I2RS working group
Internet-Draft
Intended status: Standards Track
Expires: November 26, 2016

J. Haas
Juniper
S. Hares
Huawei
May 25, 2016

I2RS Ephemeral State Requirements
draft-ietf-i2rs-ephemeral-state-07

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 26, 2016.

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. Section 2 reviews 10 key requirements related to ephemeral state.

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 re-use two existing protocols, NETCONF [RFC6241] and its similar but lighter-weight relative RESTCONF [I-D.ietf-netconf-restconf], as the protocols for carrying I2RS.

What does re-use of a protocol mean? Re-use means that while YANG, NETCONF and RESTCONF are a good starting basis for the I2RS protocol, the creation of the I2RS protocol implementations requires that the I2RS requirements

1. select features from YANG, NETCONF, and RESTCONF per version of the I2RS protocol (See sections 4, 5, and 6)

2. propose additions to YANG, NETCONF, and RESTCONF per version of the I2RS protocol for key functions (ephemeral state, protocol security, publication/subscription service, traceability),

3. suggest protocol strawman as ideas for the NETCONF, RESTCONF, and YANG changes.

The purpose of these requirements and the suggested protocol straw man is to provide a quick turnaround on creating the I2RS protocol.

Support for ephemeral state is I2RS protocol requirement that requires datastore changes (see section 3), Yang additions (see section 4), NETCONF additions (see section 5), and RESTCONF additions (see section 6).

Sections 7-9 provide details that expand upon the changes in sections 3-6 to clarify requirements discussed by the I2RS and NETCONF working groups. Sections 7 provide additional requirements that detail how write-conflicts should be resolved if two I2RS client write the same data. Section 8 provides an additional requirement that details on I2RS support of multiple message transactions. Section 9 highlights two requirements in the I2RS publication/subscription requirements [I-D.ietf-i2rs-pub-sub-requirements] that must be expanded for ephemeral state.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Review of Requirements from I2RS architecture document

The I2RS architecture defines important high-level requirements for the I2RS protocol. The following are ten requirements that [I-D.ietf-i2rs-architecture] contains which provide context for the ephemeral data state requirements given in sections 3-8:

1. The I2RS protocol SHOULD support highly reliable notifications (but not perfectly reliable notifications) from an I2RS agent to an I2RS client.

2. The I2RS protocol SHOULD support a high bandwidth, asynchronous interface, with real-time guarantees on getting data from an I2RS agent by an I2RS client.

3. The I2RS protocol will operate on data models which MAY be protocol independent or protocol dependent.

4. I2RS Agent MUST record the client identity when a node is created or modified. The I2RS Agent SHOULD to be able to read the client identity of a node and use the client identity's associated priority to resolve conflicts. The secondary identity is useful for traceability and may also be recorded.

5. Client identity MUST have only one priority for the client's identifer. A collision on writes is considered an error, but the priority associated with each client identifier is utilized to compare requests from two different clients in order to modify an existing node entry. Only an entry from a client which is higher priority can modify an existing entry (First entry wins). Priority only has meaning at the time of use.

6. The Agent identity and the Client identity SHOULD be passed outside of the I2RS protocol in a authentication and authorization protocol (AAA). Client priority may be passed in the AAA protocol. The values of identities are originally set by operators, and not standardized.

7. An I2RS Client and I2RS Agent MUST mutually authenticate each other based on pre-established authenticated identities.

8. Secondary identity data is read-only meta-data that is recorded by the I2RS agent associated with a data model's node is written, updated or deleted. Just like the primary identity, the secondary identity SHOULD only be recorded when the data node is written or updated or deleted

9. I2RS agent MAY have a lower priority I2RS client attempting to modify a higher priority client's entry in a data model. The filtering out of lower priority clients attempting to write or modify a higher priority client's entry in a data model SHOULD be effectively handled and not put an undue strain on the I2RS agent.

10. The I2RS protocol MUST support the use of a secure transport. However, certain functions such as notifications MAY use a non-secure transport. Each model or service (notification, logging) must define within the model or service the valid uses of a non-secure transport.

3. Ephemeral State Requirements

3.1. Persistence

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.

3.2. 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 utilized temporary operational state (e.g. MPLS LSP-ID or a BGP IN-RIB) as a constraints.

Ephemeral-REQ-04: 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.

3.3. Hierarchy

Ephemeral-REQ-05: The ability to augment an object with appropriate YANG structures that have the property of being ephemeral. An object defined as Yang module, schema tree, a schema node, submodule or components of a submodule (derived types, groupings, data node, RPCs, actions, and notifications".

4. YANG Features for Ephemeral State for I2RS Protocol version 1

Ephemeral-REQ-06: Yang MUST have a way to indicate in a data model that nodes have the following properties: ephemeral, writable/not-writable, status/configuration, and secure/non-secure transport. (If you desire examples, please see [I-D.hares-i2rs-protocol-strawman] for potential yang syntax).

5. NETCONF Features for Ephemeral State for I2RS Protocol version 1

Ephemeral-REQ-07: The conceptual changes to NETCONF

1. protocol version support for I2RS modifications - (e.g. I2RS version 1)

2. support for ephemeral model scope indication - which indicates whether a module is an ephemeral-only module, mixed config module (ephemeral and config), mixed derived state (ephemeral and config).

3. multiple message support - supports the I2RS "all or nothing" concept ([I-D.ietf-i2rs-architecture] section 7.9) which is the same as NETCONF "roll-back-on-error".

4. support for the following transports protocol supported: "TCP", "SSH", "TLS", and non-secure transport (see [I-D.ietf-i2rs-protocol-security-requirements] section 3.2 in requirements SEC-REQ-09 and SEC-REQ-11 for details). NETCONF should be able to expand the number of secure transport protocols supported as I2RS may add additional transport protocols.

5. ability to restrict insecure transport support to specific portions of a data models marked as valid to transfer via insecure protocol.

6. ephemeral state overwriting of configuration state MUST be controlled by the following policy knobs (as defined by [I-D.ietf-i2rs-architecture] section 6.3 and 6.3.1):

   * ephemeral configuration overwrites local configuration (true/false; normal value: true), and

   * Update of local configuration value supercedes and overwrites the ephemeral configuration (true/false; normal value: false).

7. The ephemeral overwriting to local configuration described in (8) above is considered to be the composite of all ephemeral values by all clients. Some may consider this approach as a single pane of glass for ephemeral state.

8. The ephemeral state must support notification of write conflicts using the priority requirements defined in section 3.7 below in requirements Ephemeral-REQ-09 through Ephemeral-REQ-14).

9. Ephemeral data stores SHOULD not require support interactions with writable-running, candidate data store, confirmed commit, and a distinct start-up capability,

This list of requirements require the following the following existing features are supported:

support for the following encodings: XML or JSON.

support for the following transports protocol supported: "TCP", "SSH", "TLS".

all of the following NETCONF protocol [RFC6241] specifications:

* yang pub-sub push [I-D.ietf-netconf-yang-push],

* yang module library [I-D.ietf-netconf-yang-library],

* call-home [I-D.ietf-netconf-call-home], and

* server model [I-D.ietf-netconf-server-model] with the server module must be augmented to support mutual authentication (see [I-D.ietf-i2rs-protocol-security-requirements] section 3.1 in requirements: SEC-REQ-01 to SEC-REQ-08).

6. RESTCONF Features for Ephemeral State for I2RS Protocol version 1

Ephemeral-REQ-08: The conceptual changes to RESTCONF are:

1. protocol version support for I2RS protocol modification (e.g. I2RS-version 1).

2. ephemeral model scope allowed - ephemeral modules, mixed config module (ephemeral and config), mixed derived state (ephemeral and config).

3. support for both of the following transport protocol suites:

   * HTTP over TLS (secure HTTP as defined in RESTCONF [I-D.ietf-netconf-restconf] section 2),

   * HTTP used in a non-secure fashion (See [I-D.ietf-i2rs-protocol-security-requirements], section 3.2, requirements SEC-REQ-09 and SEC-REQ-11 for details), and

   * RESTCONF SHOULD be able to expand the transports supported as as future I2RS protocol versions may support other transports.

4. The ability to restrict insecure transports to specific portions of a data model marked as valid to transfer via an insecure protocol.

5. Support for the development of a RESTCONF based yang pub-sub push based on the requirements in [I-D.ietf-i2rs-pub-sub-requirements] and equivalent to the netconf . [I-D.ietf-netconf-yang-push]

6. ephemeral state overwriting of configuration state MUST be controlled by the following policy knobs (as defined by [I-D.ietf-i2rs-architecture] section 6.3 and 6.3.1)

   * Ephemeral configuration overwrites local configuration (true/false; normal value:true), and

   * Update of local configuration value supercedes and overwrites the ephemeral configuration (true/false; normal value:false).

7. The ephemeral state overwriting a local configuration described above is considered to be the composite of all ephemeral state values by all clients. Some may consider this a single "pane of glass" for the ephemeral values.

8. RESTCONF support ephemeral state MUST support notification of write conflicts using the priority requirements (see section 3.7 below, specifically requirements Ephemeral-REQ-09 through Ephemeral-REQ-14). Expansion of existing "edit-collision" features (timestamp and Entity tag) to include I2RS client-priorities is preferred since I2RS client-Agents exchange MAY wish to use the existing edit-collision features in RESTCONF.

9. Ephemeral data stores SHOULD not require support for interactions with writeable-running, candidate data stores, confirmed commit, and a distinct start-up capability.

This requirement also requires that RESTCONF support all of the following specifications:

1. support for the following encodings: XML or JSON.

2. all of the following curren RESTCONF specifications:

   1. RESTCONF [I-D.ietf-netconf-restconf],

   2. the yang-patch features as specified in [I-D.ietf-netconf-yang-patch],

3. yang module library [I-D.ietf-netconf-yang-library] as defined in RESTCONF [I-D.ietf-netconf-restconf] section 3.3.3),

4. call-home [I-D.ietf-netconf-call-home],

5. zero-touch [I-D.ietf-netconf-zerotouch], and

6. server modules [I-D.ietf-netconf-server-model] (server module must be augmented to support mutual authentication).

7. Requirements regarding Supporting Multi-Head Control via Client Priority

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 is per-client.

Ephemeral-REQ-09: The data nodes 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. Therefore, the 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-10, Ephemeral-REQ-11, and Ephemeral-REQ-12.

Ephemeral-REQ-10: 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 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.

Ephemeral-REQ-11: The requirement to support multi-headed control is required for collisions and the priority resolution of collisions. Multi-headed control is not tied to ephemeral state. I2RS is not mandating how AAA supports priority. Mechanisms which prevent collisions of two clients trying the same node of data are the focus.

Ephemeral-REQ-12: If two clients have the same priority, the architecture says the first one wins. The I2RS protocol has this requirement to prevent was the oscillation between clients. If one

uses the last wins scenario, you may oscillate. That was our opinion, but a design which prevents oscillation is the key point.

8. Multiple Message Transactions

Ephemeral-REQ-13: Section 7.9 of the [I-D.ietf-i2rs-architecture] states the I2RS architecture does not include multi-message atomicity and roll-back mechanisms. I2RS notes 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 data-store.

Explanation:

I2RS suggests the following are some of the potential error handling techniques for multiple message sent to the I2RS client:

1. Perform all or none: All operations succeed or none of them will be applied. This useful when there are mutual dependencies.

2. Perform until error: Operations are applied in order, and when error occurs the processing stops. This is useful when dependencies exist between multiple-message operations, and order is important.

3. Perform all storing errors: Perform all actions storing error indications for errors. This method can be used when there are no dependencies between operations, and the client wants to sort it out.

Is important to reliability of the data store that none of these error handling for multiple operations in one more multiple messages cause errors into be insert the I2RS ephemeral data-store.

Discussion of Current NETCONF/RESTCONF versus

RESTCONF does an atomic action within a http session, and NETCONF has atomic actions within a commit. These features may be used to perform these features.

I2RS processing is dependent on the I2RS model. The I2RS model must consider the dependencies within multiple operations work within a model.

9. Pub/Sub Requirements Expanded for Ephemeral State

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.

The publication/subscription requirements for I2RS are in [I-D.ietf-i2rs-pub-sub-requirements], and the following general requirements SHOULD be understood to be expanded to to include ephemeral state:

- Pub-Sub-REQ-01: The Subscription Service MUST support subscriptions against ephemeral data 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 data in operational data or configuration data, or publish both ephemeral and operational data.

10. IANA Considerations

There are no IANA requirements for this document.

11. Security Considerations

The security requirements for the I2RS protocol are covered in [I-D.ietf-i2rs-protocol-security-requirements] document. The security requirements for the I2RS protocol environment are in [I-D.ietf-i2rs-security-environment-reqs].

12. 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
- Thomas Nadeau
- Juergen Schoenwaelder
- Kent Watsen

13. References

13.1. Normative References:

[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-15 (work in progress), April 2016.

[I-D.ietf-i2rs-protocol-security-requirements]
Hares, S., Migault, D., and J. Halpern, "I2RS Security Related Requirements", draft-ietf-i2rs-protocol-security-requirements-06 (work in progress), May 2016.

[I-D.ietf-i2rs-pub-sub-requirements]
Voit, E., Clemm, A., and A. Prieto, "Requirements for Subscription to YANG Datastores", draft-ietf-i2rs-pub-sub-requirements-09 (work in progress), May 2016.

[I-D.ietf-i2rs-security-environment-reqs]
Migault, D., Halpern, J., and S. Hares, "I2RS Environment Security Requirements", draft-ietf-i2rs-security-environment-reqs-01 (work in progress), April 2016.

[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-11 (work in progress), May 2016.

[I-D.ietf-netconf-call-home]
Watsen, K., "NETCONF Call Home and RESTCONF Call Home", draft-ietf-netconf-call-home-17 (work in progress), December 2015.

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-13 (work in progress), April 2016.

[I-D.ietf-netconf-server-model]
Watsen, K. and J. Schoenwaelder, "NETCONF Server and RESTCONF Server Configuration Models", draft-ietf-netconf-server-model-09 (work in progress), March 2016.

[I-D.ietf-netconf-yang-library]
Bierman, A., Bjorklund, M., and K. Watsen, "YANG Module Library", draft-ietf-netconf-yang-library-06 (work in progress), April 2016.

[I-D.ietf-netconf-yang-patch]
Bierman, A., Bjorklund, M., and K. Watsen, "YANG Patch Media Type", draft-ietf-netconf-yang-patch-08 (work in progress), March 2016.

[I-D.ietf-netconf-yang-push]
Clemm, A., Prieto, A., Voit, E., Tripathy, A., and E. Einar, "Subscribing to YANG datastore push updates", draft-ietf-netconf-yang-push-02 (work in progress), March 2016.

[I-D.ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch-08 (work in progress), April 2016.

[I-D.ietf-netmod-yang-metadata]
Lhotka, L., "Defining and Using Metadata with YANG", draft-ietf-netmod-yang-metadata-07 (work in progress), March 2016.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

13.2. Informative References

[I-D.hares-i2rs-protocol-strawman]
Hares, S., Bierman, A., and a. amit.dass@ericsson.com, "I2RS protocol strawman", draft-hares-i2rs-protocol-strawman-02 (work in progress), May 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

Authors' Addresses

Jeff Haas
Juniper

Email: jhaas@juniper.net

Susan Hares
Huawei
Saline
US

Email: shares@ndzh.com