Netmod
Internet-Draft
Intended status: Standards Track
Expires: July 28, 2017

B. Lengyel
Ericsson

January 24, 2017

Schema Annotation for YANG Models
draft-lengyel-netmod-schema-annotation-00

Abstract

In some cases there is a need to qualify other YANG models or some schema nodes in other models with additional properties (a.k.a. annotations). This document defines a method to add annotations in the form of YANG extension statements to YANG modules. The annotations are defined in separate files without modifying the text of the original YANG module.

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 July 28, 2017.

Table of Contents

1. Introduction

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 [RFC2119].

In some cases there is a need to qualify other YANG models or some schema nodes in other models with additional properties (a.k.a. annotations). This document defines a method to add annotations in the form of YANG [RFC7950] extension statements to YANG modules. The annotations are defined in separate files without modifying the text of the original YANG module.

Annotations are often specific to the model and/or implementation so they are possible to define in design time. System integrators, people defining configuration ahead of time and others need this information (without reading any data from a live node).

Often the basic YANG model and the extra properties are defined

separately, possibly by different organizations. Vendors may qualify standard models with extra properties, or even a vendor defined base model, may need different annotations depending on the specific implementation. Vendors are reluctant to modify the standard models (messing up the original file, too many variants of the base model),

so it must be possible to define such annotations in a separate file/ module.

1.1. Use Case - Yangpush

In YangPush [I-D.ietf-netconf-yang-push] we need to specify for data nodes whether the Yang server supports sending on-change data change notifications for the specific data node or not. Some servers may not support such notifications for specific data nodes because they are meaningless or would result in too many notifications or the server might simply not have implemented them. The support for on-change notifications will be defined using a YANG extension statement.

In some cases there is a need to specify the minimum acceptable interval for a periodic subscription for a specific data node or schema branch.

1.2. Use Case - Vendor Annotation of a Standard Model

Vendors sometimes want to add extra information to the standard YANG models. Examples include instructions for automatic code generation or formal description of side-effects of manipulating a data node. It would be good for standard SW tools to be at least aware of the vendor extensions, allowing vendor specific plugins to handle these extensions. By providing a standard mechanism to annotate modules generic SW tools might provide somevisibility to vendor specific extensions e.g. providing APIs to fetch the extension, displaying the extension as is.

2. Possible solutions

All solution alternatives use an Absolute schema node identifier to identify which part of the YANG schema is annotated. Absolute schema node identifier is defined in [RFC7950].

All alternatives will have the same effect as if the YANG extension statement in the annotation was placed in the base model itself.

The process of annotation will consist of the following steps. A base model will be defined. E.g. IETF defined a standard base model like ietf-system. A vendor will define extra data like instance data or annotation/deviation YANG modules (e.g. ericsson-system-ann-router1230-release3.yang). Annotation information MAY be defined once per vendor or for every product/product release separately. Annotation information will be available both in design-time as a document and in run-time.

2.1. Instance Data based Solution

A new YANG module containing information about schema-annotations is defined and a standard way of loading initial instance data in to that module is specified. (Note the module is only an example to demonstrate the concept. If the solution is selected, further work on the module is needed.)

```
    module ietf-schema-annotations {
        list schema-annotations {
            key target;
            config false;
            leaf target  {
                type schema-node-id;
                description "identifies the node in the schema tree
                that is annotated.";
            }

            list annotation {
                key statement;
                min-elements 1;
                leaf statement {
                    type yang:yang-identifier;
                    description "The statement to be conceptually
                    added to the target schema node.";
                }
                leaf argument {
                    type string;
                    description "Argument to the annotation
```

```
                        statement if any.";
                    }
                }
            }
        }
```

A standard format to specify initial instance data for a YANG module in a file is defined. The format SHALL be the same as the XML content of the <edit-config> operation inside the <config> element. The <edit-config> format was chosen as that will allow specifying the operation attribute. Both the merge and the replace values might be needed and/or useful.

If the instance data represents configuration data it SHALL be loaded by the YANG server at startup into the running datastore of the server before any management interface like CLI, Netconf, Restconf is started. Note this MAY result in overwriting data previously stored in the datastore. If the instance data represents state data it SHALL be loaded by the YANG server at startup into the operation-

state datastore before any management interface like CLI, Netconf, Restconf is started.

Example:

```
  <model-annotations xmlns:...>
      <target xmlns:sys=...>/sys:system/sys:system-time</target>
      <annotation>
          <statement xmlns:yp=...>yp:notifiable-on-change</statement>
          <argument>false</argument>
      </annotation>
  </model-annotations>
```

Instance data for the module ietf-schema-annotations SHOULD become a design time artifact provided by the model designer just as the YANG modules themselves.

The XML instance data defining the annotations would be available in design-time as an XML document and in run-time using the <get> operation.

2.2. Extension based Solution

A new YANG extension "schema-annotation" is defined, that allows adding statement to another module.


```
    extension schema-annotation {
        description
            "Annotates an existing statement with a YANG extension
            statement defined in some module. Useful for
            adding extension statements to a module without touching the
            module source.

            Substatements to this statements are conceptually added
            to the target statement.

            The argument is a 'absolute schema node identifier'.
            It identifies a target node in the schema tree to
            annotate with new statements.";

        argument target;
```

```
}
```

Example:

```
sann:schema-annotation /sys:system/sys:system-time {
    yp:notifiable-on-change false;
}
```

The annotating module would be available in design-time as a document and in run-time using the <get-schema> operation.

2.3. Deviation based Solution

Deviations already allow adding/modifying model information for another module. If IETF would allow the "deviate" statement to handle YANG extension statements this would allow us to add/remove/modify annotations.

2.4. Pros and Cons

The Instance Data based Solution is the proposed solution.

- It could be usable for other purposes as well including but not restricted to defining default security rules for access control [RFC6536] or defining yang-schema-mounts [I-D.ietf-netmod-schema-mount] in design time.
- It is more simple to read in runtime for management systems (MS).
- If annotation data was ever changed the MS could get notifications about that.

- It could easily be linked with references to the ietf-yang-library
- It introduces a bigger new concept (instance data) although a number of vendors are already using it.

The Extension Based Solution works.

We do not recommend using the Deviation based Solution as deviations

are for documenting unwelcome implementation limitations and are highly discouraged. Adding our extra information is not just acceptable, it is the recommended thing to do. Extending the allowed substatements for the deviate statement can also be considered changing [RFC7950].

3. How is this different from RFC7952 "Defining and Using Metadata with YANG?"

[RFC7952] defines metadata on the instance data level, here we need metadata on the model/schema level.

[RFC7952] metadata is transferred in Netconf/Restconf RPCs, here we usually do not need that.

[RFC7952] metadata is only available in runtime, here we need it to be available in design time.

4. Open Issues

- Set of statements that can be the target. The initial proposal is to allow the same set of schema nodes as "deviation" statement. Is that OK?

- Set of statements that can be added. The initial proposal allows only YANG extension statements. Do we want to allow anything else like additional "must" statements or restrictions? Probably not as that would allow redefining the base model, which we don't want. We only want to add information to it.

- Exact data model for module ietf-schema-annotations.

- Do we need replace and delete operations beside the add? The initial proposal only allows adding annotations. Do we need to be able to replace and remove them as well?

5. Acknowledgements

The author wishes to thank Eric Voit, Einar Nilsen-Nygaard and Alexander Clemm for their helpful comments contributing to this draft.

6. IANA Considerations

No IANA considerations

7. Security Considerations

This document introduces a mechanism for attaching metadata defined in the form of YANG extensions to the YANG schema of another YANG module. By itself, this mechanism represents no security threat. Security implications of a particular extension and attaching it to a particular YANG module shall be considered in the document defining/ attaching the extension.

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>.

[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>.

[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>.

8.2. Informative References

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

Author's Address

Balazs Lengyel
Ericsson

Email: balazs.lengyel@ericsson.com