I2RS working group
Internet-Draft
Intended status: Informational
Expires: September 12, 2017

S. Hares
Huawei
A. Dass
Ericsson
March 11, 2017

Yang for I2RS Protocol
draft-hares-netmod-i2rs-yang-04.txt

Abstract

This document requests yang language additions for the data models that exist as part of the I2RS control plane datastore. One of these additions is the ability to mark a portion of the model as having ephemeral state.

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 September 12, 2017.

Table of Contents

1. Introduction

This a proposal for additions to yang 1.1 [RFC7950] to support 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 IETF existing protocols

(NETCONF [RFC6241], RESTCONF [RFC8044], and others) 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].

This document suggests the following additions to Yang to support the I2RS control plane datastore. [I-D.ietf-i2rs-ephemeral-state] specifies the I2RS requirements for the ephemeral state.

Section 3 of this document defines optional additions to yang 1.1 to support I2RS ephemeral control plane datastore. The main addition is the datastore statement with four new substatements (dstype, ephemeral, protosup, validation). The protosup substatement has two valid substatements (protobase, protoadd). The validation substatement has has three new substatements: bulkchecks, caching, and nstransport.

Section 4 provides the augmentation to RFC7950 tables for these optional features.

2. Definitions

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.ietf-netmod-revised-datastores] for discussion of how the ephemeral datastore as a control plane datastore interacts with intended configuration datstore, the dynamic configuration protocols, and control planes datastore to create the applied datastore and operational state 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 is defined as the intended datastore as defined in [I-D.ietf-netmod-revised-datastores].

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.

applied datastore Read only datastore regarding configuration state installed in the routing system as defined in [I-D.ietf-netmod-revised-datastores].

operational state Read only datastore that combines applied datastore and operational state as defined in [I-D.ietf-netmod-revised-datastores].

operator-applied policy: is a policy that an operator sets that determines how the ephemeral datastore as a control plane data store interacts with intended configuration (see [I-D.ietf-netmod-revised-datastores]). This operator policy consists of setting a priority for each of the following (per [I-D.ietf-i2rs-ephemeral-state]):

* intended configuration,

* any dynamic configuration protocols,

* any control plane datastores (one of which is ephemeral.)

3. yang additions

3.1. datastoredef

The "datastoredef" is a statement that defines a datastore provides the ability to describe which datastore a module or submodule may be loaded into. Each datastore statement must refer to a name defined in a datastoredef statement.

The new substatements for the datstoredef command are dstype, ephemeral, module-list, precedence, protosup, and validation. The dstype provides information on the type of the datastore. The ephemeral flag indicates the datastore is ephemeral. The module-list provides a list of modules included in this datastore. the protosup indicates the protocol support for this datastore, and the validation provides information on the validation.

The "dsname" must be MUST be a nmae registered with IANA (see [I-D.ietf-netmod-revised-datastores]).

Data store syntax:

```
datastoredef <dsname> {
   <sub-statements>
};
```

dsname - Must be registered name for datastore

Figure 1

The substatements for the datastore substatement are listed below:

Table 1

substatement	This document section	RFC7960 section	cardinality
description	-	7.21.3	0..1
dstype	3.3	-	1
ephemeral	3.4	-	0..n
module		7.1	0..n
module-list	3.5	-	0..n
organization	-	7.1.7	0..1
precedence	3.6	-	0..n
protosup	3.7	-	0..n
reference	-	7.21.4	0..1
revision	-	7.1.9	0..n
revision-date	-	7.1.5.1	0..1
validation	3.8	-	0..n
version		7.1.9	0..n

Note:There is a variance with ephemeral control-plane datastore example in [I-D.ietf-netmod-revised-datastores] which uses "module" to define a datastore rather than datastore. Rather than assume the "module" will be re-used this document uses "datastoredef".

Example of use:

```
datastoredef i2rs-agent {
         dstype control-plane;
     description {"I2RS Agent datastore "};
         ephemeral true;
         module-list ietf-i2rs-rib, ietf-network, ietf-network-topology,
          ietf-l3-unicast-topology;
         protosup {
                protobase netconf;
                protoadd control-plane;
            protoadd ephemeral;
          }
         precedence applied {
          precedenceval 5; //set to high value//
         }
     precedence opstate {
          precedence 5;   //set to high value//
         }
         revision 0.0;
         version 1.0;

}
```

3.2. datastore

The "datastore" can be a substatement for the Yang Module statement provides the ability to describe which datastore a module or submodule may be loaded into. If no "datastore" statement exists, there is no restriction on the datastores a module or submodule can be loaded into.

The argument the datastore is a datastore name denoted as "dsname". The "dsname" must be MUST be a nmae registered with IANA (see [I-D.ietf-netmod-revised-datastores]).

The vaid substatements for the datastore statement are in Table 1. The "description" substatement provides a description of the datastore. The "dstype" provides information on the class (e.g., config or control plane) and the subclass of the datastore (e.g., i2rs). The ephemeral indicates that entire datastore is ephemeral. The validation provides alternate validation rules for the datastore.

Data store syntax:

```
datastore <dsname> {
   <sub-statements>
};
```

dsname - must be defined in a datastoredef statement

Figure 2

The substatements for the datastore substatement are listed below:

Table 2

substatement	This document section	RFC7960 section	cardinality
description	-	7.21.3	0..1
dstype	3.4	-	1
ephemeral	3.5	-	0..n
protosup	3.7	-	0..n
reference	-	7.21.4	0..1
revision	-	7.1.9	0..n
revision-date	-	7.1.5.1	0..1
validation	3.8	-	0..n
version		7.1.9	0..n

Application Comments:

A module may be mounted into different datastores. The datastore statement indicates which datastores a module may be mounted in, and the characteristics of each datastore.

Example of use where a module is utilized
in two different datastores.

```
module ietf-i2rs-rib {
      yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-i2rs-rib";
     // replace with iana namespace when assigned
         prefix "iir";
         import ietf-inet-types {
            prefix inet;
                //rfc6991
                }
          ....
     organization
       "IETF I2RS (Interface to Routing System) Working Group";
          ....

      datastore i2rs-agent {
            dstype "control-plane" "i2rs-vo";
                ephemeral true;
                protosup {
                 protobase netconf;
                 protoadd control-plane;
                 protoadd ephemeral;
                 }
           }
           datastore config {
                 dstype config;
                 ephemeral false;
                 protosup {
                  protobase restconf;
                 }
                 protosup {
                  protobase netconf;
                  }
           }

}
```

3.3. dstype

They substatement dstype indicates the datastore class and subclass of the datastore. A dstype substatement may only exist within a datastore statement.

Syntax of the dstype datastore is:

```
dstype <dsclass> <dsname>;

where:
    dsclass: ["config" | "control-plane ]
    dssubclass [ "i2rs-v0" ]
```

Figure 3

3.4. ephemeral

The ephemeral indicates that an object is ephemeral data which does not survive a reboot (see [I-D.ietf-i2rs-ephemeral-state]). The definition of the object may be a datastore, a module, a submodule, an action, a container, a grouping, a leaf, a leaf-list, a list, or an rpc.

Syntax is the following:
ephemeral [true | false];

The value "true" indicates the object is not ephemeral.
The value "false" indicates the value is ephemeral.

Figure 4

Note: There is a variance with ephemeral functionality with [I-D.ietf-netmod-revised-datastores]. Rather than consider the keyword ephemeral as a identity, this proposes ephemeral will be a yang substatement.

3.5. module-list

The module list contains a list of module names.

Syntax is the following:
module-list <module-name-1>, ... <module-name-n>;

Each name on the list (e.g. <module-name-1>)
must be a name in a module statement.

Figure 5

3.6. precedence

The precedence provides the value for precedence insertion of the datastores (the precedence substatement is contained) versus the datastore "dsname". Examples of datastore can be applied, opstate,

or other control plane datastores. If no precedence is statement is given, the configuration datastore takes precedence.

The module-list restricts this precedence for just these modules. A submodule must belong to one of the modules in the module list, and it further restricts the precedence value to just the submodule within the module.

Syntax is the following:

```
precedence [applied | opstate | <dsname> ] {
    <<precedence-substatements>>
};

dsname - registered name for datastore
```

Figure 6

Table 3

substatement	document section	RFC7960 section	cardinality
description	-	7.21.3	0..1
module-list	3.x	-	0..n
precedenceval	3.x	-	1
sub-modules		7.2	0..n

3.6.1. precedenceval

The precedenceval provides the value for precedence.

Syntax is the following:

```
precedenceval <precedence-integer>;

<precedence-integer>; - is the integer value for precedence.
```

Figure 7

3.7. protosup statement

This indicates which protocols support this datastore. The protocols can be netconf, restconf, coap, gprc, and bgp. The substatements for protosup are protocobase and protoadd

```
Syntax is the following:
protosup  {
    <<protosup substatements>>
}
```

Figure 8

Table 4

substatement	document section	RFC7960 section	cardinality
description	-	7.21.3	0..1
protobase	3.4.1	-	1..n
protoadd	3.4.2	-	1..n

3.7.1. protobase

The protobase substatement indicates the protocol a database can be sent over. The syntax is below:

Syntax for protobase:

```
protobase  [netconf | restconf | coap
            | bgp | isis | ospf | dots
                        | <protocol-name> ]
```

```
Where protocolo-name is one of protocol names
      registered by IANA.
```

Figure 9

3.7.2. protoadd

The protoadd specifies required optional additions to a protocol that sends information to a datastore. One example of such an addition is the additions to RESTCONF to support the I2RS protocol denoted as "i2rs".

Syntax for proto add:

```
protoadd  [control-plane | i2rs | i2nsf |
          | ephemeral | <proto-add-string>]
```

Figure 10

The protocol additions is the name of a capability or grouping of capabilities for support. For example, the "i2rs" capability is a capability which combines the capabilities the "control-plane" netconf capability with the netconf ephemeral capability.

3.8. validation

The validation keyword indicates that the object uses alternate validation besides the mechanisms defined by the configuration datastore as defined in [RFC7950]. The validation subcommand is invalid in any module or submodule which is only defined for the configuration datastore. Unless the module has a datastore statement which includes a datastore other than config, all validation statements in the module are ignored. Unless the submodule has a datastore statement which includes a datastore other than config, all validation statements are ignored.

The validation can be set on a datastore command in a a module, a submodule, an action, a container, a grouping, a leaf, a leaf-list, a list, or an rpc. The validation substatements include nstransport and bulk-checks as shown in table 3.

Syntax of the validation is:

```
 validation {
    <<validation-substatements>>
   };
```

Figure 11

Table 5

substatement	document section	RFC7960 section	cardinality
description	-	7.21.3	0..1
bulkchecks	3.8.1	-	0..1
caching	3.8.2	-	0..1
nstransport	3.8.3	-	0..1
organization	-	7.1.7	0..1
reference	-	7.21.4	0..n
revision-date	-	7.1.5.1	0..1
version		7.1.9	0..n

3.8.1. bulkcheck

The bulkcheck flag indicates whether this object uses bulk-check validation instead of the normal configuration datastore validation. The protocol updating the object must support bulk checking mechanism, or indicate that this object is not supported.

This is a new feature for control plane protocols and control plane datastores. In the configuration datastores, it is possible to support this feature at the validation level for the rpc object. Early implemementers of this feature for module which may loaded in the configuration datastore are encouraged to place bulkcheck features within "rpc" functionality.

bulkcheck syntax:

```
bulkchecks  [yes | no];
```

Figure 12

The value "no" indicates the object does not allows "bulkchecks" of data, and uses the normal configuration datastore checking. The value "yes" indicates the object does not allows "bulkchecks" of data within this object.

3.8.2. caching

The caching flag indicates whether the I2RS support caching of multiple client information within I2RS Agents.

Application note: This feature is not supported for the I2RS protocol version 0

caching syntax:

caching [yes | no];

Figure 13

The value "no" indicates the object does not allows "bulkchecks" of data, and uses the normal configuration datastore checking. The value "yes" indicates the object does not allows "bulkchecks" of data within this object.

3.8.3. nstransport

The nstransport indicates whether this object may be sent across a non-secure transport. Sending data across a non-secure transport should be done only if the circumstances warrant it.

This is a new feature for the I2RS control plane protocols and control plane datastores.

Caution: For a description of when a on-secure transport is appropriate for I2RS control plane protocol, please refer to the I2RS protocol security requirements [I-D.ietf-i2rs-protocol-security-requirements]. Implementers of this feature in an I2RS implementation should also review the I2RS security requirements [I-D.ietf-i2rs-security-environment-reqs]. No data which reveals any identity for a person or confidential information should be sent via a non-secure transport.

Syntax is the following:

nstransport [yes | no];

Figure 14

The value "no" indicates the object does not allows "bulkchecks" of data, and uses the normal configuration datastore checking. The value "yes" indicates the object does not allows "bulkchecks" of data within this object.

4. Change to RFC7950

The optional attributes add options to the tables for substatements for the module (section 7.1.1), submodule table, action, container, grouping, leaf, leaf-list, a list, and an rpc. This section provides the revised tables.

4.1. Additions to the Module table

This is the additions to module's substatment table in section 7.1.1 of [RFC7950].

7.1.1 substatement (replacement)

substatement	RFC7950 section	cardinality
anydata	7.10	0..n
anyxml	7.11	0..n
augment	7.17	0..n
choice	7.9	0..n
contact	7.1.8	0..1
container	7.5	0..n
description	7.21.3	0..1
deviation	7.20.3	0..n
extension	7.19	0..n
feature	7.20.1	0..n
grouping	7.12	0..n
identity	7.18	0..n
import	7.1.5	0..n
include	7.1.6	0..n
leaf	7.6	0..n
leaf-list	7.7	0..n
list	7.8	0..n
namespace	7.1.3	1
notification	7.16	0..n
organization	7.1.7	0..1
prefix	7.1.4	1
reference	7.21.4	0..1
revision	7.1.9	0..n
rpc	7.14	0..n
typedef	7.3	0..n
uses	7.13	0..n
yang-version	7.1.2	1
optional Yang 1.1 substatement	This document's section	cardinality
datastore	3.2	0..n
ephemeral	3.4	0..n
validation	3.8	0..n

4.2. Additions to the submodule substatement list

Below would be the replacement for the substatement table in setion 7.2.1 of [RFC7950] which lists the valid submodule statements.

7.2.1. The submodule's Substatements (replcement)

substatement	RFC7950 section	cardinality
anydata	7.10	0..n
anyxml	7.11	0..n
augment	7.17	0..n
belongs-to	7.2.2	1
choice	7.9	0..n
contact	7.1.8	0..1
container	7.5	0..n
description	7.21.3	0..1
deviation	7.20.3	0..n
extension	7.19	0..n
feature	7.20.1	0..n
grouping	7.12	0..n
identity	7.18	0..n
import	7.1.5	0..n
include	7.1.6	0..n
leaf	7.6	0..n
leaf-list	7.7	0..n
list	7.8	0..n
namespace	7.1.3	1
notification	7.16	0..n
organization	7.1.7	0..1
reference	7.21.4	0..1
revision	7.1.9	0..n
rpc	7.14	0..n
typedef	7.3	0..n
uses	7.13	0..n
yang-version	7.1.2	1
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.3. Additions to the container substatement list

Below would be the replacement for the substatement table in section 7.5.2 of [RFC7950] that lists the legal container substatements.

7.5.2. The container Substatements (replacement)

substatement	RFC7950 section	cardinality
action	7.15	0..n
anydata	7.10	0..n
anyxml	7.11	0..n
choice	7.9	0..n
config	7.21.1	0..1
description	7.21.3	0..1
grouping	7.12	0..n
if-feature	7.20.2	0..n
leaf	7.6	0..n
leaf-list	7.7	0..n
list	7.8	0..n
must	7.5.3	0..n
notification	7.16	0..n
presennce	7.5.5	0..1
reference	7.21.4	0..1
status	7.1.9	0..1
typedef	7.3	0..n
uses	7.13	0..n
when	7.21.5	0..1
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.4. Additions to leaf substatement list

Below would be replacement for the substatement table in section 7.6.2 of [RFC7950] which provides the leaf's substatements.

7.6.2 The leaf's Substatements (replacement)

substatement	RFC7950 section	cardinality
config	7.21.1	0..1
default	7.6.4	0..1
description	7.21.3	0..1
if-feature	7.20.2	0..n
mandatory	7.6.5	0..1
must	7.5.3	0..n
reference	7.21.4	0..1
status	7.21.2	0..1
type	7.6.3	1
units	7.3.3	0..1
when	7.21.5	0..1
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.5. Additions to leaf-list substatement list

Below would be the replacement for the substatement table in section 7.7.2 in [RFC7950] which provides the list of the leaf-lists substatements.

7.7.2 The leaf's Substatements (replacement)

substatement	RFC7950 section	cardinality
config	7.21.1	0..1
default	7.6.4	0..1
description	7.21.3	0..1
if-feature	7.20.2	0..n
max-elements	7.7.6	0..1
min-elements	7.7.5	0..1
must	7.5.3	0..n
ordered-by	7.7.7	0..1
reference	7.21.4	0..1
status	7.21.2	0..1
type	7.6.3	1
units	7.3.3	0..1
when	7.21.5	0..1

optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.6. Additions to list substatement list

Below would be the replacement for the table in section 7.8.1 in [RFC7950] which provides the list's substatements.

7.8.1 The list's Substatements (replacement)

substatement	RFC7950 section	cardinality
action	7.15	0..n
anydata	7.10	0..n
anyxml	7.11	0..n
choice	7.9	0..n
config	7.21.1	0..1
container	7.5	0..n
description	7.21.3	0..1
grouping	7.12	0..n
if-feature	7.20.2	0..n
key	7.8.2	0..n
leaf	7.6	0..n
leaf-list	7.7	0..n
list	7.8	0..n
max-elements	7.7.6	0..1
min-elements	7.7.5	0..1
must	7.5.3	0..n
notification	7.16	0..n }
ordered-by	7.7.7	0..1
reference	7.21.4	0..1
status	7.21.2	0..1
typedef	7.3	0..n
uses	7.13	0..n
when	7.21.5	0..1
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.7. Additions to the grouping substatement table

Below would be the replacement for the table 7.12.1 of [RFC7950] that lists the vaid substatments for the container substatements.

7.12.1. The grouping's Substatements (replacement)

substatement	RFC7950 section	cardinality
action	7.15	0..n
anydata	7.10	0..n
anyxml	7.11	0..n
choice	7.9	0..n
description	7.21.3	0..1
grouping	7.12	0..n
leaf	7.6	0..n
leaf-list	7.7	0..n
list	7.8	0..n
notification	7.16	0..n
reference	7.21.4	0..1
status	7.1.9	0..1
typedef	7.3	0..n
uses	7.13	0..n
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.8. Additions to the rpc substatement list

Below would be the replacement for the table in section 7.14.1 of [RFC7950] that lists the legal rpc substatements.

7.5.2. The rpc Substatements

substatement	RFC7950 section	cardinality
description	7.21.3	0..1
grouping	7.12	0..n
if-feature	7.20.2	0..n
input	7.14.2	0..1
output	7.14.3	0..1
reference	7.21.4	0..1
status	7.1.9	0..1
typedef	7.3	0..n
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

4.9. Additions to the action substatement list

Below would be the replacement for the table 7.15.1 of [RFC7950] that lists the legal action substatements.

7.5.2. The action Substatements

substatement	RFC7950 section	cardinality
description	7.21.3	0..1
grouping	7.12	0..n
if-feature	7.20.2	0..n
input	7.14.2	0..1
output	7.14.3	0..1
reference	7.21.4	0..1
status	7.1.9	0..1
typedef	7.3	0..n
optional Yang 1.1 substatement	This document's section	cardinality
ephemeral	3.4	0..n
validation	3.8	0..n

Figure 2

5. IANA Considerations

The additions for registries go here.

6. 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 these yang additions for an I2RS protocol should consider both security requirements.

7. Acknowledgements

tBD

8. References

8.1. 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>.

[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>.

[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>.

[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>.

[RFC8044] DeKok, A., "Data Types in RADIUS", RFC 8044, DOI 10.17487/RFC8044, January 2017, <http://www.rfc-editor.org/info/rfc8044>.

8.2. Informative References

[I-D.ietf-i2rs-ephemeral-state]
Haas, J. and S. Hares, "I2RS Ephemeral State Requirements", draft-ietf-i2rs-ephemeral-state-23 (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-security-environment-reqs]
Migault, D., Halpern, J., and S. Hares, "I2RS Environment Security Requirements", draft-ietf-i2rs-security-environment-reqs-03 (work in progress), March 2017.

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

Authors' Addresses

Susan Hares
Huawei
Saline
US

Email: shares@ndzh.com

Amit Daas
Ericsson

Email: amit.dass@ericsson.com