Lab - Explore YANG Models

Objectives

Part 1: Launch the DEVASC VM

Part 2: Explore a YANG Model on GitHub

Part 3: Explore a YANG Model Using pyang

Background / Scenario

YANG models define the exact structure, data types, syntax and validation rules for the content of messages exchanged between a managed device and another system communicating with the device. Working with files using the YANG language can be a bit overwhelming for the level of details in these files.

In this lab, you will learn how to use the open source **pyang** tool to transform YANG data models from files using the YANG language, into a much easier to read format. Using the “tree” view transformation, you will identify what the key elements of the ietf-interfaces YANG model are.

Required Resources

- 1 PC with operating system of your choice
- Virtual Box or VMWare
- DEVASC Virtual Machine

Instructions

Part 1: Launch the DEVASC VM

If you have not already completed the **Lab - Install the DEVASC-LAB**, do so now. If you have already completed that lab, launch the DEVASC VM now.

Part 2: Explore a YANG Model on GitHub

In this Part, you will install pyang module into your DEVASC VM and explore how it transforms YANG files. Pyang simplifies working with YANG files. The module comes with a pyang command line executable that transforms YANG files into a more human-readable format.

Step 1: Explore Cisco IOS XE YANG models in the GitHub repository.

a. Open Chromium and navigate to https://github.com/YangModels/yang.

b. Under the **master** branch, navigate to the YANG models for the Cisco IOS XE version 16.9.3 by clicking the following directories: **vendor > cisco > xe > 1693**.

c. Scroll down below all the Cisco YANG models and find where the IETF models begin. Look for **ietf-interfaces.yang**.

d. Click **ietf-interfaces.yang** and scroll through all the container nodes, leaf nodes, and list nodes. If you are familiar with output from the IOS command show interfaces, then you should recognize some or all of the nodes. For example, around line 221 you will see the leaf enabled.

```
leaf enabled {
  type boolean;
```

```
        default "true";
        description
          "This leaf contains the configured, desired state of the
           interface.
           Systems that implement the IF-MIB use the value of this
           leaf in the 'running' datastore to set
           IF-MIB.ifAdminStatus to 'up' or 'down' after an ifEntry
           has been initialized, as described in RFC 2863.
           Changes in this leaf in the 'running' datastore are
           reflected in ifAdminStatus, but if ifAdminStatus is
           changed over SNMP, this leaf is not affected.";
        reference
          "RFC 2863: The Interfaces Group MIB - ifAdminStatus";
      }
```

Step 2: Copy the ietf-interfaces.yang model to a folder on your VM.

a. Open VS code.

b. Click **File > Open Folder...** and navigate to the **devnet-src** directory.

c. Click **OK**.

d. Open a terminal window in VS Code: **Terminal > New Terminal**.

e. Create a subdirectory called **pyang** in the **/devnet-src** directory.

```
devasc@labvm:~/labs/devnet-src$ mkdir pyang
devasc@labvm:~/labs/devnet-src$
```

f. Return to your Chromium tab where the **ietf-interfaces.yang** model is still open. Scroll back to the top, if necessary, and click Raw to display just the YANG model data.

g. Select and copy the URL.

h. In the terminal, go to the pyang folder.

i. Use **wget** to save the raw ietf-interfaces.yang file.

```
devasc@labvm:~/labs/devnet-src/pyang$ wget
https://raw.githubusercontent.com/YangModels/yang/master/vendor/cisco/xe/1693
/ietf-interfaces.yang
--2020-06-22 20:42:20--
https://raw.githubusercontent.com/YangModels/yang/master/vendor/cisco/xe/1693/ietf-
interfaces.yang
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133,
151.101.192.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|151.101.0.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24248 (24K) [text/plain]
Saving to: 'ietf-interfaces.yang'

ietf-interfac 100%  23.68K  --.-KB/s    in 0.05s

2020-06-22 20:42:21 (439 KB/s) - 'ietf-interfaces.yang' saved [24248/24248]

devasc@labvm:~/labs/devnet-src/pyang$
```

You now have a local version of the **ietf-interfaces.yang** model that you can manipulate with **pyang**.

Part 3: Explore a YANG Model Using pyang

In this Part, you will install the **pyang** module into your DEVASC VM and explore how it transforms the YANG model you copied from GitHub. Pyang simplifies working with YANG files. The module comes with a **pyang** command line executable that transforms YANG files into a more human readable format.

Step 1: Verify pyang is installed and up to date.

a. In VS Code, open a terminal window.

b. Verify that pyang is already installed with the **pyang -v** command. Your version number may be different than the one shown here. You can also

```
devasc@labvm:~/labs/devnet-src$ pyang -v
pyang 2.2.1
devasc@labvm:~/labs/devnet-src$
```

c. (Optional) You can verify that you have the latest pyang updates using the following **pip3** command. Any updates after this lab was written will be downloaded and installed.

```
devasc@labvm:~/labs/devnet-src$ pip3 install pyang --upgrade
Requirement already up-to-date: pyang in ./.local/lib/python3.8/site-packages (2.2.1)
Requirement already satisfied, skipping upgrade: lxml in ./.local/lib/python3.8/site-
packages (from pyang) (4.5.0)
devasc@labvm:~/labs/devnet-src$
```

Step 2: Transform the ietf-interfaces.yang model.

a. Navigate to the **pyang** directory.

```
devasc@labvm:~/labs/devnet-src$ cd pyang
devasc@labvm:~/labs/devnet-src/pyang$
```

b. Enter pyang -h | more to explore the options for transforming the YANG model. Look for the **-f** option as shown below. You will use the **tree** formatting option.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -h | more
Usage: pyang [options] [<filename>...]

Validates the YANG module in <filename> (or stdin), and all its dependencies.

Options:
  -h, --help            Show this help message and exit
  -v, --version         Show version number and exit
<output omitted>
  -f FORMAT, --format=FORMAT
                        Convert to FORMAT.  Supported formats are: yang, yin,
                        dsdl, jstree, jsonxsl, capability, identifiers, jtox,
                        uml, name, omni, tree, depend, sample-xml-skeleton
<output omitted>
devasc@labvm:~/labs/devnet-src/pyang$
```

c. Transform the **ietf-interfaces.yang** model into a tree format with the following command. Notice that the **leaf enabled** is much easier to find and read in this format.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -f tree ietf-interfaces.yang
```

```
ietf-interfaces.yang:6: error: module "ietf-yang-types" not found in search path
module: ietf-interfaces
  +--rw interfaces
  |  +--rw interface* [name]
  |     +--rw name                          string
  |     +--rw description?                  string
  |     +--rw type                          identityref
  |     +--rw enabled?                      boolean
  |     +--rw link-up-down-trap-enable?     enumeration {if-mib}?
  +--ro interfaces-state
     +--ro interface* [name]
        +--ro name                string
        +--ro type                identityref
        +--ro admin-status        enumeration {if-mib}?
        +--ro oper-status         enumeration
        +--ro last-change?        yang:date-and-time
        +--ro if-index            int32 {if-mib}?
        +--ro phys-address?       yang:phys-address
        +--ro higher-layer-if*    interface-state-ref
        +--ro lower-layer-if*     interface-state-ref
        +--ro speed?              yang:gauge64
        +--ro statistics
           +--ro discontinuity-time    yang:date-and-time
           +--ro in-octets?            yang:counter64
           +--ro in-unicast-pkts?      yang:counter64
           +--ro in-broadcast-pkts?    yang:counter64
           +--ro in-multicast-pkts?    yang:counter64
           +--ro in-discards?          yang:counter32
           +--ro in-errors?            yang:counter32
           +--ro in-unknown-protos?    yang:counter32
           +--ro out-octets?           yang:counter64
           +--ro out-unicast-pkts?     yang:counter64
           +--ro out-broadcast-pkts?   yang:counter64
           +--ro out-multicast-pkts?   yang:counter64
           +--ro out-discards?         yang:counter32
           +--ro out-errors?           yang:counter32
devasc@labvm:~/labs/devnet-src/pyang$
```