Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: November 27, 2015

J. Haas
Juniper Networks

May 26, 2015

I2RS Ephemeral State Requirements
draft-haas-i2rs-ephemeral-state-reqs-00

Abstract

This document covers requests to the netmod and netconf Working Groups for functionality to support the ephemeral state requirements to implement the I2RS architecture.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 27, 2015.

Table of Contents

1. Introduction

The Interface to the Routing System (I2RS) Working Group is chartered with providing architecture and mechanisms to inject into and retrieve information from the routing system. The I2RS Architecture document [I-D.ietf-i2rs-architecture] abstractly documents a number of requirements for implementing the I2RS requirements.

The I2RS Working Group has chosen to use the YANG data modeling language [RFC6020] as the basis to implement its mechanisms.

Additionally, the I2RS Working group has chosen to use the NETCONF [RFC6241] and its similar but lighter-weight relative RESTCONF [I-D.bierman-netconf-restconf] as the protocols for carrying I2RS.

While YANG, NETCONF and RESTCONF are a good starting basis for I2RS, there are some things needed from each of them in order for I2RS to be implemented.

2. Ephemeral State Requirements

2.1. Persistence

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 datastore in NETCONF, running-config can be copied to a persistant data store, like startup config. I2RS ephemeral state MUST NOT be persisted.

2.2. Constraints

Ephemeral state MAY refer to non-ephemeral state for purposes of implementing constraints. The designer of ephemeral state modules are advised that such constraints may impact the speed of processing ephemeral state commits and should avoid them when speed is essential.

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

2.3. Hierarchy

Similar to configuration state (config true, see [RFC6020], section 7.19.1), ephemeral state is not permitted to be configured underneath nodes that are "config false" (state data).

Configuration of ephemeral state underneath "config true" is permitted. This permits augmentation of configuration state with ephemeral nodes.

Configuration of "config true" state underneath ephemeral state MUST NOT be done.

State data, "config false", is permitted underneath ephemeral state. (XXX JMH - should there be a requirement that such state data be part of an ephemeral module and perhaps become similarly inaccessible if the ephemeral module reboots?)

3. Changes to YANG

The YANG "config" keyword ([RFC6020], section 7.19.1) is extended to support the keyword "ephemeral" in addition to "true" and "false". "config ephemeral" declares the nodes underneath to be ephemeral configuration.

4. Changes to NETCONF

A capability is registered declaring that the server supports ephemeral configuration. E.g.:

:ephemeral-config
urn:ietf:params:netconf:capability:ephemeral-config:1.0

<get-config> will normally return "config ephemeral" nodes as it is a form of configuration. It is further extended to add a new parameter, "filter-ephemeral". This parameter accepts the following arguments:

- none (default): No filtering of persistent or ephemeral state is done.
- ephemeral-only: Only nodes representing ephemeral state are returned.
- exclude-ephemeral: Only persistent configuration is returned.

<get> is similarly extended to support "filter-ephemeral".

When a <copy-config> is done, regardless of datastore, nodes that are "config ephemeral" are excluded from the target output.

5. Identity, Secondary-Identity Requirements; Priority Requirements; Implications

5.1. Identity Requirements

I2RS requires clients to have an identity. This identity will be used by the Agent authentication mechanism over the appropriate protocol.

I2RS also permits clients to have a secondary identity which may be used for troubleshooting. This secondary identity is an opaque value. [I-D.ietf-i2rs-traceability] provides an example of how the secondary identity can be used for traceability.

The secondary identity is carried in the configuration operation using a new parameter to <edit-config>. E.g.:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
    <i2rs:irs-secondary-identity>user1</i2rs>
    <target>
        <running/>
    </target>
    <config>
        <top xmlns="http://example.com/schema/1.2/config">
        <interface>
            <name>Ethernet0/0</name>
            <mtu>1500</mtu>
        </interface>
        </top>
    </config>
    </edit-config>
</rpc>
```

"config ephemeral" nodes that are created or altered as part of the config operation will carry the secondary-identity as read-only metadata.

5.2. Priority Requirements

To support Multi-Headed Control, I2RS requires that there be a decidable means of arbitrating the correct state of data when multiple clients attempt to manipulate the same piece of data. This is done via a priority mechanism with the highest priority winning. This priority may vary on a per-node or sub-tree basis based for a given identity.

This further implies that priority is an attribute that is stored in the NETCONF Access Control Model [RFC6536] as part of a rule-list. E.g.:

```
+--rw rule-list [name]
   +--rw name     string
   +--rw group*   union
   +--rw rule [name]
      +--rw name string
      +--rw module-name?  union
      +--rw (rule-type)?
      |  +--:(protocol-operation)
      |  |  +--rw rpc-name?  union
      |  +--:(notification)
      |  |  +--rw notification-name?  union
      |  +--:(data-node)
      |     +--rw path node-instance-identifier
      +--rw access-operations?  union
      +--rw action action-type
      +--rw comment?  string
      +--rw i2rs:i2rs-priority i2rs-priority-type
```

Ephemeral configuration state nodes that are created or altered by users that match a rule carrying i2rs-priority will have those nodes annotated with metadata. Additionally, during commit processing, if nodes are found where i2rs-priority is already present, and the priority is better than the transaction's user's priority for that node, the commit SHALL fail. An appropriate error should be returned to the user stating the nodes where the user had insufficient priority to override the state.

5.3. Representing I2RS Attributes in ephemeral configuration state

I2RS attributes may be modeled as meta-data, [I-D.ietf-netmod-yang-metadata]. This meta-data MUST be read-only; operations attempting to alter it MUST be silently ignored. An I2RS module will be defined to document this meta data. An example of its use:

```
<foo xmlns:i2rs="https://ietf.example.com/i2rs"
     i2rs:i2rs-secondary-identity="user1" i2rs:i2rs-priority="47">
    ...
</foo>
```

6. Subscriptions to Changed State Requirements

I2RS clients require the ability to monitor changes to ephemeral state. While subscriptions are well defined for receiving notifications, the need to create a notification set for all ephemeral configuration state may be overly burdensome to the user.

There is thus a need for a general subscription mechanism that can provide notification of changed state, with sufficient information to permit the client to retrieve the impacted nodes. This should be doable without requiring the notifications to be created as part of every single I2RS module.

7. Previously Considered Ideas

7.1. A Separate Ephemeral Datastore

The primary advantage of a fully separate datastore is that the semantics of its contents are always clearly ephemeral. It also provides strong segregation of I2RS configuration and operational state from the rest of the system within the network element.

The most obvious disadvantage of such a fully separate datastore is that interaction with the network element's operational or configuration state becomes significantly more difficult. As an example, a BGP I2RS use case would be the dynamic instantiation of a BGP peer. While it is readily possible to re-use any defined groupings from an IETF-standardized BGP module in such an I2RS ephemeral datastore's modules, one cannot currently reference state from one datastore to another.

For example, XPath queries are done in the context document of the datastore in question and thus it is impossible for an I2RS model to fulfil a "must" or "when" requirement in the BGP module in the standard data stores. To implement such a mechanism would require appropriate semantics for XPath.

7.2. Panes of Glass/Overlay

I2RS ephemeral configuration state is generally expected to be disjoint from persistent configuration. In some cases, extending persistent configuration with ephemeral attributes is expected to be useful. A case that is considered potentially useful but problematic was explored was the ability to "overlay" persistent configuration with ephemeral configuration.

In this overlay scenario, persistent configuration that was not shadowed by ephemeral configuration could be "read through".

There were two perceived disadvantages to this mechanism:

1. The general complexity with managing the overlay mechanism itself.

2. Consistency issues with validation should the ephemeral state be lost, perhaps on reboot. In such a case, the previously shadowed persistent state may no longer validate.

8. Actions Required to Implement this Draft

- Draft for adding "config ephemeral" to YANG.
- Draft defining NETCONF changes including capability, RPC operation changes and support of secondary identity, RPC changes to support priority.
- I2RS draft to define meta-data for priority and secondary-identity.

9. IANA Considerations

TBD

10. Security Considerations

TBD

11. Acknowledgements

This document is an attempt to distill lengthy conversations on the I2RS mailing list for an architecture that was for a long period of time a moving target. Some individuals in particular warrant specific mention for their extensive help in providing the basis for this document:

- Alia Atlas
- Andy Bierman
- Martin Bjorklund
- Dean Bogdanavich
- Rex Fernando
- Joel Halpern
- Susan Hares
- Thomas Nadeau
- Juergen Schoenwaelder
- Kent Watsen

12. Normative References

[I-D.bierman-netconf-restconf]
Bierman, A., Bjorklund, M., Watsen, K., and R. Fernando, "RESTCONF Protocol", draft-bierman-netconf-restconf-04 (work in progress), February 2014.

[I-D.ietf-i2rs-architecture]
Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-05 (work in progress), July 2014.

[I-D.ietf-i2rs-traceability]
Clarke, J., Salgueiro, G., and C. Pignataro, "Interface to the Routing System (I2RS) Traceability: Framework and Information Model", draft-ietf-i2rs-traceability-02 (work in progress), March 2015.

[I-D.ietf-netmod-yang-metadata]
Lhotka, L., "Defining and Using Metadata with YANG", draft-ietf-netmod-yang-metadata-00 (work in progress), April 2015.

[I-D.rfernando-i2rs-yang-mods]
Fernando, R., pals, p., Madhayyan, M., and A. Clemm, "YANG modifications for I2RS", draft-rfernando-i2rs-yang-mods-00 (work in progress), February 2013.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

Author's Address

Jeffrey Haas
Juniper Networks

Email: jhaas@juniper.net