NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

H. Birkholz
Fraunhofer SIT

October 30, 2017

YANG module for Software Identifiers draft-birkholz-yang-swid-00

Abstract

This document provides a YANG module definition that enables a system entity to provide detailed information about installed software components. The structure of the module is based on the Concise Software Identifier draft and therefore also strongly related to the ISO 19770-2:2015 Software Identifiers standard. Both standards provide no interface to transport the SWID tag information between system entities and this document leverages the wide adoption of YANG based management interfaces.

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 May 3, 2018.

Copyright Notice

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

Software Identification Tags (SWID tags [SWID]) or their binary equivalent - the CoSWID tags [I-D.ietf-sacm-coswid] - provide a versatile document standard that can be installed in conjunction with a software component on a system entity. There is no standard interface to access, export, or transfer these tag document between system entities. The following YANG module enables full, fine-grained access to every attribute and metadata defined in the SWID standards via a YANG-based management interface.

2. YANG SWID module

Every node defined is read-only, as there is no installation or deployment capability associated with tag information. The descriptions of each attribute are derived from the SWID XML schema definition provided by ISO:

http://standards.iso.org/iso/19770/-2/2015-current/schema.xsd

The definitions were adapted and modified if appropriate.

The following summary illustrates the module in tree format. The complete YANG module can be found in Appendix A.

```
<CODE BEGINS>
module: yang-software-identity
    +--ro concise-software-identities*
       +--ro concise-software-identity
          +--ro lang?                              string
          +--ro any-element*
          |  +--ro any-attribute
```

```
|     +--ro attribute-name?    string
|     +--ro attribute-value?   anydata
+--ro tag-id                            string
+--ro swid-name                         string
+--ro (major-ressource-collection)?
|  +--:(payload)
|  |  +--ro payload
|  |     +--ro lang?          string
|  |     +--ro any-element*
|  |     |  +--ro any-attribute
|  |     |     +--ro attribute-name?    string
|  |     |     +--ro attribute-value?   anydata
|  |     +--ro (item-type)?
|  |        +--:(directory)
|  |        |  +--ro directory
|  |        |     +--ro directory-path?   string
|  |        +--:(file)
|  |        |  +--ro file
|  |        |     +--ro directory-path?   string
|  |        |     +--ro size?             uint32
|  |        |     +--ro file-version?     string
|  |        |     +--ro file-hash
|  |        |        +--ro hash-algo?    int16
|  |        |        +--ro hash-value?   binary
|  |        +--:(key)
|  |        |  +--ro key?           boolean
|  |        +--:(location)
|  |        |  +--ro location?      string
|  |        +--:(fs-name)
|  |        |  +--ro fs-name        string
|  |        +--:(root)
|  |           +--ro root?         string
|  +--:(evidence)
|     +--ro evidence
|        +--ro lang?          string
|        +--ro any-element*
|        |  +--ro any-attribute
|        |     +--ro attribute-name?    string
|        |     +--ro attribute-value?   anydata
|        +--ro (item-type)?
|        |  +--:(directory)
|        |  |  +--ro directory
|        |  |     +--ro directory-path?   string
|        |  +--:(file)
|        |  |  +--ro file
|        |  |     +--ro directory-path?   string
|        |  |     +--ro size?             uint32
|        |  |     +--ro file-version?     string
```

```
|        |  |     +--ro file-hash
|        |  |        +--ro hash-algo?    int16
|        |  |        +--ro hash-value?   binary
|        |  +--:(key)
|        |  |  +--ro key?           boolean
|        |  +--:(location)
|        |  |  +--ro location?      string
|        |  +--:(fs-name)
|        |  |  +--ro fs-name        string
|        |  +--:(root)
|        |     +--ro root?          string
|        +--ro date?          string
|        +--ro device-id?     string
+--ro additional-resource-collection*
|  +--ro process
|  |  +--ro lang?           string
|  |  +--ro any-element*
|  |  |  +--ro any-attribute
|  |  |     +--ro attribute-name?    string
|  |  |     +--ro attribute-value?   anydata
|  |  +--ro process-name    string
|  |  +--ro pid?            uint16
|  +--ro resource
|  |  +--ro lang?          string
|  |  +--ro any-element*
|  |  |  +--ro any-attribute
|  |  |     +--ro attribute-name?    string
|  |  |     +--ro attribute-value?   anydata
|  |  +--ro type?          string
|  +--ro entity
|  |  +--ro lang?          string
|  |  +--ro any-element*
|  |  |  +--ro any-attribute
|  |  |     +--ro attribute-name?    string
|  |  |     +--ro attribute-value?   anydata
|  |  +--ro entity-name    string
|  |  +--ro reg-id?        string
|  |  +--ro role?          string
|  |  +--ro thumbprint
|  |     +--ro hash-algo?          int16
|  |     +--ro thumbprint-value?   binary
|  +--ro link
|  |  +--ro lang?          string
|  |  +--ro any-element*
|  |  |  +--ro any-attribute
|  |  |     +--ro attribute-name?    string
|  |  |     +--ro attribute-value?   anydata
|  |  +--ro artifact?      string
```

```
            |  |  +--ro href           string
            |  |  +--ro media?         string
            |  |  +--ro ownership?     string
            |  |  +--ro rel            string
            |  |  +--ro type?          string
            |  |  +--ro use?           string
            |  +--ro software-meta
            |     +--ro lang?                          string
            |     +--ro any-element*
            |     |  +--ro any-attribute
            |     |     +--ro attribute-name?    string
            |     |     +--ro attribute-value?   anydata
            |     +--ro activation-status?           string
            |     +--ro channel-type?                string
            |     +--ro colloquial-version?          string
            |     +--ro description?                 string
            |     +--ro edition?                     string
            |     +--ro entitlement-data-required?   boolean
            |     +--ro entitlement-key?             string
            |     +--ro generator?                   string
            |     +--ro persistent-id?               string
            |     +--ro product?                     string
            |     +--ro product-family?              string
            |     +--ro revision?                    string
            |     +--ro summary?                     string
            |     +--ro unspsc-code?                 string
            |     +--ro unspsc-version?              string
            +--ro corpus?                            boolean
            +--ro patch?                             boolean
            +--ro media?                             boolean
            +--ro supplemental?                      boolean
            +--ro tag-version?                       string
            +--ro software-version?                  string
            +--ro version-scheme?                    string
```

<CODE ENDS>

3. IANA considerations

This document includes no requests to IANA.

4. Security Considerations

This document includes no security considerations yet, but will act as an incubator to create them

5. Acknowledgements

Eric Voit

6. Change Log

First version -00

7. Normative References

[I-D.ietf-sacm-coswid]
Birkholz, H., Fitzgerald-McKay, J., Schmidt, C., and D. Waltermire, "Concise Software Identifiers", draft-ietf-sacm-coswid-02 (work in progress), July 2017.

[SWID] "Information technology - Software asset management - Part 2: Software identification tag'", ISO/IEC 19770-2:2015, October 2015.

Appendix A. Detailed YANG SWID module

```
<CODE BEGINS>
module yang-software-identity {

  namespace "urn:ietf:params:xml:ns:yang:swid";
  prefix "yang-swid";
  import ietf-yang-types { prefix "yang"; }
  organization
   "Fraunhofer SIT";
  contact
   "Henk Birkholz
    Fraunhofer Institute for Secure Information Technology
    Email: henk.birkholz@sit.fraunhofer.de";
  description
    "The YANG module to provide SWID tags via YANG modeled
     management interfaces.
     Copyright (C) Fraunhofer SIT (2017).";
  revision "2017-10-30" {
  description
    "Initial version";
  reference
    "draft-birkholz-yang-swid-00";
  }

  grouping global-attributes {
    leaf lang {
      type string;
      description
```

```
        "An RFC5646 conferment language tag";
    }
    list any-element {
      container any-attribute {
        leaf attribute-name {
          type string;
          description
            "The name of the custom attribute.";
        }
        leaf attribute-value {
          type anydata;
          description
            "The value of the custom attribute.";
        }
      }
    }
  }

  grouping relative-path {
    leaf directory-path {
      type string;
      description
        "A file-system path expression relative to the SWID tag document,";
    }
  }

  grouping filesystem-item {
    uses global-attributes;
    choice item-type {
      container directory {
        uses relative-path;
      }
      container file {
        uses relative-path;
        leaf size {
          type uint32;
          description
            "The file size in bytes of the file.";
        }
        leaf file-version {
          type string;
          description
            "The file version.";
        }
        container file-hash {
          leaf hash-algo {
            type int16;
            description
```

```
            "The integer index of the IANA Named Information Hash Algorithm
             Registry table";
          }
          leaf hash-value {
            type binary;
            description
              "The binary hash value of the file";
          }
        }
      }
      leaf key {
        type boolean;
        description
          "Files that are considered important or required
           for the use of a software component.  Typical key files
           would be those which, if not available on a system entity,
           would cause the software component not to execute or
           function properly.
           Key files will typically be used to validate that
           a software component referenced by the CoSWID tag document
           is actually installed on a specific system
           entity.";
      }
      leaf location {
        type string;
        description
          "The directory or location where a file was found
           or can expected to be located.  This text-string is intended
           to include the filename itself. This SHOULD be the relative
           path represented by the root item.";
      }
      leaf fs-name {
        type string;
        mandatory true;
        description
          "The file name or directory name without any path characters.";
      }
      leaf root {
        type string;
        description
          "A system-specific root folder that the location
           item is an offset from. If this is not specified the
           assumption is the root is the same folder as the location of
           the CoSWID tag. The text-string value represents a path
           expression relative to the CoSWID tag document location in
           the (composite) file-system hierarchy.";
      }
    }
```

```
  }

  list concise-software-identities {
    config false;
    container concise-software-identity {
      uses global-attributes;
      leaf tag-id {
        type string;
        mandatory true;
        description
          "An identifier uniquely referencing a (composite)
           software component. The tag identifier is intended to be
           globally unique. There are no strict guidelines on how this
           identifier is structured, but examples include a 16 byte
           GUID (e.g. class 4 UUID).";
      }
      leaf swid-name {
        type string;
        mandatory true;
        description
          "This item provides the software component name as
           it would typically be referenced.  For example, what would
           be seen in the add/remove dialog on a Windows device, or
           what is specified as the name of a packaged software product
           or a patch identifier name on a Linux device.";
      }
      choice major-ressource-collection {
        container payload {
          uses filesystem-item;
        }
        container evidence {
          uses filesystem-item;
          leaf date {
            type string;
            description
              "The sate and time evidence represented by an
               evidence item was gathered.";
          }
          leaf device-id {
            type string;
            description
              "A text-string identifier for a device
               evidence was gathered from.";
          }
        }
      }
      list additional-resource-collection {
        container process {
```

```
          uses global-attributes;
          leaf process-name {
            type string;
            mandatory true;
            description
              "The process name as it will be found in the
               system entity's process table.";
          }
          leaf pid {
            type uint16;
            description
              "The process ID for the process in execution
               that can be included in the process item as part of an
               evidence tag.";
          }
        }
        container resource {
          uses global-attributes;
          leaf type {
            type string;
            description
              "The type of resource represented via a
               text-string  (typically, registry-key, port
               or root-uri).";
          }
        }
        container entity {
          uses global-attributes;
          leaf entity-name {
            type string;
            mandatory true;
            description
              "The text-string name of the organization
               claiming a particular role in the SWID tag";
          }
          leaf reg-id {
            type string;
            description
              "The registration id is intended to uniquely
               identify a naming authority in a given scope (e.g. global,
               organization, vendor, customer, administrative domain, etc.)
               that is implied by the referenced naming authority. The
               value of an registration ID MUST be a RFC 3986 URI. The
               scope SHOULD be the scope of an organization. In a given
               scope, the registration id MUST be used consistently.";
          }
          leaf role {
            type string;
```

```
            description
              "The relationship between this organization
               and this tag. The role of tag creator is required for every
               SWID tag. The role of an entity may include any role value,
               but the per-defined roles include: "aggregator",
               "distributor", "licensor", "software-creator",
               "tag-creator". The enumerations of this will include a
               request to IANA in order to be reference-able via an integer
               index.";
          }
          container thumbprint {
            leaf hash-algo {
              type int16;
              description
                "The integer index of the IANA Named Information Hash Algorithm
                 Registry table that is used to create the
                 thumbprint.";
            }
            leaf thumbprint-value {
              type binary;
              description
                "This value provides a hexadecimal string
                 that contains a hash (i.e. the thumbprint) of the signing
                 entities certificate.";
            }
          }
        }
        container link {
          uses global-attributes;
          leaf artifact {
            type string;
            description
              "For installation media
               (rel=installation-media); dictates the canonical name for
               the file.
               Items with the same artifact name should be considered
               mirrors of each other (so download from
               wherever works).";
          }
          leaf href {
            type string;
            mandatory true;
            description
              "An URI pointing to the resource referenced
               using a system-acceptable URI scheme (e.g., file:// http://
               https:// ftp://), including yang+swid://";
          }
          leaf media {
```

```
            type string;
            description
              "This text value is a hint to the tag consumer
               to understand what this SWID tag applies to. This item can
               also be included in the link item to represent a attributes
               defined by the W3C Media Queries Recommendation (see
               http://www.w3.org/TR/css3-mediaqueries/). A hint to the
               consumer of the link to what the target item is applicable
               for.";
          }
          leaf ownership {
            type string;
            description
              "Determines the relative strength of ownership
               of the software components. Valid enumerations are: abandon,
               private, shared.";
          }
          leaf rel {
            type string;
            mandatory true;
            description
              "The relationship between this SWID and the
               target file. Relationships can be identified by referencing
               the IANA registration library:
               https://www.iana.org/assignments/link-relations/link-
relations.xhtml.";
          }
          leaf type {
            type string;
            description
              "A longer, detailed description of the
               software.  This description can be multiple sentences
               (differentiated from summary, which is a very short,
               one-sentence description).";
          }
          leaf use {
            type string;
            description
              "Determines if the target software is a hard
               requirement or not. Valid enumerations are: required,
               recommended, optional.";
          }
        }
        container software-meta {
          uses global-attributes;
          leaf activation-status {
            type string;
            description
```

"Identification of the activation status of

```
          this software title (e.g. Trial, Serialized, Licensed,
          Unlicensed, etc).  Typically, this is used in supplemental
          tags.";
      }
      leaf channel-type {
        type string;
        description
          "Provides information on which channel this
           particular software was targeted for (e.g. Volume, Retail,
           OEM, Academic, etc). Typically used in supplemental tags.";
      }
      leaf colloquial-version {
        type string;
        description
          "The informal or colloquial version of the
           product (i.e. 2013). Note that this version may be the same
           through multiple releases of a software product where the
           version specified in entity is much more specific and will
           change for each software release.
           Note that this representation of version is typically used
           to identify a group of specific software releases that are
           part of the same release/support infrastructure (i.e.
           Fabrikam Office 2013).  This version is used for string
           comparisons only and is not compared to be an earlier or
           later release (that is done via the entity
           version).";
      }
      leaf description {
        type string;
        description
          "A longer, detailed description of the
           software.  This description can be multiple sentences
           (differentiated from summary, which is a very short,
           one-sentence description).";
      }
      leaf edition {
        type string;
        description
          "The variation of the product (Extended,
           Enterprise, Professional, Standard etc).";
      }
      leaf entitlement-data-required {
        type boolean;
        description
          "An indicator to determine if there should be
           accompanying proof of entitlement when a software license
           reconciliation is completed.";
      }
```

```
          leaf entitlement-key {
            type string;
            description
              "A vendor-specific textual key that can be
               used to reconcile the validity of an entitlement. (e.g.
               serial number, product or license key).";
          }
          leaf generator {
            type string;
            description
              "The name of the software tool that created a
               SWID tag.  This item is typically used if tags are created
               on the fly or via a catalog-based analysis for data found on
               a computing device.";
          }
          leaf persistent-id {
            type string;
            description
              "A GUID used to represent products installed
               where the product are related, but may be different
               versions.  For example, an "upgradeCode" (see
               http://msdn.microsoft.com/en-us/library/aa372375(v=vs.85).aspx
               as an reference for this example).";
          }
          leaf product {
            type string;
            description
              "The base name of the product (e.g. Office,
               Creative Suites, Websphere, etc).";
          }
          leaf product-family {
            type string;
            description
              "The overall product family this software
               belongs to.  Product family is not used to identify that a
               product is part of a suite, but is instead used when a set
               of products that are all related may be installed on
               multiple different devices.
               For example, an enterprise backup system may consist of a
               backup services, multiple different backup services that
               support mail services, databases and ERP systems, as well as
               individual software components that backup client system
               entities. In such an usage scenario, all software components
               that are part of the backup system would have the same
               product-family name so they can be grouped together in
               respect to reporting systems.";
          }
          leaf revision {
```

```
            type string;
            description
              "The informal or colloquial representation of
               the sub-version of the given product (ie, SP1, R2, RC1, Beta
               2, etc).  Note that the SoftwareIdentity.version will
               provide very exact version details,
               the revision is intended for use in environments where
               reporting on the informal or colloquial representation of
               the software is important (for example, if for a certain
               business process, an organization recognizes that it must
               have, for example "ServicePack 1" or later of a specific
               product installed on all devices, they can use the revision
               data value to quickly identify any devices that do not meet
               this requirement).
               Depending on how a software organizations distributes
               revisions, this value could be specified in a primary (if
               distributed as an upgrade) or supplemental (if distributed
               as a patch) SWID tag.";
          }
          leaf summary {
            type string;
            description
              "A short (one-sentence) description of the
               software.";
          }
          leaf unspsc-code {
            type string;
            description
              "An 8 digit code that provides UNSPSC
               classification of the software product this SWID tag
               identifies.  For more information see,
               http://www.unspsc.org/.";
          }
          leaf unspsc-version {
            type string;
            description
              "An 8 digit code that provides UNSPSC
               classification of the software product this SWID tag
               identifies. For more information see,
               http://www.unspsc.org/.";
          }
        }
      }
      leaf corpus {
        type boolean;
        description
          "Set to true, if this attribute specifies that this SWID tag is a
           collection of information that describes the pre-installation
```

```
          data of software component.";
      }
      leaf patch {
        type boolean;
        description
          "A set of files that is intended to modify an
           existing set of files (including configuration files,
           scripts and corresponding environment variables that are
           create by the OS for the runtime environment) that composes
           a software component. A software component patch does
           neither alter the version number (see 13) nor the release
           details (descriptive english text, see 44) of a software
           components.
           If a SWID tag is a patch, it MUST
           contain the patch item and its value MUST be set
           to true.

           It is recommended but not required to include a
           rel(ation) item in a patch CoSWID. If a CoSWID includes a
           patch member, but not a rel member, it is implied that it
           SHOULD be installed independently of any other CoSWID tag
           document  -- even if an effective but not explicit
           relationship exists.";
      }
      leaf media {
        type boolean;
        description
          "This text value is a hint to the tag consumer to
           understand what this SWID tag applies to. This item can also
           be included in the link item to represent a attributes
           defined by the W3C Media Queries Recommendation (see
           http://www.w3.org/TR/css3-mediaqueries/). A hint to the
           consumer of the link to what the target item is applicable
           for.";
      }
      leaf supplemental {
        type boolean;
        description
          "Specifies that this tag provides supplemental tag
           data that can be merged with primary tag data to create a
           complete record of the software information. Supplemental
           tags will often be provided at install time and may be
           provided by different entities (such as the tag consumer, or
           a Value Added Reseller).";
      }
      leaf tag-version {
        type string;
        description
```

```
          "This item indicates if a specific release of a
           software component has more than one tag that can represent
           that specific release. This may be the case if a CoSWID tag
           producer creates and releases an incorrect tag that they
           subsequently want to fix,  but with no underlying changes to
           the product the CoSWID tag represents. This could happen if,
           for example, a patch is distributed that has a link
           reference that does not cover all the various software
           releases it can patch. A newer CoSWID tag for that patch can
           be generated and the tag-version value incremented to
           indicate that the data is updated.";

      }
      leaf software-version {
        type string;
        description
          "Underlying development version for the software
           component.";
      }
      leaf version-scheme {
        type string;
        description
           "Scheme used for the version number. Valid
            enumerations are :
            * alphanumeric: strictly a string, sorting alphanumerically
            * decimal: a floating point number (i.e., 1.25 is less than
              1.3 )
            * multipartnumeric: numbers separated via dots, where the
              numbers are interpreted as integers (ie, 1.2.3 ,  1.4.5.6
              , 1.2.3.4.5.6.7). This string convention is similar to
              OIDs.
            * multipartnumeric+suffix: numbers separated via dots, where
              the numbers are interpreted as integers with an additional
              string suffix (e.g., 1.2.3a).
            * semver: a string as defined by the semver.org spec [FiXME:
              reference]
            * unknown: the last resort choice, no attempt should be made
              to order these";
      }
    }
  }
}
<CODE ENDS>
```

Author's Address

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
Darmstadt 64295
Germany

Email: henk.birkholz@sit.fraunhofer.de