Review on Technologies Used in Developing a Tool for Automatic Software Upgrades at Data Centers

Niveditha.V.K[1], Dr. Kiran.V[2], Avinash Pathak[3]

[1,2] RV College of Engineering, Bengaluru, India
[3] Cisco, Bengaluru, India
[1] nivedithavk.ec17.@rvce.edu.in , [2] kiranv.@rvce.edu.in

Abstract: *With the fast evolution pace of various technologies such as Internet of Things (IoT), Cloud Computing and the world moving towards digitalization created an increased need for data centers than ever before. Data centers support a wide range of internet services, including web hosting, e-commerce, and social networking. In recent years huge datacenters have been owned and run by tech giants like Google, Facebook, Microsoft etc. and these firms are known as Hyper-scalers. Hyper-scalers are the next big thing, ready to fundamentally alter the internet world for data storage through a variety of services supplied by them across all technological domains. The tool for automatic software upgrade focuses on having a seamless upgrade for the devices in the datacenters mainly in huge datacenters owned by the hyper-scalers. This paper mainly focuses on the technologies used in developing the tool for automatic software upgrade, an overview on how the tool is developed and its features. By deploying this tool in the datacenters, it supports them in delivering more efficient services.*

Keywords: CLI, Yang, Netconf, gRPC, SSH, Netmiko, Ncclient, docker.

1. INTRODUCTION

In simplest terms, a data center is a physical facility that organizations use to house their critical applications and data. The data center design is built on a network of computer and storage resources that allow shared applications and data to be delivered. Data Centers are playing a vital role in supporting enterprise services and cloud computing services such as social networking, large-scale computations, web searching, online gaming etc. Hyperscalers are the organizations who own huge data centers to support their services. The key components of the data center design include routers, servers, switches, storage systems and application – delivery controllers. All these networking devices need periodic upgrade to improve the performance. Moving the networking device from an older software version to a new software version is known as software upgrade. Hyperscalers maintain huge data centers which house thousands of networking devices, performing upgrade on such huge lot has not been a smooth experience for them. In order to overcome the hurdles and to give them a seamless upgrade experience without hindering the services this tool is developed. It mainly aims at automatic and smooth upgrades; it just needs onetime enrollment of the device on the tool. It can perform a seamless upgrade on many devices at once. The major concern in upgrading the devices is to fix the bugs and to enable more features on the device which in turn increases the performance of the networking device.

In the following sections different technologies used in developing the tool and the overview on how the tool is developed is discussed.

2. Technologies Used in the Tool

The tool is developed as a combination of CLI and Yang model. Using CLI alone has its own disadvantages and Yang alone could not provide all the necessary support. Therefore, the combination of both the models is chosen. Netconf protocol is used to configure the devices and communicate with them. SSH is used to get connected to the remote devices that are added on to the tool. Different python libraries like Netmiko, ncclient are used in the developing the tool. The tool is hosted as a docker application

2.1. CLI

CLI is known as command line interface. As the name itself suggests it is a command line program that allows users to type text commands instructing the device to do specific tasks. CLI was developed in 1960s to interact with the operating system or application. Prior to the invention of the mouse this was the only way to interact with the computer. There are different types of CLIs such as operating systems command line interface, application command line interfaces.

The user can interact with the devices using CLI. If the IP address of the device the user wants to get connected to is known the user can either use ssh or telnet to connect to the device. If the device does not support IP, the user has to use terminal server to access the console of the device. CLI has its share of advantages due to which it is being used till date. One of the major advantages of CLI is the results are quick and consumes less memory and processing power when compared to the other interfaces. It needs fewer resources and is highly precise. If the commands are known CLI is the simple and the most efficient.

On the other hand, the major disadvantage of the CLI is the users must be familiar with the commands and the lack of standards because each vendor has its own command sets and procedure for device configuration. The lack of standard CLls prevents interoperability of equipment from different vendors. Although there are many issues with CLI it is still one of the most common approaches to network configuration.

In the tool the CLI approach is used to get connected to the device. It is used to interact with the console of the remote device, to communicate and exchange information. For several commands which do not have an alternative to CLI are executed using CLI.

2.2. YANG

YANG is known as yet another next generation. Yang is a data modelling language used to represent both network element configuration and state data. Yang was developed in 2010 and is updated based on the real-world user experience in 2016. The language is protocol agnostic can be converted into any encoding format like XML, JSON. YANG organizes data descriptions into tree structures and includes a type of system that may be extended, a formal separation of state and configuration data, and a range of syntactic and semantic restrictions. Furthermore, YANG allows data modelers to specify the signature of remote procedure calls that may be called on network elements through the NETCONF protocol, as well as the format of event notifications sent by network components.

Yang has its own set of capabilities. It is human readable and simple to understand. In yang the data models use hierarchical configurations. Modules and sub modules provide data modularity. Extensibility can be achieved. YANG is a full, formal contract language with rich syntax and semantics to build applications on.

In the tool the Yang model is used with Netconf Protocol to interact with the remote networking devices through RPCs.

2.3. NETCONF

Netconf is a network configuration protocol. NETCONF is a protocol for managing, configuring, and installing new network device configuration. Netconf first version is developed in 2006 and has gone through many changes till 2012. Simple Network Management Protocol (SNMP) was a very popular network management protocol. The SNMP is mostly used for network fault management and performance management, while its application in configuration management is very limited, especially in system configuration (involving multiple nodes) and service provisioning. The few drawbacks in SNMP urged the need for new alternatives to network management. As a result of this necessity, the Netconf protocol was created.

For protocol communications, NETCONF uses Extensible Markup Language (XML) based on data encoding. On top of a secure transport protocol, protocol messages are exchanged. NETCONF is often sent via SSH utilizing the Netconf sub-system, and it closely resembles the device's native proprietary CLI via SSH interface in many aspects. NETCONF is the (sole) candidate to replace CLI for programmable network configuration management. NETCONF provides the fundamental programming features for comfortable and robust automation of network services.

In the tool Netconf is used to send an RPC to the remote networking device. The steps involved are:

1. A NETCONF client creates an SSH connection with the managed device's NETCONF server
2. The client must specify valid SSH username and password credentials.
3. Messages are used to communicate between the client application and the device.
4. The client sends Remote Procedure Call (RPC) messages to the device, which include standard or operations as well as any vendor-specific operations defined for the device.
5. Within the kind of RPC reply messages, the device answers with the outcomes of the operations.
6. The client application sends a <close session> RPC message to the device once it has done issuing requests and processing responses.
7. The device responds to the RPC request with an <ok> response.
8. Once the device responds with <ok> message to <close-session> the ssh connections on both the ends is closed.

Protocol Operations	Description
<get>	Gets information of device state information and running configuration
<get-config>	Gets a part or complete specified configuration data
<copy-config>	Copies the entire configuration

<delete-config>	Deletes the configuration

Table 2.1: Netconf protocol operations and description

Netconf provides several operations for manipulating configuration devices. The set of basic protocol operations consists of get-config, edit-config, copy-config, delete-config, lock, unlock, get, close-session, kill-session, discard-changes, validate and commit.

2.4. SSH

SSH (Secure Shell or Secure Socket Shell) was created to allow clients to safely establish a connection to a server present in remote location. Secure Shell gives solid passwords and public key verification and moves scrambled information between two PCs that interface over an open organization like the Internet. As well as giving solid encryption, network overseers use SSH widely to distantly control frameworks and applications, permitting them to sign into different PCs over the organization, run orders, and move records starting with one PC then onto the next.

SSH is network protocol that is made up of cryptography as a major concept and the tools required to execute. Client Server model is the sole of SSH. The concept is that a client can safely connect to a server present in a remote location during and after the session. Few applications which yield good results when SSH is applied is during file transfer from one workstation to another and during terminal simulation. While Telnet is still being used today, it too has a lot of security issues. To counter the lack of security features in the aforementioned application SSH was developed. It contains enhanced security which includes strong encryption and decryption while retaining the easy-to-use login and logout feature, initializing and disconnecting sessions which need to run on remote servers. Due to its easy-to-use and user-friendly interface, SSH was able to replace RCP and FTP (File Transfer Protocol).

In the tool SSH plays a vital role in connecting and communicating with the remote devices securely. It helps Netconf to send RPC to remote device and to perform necessary operations on the remote networking device.

2.5. gRPC

gRPC is a language independent high performance Remote Procedure Call (RPC) framework. A client programme can call a method on a server application on another computer as if it were a local object using gRPC, making it easier to construct distributed applications and services. gRPC, like many RPC systems, is built on the concept of creating a service, which includes describing the methods that may be called remotely, as well as their parameters and return types. With pluggable support for load balancing, tracing, health checking, and authentication, it can efficiently link services in and across data centers. It may also be used to link devices, mobile applications, and browsers to backend services in the last mile of distributed computing.

gRPC framework is modern, high-performance, and lightweight. API creation is using Protocol Buffers as the default allows for language independent implementations. Many languages include tools for creating strongly typed servers and clients. Client, server, and bi-directional streaming calls are all supported. Protobuf binary serialization reduces network utilization. Due to the above-mentioned advantages gRPC is deployed on the tool.

2.6. DOCKER

Docker is a container engine that creates containers on top of an operating system by utilizing Linux Kernel capabilities such as name spaces and control groups. Containers are an abstraction that groups code and dependencies together at the app layer. Multiple containers can operate on the same computer and share the operating system kernel, each executing as a separate process in user space. Containers take up less space than virtual machines (container images are often in the tens of megabytes), can accommodate more applications, and need fewer virtual machines.

Docker features a simple approach for transferring an app from a developer's laptop to a test environment and finally to production. It is extremely fast and can operate on any host with a Linux kernel that is compatible with it. (It also works with Windows.) Docker is a technology that benefits both developers and system administrators, and it's a key component of many DevOps (developers + operations) toolchains. For developers, this can be a big relief as they can concentrate on developing code rather than worrying about the system on which it will eventually execute.

The tool can be hosted as a docker application on any Cisco Linux VM. Being platform independent is the major advantage that docker provides due to which it is incorporated in the tool to avoid issues in portability.

2.7. PYTHON LIBRARIES

Few python libraries like Netmiko, Ncclient are used in developing the tool.

2.7.1 Netmiko

Netmiko is a open source python library for interacting with SSH. The Netmiko model provides both client (the system which requests for pages from a remote server, AKA user) and server (the system which holds information needed by a user. Usually placed in a remote location hosting multiple websites in one machine) features. Netmiko is a common alternative to Paramiko, but there are a few key differences: device support and performance. Paramiko supports a small range of devices which results in failure in connection. In addition, it is much slower than Netmiko.

In the tool the Netmiko library is used to connect to the device and to interact with the console of the remote device. This is done by using a command called send_ command whose arguments contain the command to be executed on the console of the remote device. Packages like Connection Handler and file transfer are imported from Netmiko.

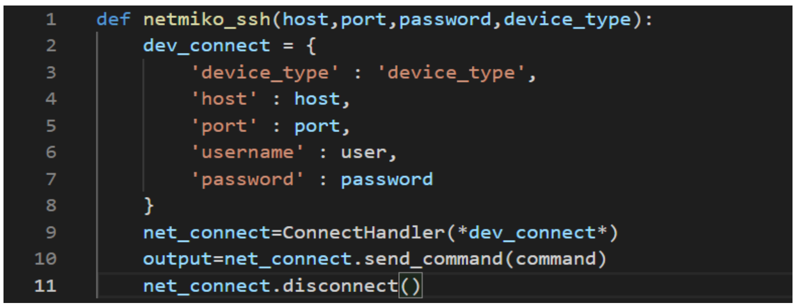

```
def netmiko_ssh(host,port,password,device_type):
    dev_connect = {
        'device_type' : 'device_type',
        'host' : host,
        'port' : port,
        'username' : user,
        'password' : password
    }
    net_connect=ConnectHandler(*dev_connect*)
    output=net_connect.send_command(command)
    net_connect.disconnect()
```

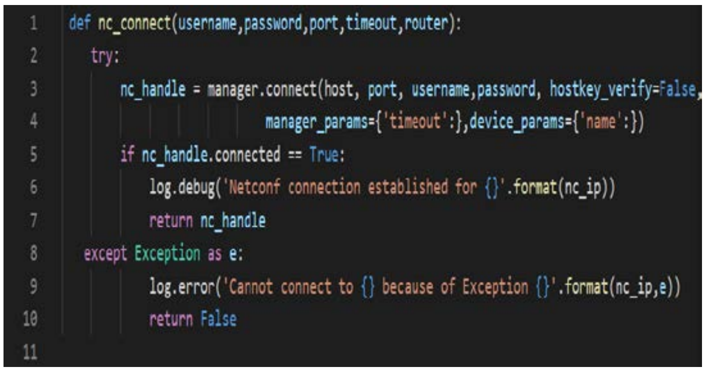
Fig 2.1: Sample Program using Netmiko send_command

2.7.2 Ncclient

As discussed in the previous sections NETCONF protocol provides efficient mechanisms for networking devices. There is a lack of supporting tools and libraries for this protocol. Ncclient is an opensource python library which is developed to support NETCONF in scripting and application development.

The main goal is to provide a straightforward API that intelligently transfers NETCONF's XML-encoded nature to Python constructs and idioms, making it easier to write network-management programmes.

In the tool few packages from ncclient like manager are imported. Ncclient is used to establish the Netconf connection with the remote networking devices.



```python
def nc_connect(username,password,port,timeout,router):
  try:
      nc_handle = manager.connect(host, port, username,password, hostkey_verify=False,
                          manager_params={'timeout':},device_params={'name':})
      if nc_handle.connected == True:
          log.debug('Netconf connection established for {}'.format(nc_ip))
          return nc_handle
   except Exception as e:
          log.error('Cannot connect to {} because of Exception {}'.format(nc_ip,e))
          return False

```

Fig 2.2: Program for Netconf connection using ncclient

3. Overview of The Tool

The tool is a web UI tool which can be used to perform software upgrade/downgrade for all Cisco XR platforms. This can be hosted as docker application on any Cisco Linux VM. The frontend of the tool is developed using React JS. The backend of the tool is developed using python, MySQL, mongo DB. The upgrade will be performed over Netconf, gRPC protocol with GISO image. The tool has enhanced pre-checks and post-checks. This is hosted on several instances across the labs making it a decentralized system but have a centralized database. The action flow in the installation process is shown in figure.

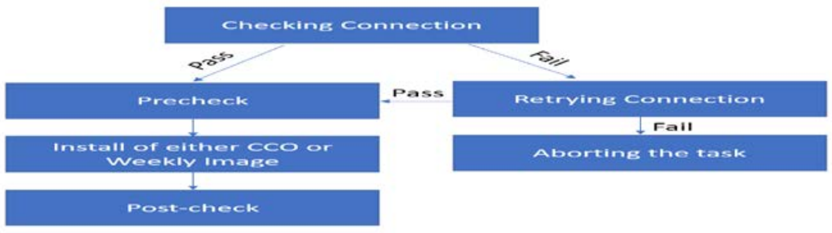


Fig 3.1: Flow diagram of the sequence of actions during install operation

4. Conclusion

With the greater dependency on the datacenters, raised a concern on maintenance of the datacenters. Continuous updation is needed for the data centers to provide efficient services. This includes the upgrading of the networking devices in the data centers. The tool because of its ability to perform automatic software upgrade made the upgrading process simple and effortless. Various technologies used in developing the tool are discussed in brief and an overview is given on how the tool is developed. The tool is developed by Cisco and is currently being used as an inhouse tool at Cisco. Vigorous work is being done on the improvements needed in the tool to improve the customer experience.

REFERENCES

[1] *J. Yu and I. Al Ajarmeh, "An Empirical Study of the NETCONF Protocol," 2010 Sixth International Conference on Networking and Services, 2010, pp. 253-258, doi: 10.1109/ICNS.2010.41.*

[2] *Baozhen Wu and Y. Chang, "Integrating SNMP agents and CLI with NETCONF-based network management systems," 2010 3rd International Conference on Computer Science and Information Technology, 2010, pp. 81-84, doi: 10.1109/ICCSIT.2010.5563913.*

[3] *Hui Xu and Debao Xiao, "Data modeling for NETCONF-based network management: XML schema or YANG," 2008 11th IEEE International Conference on Communication Technology, 2008, pp. 561-564, doi: 10.1109/ICCT.2008.4716122.*

[4] *K. Elbadawi and J. Yu, "Improving Network Services Configuration Management," 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN), 2011, pp. 1-6, doi: 10.1109/ICCCN.2011.6006050.*

[5] *R. Birke, L. Y. Chen and E. Smirni, "Data Centers in the Cloud: A Large Scale Performance Study," 2012 IEEE Fifth International Conference on Cloud Computing, 2012, pp. 336-343, doi: 10.1109/CLOUD.2012.87.*

[6] *Mihăilă, Paul & Balan, Titus & Curpen, Radu & Sandu, Florin. (2017). Network Automation and Abstraction using Python Programming Methods. MACRo 2015. 2. 10.1515/macro-2017-0011.*

[7] *J. Akhtar, "YANG modeling of network elements for the management and monitoring of Elastic Optical Networks," 2015 IEEE International Conference on Telecommunications and Photonics (ICTP), 2015, pp. 1-5, doi: 10.1109/ICTP.2015.7427931.*

[8] *S. Singh and N. Singh, "Containers & Docker: Emerging roles & future of Cloud technology," 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), 2016, pp. 804-807, doi: 10.1109/ICATCCT.2016.7912109.*

[9] *O. Gasser, R. Holz and G. Carle, "A deeper understanding of SSH: Results from Internet-wide scans," 2014 IEEE Network Operations and Management Symposium (NOMS), 2014, pp. 1-9, doi: 10.1109/NOMS.2014.6838249.*

[10] *M. Panjwani and S. De, "Study of Cloud Security in Hyper-scalers," 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom), 2020, pp. 29-34, doi: 10.23919/INDIACom49435.2020.9083727.*