Network Working Group
Internet-Draft
Obsoletes: rfc7895 (if approved)
Intended status: Standards Track
Expires: May 3, 2018

A. Bierman
YumaWorks
M. Bjorklund
Tail-f Systems
K. Watsen
Juniper Networks
October 30, 2017

YANG Library
draft-ietf-netconf-rfc7895bis-02

Abstract

This document describes a YANG library that provides information about all the YANG modules and datastores used by a network management server (e.g., a Network Configuration Protocol (NETCONF) server). Simple caching mechanisms are provided to allow clients to minimize retrieval of this information.

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 3, 2018.

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

There is a need for standard mechanisms to provide the operational state of a network management server. This includes, for instance, identifying the YANG modules and datastores that are in use by a server and how they relate to each other.

This document defines a YANG module that can be used to provide this informaton, in a way that is compatible with the NMDA [I-D.ietf-netmod-revised-datastores], but that is also backwards compatible with the "YANG Module Library" YANG module defined in [RFC7895].

If a large number of YANG modules are utilized by the server, then the YANG library contents needed can be relatively large. This information changes very infrequently, so it is important that clients be able to cache the YANG library contents and easily identify whether their cache is out of date.

YANG library information can be different on every server and can change at runtime or across a server reboot.

If the server implements multiple protocols to access the YANG-defined data, each such protocol has its own conceptual instantiation of the YANG library.

The following information is needed by a client application (for each YANG module in the library) to fully utilize the YANG data modeling language:

- identifier: a unique identifier for the module that includes the module's name, revision, submodules, features, and deviations.
- name: The name of the YANG module.
- revision: Each YANG module and submodule within the library SHOULD have a revision. This is derived from the most recent revision statement within the module or submodule.
- submodule list: The name, and if defined, revision of each submodule used by the module MUST be identified.
- feature list: The name of each YANG feature supported by the server, in a given datastore schema, MUST be identified.
- deviation list: The name of each YANG module used for deviation statements, in a given datastore schema, MUST be identified.

The following information is needed by a client application (for each datastore supported by the server) to fully access all the YANG-modelled data available on the server:

- identity: the YANG identity for the datastore.
- modules: modules supported by the datastore, including any features and deviations.

1.1. Terminology

The keywords "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].

The following terms are defined in [RFC6241]:

- client
- server

The following terms are defined in [RFC7950]:

- module
- submodule

The following terms are defined in ^NMDA":

- conventional configuration datastore
- operational datastore
- datastore schema

The following terms are used within this document:

- YANG library: A collection of metadata describing the server's operational state.

1.2. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

1.3. Motivation for rfc7895bis

RFC Ed.: delete this section, including this note, at time of publication.

All NETCONF servers supporting YANG 1.1 [RFC7950] MUST support YANG Library (see Section 5.6.4 of RFC 7950). Similarly, all RESTCONF servers MUST support YANG Library (see Section 10 of RFC 8040).

These requirements are independent of if the server supports NMDA or not.

RFC 7895 has a mandatory to implement 'modules-state' tree that a server uses to advertise all the modules it supports. However, this module was designed assuming the all modules would be in all datastores, and with the same number of features and deviations. However, this is not the case with NMDA-compatible servers that may have some modules that only appear in <operational> (e.g., ietf-network-topo) or only also appear in a dynamic datastore (e.g., i2rs-ephemeral-rib). It is also possible that a server only implements a module in <running>, as it hasn't yet coded support for returning the module's opstate yet. Presumably, an NMDA-supporting server would return all modules implemented in every datastore, but this would be misleading to existing clients and unhelpful to NMDA-aware clients.

In the end, it appears that the 'modules-state' node should be for non-NMDA aware clients. For backwards compatability, an NMDA-supporting server SHOULD populate 'modules-state' in a backwards-compatible manner. The new 'yang-library' node would be ignored by legacy clients, while providing all the data needed for NMDA-aware clients, which would themselves ignore the 'modules-state' tree.

The solution presented in this document is further motivated by the following desires:

- leverage Section 5.6.4 of RFC 7950 and Section 10 of RFC 8040.
- indicate which modules are supported by each datastore
- enable the features and deviations to vary by datastore
- structure extensible to support schema-mount
- provide a top-level container for all server metadata

1.4. Summary of Changes from RFC 7895

This document updates [RFC7895] in the following ways:

- Renames document title from "YANG Module Library" to "YANG Library".
- Adds a new top-level "yang-library" container to hold many types of server metadata: modules supported, datastores supported, relationships between datastores and modules, etc.

- Adds a set of new groupings as replacements for the deprecated "module-list" grouping.
- Adds a "yang-library-update" notification as a replacement for the deprecated "yang-library-change" notification.
- Deprecates the "modules-state" tree.
- Deprecates the "module-list" grouping.
- Deprecates the "yang-library-change" notification.

2. YANG Library

The "ietf-yang-library" module provides information about the modules and datastores supported by a server. This module is defined using YANG version 1.1, but it supports the description of YANG modules written in any revision of YANG.

Following is the YANG Tree Diagram for the "ietf-yang-library" module, excluding the deprecated 'modules-state' tree:

```
module: ietf-yang-library
    +--ro yang-library
       +--ro modules
       |  +--ro module* [id]
       |     +--ro id                  string
       |     +--ro name                yang:yang-identifier
       |     +--ro revision?           revision-identifier
       |     +--ro schema?             inet:uri
       |     +--ro namespace           inet:uri
       |     +--ro feature*            yang:yang-identifier
       |     +--ro deviation* [module]
       |     |  +--ro module    -> ../../id
       |     +--ro conformance-type    enumeration
       |     +--ro submodule* [name]
       |        +--ro name        yang:yang-identifier
       |        +--ro revision?   revision-identifier
       |        +--ro schema?     inet:uri
       +--ro module-sets
       |  +--ro module-set* [id]
       |     +--ro id        string
       |     +--ro module*   -> ../../../modules/module/id
       +--ro datastores
       |  +--ro datastore* [name]
       |     +--ro name          identityref
       |     +--ro module-set
       |             -> ../../../module-sets/module-set/id
       +--ro checksum       string

  notifications:
    +---n yang-library-update
```

2.1. yang-library

This container holds all of the server's metadata.

2.1.1. yang-library/modules/module

This list contains one entry for each unique instance of a module in use by the server. Each entry is distinguished by the module's name, revisions, features, and deviations.

A server cannot implement different revisions of the same module in different datastores, so there MUST only be a single revision of a given module in the "module" list that has a "conformance-type" leaf of "implement".

Although the server is at liberty to choose the unique "id" for each entry in the "module" list, it is suggested to use an "id" that would

be meaningful to clients. For example, "ietf-interfaces@2017-08-17" could be used as the "id" for a particular revision of the IETF interfaces YANG module. For a module that holds deviations or features specific to a datastore perhaps an "id" that also contains the datastore name, like "ietf-interfaces-deviations@2017-08-17/operational" could be used.

2.1.2. yang-library/module-sets/module-set

A "module-set" represents the datastore schema that is used by one or more datastores.

This list contains one entry for each module set in use by the server (e.g., presented by a datastore).

All conventional configuration datastores use the same datastore schema, which can normally be modelled using a single module set.

A dynamic configuration datastore may use a different datastore schema from the conventional configuration datastores, and hence may require a separate module set definition.

The datastore schema for the operational datastore is the superset of the datastore schema of all the configuration datastores, but can have unsupported nodes removed via datastore specific deviations or by omitting one or more modules from the module set associated with the operational datastore.

However, where possible, the same module set used for conventional configuration datatstores should also be used for the operational datastore. For example, deviations that are specific to "config false" nodes could still be applied to a common module instance that is also included in the module set used for the conventional datastores.

Although the server is at liberty to choose the unique "id" for the "module-set" list entry, it is suggested to choose an "id" that would be meaningful to clients, perhaps including the datastore name if the module set is only relevant to a single datastore.

2.1.3. yang-library/datastores/datastore

This list contains one entry for each datastore supported by the server, and identifes the datastore schema associated with a datastore via a reference to a module-set. Each supported conventional configuration datastore has a separate entry.

2.2. YANG Library Module

The "ietf-yang-library" module defines monitoring information for the YANG modules used by a server.

The modules "ietf-yang-types" and "ietf-inet-types" from [RFC6991] and the module "ietf-datastores" from [I-D.ietf-netmod-revised-datastores] are used by this module for some type definitions.

RFC Ed.: update the date below with the date of RFC publication and remove this note.

<CODE BEGINS> file "ietf-yang-library@2017-10-30.yang"

```
module ietf-yang-library {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-library";
  prefix "yanglib";

  import ietf-yang-types {
    prefix yang;
    reference "RFC 6991: Common YANG Data Types.";
  }
  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991: Common YANG Data Types.";
  }
  import ietf-datastores {
    prefix ds;
    reference "I-D.ietf-revised-datastores:
               Network Management Datastore Architecture.";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";

  contact
    "WG Web:   <http://tools.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>

     Author:   Andy Bierman
               <mailto:andy@yumaworks.com>

     Author:   Martin Bjorklund
               <mailto:mbj@tail-f.com>

     Author:   Kent Watsen
```

```
                <mailto:kwatsen@juniper.net>";

    description
      "This module contains information about the YANG server
       instance, including the modules and datastores the
       server supports, and which modules are present in
       which datastores.

       Copyright (c) 2017 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Simplified BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX; see
       the RFC itself for full legal notices.";

    // RFC Ed.: update the date below with the date of RFC publication
    // and remove this note.
    // RFC Ed.: replace XXXX with actual RFC number and remove this
    // note.
    revision 2017-10-30 {
      description
        "Updated revision.";
      reference
        "RFC XXXX: YANG Library.";
    }
    revision 2016-04-09 {
      description
        "Initial revision.";
      reference
        "RFC 7895: YANG Module Library.";
    }

    /*
     * Typedefs
     */

    typedef revision-identifier {
      type string {
        pattern '\d{4}-\d{2}-\d{2}';
      }
      description
        "Represents a specific date in YYYY-MM-DD format.";
```

```
    }

    /*
     * Groupings
     */

    grouping module-identification-leafs {
      description
        "Parameters for identifying YANG modules and submodules.";

      leaf name {
        type yang:yang-identifier;
        mandatory true;
        description
          "The YANG module or submodule name.";
      }
      leaf revision {
        type revision-identifier;
        description
          "The YANG module or submodule revision date.  If no revision
           statement is present in the YANG module or submodule, this
           leaf is not instantiated.";
      }
    }

    grouping schema-leaf {
      description
        "Common schema leaf parameter for modules and submodules.";

      leaf schema {
        type inet:uri;
        description
          "Contains a URL that represents the YANG schema
           resource for this module or submodule.

           This leaf will only be present if there is a URL
           available for retrieval of the schema for this entry.";
      }
    }

    grouping implementation-parameters {
      description
        "Parameters for describing the implementation of a module.";

      leaf-list feature {
        type yang:yang-identifier;
        description
          "List of YANG feature names from this module that are
```

```
           supported by the server, regardless whether they are defined
           in the module or any included submodule.";
       }
       list deviation {
         key "module";
         description
           "List of YANG deviation modules used by
            this server to modify the conformance of the module
            associated with this entry.  Note that the same module can
            be used for deviations for multiple modules, so the same
            entry MAY appear within multiple 'module' entries.";

         leaf module {
           type leafref {
             path "../../id";
           }
           description
             "The module that deviates the module associated with this
              entry.  The deviation modules MUST be part of the same
              module-sets as the module it deviates.";
         }
       }
       leaf conformance-type {
         type enumeration {
           enum implement {
             description
               "Indicates that the server implements one or more
                protocol-accessible objects defined in the YANG module
                identified in this entry.  This includes deviation
                statements defined in the module.

                For YANG version 1.1 modules, there is at most one
                module entry with conformance type 'implement' for a
                particular module name, since YANG 1.1 requires that at
                most one revision of a module is implemented.

                For YANG version 1 modules, there SHOULD NOT be more
                than one module entry for a particular module name.";
           }
           enum import {
             description
               "Indicates that the server imports reusable definitions
                from the specified revision of the module, but does not
                implement any protocol accessible objects from this
                revision.

                Multiple module entries for the same module name MAY
                exist. This can occur if multiple modules import the
```

```
              same module, but specify different revision-dates in the
              import statements.";
          }
        }
        mandatory true;
        description
          "Indicates the type of conformance the server is claiming
           for the YANG module identified by this entry.";
      }
    }

    grouping yang-library-parameters {
      description
        "The YANG library data structure is represented as a grouping
         so it can be reused in configuration or another monitoring
         data structure.";

      container modules {
        description
          "A container holding a list of modules.  Modules being
           listed here does not necessarily mean that they are
           supported by any particular datastore.

           If a module has the value 'implemented' for it's
           'conformance-type' leaf in the 'module' list, it means that
           the server supports the rpcs and notifications defined in
           the module (adjusted for features and deviations), even if
           the module is not supported in any particular datastore.";

        list module {
          key "id";
          description
            "Each entry represents a revision of a module currently
             supported by the server with a particular set of supported
             features and deviations";

          leaf id {
            type string;
            description
              "A unique identifier, independent of any other part
               of this module instance.";
          }

          uses module-identification-leafs;
          uses schema-leaf;

          leaf namespace {
            type inet:uri;
```

```
          mandatory true;
          description
            "The XML namespace identifier for this module.";
        }

        uses implementation-parameters;

        list submodule {
          key "name";
          description
            "Each entry represents one submodule within the
             parent module.";
          uses module-identification-leafs;
          uses schema-leaf;
        }
      }
    }

    container module-sets {
      description
        "A container for the list of module-sets supported by the
         server";

      list module-set {
        key "id";
        description
          "An arbitrary module-set definition provided by the
           server.

           A module-set represents a datastore schema associated with
           one or more datastores.

           Module-sets being listed here does not necessarily mean
           that they are used by any particular datastore.

           In the case of a configuration datastore, only the config
           true subset of a datastore schema is applicable to the
           datastore, any config false schema nodes and any action
           statements are ignored.";

        leaf id {
          type string;
          description
            "A system-generated value that uniquely represents the
             referenced set of modules.  Any change to the number
             of modules referenced, or to the modules themselves,
             generates a different value.";
        }
```

```
          leaf-list module {
            type leafref {
              path "../../../modules/module/id";
            }
            description
              "A module-instance supported by the server, including its
               features and deviations.";
          }
        }
      }

      container datastores {
        description
          "A container for the list of datastores supported by the
           server.";

        list datastore {
          key "name";
          description
            "A datastore supported by this server.

             Each datastore indicates which module-set it supports.

             The server MUST instantiate one entry in this list per
             specific datastore it supports.

             Each datstore entry with the same datastore schema SHOULD
             reference the same module-set.";

          leaf name {
            type identityref {
              base ds:datastore;
            }
            description
              "The identity of the datastore.";
          }
          leaf module-set {
            type leafref {
              path "../../../module-sets/module-set/id";
            }
            mandatory true;
            description
              "A reference to the module-set supported by this
               datastore";
          }
        }
      }
    }
```

```
     /*
      * Top-level container
      */

     container yang-library {
       config false;
       description
         "Container providing all the YANG meta information the
          server possesses.";

       uses yang-library-parameters;

       leaf checksum {
         type string;
         config false;
         mandatory true;
         description
           "A server-generated checksum of the contents of the
            'yang-library' tree.  The server MUST change the value of
            this leaf if the information represented by the
            'yang-library' tree, except yang-library/checksum, has
            changed.";
       }
     }

     /*
      * Notifications
      */

     notification yang-library-update {
       description
         "Generated when any YANG library information on the
          server has changed.";
     }

     /*
      * Legacy groupings
      */

     grouping module-list {
       status deprecated;
       description
         "The module data structure is represented as a grouping
          so it can be reused in configuration or another monitoring
          data structure.";

       grouping common-leafs {
         status deprecated;
```

```
          description
            "Common parameters for YANG modules and submodules.";

          leaf name {
            type yang:yang-identifier;
            status deprecated;
            description
              "The YANG module or submodule name.";
          }
          leaf revision {
            type union {
              type revision-identifier;
              type string {
                length 0;
              }
            }
            status deprecated;
            description
              "The YANG module or submodule revision date.
               A zero-length string is used if no revision statement
               is present in the YANG module or submodule.";
          }
        }

        list module {
          key "name revision";
          status deprecated;
          description
            "Each entry represents one revision of one module
             currently supported by the server.";

          uses common-leafs {
            status deprecated;
          }
          uses schema-leaf {
            status deprecated;
          }

          leaf namespace {
            type inet:uri;
            mandatory true;
            status deprecated;
            description
              "The XML namespace identifier for this module.";
          }
          leaf-list feature {
            type yang:yang-identifier;
            status deprecated;
```

```
        description
          "List of YANG feature names from this module that are
           supported by the server, regardless whether they are
           defined in the module or any included submodule.";
      }
      list deviation {
        key "name revision";
        status deprecated;
        description
          "List of YANG deviation module names and revisions
           used by this server to modify the conformance of
           the module associated with this entry.  Note that
           the same module can be used for deviations for
           multiple modules, so the same entry MAY appear
           within multiple 'module' entries.

           The deviation module MUST be present in the 'module'
           list, with the same name and revision values.
           The 'conformance-type' value will be 'implement' for
           the deviation module.";
        uses common-leafs {
          status deprecated;
        }
      }
      leaf conformance-type {
        type enumeration {
          enum implement {
            description
              "Indicates that the server implements one or more
               protocol-accessible objects defined in the YANG module
               identified in this entry.  This includes deviation
               statements defined in the module.

               For YANG version 1.1 modules, there is at most one
               module entry with conformance type 'implement' for a
               particular module name, since YANG 1.1 requires that
               at most one revision of a module is implemented.

               For YANG version 1 modules, there SHOULD NOT be more
               than one module entry for a particular module name.";
          }
          enum import {
            description
              "Indicates that the server imports reusable definitions
               from the specified revision of the module, but does
               not implement any protocol accessible objects from
               this revision.
```

```
                Multiple module entries for the same module name MAY
                exist. This can occur if multiple modules import the
                same module, but specify different revision-dates in
                the import statements.";
            }
          }
          mandatory true;
          status deprecated;
          description
            "Indicates the type of conformance the server is claiming
             for the YANG module identified by this entry.";
        }
        list submodule {
          key "name revision";
          status deprecated;
          description
            "Each entry represents one submodule within the
             parent module.";
          uses common-leafs {
            status deprecated;
          }
          uses schema-leaf {
            status deprecated;
          }
        }
      }
    }

    /*
     * Legacy operational state data nodes
     */

    container modules-state {
      config false;
      status deprecated;
      description
        "Contains YANG module monitoring information.";

      leaf module-set-id {
        type string;
        mandatory true;
        status deprecated;
        description
          "Contains a server-specific identifier representing
           the current set of modules and submodules.  The
           server MUST change the value of this leaf if the
           information represented by the 'module' list instances
           has changed.";
```

```
      }

      uses module-list {
        status deprecated;
      }
    }

    /*
     * Legacy notifications
     */

    notification yang-library-change {
      status deprecated;
      description
        "Generated when the set of modules and submodules supported
         by the server has changed.";
      leaf module-set-id {
        type leafref {
          path "/yanglib:modules-state/yanglib:module-set-id";
        }
        mandatory true;
        status deprecated;
        description
          "Contains the module-set-id value representing the
           set of modules and submodules supported at the server
           at the time the notification is generated.";
      }
    }

  }

  <CODE ENDS>
```

3. IANA Considerations

3.1. YANG Module Registry

RFC 7895 previously registered one URI in the IETF XML registry [RFC3688]. Following the format in RFC 3688, the following registration was made:

```
     URI: urn:ietf:params:xml:ns:yang:ietf-yang-library
     Registrant Contact: The NETCONF WG of the IETF.
     XML: N/A, the requested URI is an XML namespace.
```

This document takes over this registration entry made by RFC 7895.

RFC 7895 previously registered one YANG module in the "YANG Module Names" registry [RFC6020] as follows:

```
name:         ietf-yang-library
namespace:    urn:ietf:params:xml:ns:yang:ietf-yang-library
prefix:       yanglib
reference:    RFC 7895
```

This document takes over this registration entry made by RFC 7895.

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

- /modules-state/module: The module list used in a server implementation may help an attacker identify the server capabilities and server implementations with known bugs. Although some of this information may be available to all users via the NETCONF <hello> message (or similar messages in other management protocols), this YANG module potentially exposes additional details that could be of some assistance to an attacker. Server vulnerabilities may be specific to particular modules, module revisions, module features, or even module deviations. This information is included in each module entry. For example, if a particular operation on a particular data node is known to cause a server to crash or significantly degrade device performance, then the module list information will help an attacker identify server implementations with such a defect, in order to launch a denial-of-service attack on the device.

5. Acknowledgements

Contributions to this material by Andy Bierman are based upon work supported by the The Space & Terrestrial Communications Directorate (S&TCD) under Contract No. W15P7T-13-C-A616. Any opinions, findings and conclusions or recommendations expressed in this material are

those of the author(s) and do not necessarily reflect the views of The Space & Terrestrial Communications Directorate (S&TCD).

6. References

6.1. Normative References

[I-D.ietf-netmod-revised-datastores]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-05 (work in progress), October 2017.

[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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc6242>.

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

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

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

[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>.

6.2. Informative References

[RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008, <https://www.rfc-editor.org/info/rfc5277>.

[RFC6470] Bierman, A., "Network Configuration Protocol (NETCONF) Base Notifications", RFC 6470, DOI 10.17487/RFC6470, February 2012, <https://www.rfc-editor.org/info/rfc6470>.

Authors' Addresses

Andy Bierman
YumaWorks

Email: andy@yumaworks.com

Martin Bjorklund
Tail-f Systems

Email: mbj@tail-f.com

Kent Watsen
Juniper Networks

Email: kwatsen@juniper.net