Internet Engineering Task Force (IETF)
Request for Comments: 8242
Category: Informational
ISSN: 2070-1721

J. Haas
Juniper
S. Hares
Huawei

September 2017

Interface to the Routing System (I2RS) Ephemeral State Requirements

Abstract

"An Architecture for the Interface to the Routing System" (RFC 7921) abstractly describes a number of requirements for ephemeral state (in terms of capabilities and behaviors) that any protocol suite attempting to meet the needs of the Interface to the Routing System (I2RS) protocol has to provide. This document describes, in detail, requirements for ephemeral state for those implementing the I2RS protocol.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8242.

Table of Contents

1. Introduction 3
1.1. Requirements Language 4
2. Architectural Requirements for Ephemeral State 4
3. Ephemeral State Requirements 5
3.1. Persistence 5
3.2. Constraints 6
3.3. Hierarchy 6
3.4. Ephemeral Configuration Overlapping Local Configuration 6
4. YANG Features for Ephemeral State 7
5. NETCONF Features for Ephemeral State 7
6. RESTCONF Features for Ephemeral State 7
7. Requirements regarding Supporting Multi-Head Control via Client 7
8. Multiple Message Transactions 9
9. Pub/Sub Requirements Expanded for Ephemeral State 9
10. IANA Considerations 10
11. Security Considerations 10
12. Normative References 10
Acknowledgements 12
Authors' Addresses 12

1. Introduction

The Interface to the Routing System (I2RS) Working Group (WG) is chartered with providing architecture and mechanisms to inject into and retrieve information from the routing system. The I2RS Architecture document [RFC7921] abstractly documents a number of requirements for implementing the I2RS and defines ephemeral state as "state that does not survive the reboot of a routing device or the reboot of the software handling the I2RS software on a routing device" (see Section 1.1 of [RFC7921]). Section 2 of this document describes the specific requirements that the I2RS WG has identified based on the I2RS architecture's abstract requirements. The Interface to the Routing System (I2RS) Working Group (WG) is chartered with providing architecture and mechanisms to inject into and retrieve information from the routing system. The I2RS Architecture document [RFC7921] abstractly documents a number of requirements for implementing the I2RS and defines ephemeral state as "state that does not survive the reboot of a routing device or the reboot of the software handling the I2RS software on a routing device" (see Section 1.1 of [RFC7921]). Section 2 of this document provides a summary of these abstract requirements, and section 3 recasts these abstract requirements into specific requirements for the Ephemeral state for any IETF network management system.

The I2RS WG has chosen to use the YANG data modeling language [RFC7950] as the basis to implement its mechanisms.

Additionally, the I2RS WG has chosen to reuse two existing protocols, NETCONF [RFC6241] and its similar but lighter-weight relative RESTCONF [RFC8040], as the protocols for carrying I2RS.

What does reuse of a protocol mean? Reuse means that while the combination of the YANG modeling language and the NETCONF and RESTCONF protocols is a good starting basis for the I2RS data modeling language and protocol, the requirements for I2RS protocol implementations should:

1. select features from the YANG modeling language and the NETCONF and RESTCONF protocols per version of the I2RS protocol (see Sections 4, 5, and 6), and

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).

The purpose of these requirements is to ensure clarity during I2RS protocol creation.

Support for ephemeral state is an I2RS protocol requirement that necessitates 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 WGs. Section 7 provides additional requirements that detail how write-conflicts should be resolved if two I2RS client write the same data. Section 8 describes I2RS requirements for support of multiple message transactions. Section 9 highlights two requirements for I2RS publication/subscription [RFC7923] 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", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Architectural Requirements for Ephemeral State

The I2RS architecture [RFC7921] and the I2RS problem statement [RFC7920] define the important high-level requirements for the I2RS protocol in abstract terms. This section distills this high-level abstract guidance into specific requirements for the I2RS protocol. To aid the reader, there are references back to the abstract descriptions in the I2RS architecture document and the I2RS problem statement, but the reader should note the requirements below are not explicitly stated in the I2RS architecture document or in the I2RS problem statement.

Requirements:

1. The I2RS protocol SHOULD support an asynchronous programmatic interface with properties described in Section 5 of [RFC7920] (e.g., high throughput) with support for target information streams, filtered events, and thresholded events (real-time events) sent by an I2RS agent to an I2RS client (from Section 1.1 of [RFC7921]).

2. An I2RS agent MUST record the client identity when a node is created or modified. The I2RS agent SHOULD 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 (from Section 4 of [RFC7921]).

3. An I2RS client identity MUST have only one priority for the client's identifier. 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 that is higher priority can modify an existing entry (first entry wins). Priority only has meaning at the time of use (from Section 7.8 of [RFC7921]).

4. An I2RS client's secondary identity data is read-only metadata that is recorded by the I2RS agent associated with a data model's node when the data node is written. Just like the primary client identity, the secondary identity SHOULD only be recorded when the data node is written (from Sections 7.4 of [RFC7921].)

5. An 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 SHOULD not put an undue strain on the I2RS agent. (See Section 7.8 of [RFC7921] augmented by the resource limitation language in Section 8 [RFC7921].)

3. Ephemeral State Requirements

In requirements Ephemeral-REQ-01 to Ephemeral-REQ-15, Ephemeral state is defined as potentially including in a data model ephemeral configuration and operational state which is flagged as ephemeral.

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.

At first glance, the I2RS ephemeral state may seem equivalent to the writable-running datastore in NETCONF (e.g., running-config), which can be copied to a datastore that persists across a reboot (software or hardware). However, I2RS ephemeral state MUST NOT persist across a reboot (software or hardware).

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 have constraints that refer to operational state, this includes potentially fast-changing or short-lived operational state nodes, such as MPLS LSP-ID (label-switched path ID) or a BGP Adj-RIB-IN (Adjacent RIB Inbound). 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-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 [RFC7923], tracing [RFC7922], 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.

3.3. Hierarchy

Ephemeral-REQ-06: YANG MUST have the ability to do the following:

1. define a YANG module or submodule schema that only contains data nodes with the property of being ephemeral, and

2. augment a YANG module with additional YANG schema nodes that have the property of being ephemeral.

3.4. Ephemeral Configuration Overlapping Local Configuration

Ephemeral-REQ-07: Local configuration MUST have a priority that is comparable with individual I2RS client priorities for making changes. This priority will determine whether local configuration changes or individual ephemeral configuration changes take precedence as described in [RFC7921]. The I2RS protocol MUST support this mechanism.

4. YANG Features for Ephemeral State

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 a way to indicate, config false YANG schema nodes that are writable operational state.

5. NETCONF Features for Ephemeral State

Ephemeral-REQ-09: The changes to NETCONF 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 (see requirements Ephemeral-REQ-11 through Ephemeral-REQ-14).

6. RESTCONF Features for Ephemeral State

Ephemeral-REQ-10: The conceptual changes to RESTCONF are:

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 (see requirements Ephemeral-REQ-11 through Ephemeral-REQ-14).

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-11: The following requirements must be supported by the I2RS protocol in order to support I2RS client identity and priority:

- the data nodes MUST store I2RS client identity and MAY store the effective priority at the time the data node is stored.

- Per SEC-REQ-07 in Section 4.3 of [RFC8241], an I2RS Identifier MUST have just one priority. The I2RS protocol MUST support the ability to have data nodes 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.

Ephemeral-REQ-12: When a collision occurs as two I2RS clients are trying to write the same data node, this collision is considered an error. The I2RS priorities are used to provide a deterministic resolution to the conflict. When there is a collision, and the data node is changed, a notification (which includes indicating the 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.

Explanation: RESTCONF and NETCONF updates can come in concurrently from alternative sources. Therefore, the collision detection and comparison of priority needs to occur for any type of update.

For example, RESTCONF tracks the source of configuration change via the entity-tag (see Section 3.5.2 of [RFC8040]), which the server returns to the client along with the value in GET or HEAD methods. RESTCONF requires that this resource entity-tag be updated whenever a resource or configuration resource within the resource is altered. In the RESTCONF processing, when the resource or a configuration resource within the resource is altered, the processing of the configuration change for two I2RS clients must detect an I2RS collision and resolve the collision using the priority mechanism.

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. The I2RS protocol MUST NOT mandate the internal mechanism for how AAA protocols (e.g., Radius or Diameter) or mechanisms distribute priority per identity except that any AAA protocols MUST operate over a secure transport layer (see Radius [RFC6614] and Diameter [RFC6733]). Mechanisms that 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 in which 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.

8. Multiple Message Transactions

Ephemeral-REQ-15: Section 7.9 of the [RFC7921] 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 I2RS protocol should support:

multiple operations in one message. An error in one operation MUST NOT stop additional operations from being carried out, nor can it cause previous operations to be rolled back.

multiple operations in multiple messages, but multiple message-command error handling MUST NOT insert errors into the I2RS ephemeral state.

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.

Thus, there is 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 [RFC7923], and the following general requirements SHOULD be understood to be expanded to include ephemeral state:

- Pub-Sub-REQ-01: The subscription service MUST support subscriptions against ephemeral state in operational datastores, configuration datastores, or both.

- Pub-Sub-REQ-02: The subscription service MUST support filtering so that subscribed updates under a target node might publish either:

  1. only ephemeral state in operational data or configuration data, or

  2. both ephemeral and operational data.

- Pub-Sub-REQ-03: The subscription service MUST support subscriptions that are ephemeral. (For example, an ephemeral data model that has ephemeral subscriptions.)

10. IANA Considerations

This document does not require any IANA actions.

11. Security Considerations

The security requirements for the I2RS protocol are covered in [RFC8241].

12. Normative References

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

[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, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6614] Winter, S., McCauley, M., Venaas, S., and K. Wierenga, "Transport Layer Security (TLS) Encryption for RADIUS", RFC 6614, DOI 10.17487/RFC6614, May 2012, <https://www.rfc-editor.org/info/rfc6614>.

[RFC6733] Fajardo, V., Ed., Arkko, J., Loughney, J., and G. Zorn, Ed., "Diameter Base Protocol", RFC 6733, DOI 10.17487/RFC6733, October 2012, <https://www.rfc-editor.org/info/rfc6733>.

[RFC7920] Atlas, A., Ed., Nadeau, T., Ed., and D. Ward, "Problem Statement for the Interface to the Routing System", RFC 7920, DOI 10.17487/RFC7920, June 2016, <https://www.rfc-editor.org/info/rfc7920>.

[RFC7921] Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", RFC 7921, DOI 10.17487/RFC7921, June 2016, <https://www.rfc-editor.org/info/rfc7921>.

[RFC7922] Clarke, J., Salgueiro, G., and C. Pignataro, "Interface to the Routing System (I2RS) Traceability: Framework and Information Model", RFC 7922, DOI 10.17487/RFC7922, June 2016, <https://www.rfc-editor.org/info/rfc7922>.

[RFC7923] Voit, E., Clemm, A., and A. Gonzalez Prieto, "Requirements for Subscription to YANG Datastores", RFC 7923, DOI 10.17487/RFC7923, June 2016, <https://www.rfc-editor.org/info/rfc7923>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8241] Hares, S., Migault, D., and J. Halpern, "Interface to the Routing System (I2RS) Security-Related Requirements", RFC 8241, DOI 10.17487/RFC8241, September 2017, <https://www.rfc-editor.org/info/rfc8241>.

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 Bogdanavic,
Rex Fernando,
Joel Halpern,
Thomas Nadeau,
Juergen Schoenwaelder,
Kent Watsen,
Robert Wilton, and
Joe Clarke.

Authors' Addresses

Jeff Haas
Juniper

Email: jhaas@juniper.net

Susan Hares
Huawei
Saline
United States of America

Email: shares@ndzh.com