I2RS working group
Internet-Draft
Intended status: Informational
Expires: May 19, 2017

S. Hares
Huawei
A. Dass
Ericsson
November 15, 2016

RESTCONF Changes to Support I2RS Protocol
draft-hares-netconf-i2rs-restconf-00.txt

Abstract

This document describes a RETCONF additions to support the Interface to Routing system (I2RS) protocol requirements for I2RS protocol version 1. The I2RS protocol is a re-use higher layer protocol which defines extensions to other protocols (NETCONF and RESTCONf) and extensions to the Yang Data Modeling language.

The I2RS protocol supports ephemeral state datastores as control plane datastores. Initial versions of this document contain descriptions of the ephemeral datastore. Future versions may move this description to NETMOD datastore description documents.

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 May 19, 2017.

Table of Contents

1. Introduction

This a proposal for yang additions to support the first version of the I2RS protocol.

The I2RS architecture [RFC7921] defines the I2RS interface "a programmatic interface for state transfer in and out of the Internet routing system". The I2RS protocol is a protocol designed to a higher level protocol comprised of a set of existing protocols which have been extended to work together to support a new interface to the routing system. The I2RS protocol is a "reuse" management protocol which creates new management protocols by reusing existing protocols and extending these protocols for new uses, and has been designed to be implemented in phases [RFC7921].

The first version of the I2RS protocol is comprised of extensions to existing features of NETCONF [RFC6241] and RESTCONF [I-D.ietf-netconf-restconf]. The data modeling language for the I2RS protocol will be Yang [RFC7950] with features and extensions proposed in this draft.

The structure of this document is:

Section 2 provides definitions for terms in this document.

Section 3 summarizes the changes to configuration data store, NETCONF, RESTCONF, and YANG.

Section 4 describes the RESTCONF additions to support the I2RS protocol.

[I-D.ietf-i2rs-ephemeral-state] specifies the I2RS requirements for the ephemeral state. Section 5 discusses how ephemeral state that fulfills these requirements might be implemented in a control plane datastore.

2. Definitions Related to Ephemeral Configuration

This section reviews definitions from I2RS architecture [RFC7921] and NETCONF operational state definitions

[I-D.nmdsdt-netmod-revised-datastores] before using these to construct a definition of the ephemeral data store.

2.1. 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].

2.2. I2RS Definitions

The I2RS architecture [RFC7921] defines the following terms:

ephemeral data: is data which does not persist across a reboot (software or hardware) or a power on/off condition. Ephemeral data can be configured data or data recorded from operations of the router. Ephemeral configuration data also has the property that a system cannot roll back to a previous ephemeral configuration state. (See [RFC7921] for an architectural overview, [I-D.ietf-i2rs-ephemeral-state] for requirements, and [I-D.nmdsdt-netmod-revised-datastores] for discussion of how the ephemeral datastore as a control plane datastore interacts with intended datstore and dynamic configuration protocols to form the applied datastore".

local configuration: is the data on a routing system which does persist across a reboot (software or hardware) and a power on/off condition. Local configuration has the ability to roll back to a pervious configuration state. Local configuration is defined as the intended datastore [I-D.nmdsdt-netmod-revised-datastores] which is modified by dynamic configuration protocols (such as DHCP) and the I2RS ephemeral data store

dynamic configuration protocols datastore are configuration protocols such as DHCP that interact with the intended datastore (which does persist across a reboot (software or hardware) power on/off condition), and the I2RS ephemeral state control plane datastore.

operator-applied policy: is a policy that an operator sets that determines how the ephemeral datastore as a control plane data store interacts with applied datastore (as defined in [I-D.nmdsdt-netmod-revised-datastores]). This operator policy consists of policy knobs that the operator sets to determine how the I2RS agent control plane ephemeral state datastore will interact with the intended configuration datastor and the dynamic configuration protocol datastore. Three policy knobs could be used to implement this policy:

* policy knob 1: I2RS Ephemeral control-plane datastore takes takes precedence over the intended datastore in the routing protocols.

* policy knob 2: Updated intended configuration datastore takes precedence over the I2RS ephemeral control-plane data store in the routing protocols

* policy knob 3: Ephemeral control plane datastore takes precedence over any other dynamic configuration protocols datastore.

An practical example for three states of the operator-applied policy may help the reader understand the concept. Consider the following three desired outcomes with their policy knob states:

Monitoring Features only The policy knob settings are:

Policy knob 1=false,

policy knob 2=true,

Policy knob 3=false,

Action: I2RS protocol software feature is installed, but the operator does not want the I2RS ephemarl datastore to take precedence (that is be used) on any variables in the applied configuration datastore. This policy set might be valid if I2RS is only suppose to monitor data on this node through newly defined parameters.

I2RS Agent Changes win the policy knob settings would be:

Policy knob 1=true,

policy knob 2=false,

Policy knob 3=false,

Action: This is the normal case for the I2RS Agent where the ephemeral control-plane datastore takes precedence over the intended configuration datastore and dynamic configuration datastores. The values from the I2RS ephemearl datastore are used rather than the intended configuration datastore and the dynamic configuration protocol datastore. When the ephemeral data is removed by the I2RS agent, the dyanmic configuration datastore and the intended configuration datastore state is restored, combined and passed to the routing protocols for application.

Just change until next configuration update the policy knob settings would be:

Policy knob 1=true,

policy knob 2=true,

Policy knob 3=false,

Action: This case can occur if the I2RS Client write to the ephemeral control plane data store is only suppose to take precedence until the next configuration cycle from a centralized system. Suppose the local configuration is get by the centralized system at 11:00pm each night. The I2RS Client writes temporary changes to the routing system via the I2RS agent ephemeral write. At 11:00pm, the local configuration update overwrite the ephemeral. The I2RS Agent notifies the I2RS Client which is tracking which of the ephemeral changes are being overwritten.

3. Overview of Changes

This oveview reviews the following:

- What NETCONF [RFC6241] protocol existing features required for I2RS protocol and what extension for these extension features that are needed for the I2RS protocol version 1,

- What RESTCONF [I-D.ietf-netconf-restconf] protocol existing features are required for the I2RS protocol and what extensions are needed for I2RS protocol version 1.

- An overview of the Yang 1.1 data modeling language[RFC7950] features are needed for I2RS protocol version 1.

- An overview of the extensions to Yang 1.1 data modeling language [RFC7950] that are needed for the I2RS protocol version 1.

3.1. I2RS protocol requirements

The requirements for the I2RS protocol are defined in the following documents:

- I2RS Problem Statement [RFC7920],

- I2RS Architecture [RFC7921],

- I2RS Traceability [RFC7922],

- Publication and Subscription [RFC7923],

- I2RS Ephemeral State Requrements, , [I-D.ietf-i2rs-ephemeral-state]

- I2RS Protocol Security Requirements, [I-D.ietf-i2rs-protocol-security-requirements]

The Interface to the routing System (I2RS) creates a new capability for the routing systems, and with greater capaiblities come a greater need for security. The requirements for a secure environment for I2RS is described in [I-D.ietf-i2rs-security-environment-reqs].

3.2. NETCONF Features and Extensions

The features the I2RS protocol requires are:

- NETCONF [RFC6241] with its updates [RFC7803],

- Network Access Control Model [RFC6536] with update (draft-bierman-netconf-rf6536bis)

- Running NETCONF over TLS with mutually X.509 authentication [RFC7589]

- Keystore Model [I-D.ietf-netconf-keystore],

- Subscribing to Yang Datastore updates [I-D.ietf-netconf-yang-push],

- NETCONF support for Event Notifications [I-D.ietf-netconf-netconf-event-notifications],

- Subscribing to NETCONF Events (updated) [I-D.ietf-netconf-rfc5277bis]

- Yang Patch Media type [I-D.ietf-netconf-yang-patch],

- NETCONF/RESTCONF Zero Touch provisioning [I-D.ietf-netconf-zerotouch],

- TLS Client and Server Models [I-D.ietf-netconf-tls-client-server]

- Call Home [I-D.ietf-netconf-call-home],

- Module library [RFC7895],

- NETCONF/RESTCONF Zero Touch provisioning [I-D.ietf-netconf-zerotouch],

3.3. RESTCONF features and Extensions

This protocol strawman utilizes the following existing proposed features for NETCONF and RESTCONF

- RESTCONF [I-D.ietf-netconf-restconf]
- Module library [RFC7895],
- Publication/Subscription via Push [I-D.ietf-netconf-yang-push],
- Patch [I-D.ietf-netconf-yang-patch],
- syslog yang module (both [RFC5424] and [I-D.ietf-netmod-syslog-model]

3.4. Assumptions on Data Store Model Melee

The NETMOD Working Group has been working to create new definitions of datastores based on feedback from operators on desiring a split between operational state and configuration state.

This document takes [I-D.nmdsdt-netmod-revised-datastores] as the current status of the datastore discussion on configuration state, operational state, ephemeral state changes (via I2RS), and routing protocol state. The following things need to be carefully defined in this work:

What is a dynamic configuration protocol (is it I2RS or DHCP)

What is a control-plane datastore - (ephemeral state only or others?)

How to express the policy knobs that provide preference between intended configuration, control plane datstore, and dynamic configuration protocols

How does operational state allow for operational state to be defined by ephemeral-only data models, and mixed (ephemeral + intended configuration)

[I-D.nmdsdt-netmod-revised-datastores] is making good progress, but these additional details need to be tied down.

4. RESTCONF features and Extensions

This protocol strawman utilizes the following existing proposed features for NETCONF and RESTCONF

- RESTCONF [I-D.ietf-netconf-restconf]
- Module library [RFC7895],
- Publication/Subscription via Push [I-D.ietf-netconf-yang-push],
- Patch [I-D.ietf-netconf-yang-patch],
- syslog yang module (both [RFC5424] and [I-D.ietf-netmod-syslog-model]

5. Assumptions on Data Store Model Melee

The NETMOD Working Group has been working to create new definitions of datastores based on feedback from operators on desiring a split between operational state and configuration state.

This document takes [I-D.nmdsdt-netmod-revised-datastores] as the current status of the datastore discussion on configuration state, operational state, ephemeral state changes (via I2RS), and routing protocol state. The following things need to be carefully defined in this work:

What is a dynamic configuration protocol (is it I2RS or DHCP)

What is a control-plane datastore - (ephemeral state only or others?)

How to express the policy knobs that provide preference between intended configuration, control plane datstore, and dynamic configuration protocols

How does operational state allow for operational state to be defined by ephemeral-only data models, and mixed (ephemeral + intended configuration)

[I-D.nmdsdt-netmod-revised-datastores] is making good progress, but these additional details need to be tied down.

6. RESTCONF protocol extensions for the ephemeral datastore

capability-name: ephemeral-datastore

6.1. Overview

This capability defines the RESTCONF protocol extensions for the ephemeral state. The ephemeral state has the features described in the previous section on NETCONF.

6.2. Dependencies

The ephemeral capabilities have the following dependencies:

- The Yang modules must support the ephemeral state keyword.
- The following features must be supported by RESTCONF
  - Module library [RFC7895],
  - Publication/Subscription via Push [I-D.ietf-netconf-yang-push],
  - Patch [I-D.ietf-netconf-yang-patch],
  - syslog yang module (both [RFC5424] and [I-D.ietf-netmod-syslog-model]

6.3. Capability identifier

The ephemeral-datastore capability is identified by the following capability string: (capability uri)

6.4. New Operations

none

6.5. modification to data resources

RESTCONF must be able to support the ephemeral datstore as a context with its rules as part of the "{+restconf}/data" subtree. The "edit collision" features in RESTCONF must be able to provide notification to I2RS read functions or to rpc functions. The "timestamp" with a last modified features must support the traceability function.

The "Entity Tag" could support saving a client-priority tuple as a opaque string, but it is important that that additions be made to restore client-priority so it can be compared with strimgs can be done to determine the comparison of two I2RS client-priorities.

6.6. Modification to existing operations

The current operations in RESTCONF are: OPTIONS, HEAD, GET, POST, PUT, PATCH, and DELETE. This section describes the modification to these exiting operations.

6.6.1. OPTIONS changes

The options methods should be augmented by the [RFC7895] information that will provide an indication of what ephemeral state exists in a data modules, or a data modules sub-modules or nodes.

6.6.2. HEAD changes

The HEAD in retrieving the headers of a resources. It would be useful to changes these headers to indicate the datastore a node or submodule or module is in (ephemeral or normal), and allow filtering on ephemeral nodes or trees, submodules or module.

6.6.3. GET changes

GET must be able to read from the URL and a context ("?context=ephemeral"). Similarly, it is important the Get be able to determine if the context=ephemeral.

6.6.4. POST changes

POST must simply be able to create resources in ephemeral datastores ("context=ephemeral") and invoke operations defined in ephemeral data models.

6.6.5. PUT changes

PUT must be able to reference an ephemeral module, sub-module, and nodes ("?context=ephemeral").

6.6.6. PATCH changes

Plain PATCH must be able to update or create child resources in an ephemeral context ("?context=ephemeral") The PATCH for the ephemeral state must be change to provide a merge or update of the original data only if the client’s using the patch has a higher priority than an existing datastore’s client, or if PATCH requests to create a new node, sub-module or module in the datastore.

6.6.7. DELETE changes

The phrase "?context=ephemeral" following an element will specify the ephemeral data store when deleting an entry.

6.6.8. Query Parameters

The query parameters (content, depth, fields, insert, point, start-time, stop-time, and with-defaults (report-all, trim, explicit, report-all-tagged) must support ephemeral context ("?context=ephemeral") described above.

6.7. Interactions with Notifications

The ephemeral database must support the ability to publish notifications as events and the I2RS clients being able to receiving notifications as Event stream. The event error stream processing should support the publication/subscription mechanisms for ephemeral state defined in [I-D.ietf-netconf-yang-push].

6.8. Interactions with Error Reporting

The ephemeral database must support in RESTCONF must also support passing error information regarding ephemeral data access over to RESTCONF equivalent of the and traceability client.

7. Ephemeral Data (Background)

Note: This section probably goes with the definition of ephemeral state or as its own Draft

This section provides an overview of the ephemeral data store as a control plane datastore and discusses several concepts that implementers need to consider and provide feedback on. The concepts include basic ephemeral datastore concepts, I2RS caching of ephemeral data, issues for massive data flow, error handling (normal and reduced), use of IPFIX or Binary for carrying I2RS ephemeral data, and ephemeral state

This section augments [I-D.nmdsdt-netmod-revised-datastores] to begin to discuss how the ephemeral state comtrol-plane datastore might be implemented. The purpose of this section is to gather implementer wisdom on the ephemeral datastore into one place. This section discusses:

Ephemeral state as a control plane data store

Qualities of ephemeral datastores

Need to support Massive amounts of configuration data,

Two types of Error handling (regular, reduced)

Should we support link to IPFIX in I2RS protocol and ephemeral state?

Binary encoding for RESTCONF/NETCONF

Ephemeral state in DDoS environments.

[I-D.ietf-i2rs-ephemeral-state] describes the requirements for I2RS ephemeral state.

This section augments [I-D.nmdsdt-netmod-revised-datastores] to begin to discuss how the ephemeral state comtrol-plane datastore might be implemented. This initial draft refines the general description so that early I2RS ephemeral state implementations may progress.

7.1. Ephemeral Control Plane Datastore

[I-D.nmdsdt-netmod-revised-datastores] architecture suggests that the applied configuration is the combination of intended datastore, the dynamic configuration protocols, and the control-plane datastores. As described above, there are policy knobs which allow the I2RS Agent to handle deciding what specific configuration variables is installed in protocols (E.g BGP) or protocol independent functions (RIB or Filters). In addition, the control-plane datastore may store the parameters need to provide publication of events, statistics, telementry within the ephemeral control-plane datastore.

The ephemeral data-store may have models which learn operational state and augment it by configuration. For example [I-D.ietf-i2rs-yang-l3-topology] uploads ospf and isis topology information from the routing system and allows configuration of additional links or nodes.

This new architecture is a multiple panes-of-glass model where the decision on what value is chosen is based on policy. The extension of this model is that it is possible for two or more of the control-plane datastores to be ephemeral. If this occurs, then the policy knobs must define the how the 2+ ephemeral datastores interact with each other and the configuration state.

7.2. Qualities of Ephemeral Datastore

Note: The requirements for ephemeral state are in: [I-D.ietf-i2rs-ephemeral-state]).

This section provides a discussion so that implementers writing code for these datastores can discuss what needs to be standardized and what does not need to be standardized.

The ephemeral data store has the following general qualities:

1. Ephemeral state is not unique to I2RS work.

2. The ephemeral datastore is never locked.

3. The ephemeral portion of the intended configuration, applied state, and derived state does not persist over a reboot,

4. an ephemeral node cannot roll-back to its previous value,

5. Since ephemeral data store is just data that does not presist over a reboot, then in theory any node or group of nodes in a YANG data model could be ephemeral. The YANG data module must indicate what portion of the data model (if any) is ephemeral.

    * A YANG data module could be all ephemeral (e.g. [I-D.ietf-i2rs-rib-data-model]) with no directly associated configuration models,

    * A YANG model could be all ephemeral but associated with a configuration model

    * or a single data node or data tree could be made ephemeral.

6. The management protocol (NETCONF/RESTCONF) needs to signal which poritons of a data model(node, tree, or data model) are ephemeral in the module library [RFC7895].

7.3. I2RS Agent Caching of Ephemeral Data

The multiple control-plane datastore model [I-D.nmdsdt-netmod-revised-datastores] architecture allows multiple datastores which could allow an implementation of caching of ephemeral data in the I2RS Agent by having a main and a backup I2RS agent. Early implementations should at least support the single ephemeral data model, but MAY support the multiple datastore mode. It is important that these early implementations provide feedback for standardization on the following:

the policy knobs needed to make single ephemeral control planes datastores function,

the policy knobs neeed to make multiple ephemeral control plane datastores which support caching work.

7.4. Massive Amounts of Configuration Data

Large amounts of data can flow from the I2RS agent to the I2RS client, or from the I2RS client to the I2RS Agent. The I2RS client may set or query ephemeral configuration in the routing system via the I2RS agent and receive operational state, notifications, or logging from the I2RS Agent on behalf of the I2RS routing system. I2RS Clients can send large amount of ephemeral configuration data to the I2RS Agent. The writes may be done via NETCONF (<edit-config> or an rpc function), or via RESTCONF (PUT, PATCH, POST). Reads can be done via NETCONF <get-config> or RESTCONF GET or query.

The I2RS RIB Data Model [I-D.ietf-i2rs-rib-data-model] also supports the use of rpc to add/delete RIBs, add/delete/update routes, and add/delete nexthops. If the I2RS client does a small to medium number of writes to the I2RS ephemeral state in the I2RS Agent in a routing system, the full validation that NETCONF or RESTCONF does will be able to be done without any reduction in speed to the I2RS high-performance system. For example, if the I2RS RIB Data Model has adds a 1000 routes, the I2RS RIB use of rpc to add/delete/update routes should be able to provide a high-performance system. Alternatively the NETCONF <edit-config> could update these 1000 routes with a write, or the RESTCONF POST, PUT or PATCH should be able to add the 1000 routes.

If a large number of ephemeral routes or filters are written (updates or new) by the I2RS Client to the ephemeral state in the I2RS agent, one of the key issues for a high performance interface is the time it takes to validate routes. Due to this concern, the I2RS architecture was design to allow less than the full NETCONF or RESTCONF validation. The concept is that the I2RS routes would be validated within the I2RS client and sent via a 99.999% reliable connection. In this scenario, the I2RS Agent would trust the validation that the I2RS Client did, and the communication of the route additions via the network connection.

An experiment regarding this has been done with the ODL code base update of ephemeral routes, but additional experimentation needs to be done prior to finalizing this design. Section 3.4.2 reviews how this process might be done, but many open issues exist in implementing this "low-validation" interface. Without additional experimentation and prototype code, this type of "low-validation",

7.5. Write Error handling

This section reviews I2RS normal error handling and error handling for rpc with no validation checks.

7.5.1. Normal validation checks

An I2RS agent validates an I2RS client’s information by examining the following:

- message syntax validation,
- syntax validation for nodes of data model,
- referential checks (leafref checks MUST clauses, and instance indentifier),
- checks groups of data within a data model or groups of data across data models,
- write access to data,
- if write access and values already exist, if I2RS client write access is higher than existing priority.

7.5.1.1. Reduced Validation (Experimental)

Can the I2RS protocol allow for reduced error checking? The need for speed in the I2RS protocol insertions in to the I2RS RIB suggest that it is worth experimenting for reduced validation in order to obtain high levels of throughput. If NETCONF or RESTCONf streams pre-checked routes to the datastore, what happens? Implementation experience is needed to determine the feasibility of this approach.

This feature may require a operator-applied policy knob swith a "no validation" feature

- operator-applied policy knob enabling this feature;
- rpc in a data model with the yang "ephemeral-validation no-check;"

7.6. IPFIX for traffic monitoring

Due to the potentially large data flow the traffic measurment statistics generate, these statistics are best handled by publication techniques within NETCONF or a separate protocol such as IPFIX. In the future version of the I2RS protocol may desire to support a data

stream outbound from the I2RS Agent to an I2RS client via the IPFIX protocol.

7.7. Binary encoding of RESTCONF/NETCONF

The binary encoding of JSON or XML encodnig in RESTCONF or NETCONF may provide a better throughput. Research needs to be done on what is the appropriate binary encoding.

7.8. Ephemeral state in DDoS environments

I2RS ephemeral state may operate in places where there is a DDoS attacks where the network devices are attacked. Is one attack plane the ability to remove all tracing if the I2RS reboots an attack vector?

8. IANA Considerations

This is a protocol strawman - nothing is going to IANA.

9. Security Considerations

The security requirements for the I2RS protocol are covered in [I-D.ietf-i2rs-protocol-security-requirements]. The security environment the I2RS protocol is covered in [I-D.ietf-i2rs-security-environment-reqs]. Any person implementing or deploying the I2RS protocol should consider both security requirements.

10. Acknowledgements

TBD

11. References

11.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, DOI 10.17487/RFC4107, June 2005, <http://www.rfc-editor.org/info/rfc4107>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <http://www.rfc-editor.org/info/rfc4960>.

[RFC5339] Le Roux, JL., Ed. and D. Papadimitriou, Ed., "Evaluation of Existing GMPLS Protocols against Multi-Layer and Multi-Region Networks (MLN/MRN)", RFC 5339, DOI 10.17487/RFC5339, September 2008, <http://www.rfc-editor.org/info/rfc5339>.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <http://www.rfc-editor.org/info/rfc5424>.

[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>.

[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>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6244] Shafer, P., "An Architecture for Network Management Using NETCONF and YANG", RFC 6244, DOI 10.17487/RFC6244, June 2011, <http://www.rfc-editor.org/info/rfc6244>.

[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>.

[RFC7158] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7158, DOI 10.17487/RFC7158, March 2014, <http://www.rfc-editor.org/info/rfc7158>.

[RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <http://www.rfc-editor.org/info/rfc7589>.

[RFC7803] Leiba, B., "Changing the Registration Policy for the NETCONF Capability URNs Registry", BCP 203, RFC 7803, DOI 10.17487/RFC7803, February 2016, <http://www.rfc-editor.org/info/rfc7803>.

[RFC7895] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Module Library", RFC 7895, DOI 10.17487/RFC7895, June 2016, <http://www.rfc-editor.org/info/rfc7895>.

[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, <http://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, <http://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, <http://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, <http://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, <http://www.rfc-editor.org/info/rfc7950>.

[RFC7952] Lhotka, L., "Defining and Using Metadata with YANG", RFC 7952, DOI 10.17487/RFC7952, August 2016, <http://www.rfc-editor.org/info/rfc7952>.

[RFC7958] Abley, J., Schlyter, J., Bailey, G., and P. Hoffman, "DNSSEC Trust Anchor Publication for the Root Zone", RFC 7958, DOI 10.17487/RFC7958, August 2016, <http://www.rfc-editor.org/info/rfc7958>.

11.2. Informative References

[I-D.ietf-i2rs-ephemeral-state] Haas, J. and S. Hares, "I2RS Ephemeral State Requirements", draft-ietf-i2rs-ephemeral-state-22 (work in progress), November 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-17 (work in progress), September 2016.

[I-D.ietf-i2rs-rib-data-model]
Wang, L., Ananthakrishnan, H., Chen, M., amit.dass@ericsson.com, a., Kini, S., and N. Bahadur, "A YANG Data Model for Routing Information Base (RIB)", draft-ietf-i2rs-rib-data-model-06 (work in progress), July 2016.

[I-D.ietf-i2rs-rib-info-model]
Bahadur, N., Kini, S., and J. Medved, "Routing Information Base Info Model", draft-ietf-i2rs-rib-info-model-09 (work in progress), July 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-yang-l3-topology]
Clemm, A., Medved, J., Varga, R., Tkacik, T., Liu, X., Bryskin, I., Guo, A., Ananthakrishnan, H., Bahadur, N., and V. Beeram, "A YANG Data Model for Layer 3 Topologies", draft-ietf-i2rs-yang-l3-topology-04 (work in progress), September 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-keystore]
Watsen, K. and G. Wu, "Keystore Model", draft-ietf-netconf-keystore-00 (work in progress), October 2016.

[I-D.ietf-netconf-netconf-event-notifications]
Prieto, A., Clemm, A., Voit, E., Nilsen-Nygaard, E., Tripathy, A., Chisholm, S., and H. Trevino, "NETCONF Support for Event Notifications", draft-ietf-netconf-netconf-event-notifications-01 (work in progress), October 2016.

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-18 (work in progress), October 2016.

[I-D.ietf-netconf-rfc5277bis]
Clemm, A., Prieto, A., Voit, E., Nilsen-Nygaard, E., Tripathy, A., Chisholm, S., and H. Trevino, "Subscribing to Event Notifications", draft-ietf-netconf-rfc5277bis-01 (work in progress), October 2016.

[I-D.ietf-netconf-tls-client-server]
Watsen, K., "TLS Client and Server Models", draft-ietf-netconf-tls-client-server-01 (work in progress), November 2016.

[I-D.ietf-netconf-yang-patch]
Bierman, A., Bjorklund, M., and K. Watsen, "YANG Patch Media Type", draft-ietf-netconf-yang-patch-13 (work in progress), November 2016.

[I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", draft-ietf-netconf-yang-push-04 (work in progress), October 2016.

[I-D.ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch-11 (work in progress), October 2016.

[I-D.ietf-netmod-schema-mount]
Bjorklund, M. and L. Lhotka, "YANG Schema Mount", draft-ietf-netmod-schema-mount-03 (work in progress), October 2016.

[I-D.ietf-netmod-syslog-model]
Wildes, C. and K. Koushik, "A YANG Data Model for Syslog Configuration", draft-ietf-netmod-syslog-model-11 (work in progress), November 2016.

[I-D.nmdsdt-netmod-revised-datastores]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "A Revised Conceptual Model for YANG Datastores", draft-nmdsdt-netmod-revised-datastores-00 (work in progress), October 2016.

Authors' Addresses

Susan Hares
Huawei
Saline
US

Email: shares@ndzh.com

Amit Daas
Ericsson

Email: amit.dass@ericsson.com