Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: July 21, 2021

M. Veillette, Ed.
Trilliant Networks Inc.
A. Pelov, Ed.
I. Petrov, Ed.
Acklio

January 17, 2021

YANG Schema Item iDentifier (YANG SID)
draft-ietf-core-sid-15

Abstract

YANG Schema Item iDentifiers (YANG SID) are globally unique 63-bit unsigned integers used to identify YANG items. This document defines the semantics, the registration, and assignment processes of YANG SIDs. To enable the implementation of these processes, this document also defines a file format used to persist and publish assigned YANG SIDs.

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 https://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 21, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

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 3
2. Terminology and Notation 4
3. ".sid" file lifecycle 5
4. ".sid" file format 6
5. Content-Types 12
6. Security Considerations 12
7. IANA Considerations 13
7.1. YANG Namespace Registration 13
7.2. Register ".sid" File Format Module 13
7.3. CoAP Content-Formats Registry 13
7.4. Create new IANA Registry: "YANG SID Mega-Range" registry 14
7.4.1. Structure 14
7.4.2. Allocation policy 14
7.4.2.1. First allocation 15
7.4.2.2. Consecutive allocations 15
7.4.3. Initial contents of the Registry 15
7.5. Create a new IANA Registry: IETF YANG SID Range Registry (managed by IANA) 15
7.5.1. Structure 15
7.5.2. Allocation policy 16
7.5.3. Initial contents of the registry 17
7.6. Create new IANA Registry: "IETF YANG SID Registry" . . . 18
7.6.1. Structure 18
7.6.2. Allocation policy 18
7.6.3. Recursive Allocation of YANG SID Range at Document Adoption 19
7.6.4. Initial contents of the registry 20
8. Acknowledgments 21
9. References 21
9.1. Normative References 21
9.2. Informative References 22
Appendix A. ".sid" file example 23
Appendix B. SID auto generation 32
Appendix C. ".sid" file lifecycle 33
C.1. ".sid" File Creation 33
C.2. ".sid" File Update 35
Authors' Addresses 36

1. Introduction

Some of the items defined in YANG [RFC7950] require the use of a unique identifier. In both NETCONF [RFC6241] and RESTCONF [RFC8040], these identifiers are implemented using names. To allow the implementation of data models defined in YANG in constrained devices and constrained networks, a more compact method to identify YANG items is required. This compact identifier, called YANG SID or simply SID in this document and when the context is clear, is encoded using a 63-bit unsigned integer. The limitation to 63-bit unsigned integers allows SIDs to be manipulated more easily on platforms that might otherwise lack native 64-bit unsigned arithmetic. The loss of a single bit of range is not significant given the size of the remaining space.

The following items are identified using SIDs:

- identities

- data nodes (Note: including those nodes defined by the 'yang-data' extension.)

- RPCs and associated input(s) and output(s)

- actions and associated input(s) and output(s)

- notifications and associated information

- YANG modules, submodules and features

It is possible that some protocols use only a subset of the assigned SIDs, for example, for protocols equivalent to NETCONF [RFC6241] like [I-D.ietf-core-comi] the transportation of YANG modules SIDs might be unnecessary. Others protocols might need to be able to transport this information, for example protocols related to discovery such as Constrained YANG Module Library [I-D.ietf-core-yang-library].

SIDs are globally unique integers, a registration system is used in order to guarantee their uniqueness. SIDs are registered in blocks called "SID ranges".

Assignment of SIDs to YANG items can be automated. For more details how this could be achieved, please consult Appendix B.

SIDs are assigned permanently, items introduced by a new revision of a YANG module are added to the list of SIDs already assigned. If the meaning of an item changes, for example as a result from a non-

backward compatible update of the YANG module, a new SID should be assigned to it.

Section 3 provides more details about the registration process of YANG modules and associated SIDs. To enable the implementation of this registry, Section 4 defines a standard file format used to store and publish SIDs.

2. Terminology and Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are defined in [RFC7950]:

- action
- feature
- module
- notification
- RPC
- schema node
- schema tree
- submodule

The following term is defined in [RFC8040]:

- yang-data extension

This specification also makes use of the following terminology:

- item: A schema node, an identity, a module, a submodule or a feature defined using the YANG modeling language.
- schema-node path: A schema-node path is a string that identifies a schema node within the schema tree. A path consists of the list of consecutive schema node identifier(s) separated by slashes ("/"). Schema node identifier(s) are always listed from the top-level schema node up to the targeted schema node and could contain

namespace information. (e.g. "/ietf-system:system-state/clock/current-datetime")

- o Namespace-qualified form - a schema node identifier is prefixed with the name of the module in which the schema node is defined, separated from the schema node identifier by the colon character (":").

- o YANG Schema Item iDentifier (YANG SID or simply SID): Unsigned integer used to identify different YANG items.

3. ".sid" file lifecycle

YANG is a language designed to model data accessed using one of the compatible protocols (e.g. NETCONF [RFC6241], RESTCONF [RFC8040] and CORECONF [I-D.ietf-core-comi]). A YANG module defines hierarchies of data, including configuration, state data, RPCs, actions and notifications.

Many YANG modules are not created in the context of constrained applications. YANG modules can be implemented using NETCONF [RFC6241] or RESTCONF [RFC8040] without the need to assign SIDs.

As needed, authors of YANG modules can assign SIDs to their YANG modules. In order to do that, they should first obtain a SID range from a registry and use that range to assign or generate SIDs to items of their YANG module. The assignments can then be stored in a ".sid" file. For example on how this could be achieved, please refer to Appendix C.

Registration of the ".sid" file associated to a YANG module is optional but recommended to promote interoperability between devices and to avoid duplicate allocation of SIDs to a single YANG module. Different registries might have different requirements for the registration and publication of the ".sid" files. For a diagram of one of the possibilities, please refer to the activity diagram on Figure 1 in Appendix C.

Each time a YANG module or one of its imported module(s) or included sub-module(s) is updated, a new ".sid" file MAY need to be created. All the SIDs present in the previous version of the ".sid" file MUST be present in the new version as well. The creation of this new version of the ".sid" file SHOULD be performed using an automated tool.

If a new revision requires more SIDs than initially allocated, a new SID range MUST be added to the 'assignment-ranges' as defined in Section 4. These extra SIDs are used for subsequent assignments.

For an example of this update process, see activity diagram Figure 2 in Appendix C.

4. ".sid" file format

".sid" files are used to persist and publish SIDs assigned to the different YANG items of a specific YANG module. It has the following structure.

```
module: ietf-sid-file
  +--rw module-name?              yang:yang-identifier
  +--rw module-revision?          revision-identifier
  +--rw sid-file-version?         sid-file-version-identifier
  +--rw description?              string
  +--rw dependency-revision* [module-name]
  |  +--rw module-name        yang:yang-identifier
  |  +--rw module-revision    revision-identifier
  +--rw assigment-ranges* [entry-point]
  |  +--rw entry-point    sid
  |  +--rw size           uint64
  +--rw items* [namespace identifier]
     +--rw namespace     enumeration
     +--rw identifier    union
     +--rw sid           sid
```

The following YANG module defined the structure of this file, encoding is performed using the rules defined in [RFC7951]. It references ietf-yang-types defined in [RFC6991] and ietf-restconf defined in [RFC8040].

RFC Ed.: please update the date of the module and Copyright if needed and remove this note.

```
<CODE BEGINS> file "ietf-sid-file@2020-02-05.yang"
module ietf-sid-file {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-sid-file";
  prefix sid;

  import ietf-yang-types {
    prefix yang;
    reference "RFC 6991: Common YANG Data Types.";
  }
  import ietf-restconf {
    prefix rc;
    reference "RFC 8040: RESTCONF Protocol.";
  }
```

```
  organization
    "IETF Core Working Group";

  contact
    "WG Web:   <http://datatracker.ietf.org/wg/core/>

     WG List:  <mailto:core@ietf.org>

     Editor:   Michel Veillette
               <mailto:michel.veillette@trilliant.com>

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

     Editor:   Alexander Pelov
               <mailto:a@ackl.io>

     Editor:   Ivaylo Petrov
               <mailto:ivaylo@ackl.io>";

  description
    "Copyright (c) 2021 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
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     for full legal notices.

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
     NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
     'MAY', and 'OPTIONAL' in this document are to be interpreted as
     described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
     they appear in all capitals, as shown here.

     This module defines the structure of the .sid files.

     Each .sid file contains the mapping between the different
     string identifiers defined by a YANG module and a
     corresponding numeric value called YANG SID.";

  revision 2020-02-05 {
```

```
    description
      "Initial revision.";
    reference
      "[I-D.ietf-core-sid] YANG Schema Item iDentifier (YANG SID)";
  }

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

  typedef sid-file-version-identifier {
    type uint64;
    description
      "Represents the version of a .sid file.";
  }

  typedef sid {
    type uint64 {
      range "0..9223372036854775807";
    }
    description
      "YANG Schema Item iDentifier";
    reference
      "[I-D.ietf-core-sid] YANG Schema Item iDentifier (YANG SID)";
  }

  typedef schema-node-path {
    type string {
      pattern
        '/[a-zA-Z_][a-zA-Z0-9\-_.]*:[a-zA-Z_][a-zA-Z0-9\-_.]*' +
        '(/[a-zA-Z_][a-zA-Z0-9\-_.]*(:[a-zA-Z_][a-zA-Z0-9\-_.]*)?)*';
    }
    description
      "A schema-node path is an absolute YANG schema node identifier as
      defined by the YANG ABNF rule absolute-schema-nodeid, except
      that module names are used instead of prefixes.

      This string additionally follows the following rules:

       o  The leftmost (top-level) data node name is always in the
          namespace-qualified form.
       o  Any subsequent schema node name is in the namespace-qualified form
          if the node is defined in a module other than its parent node, and
          the simple form is used otherwise. No predicates are allowed.";
```

```
    reference
      "RFC 7950, The YANG 1.1 Data Modeling Language;
       Section 6.5: Schema Node Identifier;";
  }

  rc:yang-data sid-file {
      uses sid-file;
  }

  grouping sid-file {
    description "A grouping that contains a YANG container representing the
      file structure of a .sid files.";

    container sid-file {
      description
        "A Wrapper container that together with the rc:yang-data extension
         marks the YANG data structures inside as not being intended to be
         implemented as part of a configuration datastore or as an operational
         state within the server.";
      leaf module-name {
        type yang:yang-identifier;
        description
          "Name of the YANG module associated with this .sid file.";
      }

      leaf module-revision {
        type revision-identifier;
        description
          "Revision of the YANG module associated with this .sid file.
          This leaf is not present if no revision statement is
          defined in the YANG module.";
      }

      leaf sid-file-version {
        type sid-file-version-identifier;
        description
          "Optional leaf that specifies the version number of the .sid file.
          .sid files and the version sequence are specific to a given YANG
          module revision. This number starts at zero when there is a new YANG
          module revision and increases monotonically.  This number can
          distinguish updates to the .sid file which are the result of new
          processing, or the result of reported errata.";
      }

      leaf description {
        type string;
        description
          "Free-form meta information about the generated file. It might
```

```
          include .sid file generation tool and time among other things.";
      }

      list dependency-revision {
        key "module-name";

        description
          "Information about the used revision during the .sid file generation
          of each YANG module that the module in 'module-name' imported.";

        leaf module-name {
          type yang:yang-identifier;
          mandatory true;
          description
            "Name of the YANG module, dependency of 'module-name', for which
            revision information is provided.";
        }
        leaf module-revision {
          type revision-identifier;
          mandatory true;
          description
            "Revision of the YANG module, dependency of 'module-name', for
which
            revision information is provided.";
        }
      }

      list assigment-ranges {
        key "entry-point";
        description
          "YANG SID range(s) allocated to the YANG module identified by
          'module-name' and 'module-revision'.

          - The YANG SID range first available value is entry-point and the the
            last available value in the range is (entry-point + size - 1).
          - The YANG SID ranges specified by all assignment-rages MUST NOT
            overlap.";

        leaf entry-point {
          type sid;
          mandatory true;
          description
            "Lowest YANG SID available for assignment.";
        }

        leaf size {
          type uint64;
          mandatory true;
```

description

```
            "Number of YANG SIDs available for assignment.";
        }
      }

      list items {
        key "namespace identifier";
        description
          "Each entry within this list defined the mapping between
          a YANG item string identifier and a YANG SID. This list MUST
          include a mapping entry for each YANG item defined by
          the YANG module identified by 'module-name' and
          'module-revision'.";

        leaf namespace {
          type enumeration {
            enum module {
              value 0;
              description
                "All module and submodule names share the same
                global module identifier namespace.";
            }
            enum identity {
              value 1;
              description
                "All identity names defined in a module and its
                submodules share the same identity identifier
                namespace.";
            }
            enum feature {
              value 2;
              description
                "All feature names defined in a module and its
                submodules share the same feature identifier
                namespace.";
            }
            enum data {
              value 3;
              description
                "The namespace for all data nodes, as defined in YANG.";
            }
          }
          description
            "Namespace of the YANG item for this mapping entry.";
        }

        leaf identifier {
          type union {
            type yang:yang-identifier;
```

```
            type schema-node-path;
          }
          description
            "String identifier of the YANG item for this mapping entry.

            If the corresponding 'namespace' field is 'module',
            'feature', or 'identity', then this field MUST
            contain a valid YANG identifier string.

            If the corresponding 'namespace' field is 'data',
            then this field MUST contain a valid schema node
            path.";
        }

        leaf sid {
          type sid;
          mandatory true;
          description
            "YANG SID assigned to the YANG item for this mapping entry.";
        }
      }
    }
  }
}
<CODE ENDS>
```

5. Content-Types

The following Content-Type is defined:

application/yang-data+cbor; id=sid: This Content-Type represents a CBOR YANG document containing one or multiple data node values. Each data node is identified by its associated SID.

FORMAT: CBOR map of SID, instance-value

The message payload of Content-Type 'application/yang-data+cbor' is encoded using a CBOR map. Each entry within the CBOR map contains the data node identifier (i.e. SID) and the associated instance-value. Instance-values are encoded using the rules defined in [I-D.ietf-core-yang-cbor] section 4.

6. Security Considerations

This document defines a new type of identifier used to encode data models defined in YANG [RFC7950]. As such, this identifier does not contribute to any new security issues in addition of those identified for the specific protocols or contexts for which it is used.

7. IANA Considerations

7.1. YANG Namespace Registration

This document registers the following XML namespace URN in the "IETF XML Registry", following the format defined in [RFC3688]:

URI: please assign urn:ietf:params:xml:ns:yang:ietf-sid-file

Registrant Contact: The IESG.

XML: N/A, the requested URI is an XML namespace.

Reference: RFC XXXX

// RFC Ed.: please replace XXXX with RFC number and remove this note

7.2. Register ".sid" File Format Module

This document registers one YANG module in the "YANG Module Names" registry [RFC6020]:

- o name: ietf-sid-file
- o namespace: urn:ietf:params:xml:ns:yang:ietf-sid-file
- o prefix: sid
- o reference: [[THISRFC]]

7.3. CoAP Content-Formats Registry

This document adds the following Content-Format to the "CoAP Content-Formats", within the "Constrained RESTful Environments (CoRE) Parameters" registry.

Media Type	Content Coding	ID	Reference
application/yang-data+cbor; id=sid		TBD1	RFC XXXX

// RFC Ed.: replace TBD1 with assigned IDs and remove this note. // RFC Ed.: replace RFC XXXX with this RFC number and remove this note.

7.4. Create new IANA Registry: "YANG SID Mega-Range" registry

The name of this registry is "YANG SID Mega-Range". This registry is used to record the delegation of the management of a block of SIDs to third parties (such as SDOs or registrars).

7.4.1. Structure

Each entry in this registry must include:

- o The entry point (first SID) of the registered SID block.
- o The size of the registered SID block. The size MUST be one million (1 000 000) SIDs.
- o The contact information of the requesting organization including:
  - * The policy of SID range allocations: Public, Private or Both.
  - * Organization name
  - * URL

The information associated to the Organization name should not be publicly visible in the registry, but should be available. This information includes contact email and phone number and change controller email and phone number.

7.4.2. Allocation policy

The IANA policy for future additions to this registry is "Expert Review" [RFC8126].

An organization requesting to manage a YANG SID Range (and thus have an entry in the YANG SID Mega-Range Registry), must ensure the following capacities:

- o The capacity to manage and operate a YANG SID Range Registry. A YANG SID Range Registry MUST provide the following information for all YANG SID Ranges allocated by the Registry:
  - * Entry Point of allocated YANG SID Range
  - * Size of allocated YANG SID Range
  - * Type: Public or Private

+ Public Ranges MUST include at least a reference to the YANG module and ".sid" files for that YANG SID Range.

+ Private Ranges MUST be marked as "Private"

o A Policy of allocation, which clearly identifies if the YANG SID Range allocations would be Private, Public or Both.

o Technical capacity to ensure the sustained operation of the registry for a period of at least 5 years. If Private Registrations are allowed, the period must be of at least 10 years.

7.4.2.1. First allocation

For a first allocation to be provided, the requesting organization must demonstrate a functional registry infrastructure.

7.4.2.2. Consecutive allocations

On subsequent allocation request(s), the organization must demonstrate the exhaustion of the prior range. These conditions need to be asserted by the assigned expert(s).

If that extra-allocation is done within 3 years from the last allocation, the experts need to discuss this request on the CORE working group mailing list and consensus needs to be obtained before allocating a new Mega-Range.

7.4.3. Initial contents of the Registry

The initial entry in this registry is allocated to IANA:

Entry Point	Size	Allocation	Organization name	URL
0	1000000	Public	IANA	iana.org

7.5. Create a new IANA Registry: IETF YANG SID Range Registry (managed by IANA)

7.5.1. Structure

Each entry in this registry must include:

o The SID range entry point.

- o The SID range size.
- o The YANG module name.
- o Document reference.

7.5.2. Allocation policy

The first million SIDs assigned to IANA is sub-divided as follows:

- o The range of 0 to 999 (size 1000) is "Reserved" as defined in [RFC8126].
- o The range of 1000 to 59,999 (size 59,000) is reserved for YANG modules defined in RFCs.
  - * The IANA policy for additions to this registry is either:
    - + "Expert Review" [RFC8126] in case the ".sid" file comes from a YANG module from an existing RFC, or
    - + "RFC Required" [RFC8126] otherwise.
  - * The Expert MUST verify that the YANG module for which this allocation is made has an RFC (existing RFC) OR is on track to become RFC (early allocation with a request from the WG chairs as defined by [RFC7120]).
- o The range of 60,000 to 99,999 (size 40,000) is reserved for experimental YANG modules. This range MUST NOT be used in operational deployments since these SIDs are not globally unique which limit their interoperability. The IANA policy for this range is "Experimental use" [RFC8126].
- o The range of 100,000 to 999,999 (size 900,000) is "Reserved" as defined in [RFC8126].

Entry Point	Size	IANA policy
0	1,000	Reserved
1,000	59,000	Expert Review or RFC Required
60,000	40,000	Experimental use
100,000	900,000	Reserved

The size of the SID range allocated for a YANG module is recommended to be a multiple of 50 and to be at least 33% above the current

number of YANG items. This headroom allows assignment within the same range of new YANG items introduced by subsequent revisions. The maximum SID range size is 1000. A larger size may be requested by the authors if this recommendation is considered insufficient. It is important to note that an additional SID range can be allocated to an existing YANG module if the initial range is exhausted.

In case a SID range is allocated for an existing RFC through the "Expert Review" policy, the Document reference field for the given allocation should point to the RFC that the YANG module is defined in.

In case a SID range is required before publishing the RFC due to implementations needing stable SID values, early allocation as defined in [RFC7120] can be employed. As specified in section 4.6 of [RFC8126], RFCs and by extension documents that are expected to become an RFC fulfill the requirement for "Specification Required" stated in section 2 of [RFC7120], which allows for the early allocation process to be employed.

7.5.3. Initial contents of the registry

Initial entries in this registry are as follows:

Entry Point	Size	Module name	Document reference
1000	100	ietf-coreconf	[I-D.ietf-core-comi]
1100	50	ietf-yang-types	[RFC6991]
1150	50	ietf-inet-types	[RFC6991]
1200	50	iana-crypt-hash	[RFC7317]
1250	50	ietf-netconf-acm	[RFC8341]
1300	50	ietf-sid-file	RFCXXXX
1500	100	ietf-interfaces	[RFC8343]
1600	100	ietf-ip	[RFC8344]
1700	100	ietf-system	[RFC7317]
1800	400	iana-if-type	[RFC7224]
2400	50	ietf-voucher	[RFC8366]
2450	50	ietf-constrained-voucher	[I-D.ietf-anima-constrained-voucher]
2500	50	ietf-constrained-voucher-request	[I-D.ietf-anima-constrained-voucher]

// RFC Ed.: replace XXXX with RFC number assigned to this draft.

For allocation, RFC publication of the YANG module is required as per [RFC8126]. The YANG module must be registered in the "YANG module Name" registry according to the rules specified in section 14 of [RFC6020].

7.6. Create new IANA Registry: "IETF YANG SID Registry"

The name of this registry is "IETF YANG SID Registry". This registry is used to record the allocation of SIDs for individual YANG module items.

7.6.1. Structure

Each entry in this registry must include:

- o The YANG module name. This module name must be present in the "Name" column of the "YANG Module Names" registry.

- o A link to the associated ".yang" file. This file link must be present in the "File" column of the "YANG Module Names" registry.

- o The link to the ".sid" file which defines the allocation. The ".sid" file is stored by IANA.

- o The number of actually allocated SIDs in the ".sid" file.

7.6.2. Allocation policy

The allocation policy is Expert review. The Expert MUST ensure that the following conditions are met:

- o The ".sid" file has a valid structure:

  - * The ".sid" file MUST be a valid JSON file following the structure of the module defined in RFCXXXX (RFC Ed: replace XXX with RFC number assigned to this draft).

- o The ".sid" file allocates individual SIDs ONLY in the YANG SID Ranges for this YANG module (as allocated in the IETF YANG SID Range Registry):

  - * All SIDs in this ".sid" file MUST be within the ranges allocated to this YANG module in the "IETF YANG SID Range Registry".

- o If another ".sid" file has already allocated SIDs for this YANG module (e.g. for older or newer versions of the YANG module), the YANG items are assigned the same SIDs as in the other ".sid" file.

- o If there is an older version of the ".sid" file, all allocated SIDs from that version are still present in the current version of the ".sid" file.

7.6.3. Recursive Allocation of YANG SID Range at Document Adoption

Due to the difficulty in changing SID values during IETF document processing, it is expected that most documents will ask for SID allocations using Early Allocations [RFC7120]. The details of the Early Allocation should be included in any Working Group Adoption call. Prior to Working Group Adoption, an internet draft authors can use the experimental SID range (as per Section 7.5.2) for their SIDs allocations or other values that do not create ambiguity with other SID uses (for example they can use a range that comes from a non-IANA managed "YANG SID Mega-Range" registry).

After Working Group Adoption, any modification of a ".sid" file is expected to be discussed on the mailing list of the appropriate Working Groups. Specific attention should be paid to implementers' opinion after Working Group Last Call if a SID value is to change its meaning. In all cases, a ".sid" file and the SIDs associated with it are subject to change before the publication of an internet draft as an RFC.

During the early use of SIDs, many existing, previously published YANG modules will not have SID allocations. For an allocation to be useful the included YANG modules may also need to have SID allocations made.

The Expert Reviewer who performs the (Early) Allocation analysis will need to go through the list of included YANG modules and perform SID allocations for those modules as well.

- o If the document is a published RFC, then the allocation of SIDs for its referenced YANG modules is permanent. The Expert Reviewer provides the generated SID file to IANA for registration. This process may be time consuming during a bootstrap period (there are over 100 YANG modules to date, none of which have SID allocations), but should quiet down once needed entries are allocated.

- o If the document is an unprocessed Internet-Draft adopted in a WG, then an Early Allocation is performed for this document as well. Early Allocations require approval by an IESG Area Director. An early allocation which requires additional allocations will list the other allocations in it's description, and will be cross-posted to the any other working group mailing lists.

- o A YANG module which references a module in an document which has not yet been adopted by any working group will be unable to perform an Early Allocation for that other document until it is adopted by a working group. As described in [RFC7120], an AD Sponsored document acts as if it had a working group. The approving AD may also exempt a document from this policy by agreeing to AD Sponsor the document.

At the end of the IETF process all the dependencies of a given module for which SIDs are assigned, should also have SIDs assigned. Those dependencies' assignments should be permanent (not Early Allocation).

A previously SID-allocated YANG module which changes its references to include a YANG module for which there is no SID allocation needs to repeat the Early Allocation process.

Early Allocations are made with a one-year period, after which they are expired. [RFC7120] indicates that at most one renewal may be made. For the SID allocation a far more lenient stance is desired.

1. An extension of a referencing documents Early Allocation should update any referenced Early Allocations to expire no sooner than the referencing document.

2. The [RFC7120] mechanism allows the IESG to provide a second renewal, and such an event may prompt some thought about how the collection of documents are being processed.

This is driven by the very generous size of the SID space and the often complex and deep dependencies of YANG modules. Often a core module with many dependencies will undergo extensive review, delaying the publication of other documents.

[RFC7120] also says:

Note that if a document is submitted for review to the IESG and at the time of submission some early allocations are valid (not expired), these allocations should not be expired while the document is under IESG consideration or waiting in the RFC Editor's queue after approval by the IESG.

7.6.4. Initial contents of the registry

None.

8. Acknowledgments

The authors would like to thank Andy Bierman, Carsten Bormann, Michael Richardson, Abhinav Somaraju, Peter van der Stok, Laurent Toutain and Randy Turner for their help during the development of this document and their useful comments during the review process.

9. References

9.1. Normative References

[I-D.ietf-core-yang-cbor]
Veillette, M., Petrov, I., and A. Pelov, "CBOR Encoding of Data Modeled with YANG", draft-ietf-core-yang-cbor-13 (work in progress), July 2020.

[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>.

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

[RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <https://www.rfc-editor.org/info/rfc7120>.

[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>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

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

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

9.2. Informative References

[I-D.ietf-anima-constrained-voucher]
Richardson, M., Stok, P., and P. Kampanakis, "Constrained Voucher Artifacts for Bootstrapping Protocols", draft-ietf-anima-constrained-voucher-09 (work in progress), November 2020.

[I-D.ietf-core-comi]
Veillette, M., Stok, P., Pelov, A., Bierman, A., and I. Petrov, "CoAP Management Interface (CORECONF)", draft-ietf-core-comi-10 (work in progress), July 2020.

[I-D.ietf-core-yang-library]
Veillette, M. and I. Petrov, "Constrained YANG Module Library", draft-ietf-core-yang-library-03 (work in progress), January 2021.

[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>.

[RFC7224] Bjorklund, M., "IANA Interface Type YANG Module", RFC 7224, DOI 10.17487/RFC7224, May 2014, <https://www.rfc-editor.org/info/rfc7224>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

[RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <https://www.rfc-editor.org/info/rfc8366>.

Appendix A. ".sid" file example

The following ".sid" file (ietf-system@2014-08-06.sid) have been generated using the following yang modules:

- ietf-system@2014-08-06.yang (defined in [RFC7317])
- ietf-yang-types@2013-07-15.yang (defined in [RFC6991])
- ietf-inet-types@2013-07-15.yang (defined in [RFC6991])
- ietf-netconf-acm@2018-02-14.yang (defined in [RFC8341])
- iana-crypt-hash@2014-04-04.yang (defined in [RFC7317])


```
{
  "ietf-sid-file:sid-file" : {
    "module-name": "ietf-system",
    "module-revision": "2020-02-05",
    "dependency-revision": [
      {
        "module-name": "ietf-yang-types",
        "module-revision": "2013-07-15.yang"
      },
      {
        "module-name": "ietf-inet-types",
        "module-revision": "2013-07-15.yang"
      },
      {
        "module-name": "ietf-netconf-acm",
        "module-revision": "2018-02-14.yang"
      },
      {
        "module-name": "iana-crypt-hash",
        "module-revision": "2014-04-04.yang"
```

```
        }
      ],
      "description": "Example sid file",
      "assignment-ranges": [
        {
          "entry-point": 1700,
          "size": 100
        }
      ],
      "items": [
        {
          "namespace": "module",
          "identifier": "ietf-system",
          "sid": 1700
        },
        {
          "namespace": "identity",
          "identifier": "authentication-method",
          "sid": 1701
        },
        {
          "namespace": "identity",
          "identifier": "local-users",
          "sid": 1702
        },
        {
          "namespace": "identity",
          "identifier": "radius",
          "sid": 1703
        },
        {
          "namespace": "identity",
          "identifier": "radius-authentication-type",
          "sid": 1704
        },
        {
          "namespace": "identity",
          "identifier": "radius-chap",
          "sid": 1705
        },
        {
          "namespace": "identity",
          "identifier": "radius-pap",
          "sid": 1706
        },
        {
          "namespace": "feature",
          "identifier": "authentication",
```

```
          "sid": 1707
        },
        {
          "namespace": "feature",
          "identifier": "dns-udp-tcp-port",
          "sid": 1708
        },
        {
          "namespace": "feature",
          "identifier": "local-users",
          "sid": 1709
        },
        {
          "namespace": "feature",
          "identifier": "ntp",
          "sid": 1710
        },
        {
          "namespace": "feature",
          "identifier": "ntp-udp-port",
          "sid": 1711
        },
        {
          "namespace": "feature",
          "identifier": "radius",
          "sid": 1712
        },
        {
          "namespace": "feature",
          "identifier": "radius-authentication",
          "sid": 1713
        },
        {
          "namespace": "feature",
          "identifier": "timezone-name",
          "sid": 1714
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:set-current-datetime",
          "sid": 1715
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:set-current-datetime/
                         current-datetime",
          "sid": 1716
        },
```

```
        {
          "namespace": "data",
          "identifier": "/ietf-system:system",
          "sid": 1717
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-restart",
          "sid": 1718
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-shutdown",
          "sid": 1719
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state",
          "sid": 1720
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/clock",
          "sid": 1721
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/clock/boot-datetime",
          "sid": 1722
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/clock/
                         current-datetime",
          "sid": 1723
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/platform",
          "sid": 1724
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/platform/machine",
          "sid": 1725
        },
        {
          "namespace": "data",
```

```
          "identifier": "/ietf-system:system-state/platform/os-name",
          "sid": 1726
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/platform/os-release",
          "sid": 1727
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system-state/platform/os-version",
          "sid": 1728
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication",
          "sid": 1729
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user",
          "sid": 1730
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/
                         user-authentication-order",
          "sid": 1731
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         authorized-key",
          "sid": 1732
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         authorized-key/algorithm",
          "sid": 1733
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         authorized-key/key-data",
          "sid": 1734
        },
        {
```

```
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         authorized-key/name",
          "sid": 1735
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         name",
          "sid": 1736
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/authentication/user/
                         password",
          "sid": 1737
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/clock",
          "sid": 1738
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/clock/timezone-name",
          "sid": 1739
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/clock/timezone-utc-offset",
          "sid": 1740
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/contact",
          "sid": 1741
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver",
          "sid": 1742
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/options",
          "sid": 1743
        },
        {
```

```
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/options/
                         attempts",
          "sid": 1744
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/options/
                         timeout",
          "sid": 1745
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/search",
          "sid": 1746
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/server",
          "sid": 1747
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/server/name",
          "sid": 1748
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/server/
                         udp-and-tcp",
          "sid": 1749
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/server/
                         udp-and-tcp/address",
          "sid": 1750
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/dns-resolver/server/
                         udp-and-tcp/port",
          "sid": 1751
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/hostname",
          "sid": 1752
```

```
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/location",
          "sid": 1753
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp",
          "sid": 1754
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/enabled",
          "sid": 1755
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server",
          "sid": 1756
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/
                         association-type",
          "sid": 1757
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/iburst",
          "sid": 1758
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/name",
          "sid": 1759
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/prefer",
          "sid": 1760
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/udp",
          "sid": 1761
        },
        {
```

```
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/udp/address",
          "sid": 1762
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/ntp/server/udp/port",
          "sid": 1763
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius",
          "sid": 1764
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/options",
          "sid": 1765
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/options/attempts",
          "sid": 1766
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/options/timeout",
          "sid": 1767
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server",
          "sid": 1768
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/
                         authentication-type",
          "sid": 1769
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/name",
          "sid": 1770
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/udp",
```

```
          "sid": 1771
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/udp/
                         address",
          "sid": 1772
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/udp/
                         authentication-port",
          "sid": 1773
        },
        {
          "namespace": "data",
          "identifier": "/ietf-system:system/radius/server/udp/
                         shared-secret",
          "sid": 1774
        }
      ]
    }
  }
```


Appendix B. SID auto generation

Assignment of SIDs to YANG items can be automated, the recommended process to assign SIDs is as follows:

1. A tool extracts the different items defined for a specific YANG module.

2. The list of items is sorted in alphabetical order, 'namespace' in descending order, 'identifier' in ascending order. The 'namespace' and 'identifier' formats are described in the YANG module 'ietf-sid-file' defined in Section 4.

3. SIDs are assigned sequentially from the entry point up to the size of the registered SID range. This approach is recommended to minimize the serialization overhead, especially when delta between a reference SID and the current SID is used by protocols aiming to reduce message size.

4. If the number of items exceeds the SID range(s) allocated to a YANG module, an extra range is added for subsequent assignments.

5. The "dependency-revision" should reflect the revision numbers of each YANG module that the YANG module imports at the moment of the generation.

In case of an update to an existing ".sid" file, an additional step is needed that increments the ".sid" file version number. If there was no version number in the previous version of the ".sid" file, 0 is assumed as the version number of the old version of the ".sid" file and the version number is 1 for the new ".sid" file. Apart from that, changes of ".sid" files can also be automated using the same method described above, only unassigned YANG items are processed at step #3. Already existing items in the ".sid" file should not be given new SIDs.

Note that ".sid" file versions are specific to a YANG module revision. For each new YANG module or each new revision of an existing YANG module, the version number of the initial ".sid" file should either be 0 or should not be present.

Note also that RPC or action "input" and "output" data nodes MUST always be assigned SID even if they don't contain data nodes. The reason for this requirement is that other modules can augment the given module and those SIDs might be necessary.

Appendix C. ".sid" file lifecycle

Before assigning SIDs to their YANG modules, YANG module authors must acquire a SID range from a "YANG SID Range Registry". If the YANG module is part of an IETF draft or RFC, the SID range need to be acquired from the "IETF YANG SID Range Registry" as defined in Section 7.5. For the other YANG modules, the authors can acquire a SID range from any "YANG SID Range Registry" of their choice.

Once the SID range is acquired, the owner can use it to generate ".sid" file/s for his YANG module/s. It is recommended to leave some unallocated SIDs following the allocated range in each ".sid" file in order to allow better evolution of the YANG module in the future. Generation of ".sid" files should be performed using an automated tool. Note that ".sid" files can only be generated for YANG modules and not for submodules.

C.1. ".sid" File Creation

The following activity diagram summarizes the creation of a YANG module and its associated ".sid" file.

```
      +---------------+
  O   | Creation of a |
```

```
 -|- ->| YANG module   |
 / \   +---------------+
               |
               V
        /-------------\
       / Standardized  \     yes
       \ YANG module ? /-------------+
        \-------------/              |
               | no                  |
               V                     V
        /-------------\      +---------------+
       / Constrained   \ yes | SID range     |
   +-->\ application ? /---->| registration  |<----------+
   |    \-------------/      +---------------+           |
   |           | no                  |                   |
   |           V                     V                   |
   |   +---------------+     +---------------+           |
   +---| YANG module   |     | SID sub-range |           |
       | update        |     | assignment    |<----------+
       +---------------+     +---------------+           |
                                     |                   |
                                     V                   |
                             +---------------+    +-------------+
                             | ".sid" file   |    | Rework YANG |
                             | generation    |    |    model    |
                             +---------------+    +-------------+
                                     |                   ^
                                     V                   |
                               /----------\  yes        |
                              /  Work in   \ -----------+
                              \  progress  /
                               \----------/
                                     | no
                                     V
                              /-------------\       /-------------\
                             /      RFC      \ no  /     Open      \ no
                             \  publication? /---->\ specification?/---+
                              \-------------/       \-------------/    |
                                     | yes                 | yes       |
                                     |     +---------------+           |
                                     V     V                           V
                             +---------------+
+---------------+
                             |     IANA      |                 | Third party
|
                             | registration  |                 | registration
|
                             +-------+-------+                 +-------
```

```
+-------+
                                        |                                 |
                                        +---------------------------------+
                                        V
```

[DONE]

Figure 1: SID Lifecycle

C.2. ".sid" File Update

The following Activity diagram summarizes the update of a YANG module and its associated ".sid" file.

```
          +---------------+
     O    | Update of the |
    -|- ->| YANG module   |
    / \   | or include(s) |
          | or import(s)  |
          +---------------+
                  |
                  V
              /-------------\
             /  New items    \ yes
             \  created  ?   /------+
              \-------------/       |
                     | no           V
                     |       /-------------\      +----------------+
                     |      /  SID range    \ yes | Extra sub-range|
                     |      \  exhausted ?  /---->| assignment     |
                     |       \-------------/      +----------------+
                     |              | no                  |
                     |              +---------------------+
                     |              |
                     |              V
                     |      +---------------+
                     |      | ".sid" file   |
                     |      | update based  |
                     |      | on previous   |
                     |      | ".sid" file   |
                     |      +---------------+
                     |              |
                     |              V
                     |       /-------------\      +---------------+
                     |      /  Publicly     \ yes | YANG module   |
                     |      \  available ?  /---->| registration  |
                     |       \-------------/      +---------------+
                     |              | no                  |
                     +--------------+---------------------+
                                    |
                                 [DONE]
```

Figure 2: YANG and ".sid" file update

Authors' Addresses

Michel Veillette (editor)
Trilliant Networks Inc.
610 Rue du Luxembourg
Granby, Quebec J2J 2V2
Canada

Phone: +14503750556
Email: michel.veillette@trilliant.com

Alexander Pelov (editor)
Acklio
1137A avenue des Champs Blancs
Cesson-Sevigne, Bretagne 35510
France

Email: a@ackl.io

Ivaylo Petrov (editor)
Acklio
1137A avenue des Champs Blancs
Cesson-Sevigne, Bretagne 35510
France

Email: ivaylo@ackl.io