Model Driven Approach to Configuration And Telemetry: YANG

MINT 709: Capstone Project Report

CHAITALEE PATEL

Supervisor: MS. KANWAL CHEEMA

ABSTRACT

NETCONF/YANG has become an extremely important approach for the Network Management Configuration. The intention of this project is to carry out analysis of various Network Configuration Management approaches and revolution by NETCONF/YANG based approach. This Project also demonstrates practical configuration of Network device through NETCONF/YANG based approach.

ACKNOWLEDGEMENT

I express my cavernous sense of obligation and gratitude to my supervisor Ms. Kanwal Cheema for her genuine guidance and constant encouragement throughout this project work. I am highly obliged as my honourable supervisor has devoted her valuable time and shared her expertise knowledge.

I pay my profound gratefulness to Dr. Michael Macgregor for giving me an opportunity to carry out the project work. I must thank him for sparing his valuable time from his busy schedule.

I extend my sincere thanks to Mr. Shahnawaz Mir for providing me such an opportunity to do my project work.

I also wish to express my heartfelt appreciation to my friends, colleagues and many who have rendered their support for the successful completion of the project, both explicitly and implicitly.

CHAITALEE PATEL

INDEX

LIST OF FIGURES

LIST OF TABLES

C H A P T E R 1

INTRODUCTION

This Chapter explains description of the problem for this project. Organization of report is carried out in Section 1.2

1.1 Description of the problem

Network providers are challenged by new requirements for fast and error-free service turn-up. Device-specific adapters, CLI scripting, and entrenched commercial tools, which are traditional configuration approach used to be a restriction to meeting new requirements. Up until recently, there has been no standard way of configuring network devices other than SNMP and SNMP is not optimal for configuration management. NETCONF and YANG are standards focusing on Configuration management which released by IETF. In the NETCONF protocol, a YANG data modeling language is used to model configuration and state data manipulation.

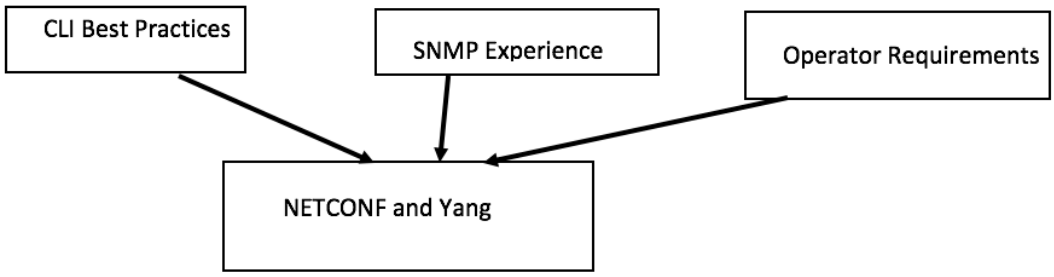


Figure 1.1.1: NETCONF and YANG

In this project, I did an analysis of the traditional approaches to network configuration and configuration through data modeling with YANG, in addition to explaining what YANG is and how it was developed.

Whenever a different and more efficient approach to something is developed, we will see the next technological evolution it will bring. For example, when object oriented programming languages were formed, it completely revolutionized programming. I did an analysis of what changes and revolutions will the YANG modeling bring into the world of networking.

Similarly, the use of YANG protocol in the management of networks instead of SMIv2 (used in SNMP) has been studied. I have also included practical based on YANG/NETCONF with results.

1.2 Organization of Report

This report consists of 5 chapters in total. The framework for the report is described as follows:

Chapter 1 provides a brief introduction of this project. NETCONF and YANG are described in details within Chapter 2. Chapter 3 discusses how testing is carried out for

this project including block diagram and installation procedure for YANG explorer. Result of this project is discussed in Chapter 4. Finally, Chapter 5 gives the conclusive remarks of work.

C H A P T E R 2

NETCONF and YANG

This chapter describes various Network configuration approaches. It also provides details about NETCONF and YANG.

2.1 List of Network Configuration Management approaches:

Current ways to deal with network configuration management has been created step by step since business networks first showed up. This section describes the various approaches used to communicate with network devices keeping in mind the end goal to get to access configuration data.

2.1.1 Manual configuration: System administrators utilize command Line Interface (CLI) to deal with their networks through manual configuration [1]. They utilize TELNET, SSH or terminal servers to access network devices.

Manual configuration is more useful for small sized company, but it is practical for large sized company only when few changes are expected over time. The main benefit of CLI is that network devices can be recorded during each step of configuration.

In contrast, CLI configuration is subject to human errors. System Administrators can do errors during configuration process. In multi-vendor networks, System Administrators must have to remember syntaxes and their sequences to configure network devices. This turns out to be particularly tidy work when the network supports complex and multiple services. At last, CLI isn't scalable inside large sized networks which leads to automatic configuration approach.

2.1.2 Template-based and Script-based configuration: Script based approach is considered as an initial move towards automation. In Script based setup, a grouping of commands is programmed within a script file, during execution it creates a device-native configuration file (or commands) which can be straightforwardly transferred to the device. This approach is valuable in automating a repetitive management task.

The script files are typically composed utilizing hybrid language; high-level language, for example, Perl, UNIX shell, Python or domain-specific language and a lower-level language which determines the real CLI commands that an administrator utilizes while configuring a network service or device [6].

It is possible to automate the backups with script based approach. By taking regular backups, automatic comparison of backup is also possible which is beneficial for reporting purpose. Bulk network configuration changes are possible with script based approach. Pushing out the changes in multiple devices is also easy with this approach [2].

In contrast, this approach rely on the device-specific template since the actual configuration is written in device-native language. It is not practical for multi-vendor network which leads to vendor-neutral configuration.

2.1.3 Vendor-neutral configuration: To determine the issue of script based approach particularly in heterogeneous condition (multi-vendor network), management framework must communicate with the fundamental devices utilizing vendor-neutral language.

This section describes two different vendor-neutral configurations: SNMP based Configuration and HTTP based Configuration.

SNMP Based Configuration:

Simple Network Management protocol (SNMP) is used for managing complex networks. Network devices send protocol data units to different parts of network using SNMP. SNMP needs three components for network management: SNMP Manager, SNMP Agent, and Management Information Base (MIB).

SNMP managers manages networks by monitoring network parameters, and optionally, settings parameters in the managed devices. SNMP manager sends read requests to SNMP Agent. SNMP agent responds these requests with requested data. SNMP Agent is a software that remains in the network, and gather data from these devices. Sometimes, SNMP agent can initiate communication and send data to SNMP manager using asynchronous events called traps [3].

SNMP Agents store the data of managed devices in a database which is called as Management Information Base (MIB). MIB is a database of all objects which are used by network management system to manage and monitor all the devices on the network. MIB can be retrieved by SNMP based network management system. Database of MIB is hierarchical who's each entry is addressed by Object identifier (OID). Objects in the MIB are specified using a subset of Abstract Syntax Notation one (ASN.1) which is called as "Structure of management information version 2 (SMIv2).

A specific advantage of SNMP is the simplicity of the protocol and low intricacy of an agent execution. This low complexity permits a SNMP agent to be inserted into even extremely restricted resource devices. This component made SNMP Ubiquitous, with the goal that today most network devices support SNMP agents.

In any case, SNMP has a few drawbacks which keep it from being the primary protocol for network management today. At first, SNMP had weak security, as SNMP was utilized basically to read data from the agents and there was assumption that only preauthorized and trusted clients had access to the network. Despite the fact that the security model was enhanced in the third version of the protocol, there is still critical security issue as follows: Vulnerability has been found in the authentication code of SNMPv3.It is due to error within the verification of the Keyed-Hash message authentication code (HMAC) digest. It can be exploited by malicious users for spoofing authenticated SNMPv3 packets. So today SNMP is utilized basically for fault management and performance management.

Another issue is the limited capacities of the protocol. As an example, SNMP can read and write single variable or set of variables referred by sequences of numbers, so it makes task of developing system complex. In spite of the fact that there are standardized sets of managed objects (e.g. MIB-II) which are prescribed to be implemented, for the most part, they don't give enough adaptability, consequently a large portion of the usefulness of numerous devices is just accessible by utilizing proprietary MIBs, which makes it hard to work with heterogeneous networks.
Generally, because of its variable-oriented nature SNMP is utilized to work the devices by management software only, and SNMP is normally unsuitable for performing operations manually by people. Because of the limitation of SNMP, numerous modern devices join SNMP based setup with Script based configuration.

HTTP Based Configuration:
HTTP-based arrangement is another arrangement that gives vendor-neutral design. HTTP-based approach utilizes HyperText Transfer Protocol (HTTP) to exchange management data amongst agents and managers.
For this situation, agents (which are normally referred as web-based agents) must have HTTP-server inserted to help HTTP based configuration.
Managers are basically web browsers. The web-based agents may send static site pages to the managers or dynamic website pages, both encoded in HyperText Markup Language (HTML). Java applets and Servlets are likewise basic techniques for collaborations in which HTTP is utilized to communicate between the manager and the web-based agent.
Various vendors have been trying to do development at XML-based management for quite a long while [4]. Juniper Networks created Junos XML API for the Junos network OS. This API gives management applications full access to the agent's management data using Junos XML management protocol, which was formally known as JUNOScript.
Notwithstanding Junos XML structure, Juniper Networks and also Cisco began to adapt **NETCONF protocol** in their devices. Section 2.2 Describes NETCONF Protocol in detail. Numerous scholastic research ventures have been proposed for XML– based network management systems.

2.1.4 Policy-based approach: Policy based management (PBM) has turned into a promising answer for managing networked frameworks. The fundamental inspiration of PBM is to decrease the human intercession by giving dynamic adaptability through reconfiguration and expansion of new strategies without hurting system operation. [2]
Administrator can manage various aspects of a network in a simple manner by deploying a set of policies which govern its behavior. Policies are not dependent on rules which can enhance the functionality of managed devices by introducing interpreted logic that can be changed dynamically without modifying underlying implementation.

This suggests PBM framework should (i) change management prerequisites to syntactical and verifiable rules which governs the function and status of the network, (ii) change such policy rules into device-native configuration, and (iii) implement these configurations to managed devices.

2.1.5 Separating Information and Control planes approaches: A few research works have tended to the network management complexity by raising two key focuses: (1) isolating the usefulness of the information plane and control plane and (2) programmability rather than configuration.
This has prompted another worldview named software defined networking (SDN). It has addressed the requirement for centralized administrative domain. It uses programmable interfaces to provision new network elements and devices, or to reconfigure existing ones. So it is easy to program network using scripting tools. Based on fluctuating needs and demands, administrators can adjust the network-wide traffic flow. It uses SDN controllers to keep global view of the Network. SDN is Vendor-Neutral and standard-based [5].

2.2 Revolution by YANG compare to traditional approach:

This section describes details about NETCONF and YANG. It also shows difference between NETCONF and SNMP.
CLIs are profoundly proprietary, and human intervention is required to comprehend and translate their text-based specification. SNMP does not recognize configurational and operational data. The solution lies in embracing a programmatic and standards-based method for writing configurations to any network device, replacing the procedure of manual configuration.
The Internet Engineering Task Force (IETF) NETCONF is an extensible protocol intended to give a generic interface to configure network devices. NETCONF is based on document-based approach to device configuration, which means that unlike CLI and SNMP it is possible to work with the device configuration.
NETCONF not only enables us to recover or submit setup data to a device, it also allows to edit arbitrary parts of a configuration in a single transaction [6]. However, NETCONF rely on secure transport protocols (SSH, Transport layer security (TLS)) for message passing, which is a good point of view in terms of security.
In addition, it is easy to integrate credential management using an existing security management system in oppose to using SNMP.
NETCONF makes utilization of extended Markup Language (XML) to represent device configuration in the form of document. The format of this document for a device is represented using a modeling language and is customizable, so vendors can specify their own configuration models based on the devices and services they offer. In addition of defining structure of the configuration data, it is also possible to define device's operational state data. IETF network configuration working group has recommended to use YANG language to specify the model.

YANG was particularly created by the IETF NETCONF working group for this reason. The inspiration driving YANG was to make an easily readable data modeling language which takes into consideration a high level of validation of a configuration data store, which is significant for automated network configuration management. YANG helps to identify an invalid configuration, e.g. locking the "disabled", (or Cisco's "shutdown") choice of a remote management interface enables us to ensure that the association between the management station and the managed device won't be lost as a result of a mistake in the configuration.

	SNMP	NETCONF
Standard	IETF	IETF
Resources	OIDs	Paths
Data Models	Defined in MIBs	YANG Models
Data Modeling Language	SMI	YANG
Management Operations	SNMP	NETCONF
Transport Stack	UDP	SSH TCP

Table 2.2.1: Difference between NETCONF and SNMP, Ref [7]

NETCONF:

NETCONF design has been influenced by proprietary protocols like Juniper Networks JUNOScript application programming interface.

- NETCONF delivers mechanisms about how to install, manipulate and delete the configuration of network devices.
- Its operations depend on the top of simple remote procedure call (RPC) layer.
- It uses XML based data encoding for the configuration data with protocol messages.
- NETCONF initially designed as replacement for CLI specific programming interfaces. NETCONF normally transported over SSH protocol.
- NETCONF supports only one session at a time and any new NETCONF connection requests are rejected once the first session is established.
- NETCONF is used to manage multiple network devices, upload and manipulate new configurations, and retrieve partial or full configuration and state data.
- NETCONF can be defined in terms of four layers as mentioned in Figure 2.2.1

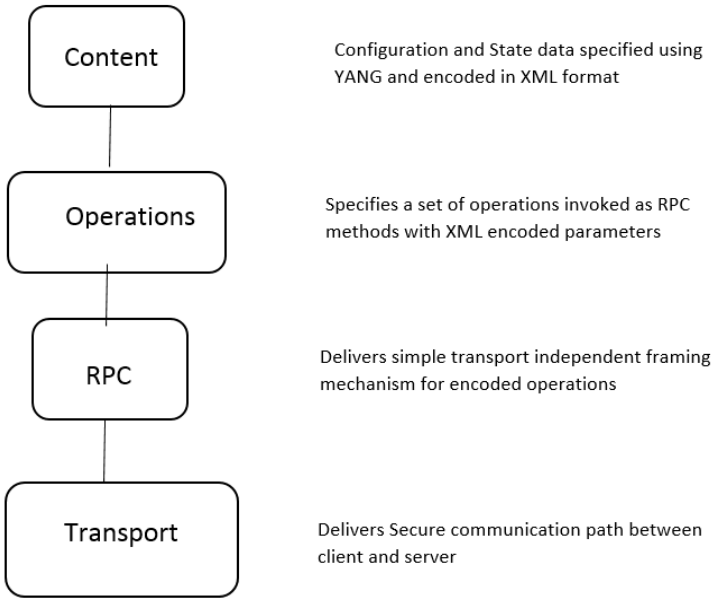


Figure 2.2.1: layers of NETCONF, Ref [7]

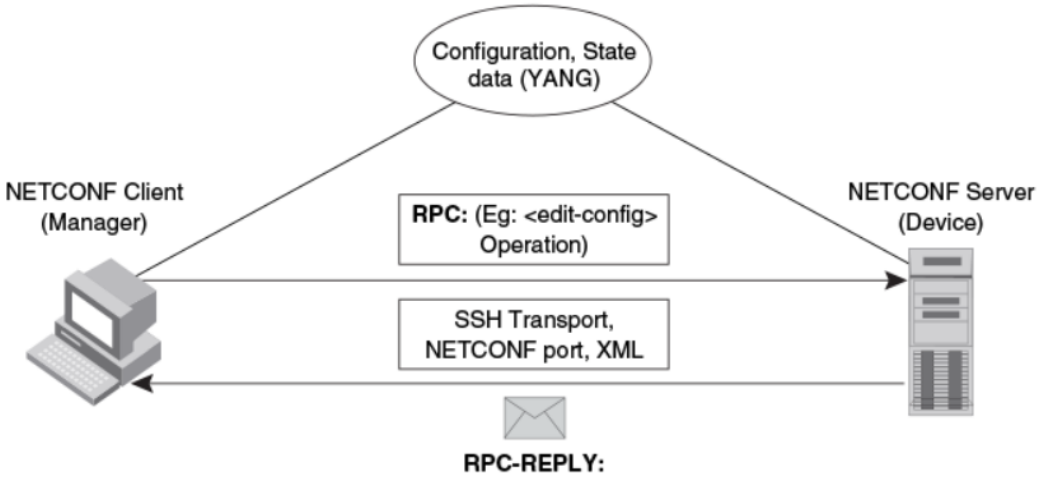


Figure 2.2.2: NETCONF Communication Ref [8]

- Figure 2.2.2 shows how NETCONF client and server communicate with each other. NETCONF utilizes Remote procedure call (RPC) to provide communication between client and server. A client encodes an RPC in XML and delivers it to a server using a secure and connection oriented session. Thereafter, server responds with a reply encoded in XML as shown in above figure.
- There are series of alternating request and reply messages for the communication between the client and server. <rpc> and <rpc-reply> elements are used by NETCONF peers to provide transport protocol independent framing of NETCONF requests and responses. RPC requests are processed sequentially by NETCONF server in the order in which they are received.

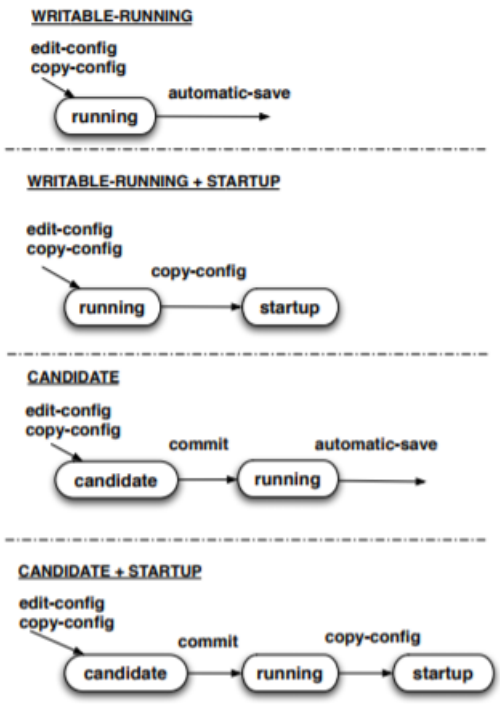


Figure 2.2.3: NETCONF Data stores, Ref [9]

- NETCONF provides a small set of low-level operations to manage the device configurations. It also retrieve device state information. NETCONF supported operations are edit-config, get, get-config, and close-session.
- NETCONF is using the concept of logical data-stores like "writable-running" or "candidate" as shown in figure 2.2.3. Operators require a method to distribute changes to the devices and validate them locally before activating them. It is shown in the last 2 options in figure 2.2.3 where configuration data can be sent to candidate databases in the devices before they are actually running in production applications [9].
- All the NETCONF devices allow the configuration data to be saved, edited, unlocked and locked. Non-volatile storage saves all the changes to the configuration data.

YANG:

After the improvement of the NETCONF protocol in the IETF, it turned out to be evident that a data modeling language was expected to characterize data models controlled by the NETCONF protocol [10].

A design group made a suggestion that turned into the premise of the YANG language. The syntactic structure and the base sort framework was basically acquired from SMIng (structure of Management Information, Next generation) [10]. SMIng can be called as Protocol-independent data definition language for management information which can be used with SNMP management framework. However, in light of the lessons gained from the SMIng project, no endeavors were made to make YANG protocol neutral. Rather, YANG ties into ideas of the

NETCONF protocol, for example, the suspicion that data model occurrences can be serialized into XML. Standardization of YANG began with the arrangement of the NETMOD working group during April 2008 [10].
The YANG 1.0 determination was distributed as RFC 6020 in October 2010. As of late, the NETMOD working group has been chipping away at YANG 1.1, which has been published in August 2016 in RFC 7950 [10].
YANG can be called as tree-structured instead of Object oriented. Here, configuration data is structured into a tree and the data can be of complex ways like unions and lists. Modules can contain definitions and augment the tree in other module. Strong revision rules can be defined for modules. YANG can be mapped with NETCONF XML representation on the wire. YANG is different than older network management data model languages by its strong support of constraints and data validation rules.

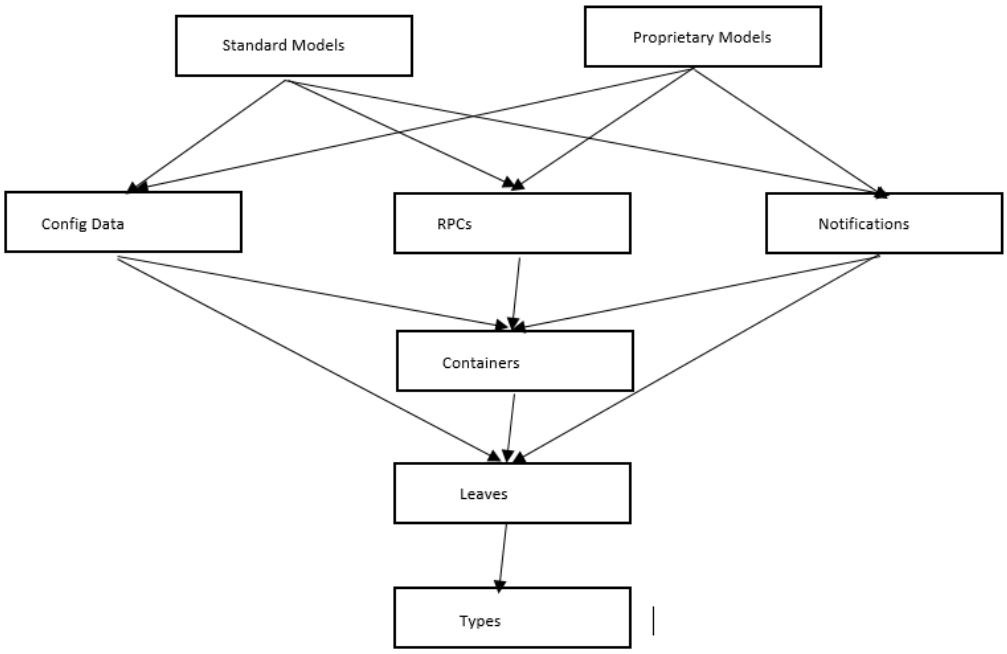


Figure 2.2.4: YANG Data Model Structure, Ref [11]

- A data model suggests how data can be represented and accessed. YANG data models contains modules and submodules. It can define configuration and state data, notifications, and RPCs for use by NETCONF specific operations.
- Figure 2.2.4 describes YANG data model structure. A YANG module characterizes a data model through its data, and the hierarchical association of and requirements on that data. Every module is particularly distinguished by a namespace URL [12].
- A module characterizes a single data model. However, a module can reference definitions in different modules and submodules by utilizing the import statement or include statement. Import Statement allows definitions from

one module to be available inside another module or submodule. Include Statement allows content from a submodule to be available to that submodule's parent module, or to another submodule of that particular parent module. [10]. Also, a module can augment another data model by utilizing the augment statement to characterize the situation of the new nodes in the data model hierarchy and the when statement to characterize the conditions under which the new nodes are legitimate. A module utilizes the feature statement to determine parts of a module that are conditional and the deviation statement to indicate where the device's implementation may deviate from the original definition.

- While you import an external module, you can characterize a prefix that is utilized while referencing definitions in the imported module. We suggest that you utilize the same prefix as that characterized in the imported module to stay away from conflicts.
- YANG models data utilizing a hierarchical, tree-based structure with nodes. YANG characterizes four nodes types. Every node has a name, and relying upon the node type, the node might either characterize a value or contain a set of child nodes. The nodes types are:
  1. leaf node: It contains a single value of a particular type
  2. leaf-list node: It carries a sequence of leaf nodes
  3. Container node: It keeps a grouping of related nodes including just child nodes, which can be any of the four node types
  4. list node: It Contains a grouping of list entries, each of which is uniquely recognized by at least one key leafs
- In YANG, each leaf and leaf-list node incorporates the type statement to recognize the data type for valid data for that node. YANG characterizes a set of built-in types and furthermore gives the typedef statement for characterizing a derived type from a base type, which can be either a built-in type or another derived type.
- A node characterizes configuration data by default. Configuration data is returned utilizing the NETCONF <get-config> operation, and state data is returned utilizing the NETCONF <get> operation.

2.3 Challenges for NETCONF/YANG Based Approach in multi-vendor and large-scale deployments:

There are number of challenges holding NETCONF/YANG back from more widespread adoption in multi-vendor and large-scale deployments as mentioned below [19]:

2.3.1 NETCONF/YANG is not supported by current installed platform

Most of current installed physical equipment doesn't support NETCONF/YANG. Cisco Provides supports for NETCONF/YANG through IOS-XR, IOS-XE and NX-OS based platform. While Huawei provides support for NETCONF/YANG across all VRP8 platform router products, ex, CX6000, NE40E. Juniper delivers

support for NETCONF/YANG for Junos 14.2 and higher OS [20].

There is need of YANG-CLI "translators," for major device models to work. Here, some vendors might argue that they can configure their devices easily by using their own tools, the disadvantage of this method is less automation and increased vendor lock-in.

2.3.2 Interoperability Issue

Implementation and interpretation of standards can be within proprietary manner, or still erroneous ways, so there is high demand for standard NETCONF/YANG validation tools which can show that vendor's models are properly standards-compliant. Vendors can take part in interoperability test with one another to generate confidence in their models.

2.3.3 Opposition to changes and Shortage of Expertise in Service Provider

Some network management teams can feel tough as they will need to change their current way of network operation and move towards NETCONF/YANG. If any person has passed so many years to obtain CLI-related, vendor certifications and scripting knowledge, it can be little motivating to begin to study new NETCONF/YANG or XML in LINUX type environment. A shortage of knowledge with NETCONF/YANG is preventing adoption in service provider environment.

2.3.4 YANG module's standardization Issue

YANG module is still not standardized although NETCONF and YANG have been standardized [21]. A module contains single YANG file, while a model carries more than one YANG modules. Example: A model can be for a switch or network router, while its component modules can be for interfaces, dynamic routing, OSPF, etc. The modules used by device vendors to describe their appliances may vary a lot and efforts to standardize modules have been very slow [21]. This diversity can delay interoperability.

CHAPTER 3

TEST SETUP

This Chapter gives a detailed description about how testing is carried out for this Project. Section 3.1 shows Block Diagram, while YANG explorer installation procedure is described in section 3.2.

3.1 Block Diagram

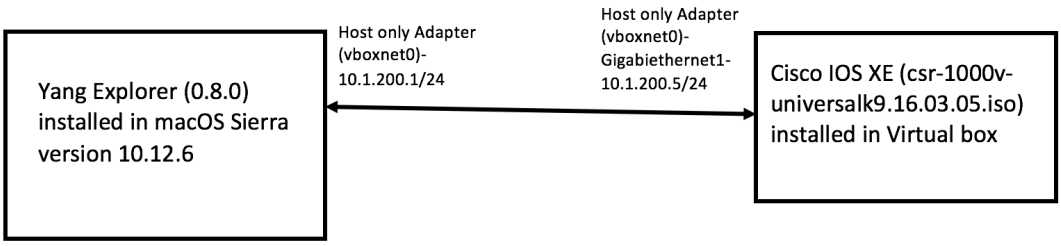


Figure 3.1.1: Block Diagram

Connections:

- As per the diagram, I have installed Yang Explorer (v0.8.0 (Beta)) within local PC (macOS). Cisco IOS XE (csr-1000v-universalk9.16.03.05.iso) is installed in Virtual box (v5.1.4)

- Host only adapter (vboxnet0) is created on virtual box which contains 10.1.200.1/24 IP address. Vboxnet0 is assigned as Network adapter on CISCO IOS XE. 3.2

- Host only Adapter creates a separate network that contains the host and a set of VMs, without the need of host's physical network interface. A separate virtual network interface is created on top of the host that provides connection among VMs and the host.

- 10.1.200.5/24 is assigned on Gigabitethernet1 as static IP address. Both the machines are able to ping with each other.

3.2 YANG Explorer Installation Procedure

Below steps are carried out for YANG explorer installation:

1. Downloaded and installed PIP package manager from below link: https://pip.pypa.io/en/stable/installing/

2. Installed virtualenv by running below command on terminal:
   sudo pip install virtualenv

3. By default brew command not found for MAC. Used below command in terminal to install it. [18]
   /usr/bin/ruby -e "$(curl -fsSL
   https://raw.githubusercontent.com/Homebrew/install/master/install)"

4. Installed graphviz by running below command on terminal:
   brew install graphviz

5. Adobe flash plugin already installed in my google chrome browser. Yang explorer setup install
   git clone https://github.com/CiscoDevNet/yang-explorer.git
   cd yang-explorer
   bash setup.sh

6. Run yang explorer, running with Local host, start server
   cd <install-root>/yang-explorer
   [sudo] ./start.sh &

7. In google chrome,
   http://localhost:8088/static/YangExplorer.html

CHAPTER 4

RESULTS ANALYSIS

This chapter shows the snapshots taken for YANG explorer and Cisco-IOS XE connectivity. It also describes how to control IOS XE from YANG

4.1 Connectivity Testing:

Below snapshots shows successful ping reply from Cisco IOS XE to Local PC.

```
vboxnet0: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        ether 0a:00:27:00:00:00
        inet 10.1.200.1 netmask 0xffffff00 broadcast 10.1.200.255
```

```
chaitalee_IOS_XE#sh ip int br
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       10.1.200.5      YES NVRAM  up                    up
GigabitEthernet2       unassigned      YES NVRAM  administratively down down
chaitalee_IOS_XE#ping 10.1.200.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.200.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/3 ms
chaitalee_IOS_XE#
```

4.2 IOS-XE 16.3.5 Configuration

Below steps shows initial configuration for Cisco IOS XE to configure NETCONF/YANG data modeling

Steps:

1. Enable the NETCONF/YANG

```
chaitalee_IOS_XE(config)#netconf-yang
chaitalee_IOS_XE(config)#% Generating 2048 bit RSA keys, keys will be non-export
able...
*Dec  7 16:29:32.585: yang-infra: netconf-yang server has been notified to start
[OK] (elapsed time was 2 seconds)

*Dec  7 16:29:34.754: %SSH-5-ENABLED: SSH 1.99 has been enabled
*Dec  7 16:29:34.871: %PKI-4-NOCONFIGAUTOSAVE: Configuration was modified.  Issu
e "write memory" to save new IOS PKI configuration
*Dec  7 16:29:39.399: %ONEP_BASE-6-SS_ENABLED: ONEP: Service set Vty was enabled
 by Platform
*Dec  7 16:29:39.803: %ONEP_BASE-6-CONNECT: [Element]: ONEP session Application:
com.cisco.nesd Host:chaitalee_IOS_XE ID:9556 User: has connected.
*Dec  7 16:29:43.657: %ONEP_BASE-6-CONNECT: [Element]: ONEP session Application:
com.cisco.vtyserverutild Host:chaitalee_IOS_XE ID:9600 User: has connected.
*Dec  7 16:29:53.609: %ONEP_BASE-6-CONNECT: [Element]: ONEP session Application:
com.cisco.syncfd Host:chaitalee_IOS_XE ID:7013 User: has connected.
*Dec  7 16:29:53.843: %DMI-5-SYNC_START: F0: syncfd:  External change to running
 configuration detected. The running configuration will be synchronized to the D
MI data store.
*Dec  7 16:29:55.330: %DMI-5-SYNC_COMPLETE: F0: syncfd:  The running configurati
on has been synchronized to the DMI data store.
```

2. Provide username/password for NETCONF-SSH Access
   Enable AAA (authentication, authorization and accounting)

```
chaitalee_IOS_XE(config)#username chaitalee privilege 15 password 0 chaitalee
chaitalee_IOS_XE(config)#aaa new-model
chaitalee_IOS_XE(config)#aaa authorization exec default local
```

3. Allow NETCONF/YANG Syslog and SNMP Event Monitoring

```
chaitalee_IOS_XE(config)#snmp-server community public RW
chaitalee_IOS_XE(config)#snmp-server trap link ietf
chaitalee_IOS_XE(config)#$ enable traps snmp authentication linkdown linkup
chaitalee_IOS_XE(config)#snmp-server enable traps syslog
chaitalee_IOS_XE(config)#snmp-server manager
chaitalee_IOS_XE(config)#logging history debugging
chaitalee_IOS_XE(config)#logging snmp-trap emergencies
chaitalee_IOS_XE(config)#logging snmp-trap alerts
chaitalee_IOS_XE(config)#logging snmp-trap critical
chaitalee_IOS_XE(config)#logging snmp-trap errors
chaitalee_IOS_XE(config)#logging snmp-trap warnings
chaitalee_IOS_XE(config)#logging snmp-trap notifications
chaitalee_IOS_XE(config)#logging snmp-trap informational
chaitalee_IOS_XE(config)#logging snmp-trap debugging
chaitalee_IOS_XE(config)#
```

4. Generate NETCONF notifications for SNMP traps

```
chaitalee_IOS_XE(config)#$ snmp-trap-control trap-list 1.3.6.1.6.3.1.1.5.3
chaitalee_IOS_XE(config-cia-trap)#$-control trap-list 1.3.6.1.6.3.1.1.5.4
chaitalee_IOS_XE(config-cia-trap)#$rap-list 1.3.6.1.4.1.9.9.41.2.0.1
chaitalee_IOS_XE(config-cia-trap)#_
```

5. Network configuration for GigabitEthernet1 which is connected with local PC via vboxnet0

```
chaitalee_IOS_XE(config)# vrf definition Mgmt-vrf
chaitalee_IOS_XE(config-vrf)#address-family ipv4
chaitalee_IOS_XE(config-vrf-af)#exit-address-family
chaitalee_IOS_XE(config-vrf)#address family ipv6
                             ^
% Invalid input detected at '^' marker.

chaitalee_IOS_XE(config-vrf)#address-family ipv6
chaitalee_IOS_XE(config-vrf-af)#exit-address-family
chaitalee_IOS_XE(config-vrf)#exit
chaitalee_IOS_XE(config)#interface gigabitethernet 1
chaitalee_IOS_XE(config-if)#vrf forwarding Mgmt-vrf
% Interface GigabitEthernet1 IPv4 disabled and address(es) removed due to enabli
ng VRF Mgmt-vrf
chaitalee_IOS_XE(config-if)#ip address 10.1.200.5 255.255.255.0
chaitalee_IOS_XE(config-if)#no shut
chaitalee_IOS_XE(config-if)#negotiation auto
chaitalee_IOS_XE(config-if)#exit
chaitalee_IOS_XE(config)#ip route vrf Mgmt-vrf 0.0.0.0 0.0.0.0 10.1.200.1
chaitalee_IOS_XE(config)#
```

6. verification of software processes needed to support the Data Model Interface (DMI) on IOS XE.

```
chaitalee_IOS_XE#show platform software yang-management process
confd            : Running
nesd             : Running
syncfd           : Running
ncsshd           : Running
dmiauthd         : Running
vtyserverutild   : Running
opdatamgrd       : Running
ngnix            : Not Running
```

7. Verification of SSH connectivity to the IOS-XE from the local PC.
   If it is connected, then YANG explorer from browser will also connect with IOS-XE

chaitalees-MacBook-Pro:~ chaitaleepatel$ ssh -s chaitalee@10.1.200.5 -p 830 netconf

chaitalee@10.1.200.5's password:

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<capabilities>
<capability>urn:ietf:params:netconf:base:1.0</capability>
<capability>urn:ietf:params:netconf:base:1.1</capability>
<capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>
<capability>urn:ietf:params:netconf:capability:xpath:1.0</capability>
<capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
<capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
<capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
<capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
<capability>urn:ietf:params:netconf:capability:interleave:1.0</capability>
<capability>http://tail-f.com/ns/netconf/actions/1.0</capability>
<capability>http://tail-f.com/ns/netconf/extensions</capability>
<capability>urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=report-all</capability>
<capability>urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults?revision=2011-06-01&amp;amp;module=ietf-netconf-with-defaults</capability>
```

…….
Cntrl+c to stop it

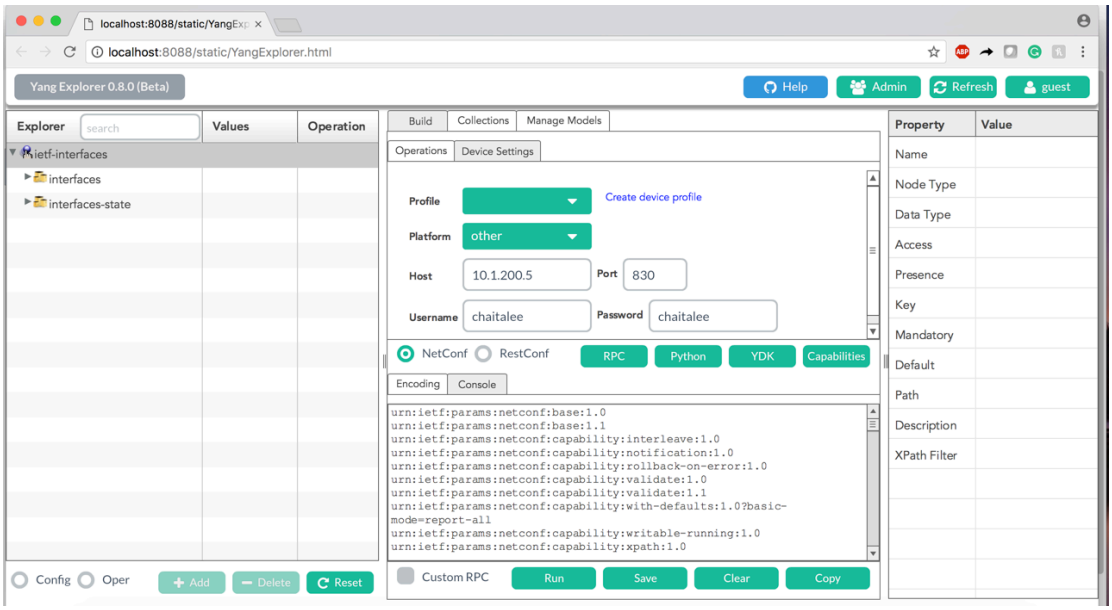
4.3 YANG Explorer Initial Configuration

Below steps shows initial configuration required for YANG Explorer to gather details from IOS XE. It also includes RPC script to subscribe Notifications.[18]

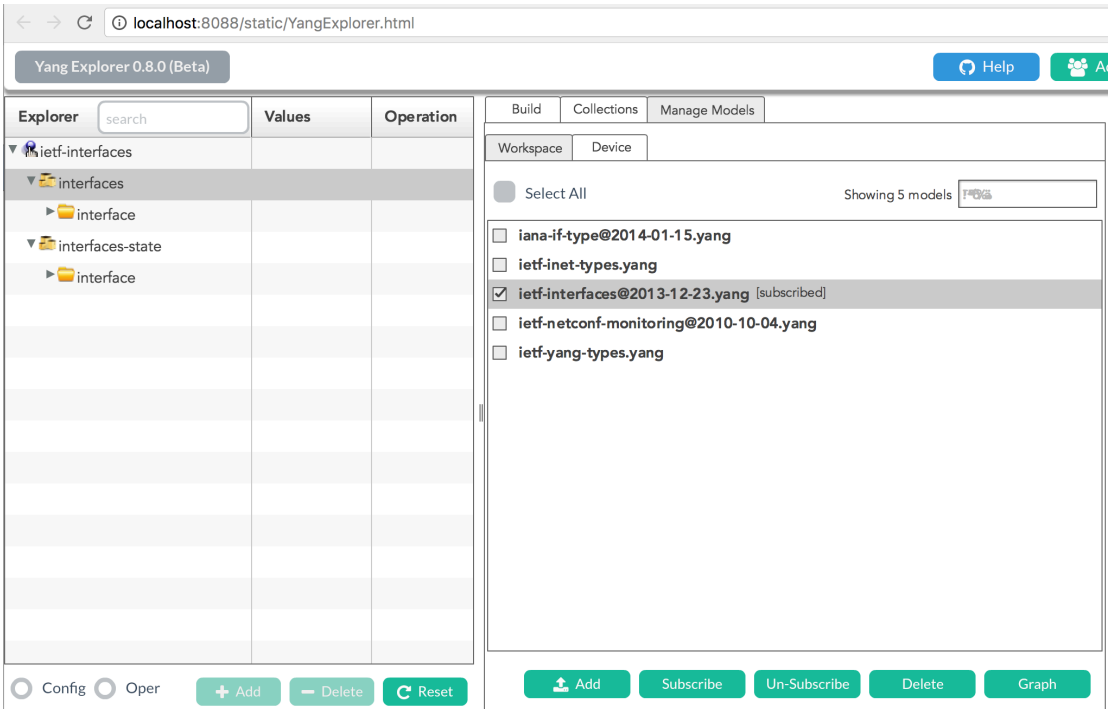
1. From the terminal use ./start.sh & command from the yang-explorer directory.

2. Yang explorer runs on port 8088. From the google chrome window use below URL:
http://localhost:8088/static/YangExplorer.html

3. When connected, enter Host IP 10.1.200.5, port:830 (NETCONF), username: chaitalee and password: chaitalee. Click on capabilities to retrieve the YANG operational capabilities list from the IOS-XE



4. Subscribe the YANG data models under Manage Models options. It is needed for NETCONF remote procedure calls (RPC) messages.

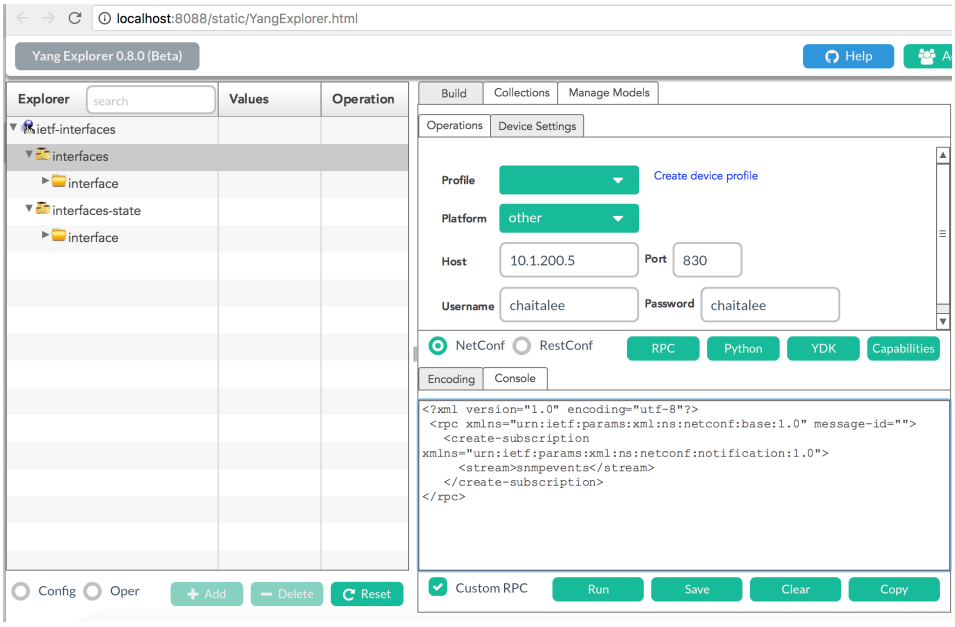


5. Subscribe to NETCONF Notifications:
A client (YANG Explorer) registers to receive NETCONF notification from a server (IOS-XE) by sending YANG formatted NETCONF RPC message. Server sends NETCONF notifications asynchronously to every client that subscribes. The NETCONF server begins to send the event notifications to the NETCONF client

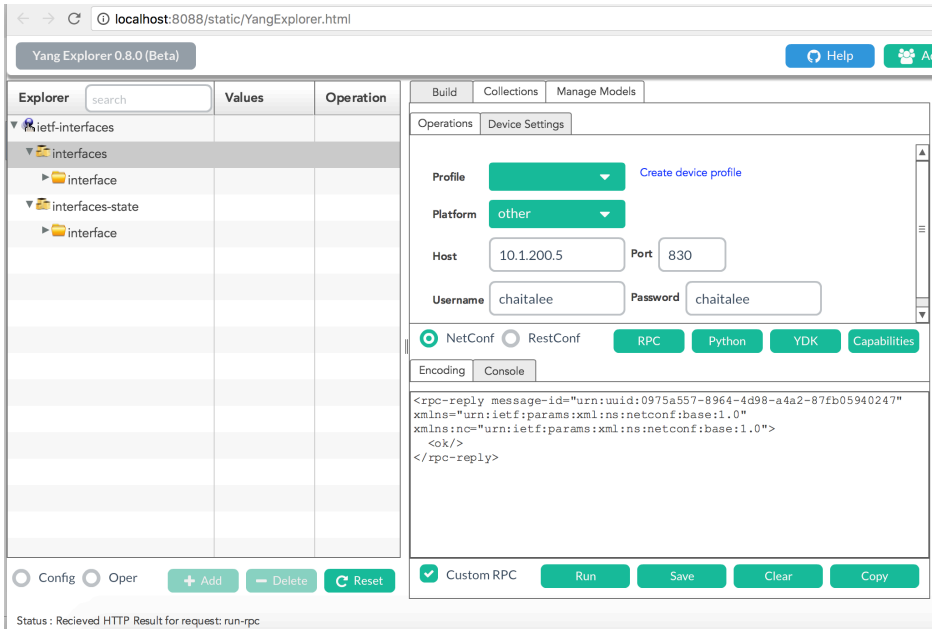
because the events occur inside the system. These event notifications will continue to be sent until either the NETCONF session is terminated or the subscription terminates for some different reason.

RPC Script to subscribe notifications:

```
<?xml version="1.0" encoding="utf-8"?>
 <rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
   <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
     <stream>snmpevents</stream>
   </create-subscription>
 </rpc>
```



Reply comes with OK message for successful operation after clicking on RUN as shown in below snapshot:



4.4 NETCONF/YANG Operation

This section provides 3 different scenarios. It shows that the YANG formatted NETCONF RPC messages sent via NETCONF from the Yang Explorer application to the IOS XE. These messages are converted to standard Cisco IOS CLI by the confd software process on the IOS XE. Also, Cisco IOS CLI data (show command data) is converted to YANG formatted data by the confd software process on the Cisco IOS XE before it is sent as NETCONF RPC message to the Yang Explorer application. This means that the regular CLI can still be used on the IOS XE to configure and collect show command data in addition to use NETCONF/YANG to do the same.

Part 1: Data retrieval

In this part, Yang explorer is requesting a list of interface names from the IOS XE. get-config operation is used to collect interface names. IOS XE has only 2 interfaces setup in practice.

RPC Script to collect interfaces:

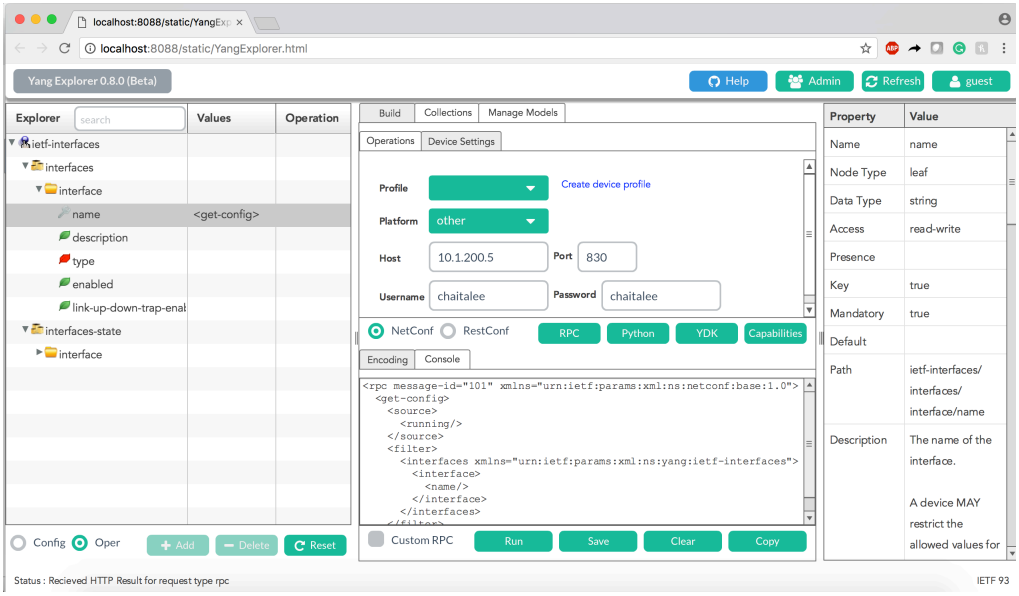
```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface/>
      </interfaces>
    </filter>
  </get-config>
</rpc>
```

IOS XE reply with int1 and int2:

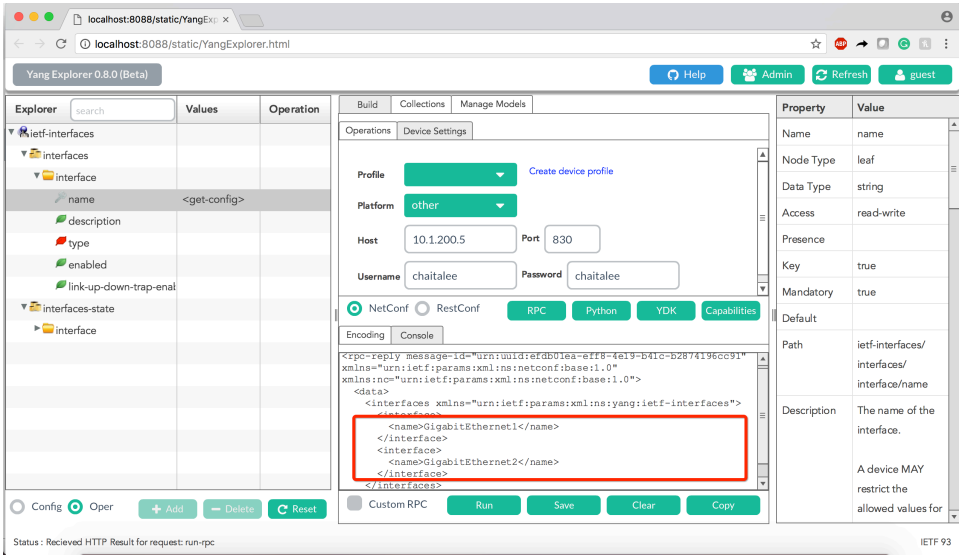
```
<rpc-reply message-id="urn:uuid:944eddef-a253-4468-a1d8-cbc2a975f42f"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-
type">ianaift:ethernetCsmacd</type>
        <enabled>true</enabled>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <address>
            <ip>10.1.200.5</ip>
            <netmask>255.255.255.0</netmask>
          </address>
```

```
      </ipv4>
      <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip"/>
    </interface>
    <interface>
      <name>GigabitEthernet2</name>
      <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-
type">ianaift:ethernetCsmacd</type>
      <enabled>false</enabled>
      <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip"/>
      <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip"/>
    </interface>
  </interfaces>
 </data>
</rpc-reply>
```

Setup on YANG Explorer:



Reply coming after clicking on RUN as below:



Part 2: IOS XE interface control from YANG Explorer:

In this part, YANG explorer is used to Shut Down an Ethernet Interface on the IOS XE. Initially Gigabit Ethernet 2 is up on IOS XE. NETCONF RPC script is applied from YANG explorer application to shut down it. Enabled=false is added in RPC script to shut down the dedicated interface.

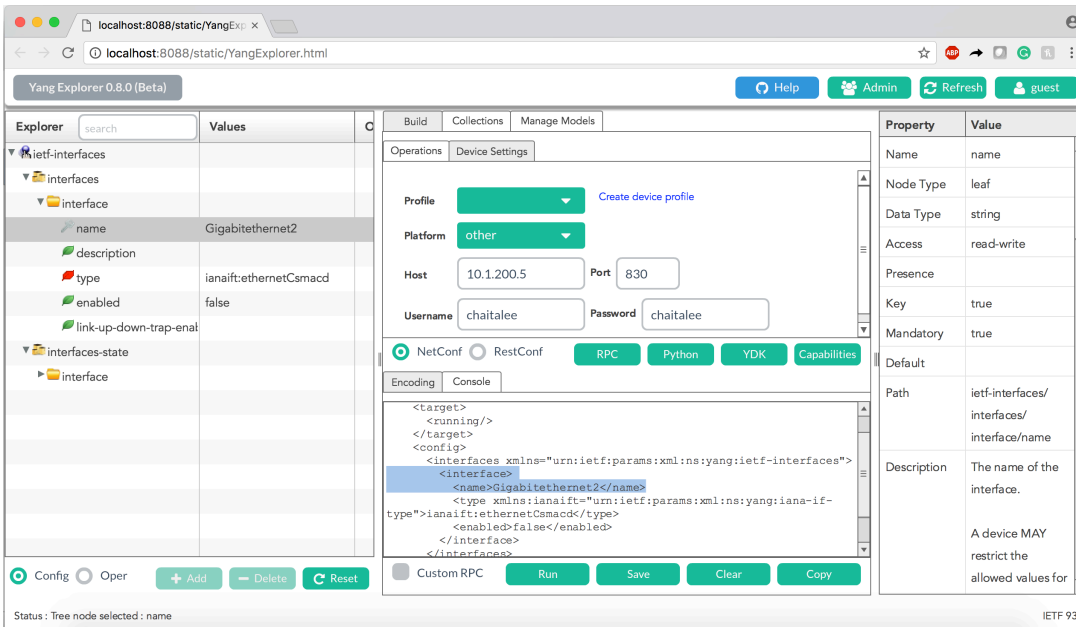
RPC script to shutdown interface:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface>
          <name>GigabitEthernet2</name>
          <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-
type">ianaift:ethernetCsmacd</type>
          <enabled>false</enabled>
        </interface>
      </interfaces>
    </config>
  </edit-config>
</rpc>
```

Reply comes with OK message after clicking RUN on YANG Explorer:

```
<rpc-reply message-id="urn:uuid:4fa1c369-201f-498d-8d66-d48d5ee6ab11"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

Setup on YANG Explorer:



Output Observed on IOS XE at same time when RPC is applied:

```
chaitalee_IOS_XE#sh ip int br
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       10.1.200.5      YES manual up                    up
GigabitEthernet2       unassigned      YES NVRAM  up                    up
chaitalee_IOS_XE#
*Dec  7 17:43:45.715: %DMI-5-CONFIG_I: F0: nesd:  Configured from NETCONF/RESTC
NF by chaitalee, transaction-id 3972
*Dec  7 17:43:47.674: %LINK-5-CHANGED: Interface GigabitEthernet2, changed stat
 to administratively down
*Dec  7 17:43:48.675: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEt
ernet2, changed state to down
chaitalee_IOS_XE#sh ip int br
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       10.1.200.5      YES manual up                    up
GigabitEthernet2       unassigned      YES NVRAM  administratively down down
chaitalee_IOS_XE#
```

Part 3: Save the configuration on IOS XE through yang explorer application

The running configuration can be saved to the startup configuration on the IOS XE by sending this YANG formatted NETCONF RPC message to the IOS XE via NETCONF.

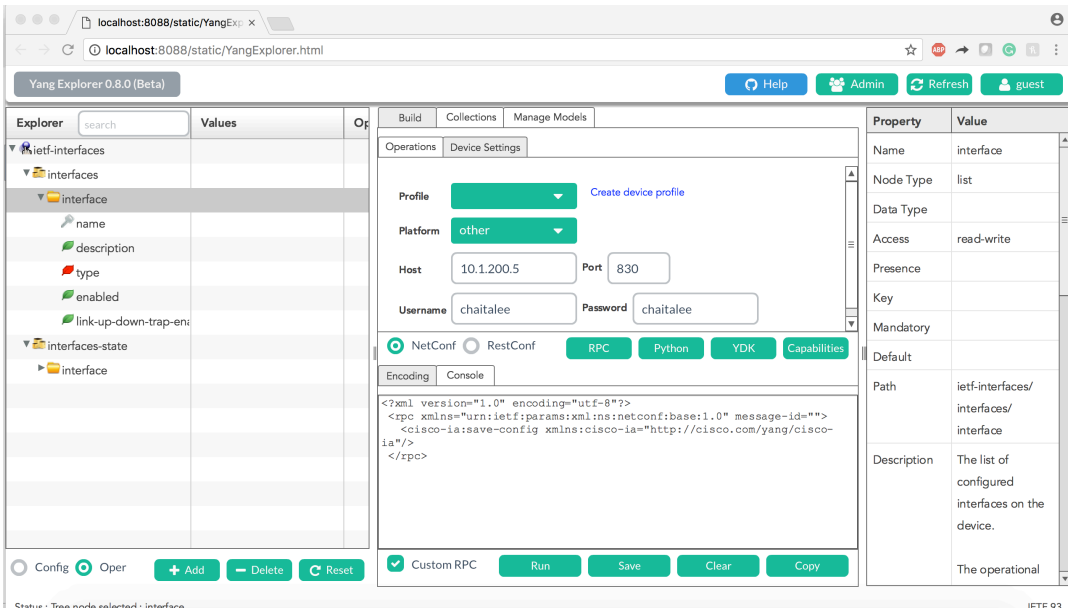
RPC Script to Save Configuration:

```
<?xml version="1.0" encoding="utf-8"?>
 <rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
   <cisco-ia:save-config xmlns:cisco-ia="http://cisco.com/yang/cisco-ia"/>
 </rpc>
```

Reply comes with 'Save running-config successful' message after clicking RUN on YANG Explorer:

<rpc-reply message-id="urn:uuid:8aa74576-340d-4559-840a-e506fe15af62"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
<result xmlns="http://cisco.com/yang/cisco-ia">**Save running-config successful</result>**
</rpc-reply>

Setup on YANG Explorer:



Below two Snapshots shows running config and startup config has same configuration for gigabiethernet2 as shutdown state which means configuration is saved.

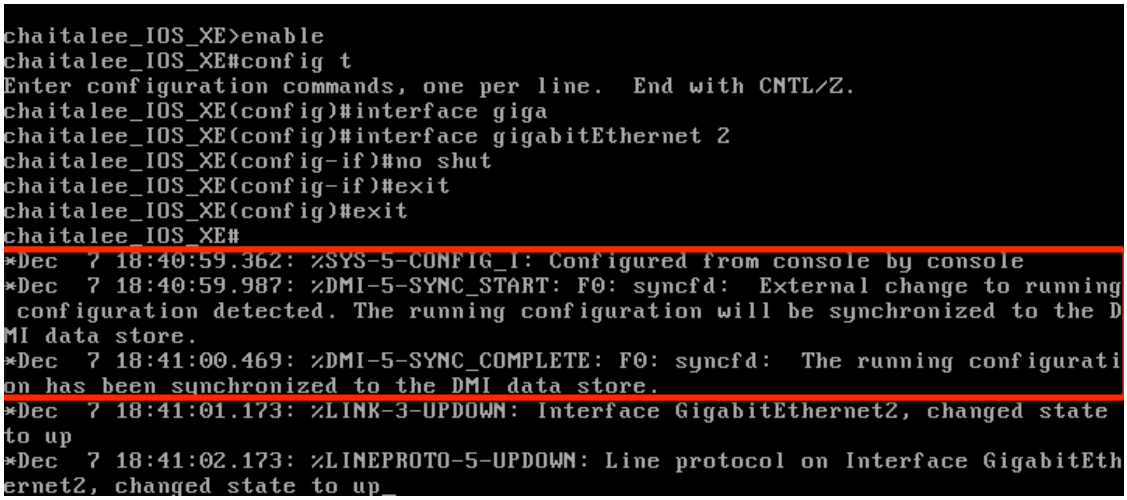
```
chaitalee_IOS_XE#show startup-config | begin interface
interface GigabitEthernet1
 vrf forwarding Mgmt-vrf
 ip address 10.1.200.5 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 no ip address
 shutdown
 negotiation auto
 no mop enabled
 no mop sysid
!
!
```

```
chaitalee_IOS_XE#show running-config | begin interface
interface GigabitEthernet1
 vrf forwarding Mgmt-vrf
 ip address 10.1.200.5 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 no ip address
 shutdown
 negotiation auto
 no mop enabled
 no mop sysid
!
!
```

Part 4: Configure IOS XE from the CLI

IOS XE CLI can still be used to configure the switch and collect show command data in addition to using NETCONF/YANG to do the same. When the IOS XE CLI is used instead of NETCONF/YANG to configure the new running-config is synchronized with the Data Model Interface (DMI) on the IOS XE via the syncfd software process.

Below snapshot shows manually bringing up the interface synchronizing with YANG Explorer:



Use get-config on yang explorer to see the changes on gigabitethernet 2:

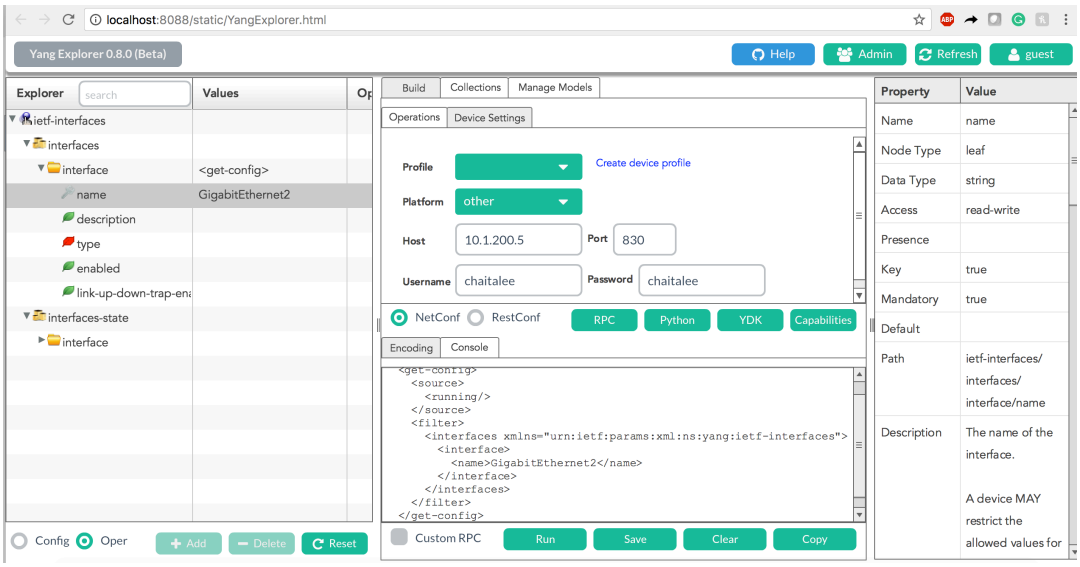
```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface>
          <name>GigabitEthernet2</name>
        </interface>
      </interfaces>
    </filter>
  </get-config>
</rpc>
```

Enable =true comes in Reply which means interface is up now:

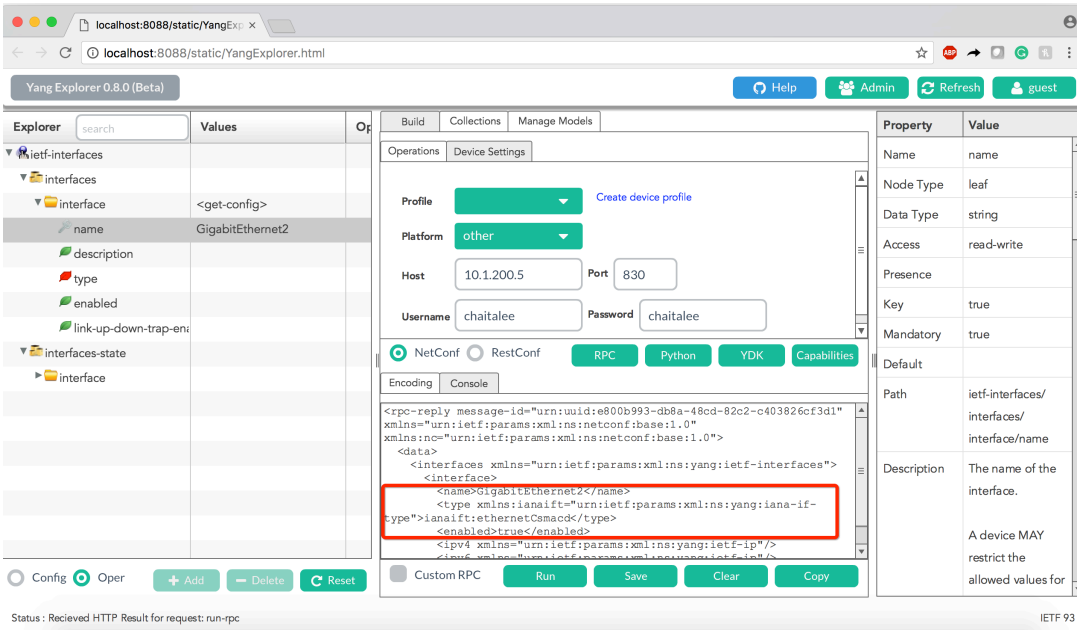
```
<rpc-reply message-id="urn:uuid:e800b993-db8a-48cd-82c2-c403826cf3d1"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
```

```
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>GigabitEthernet2</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-
type">ianaift:ethernetCsmacd</type>
        <enabled>true</enabled>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip"/>
        <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip"/>
      </interface>
    </interfaces>
  </data>
</rpc-reply>
```

Setup on YANG Explorer:



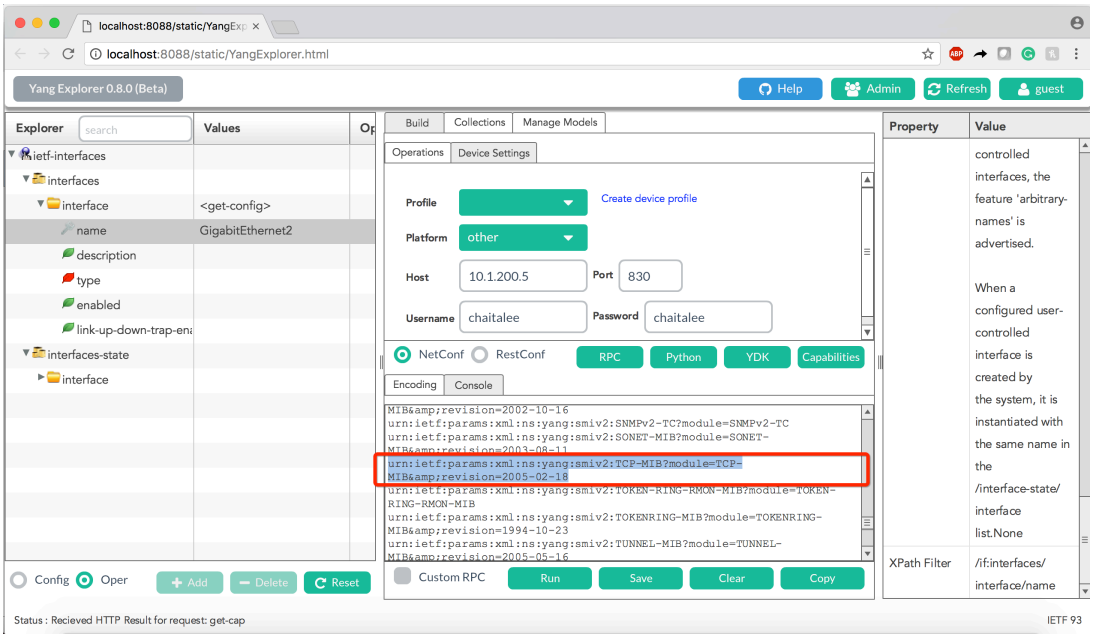
Reply comes after clicking on RUN as below:



Part 5: Check What SNMP MIB (Management Information Base) Operational Data is Available via GET Request Operations

The SNMP MIB data that can be returned with NETCONF GET operations is not user configurable. SNMP MIB has been described in section 2.1.3 in this project. All supported SNMP MIBs that are converted into structured data defined by YANG data models are part of the IOS-XE software. There are three options stated below to discover what MIB data is available in GET requests. All supported MIBs will include smiv2 in the capability response.

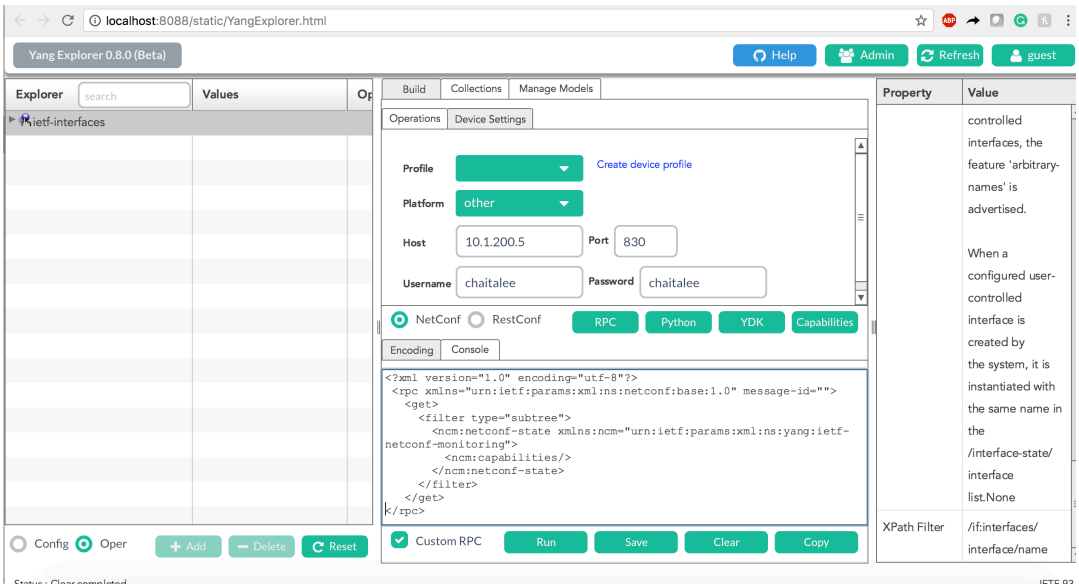
Option 1: The Capabilities button can be selected in the Yang Explorer application GUI. The IOS XE replies back with its capability list that contains smiv2 MIB entries.



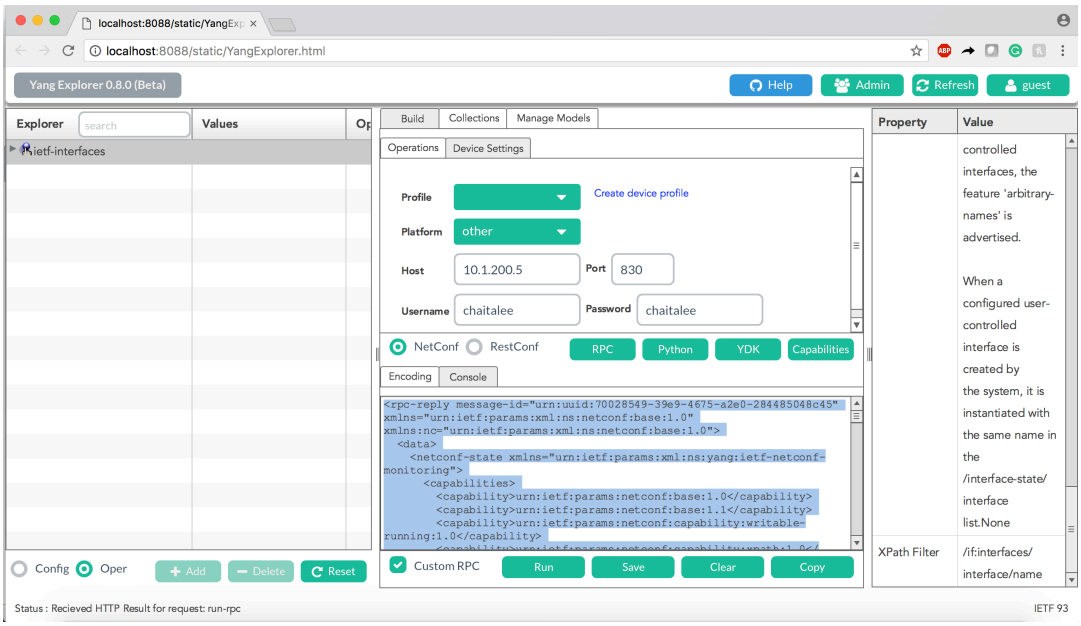
Option 2: YANG formatted NETCONF RPC message can be sent to the IOS XE via NETCONF in order to retrieve the capabilities list which includes available smiv2 MIB models.

```
<?xml version="1.0" encoding="utf-8"?>
 <rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
   <get>
     <filter type="subtree">
       <ncm:netconf-statexmlns:ncm="urn:ietf:params:xml:ns:yang:ietf-netconf-
monitoring">
         <ncm:capabilities/>
       </ncm:netconf-state>
     </filter>
   </get>
</rpc>
```

Setup on YANG Explorer



Reply coming after clicking on RUN



Option 3: A list of available MIB models can be viewed in the NETCONF capabilities and Hello message returned by the IOS XE in response to an SSH connection from the local PC.

chaitalees-MacBook-Pro:~ chaitaleepatel$ ssh -s chaitalee@10.1.200.5 -p 830 netconf
chaitalee@10.1.200.5's password:
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<capabilities>
<capability>urn:ietf:params:netconf:base:1.0</capability>
<capability>urn:ietf:params:netconf:base:1.1</capability>

<capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>
<capability>urn:ietf:params:netconf:capability:xpath:1.0</capability>
<capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
<capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
<capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</cap
……
……..
ision=2003-08-11</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:TCP-MIB?module=TCP-MIB&revision=2005-02-18</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:TOKEN-RING-RMON-MIB?module=TOKEN-RING-RMON-MIB</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:TOKENRING-MIB?module=TOKENRING-MIB&revision=1994-10-23</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:TUNNEL-MIB?module=TUNNEL-MIB&revision=2005-05-16</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:UDP-MIB?module=UDP-MIB&revision=2005-05-20</capability>
<capability>urn:ietf:params:xml:ns:yang:smiv2:VPN-TC-STD-MIB?module=VPN-TC-STD-MIB&revision=2005-11-15</capability>
</capabilities>
<session-id>5719</session-id></hello>]]>]]>

C H A P T E R 5

CONCLUSION

I discussed various Network Configuration Management approaches. From that, NETCONF/YANG based approach is turned out to be best approach. It has the key advantages in terms of selective data retrieval, multiple configuration data storage, configuration testing support and configuration validation support. 'get-config' command used in this project returns only configuration data which is unique feature of NETCONF/YANG based configuration.

NETCONF is developed to enable more effective and efficient network management by addressing the shortcomings of current network configuration protocols like SNMP and CLI. There can be good chance that NETCONF/YANG will become widely established due to its transaction based approach. YANG represents configuration data as a model to make it easy for multiple Network equipment manufacturers to implement a feature in a universal way. Since SNMP is more efficient for monitoring, it will still continue with NETCONF/YANG based approach. The network configuration management and automation issues that negatively affects service providers today are resolved by the NETCONF/YANG based approach. So it turns out to be feasible approach for device Management.

I also tried to configure CSR 1000v (IOS XE) through YANG explorer and also using CLI. Data retrieval, interface control, configuration save and SNMP operational data checking for cisco IOS XE is performed via YANG Explorer through NETCONF messages.

In future, Legacy devices are unlikely to be made NETCONF/YANG-compatible so there is still need for a CLI translator or middleware platform to connect to and configure old model devices. The skill set of operational teams requires to be developed to include NETCONF/YANG capabilities.

R E F E R E N C E S

References:

1. Research paper on “Network Configuration Management Using NETCONF and YANG”,by Jürgen Schönwälder, Jacobs University, Martin Björklund, Tail-f Systems, Phil Shafer, Juniper Networks
2. https://www.auvik.com/media/blog/network-configuration-management/
3. http://www.brocade.com/content/html/en/administration-guide/nos-601a-adminguide/GUID-37232310-C88A-4466-925C-36F990B410E1.html
4. RFC:6244: An Architecture for Network Management Using NETCONF and YANG
5. https://www.cisco.com/c/en/us/solutions/software-defined-networking/sdn-vs-nfv.html
6. NETCONF AND YANG Tutorial Series from tail –f Education http://www.tail-f.com/education/
7. http://www.ieee802.org/802_tutorials/2014-07/Tutorial_Berman_1407.pdf
8. http://www.brocade.com/content/html/en/administration-guide/netiron-05900-adminguide/GUID-3F0F0107-DF2B-4F3B-ADC8-FF41E9F1E8FE.html
9. Research Paper on “Automating Network and Service Configuration Using NETCONF and YANG”, by Stefan Wallin and Claes Wikstrom http://www.tail-f.com/wordpress/wp-content/uploads/2013/03/Tail-f-NETCONF-YANG-Service-Automation-LISA-Usenix-2011.pdf
10. RFC:6020: YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)
11. https://forums.juniper.net/jnet/attachments/jnet/Automation_Scripting/592/1/JUNOS-NETCONF-and-YANG-Tutorial.pdf
12. https://www.juniper.net/documentation/en_US/junos/topics/concept/netconf-yang-modules-overview.html
13. https://www.cisco.com/c/en/us/support/docs/storage-networking/management/200933-YANG-NETCONF-Configuration-Validation.html
14. https://www.networkcomputing.com/data-centers/network-automation-more-scripting/819125107
15. http://www.ietf.org/rfc/rfc4741.txt
16. http://www.cisco.com/c/en/us/td/docs/optical/ncs1001/b_Datamodels_cg_ncs1001/b_Datamodels_cg_ncs1000_chapter_00.html
17. https://coolestguidesontheplanet.com/installing-homebrew-on-macos-sierra-package-manager-for-unix-apps/
18. https://www.cisco.com/c/en/us/support/docs/storage-networking/management/200933-YANG-NETCONF-Configuration-Validation.html
19. RFC7223:https://datatracker.ietf.org/doc/rfc7223/
20. https://github.com/YangModels/yang/tree/master/vendor
21. https://www.oneaccess-net.com/images/public/wp_heavy_reading.pdf