YANG Catalog

Ops Area
July 18, 2017

Joe Clarke, Cisco
Introduction

• We’re beginning to have a “good” problem with YANG
  • Lots of YANG modules
  • Lots of bodies working on YANG modules
  • Lots of vendors implementing YANG modules

• Is it sufficient just to have a module developed and implemented?

• We need an industry-wide, single stop, open catalog and toolchain to help YANG consumers, YANG implementors, and YANG authors find and explore available modules
Yangcatalog.org

A YANG model catalog and registry that allows users to find models relevant to their use cases from the large and growing number of YANG modules being published.

This server is running:

- A NETCONF and REST (not RESTCONF-compliant yet) server loaded with the YANG module from draft-clacla-netmod-model-catalog. It currently only allows public read access to the content. Feel free to reach out through the github forum if you are interested in write access. The username is open and the password is open.

https://yangcatalog.org
YANG Catalog Overview

• Set of open source tools
  • YANG Keyword Search
  • YANG Module Impact Analysis
  • YANG Validators
  • YANG-compliant Regular Expression Validator
  • YANG Modules Exploration (changing soon)
  • YANG Implementation Discovery (coming soon)

• An REST-like API to fetch per-module metadata and per-vendor implementation data

• An authenticated API to upload per-module metadata and per-vendor implementation data
Yang Search

Keyword and regex searches supported

The Catalog stores multiple module revisions, but one can choose to only see the latest

Search By Schema Type (e.g., find all matching typedefs)

Find modules by YANG version
### Search Results

#### YANG DB Search Results for 'uri'

<table>
<thead>
<tr>
<th>Name</th>
<th>Revision</th>
<th>Schema Type</th>
<th>Path</th>
<th>Module</th>
<th>Origin</th>
<th>Organization</th>
<th>Maturity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>2013-07-15</td>
<td>typedef</td>
<td>/inet.uri</td>
<td>ietf-inet-types</td>
<td>IETF</td>
<td>RFC</td>
<td>188</td>
<td>N/A</td>
</tr>
</tbody>
</table>

#### YANG Definition for '/inet.uri'

```yangi
typedef uri {
  type string {
    description "The uri type represents a Uniform Resource Identifier (URI) as defined by STD 66."
    objects using the uri type MUST be in US-ASCII encoding, and MUST be normalized as described by RFC 3986 Sections 6.2.1, 6.2.2.1, and 6.2.2.3. All unnecessary percent-encoding is removed, and all case-insensitive characters are set to lowercase except for hexadecimal digits, which are normalized to uppercase as described in Section 6.2.2.3.
    The purpose of this normalization is to help provide unique URIs. Note that this normalization in not sufficient to provide uniqueness. Two URIs that are textually distinct after this normalization may still be equivalent.
    Objects using the uri type may restrict the schemes that they permit, for example, 'data' and 'uri' schemes might not be appropriate.
    A zero-length URI is not a valid URI. This can be used to express "URI absent" where required.
    In the value set and its semantics, this type is equivalent to the uri SMV2 (textual convention defined in RFC 3986)."
    references [RFC 3986: Uniform Resource Identifier (URI): Generic Syntax RFC 3986; Report from the Joint IETF/IEEE Planning Internet Group (Uniform Resource Identifiers (URIs), URIs, and Uniform Resource Names (URNs)).]
}
```
Impact Analysis

Understand the relationship of a given module (what modules does it affect, what modules affect it)

Show bottlenecks to standardization

Identify modules that do not compile.
The YANG Catalog API

Table of Contents

- What are you trying to do?
  - I am a vendor, and I want to provide my module metadata.
  - I am a vendor, and I want to provide my implementation metadata.
  - I am a Standards Development Organization, and I want to provide my module metadata.
- Hack YANG Catalog

Model Creator? Add Model Metadata

If you are a Standard Development Organization (SDO), an open source project or an organization (e.g., a vendor that creates proprietary models), and you want to add your models and related metadata to the Catalog, do the following:

1. Check your modules into GitHub. Ideally submit your modules directly to the https://github.com/YangModels/yang repository via a pull request. Alternatively, you can use any public repository, and then also add a git sub-module to the https://github.com/YangModels/yang repository via a pull request.
2. Request a new YANG Catalog API account if you do not have one already.

Documented at https://yangcatalog.org/contribute.php
Module Metadata API Structure

```plaintext
module: module-metadata
   |--rw modules
      |--rw module* [name revision]
         |--rw generated-from? enumeration
         |--rw maturity-level? enumeration
         |--rw document-name? string
         |--rw author-email? yc:email-address
         |--rw reference? inet:uri
         |--rw name yang:yang-identifier
         |--rw revision union
         |--rw organization string
      |--rw source-file
         |--rw owner string
         |--rw repository string
         |--rw path path
         |--rw branch? string
      |--rw organization-specific-metadata
         |--rw ietf
            |--rw ietf-wg? string
```
Vendor Implementation Metadata API Structure

```plaintext
module: platform-implementation-metadata
   +-rw platforms* [vendor name software-version software-flavor]
      +-rw vendor string
      +-rw name string
      +-rw models* string
      +-rw software-flavor string
      +-rw software-version string
      +-rw os-type? string
      +-rw capabilities-file
         +-rw owner? string
         +-rw repository? url
         +-rw path? path
```
Retrieving Metadata

```
1. {
   "yang-catalog:module": {
      "name": "ietf-interfaces",
      "revision": "2014-05-08",
      "generated-from": "not-applicable",
      "maturity-level": "ratified",
      "document-name": "rfc7223",
      "organization": "ietf",
      "compilation-status": "PASSED",
      "compilation-result": "",
      "prefix": "if",
      "yang-version": "1.0",
      "description": "This module contains a collection of YANG definitions for managing network interfaces. Copyright (c) 2014 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 7223; see the RFC itself for full legal notices.",
      "contact": "wg-web: http://tools.ietf.org/wg/netmod/>WG List: <mailto:netmod@IETF.org> WG Chair: Thomas Nadeau <mailto:nadeau@LucidVision.com> WG Chair: Juergen Schoenwaelder <mailto:j.schoenwaelder@jacobs-university.de> Editor: Martin Bjorkland <mailto:mbj@Mail-f.com>",
      "module-type": "module",
      "source-file": {
         "online": {
            "owner": "YangModels",
            "repository": "https://github.com/YangModels/yang",
            "path": "standard/ietf/RFC/ietf-interfaces@2014-05-08.yang"
         }
      }
   }
} 
```
Retrieving Metadata
YANG Regular Expression Validator

Validate regex patterns using YANG and W3C rules.
Results And Next Steps

• We have fully integrated IETF and BBF models
• Working to incorporate MEF models
• Working with Cisco, Huawei and Juniper to add metadata for their models
• Submitted draft-clacla-netmod-model-catalog-00 to describe the backing store for the Catalog
• Evangelizing YANG Catalog with vendors, SDOs, and customers
• Continue to develop the yang-catalog.yang model
• Take feedback to enhance the Catalog tools
• Add a report for YANG implementation discovery and revamp the YANG exploration module