ephemeral Config Overlap with Local configuration

- Ephemeral-REQ-07: Ephemeral configuration state could override overlapping local configuration state, or vice-versa. Implementations MUST provide a mechanism to choose which takes precedence. This mechanism MUST include local configuration (policy) and MAY be provided via the I2RS protocol mechanisms.

Alternative to Ephemeral-07

- Ephemeral-REQ-07: Ephemeral configuration state MUST be able to set a priority on local configuration and ephemeral state. Based on this priority implementations MUST be able to provide a mechanism to choose which takes precedence. The I2RS Protocol MUST be able to support this mechanisms.

Yang Features

- Ephemeral-REQ-08:In addition to config true/false, there MUST be a way to indicate that YANG schema nodes represent ephemeral state. It is desirable to allow for, and have to way to indicate, config false YANG schema nodes that are writable operational state.

NETCONF/RESTCONF

- Ephemeral-REQ-09/10: The changes to NETCONF /RESTCONF MUST include:
  - 1. Support for communication mechanisms to enable an I2RS client to determine that an I2RS agent supports the mechanisms needed for I2RS operation.
  - 2. The ephemeral state MUST support notification of write conflicts using the priority requirements defined in section 7 below in requirements Ephemeral-REQ-11 through Ephemeral-REQ-14).

Mulit-headed Control

- Ephemeral-REQ-11: The I2RS Protocol (e.g. NETCONF/RESTCONF + yang) MUST be able to support:
  - data nodes that MAY store I2RS client identity and not the effective priority at the time the data node is stored.
  - Per SEC-REQ-07 in section 3.1 of [I-D.ietf-i2rs-protocol-security-requirements], an identifier MUST have just one priority. The I2RS protocol MUST support the ability to have data nodes MAY store I2RS client identity and not the effective priority of the I2RS client at the time the data node is stored.
  - The priority MAY be dynamically changed by AAA, but the exact actions are part of the protocol definition as long as collisions are handled as described in Ephemeral-REQ-12, Ephemeral-REQ-13, and Ephemeral-REQ-14.

Multi-Headed control (2)

- Ephemeral-REQ-12: When a collision occurs as two clients are trying to write the same data node, this collision is considered an error and priorities were created to give a deterministic result.
  - When there is a collision, a notification (which includes indicating data node the collision occurred on) MUST BE sent to the original client to give the original client a chance to deal with the issues surrounding the collision. The original client may need to fix their state.

Note:RESTCONF and NETCONF posts can come in concurrently from alternative sources (see ETag in [I-D.ietf-netconf-restconf] section 3.4.1.2 usage). Therefore the collision detection and comparison of priority needs to occur both for both type of updates (POST or edit-config) at the point of comparison.

Mutli-Headed Control (3)

- Ephemeral-REQ-13: Multi-headed control is required for collisions and the priority resolution of collisions. Multi-headed control is not tied to ephemeral state. I2RS protocol MUST NOT mandate how AAA supports priority. Mechanisms which prevent collisions of two clients trying to modify the same node of data are the focus.

- Ephemeral-REQ-14: A deterministic conflict resolution mechanism MUST be provided to handle the error scenario that two clients, with the same priority, update the same configuration data node. The I2RS architecture gives one way that this could be achieved, by specifying that the first update wins. Other solutions, that prevent oscillation of the config data node, are also acceptable.

Multiple Transactions

- Ephemeral-REQ-15: Section 7.9 of the [I-D.ietf-i2rs-architecture] states the I2RS architecture does not include multi-message atomicity and roll-back mechanisms. The I2RS protocol implementation MUST not require the support of these features. As part of this requirement, the I2SR protocol should support:
  - Multiple operations in one or more messages handling can handle errors within the set of operations in many ways.
  - No multi-message commands SHOULD cause errors to be inserted into the I2RS ephemeral state.

  Note: "SHOULD" is specified due to a lengthy discussion regarding error that should occur.

Pub/Sub

- Pub-Sub-REQ-01: The Subscription Service MUST support subscriptions against ephemeral state in operational data stores, configuration data stores or both.
- Pub-Sub-REQ-02: The Subscription Service MUST support filtering so that subscribed updates under a target node might publish only ephemeral state in operational data or configuration data, or publish both ephemeral and operational data.
- Pub-Sub-REQ-03: The subscription service MUST support subscriptions which are ephemeral. (E.g. An ephemeral data model which has ephemeral subscriptions.)

Feedback needed