Automation of CISCO SDWAN Controllers Upgrade Process

Siddartha S S
Dept. of Electronics and Communication
RV College of Engineering
Bengaluru, India

S Praveen
Dept. of Electronics and Communication
RV College of Engineering
Bengaluru, India

***Abstract*—With more and more companies utilizing cloud capabilities and adapting hybrid work style for their employees, the traditional Wide Area Network (WAN) networks face the difficulties in terms of complexity and flexibility. Software Defined Wide Area Network (SD-WAN) provides solutions to all these problems with additional capabilities for better connectivity. Cisco SD-WAN provides Graphical User Interface (GUI) support for monitoring and configuring devices remotely. But the manual upgrade process is time expensive and hampers productivity and quick services. The process of data collection pre and post upgrade is tedious and time consuming. This paper focuses on automating the controllers upgrade process in the Cisco SDWAN fabric. The process of automation using Application Programming Interface (API)s is discussed in this work. The aim is also to develop a file comparator that takes in the log files and outputs the discrepancies or differences pre and post update. Results demonstrate that hours of manual work can be reduced to minutes using automation.**

Keywords—SDN; SDWAN; automation; API;Python

I. INTRODUCTION

With the advancement in technology, due to raising demand from customer for better experiences and increased competition, businesses are moving towards embracing digital transformation. It is necessary to adapt to better technologies for delivering increased productivity with reduced cost. As more and more businesses are increasingly adopting cloud services such as Software as a Service (SAAS), Infrastructure as a Service (IAAS), etc., across multiple cloud platforms, the IT department struggles to provide satisfactory experience for business-critical applications.

Software Defined Wide Area Network (SDWAN) offers solutions to problems faced by traditional Wide Area Network (WAN) [10], [11], [13]. SDWAN is a part of Software Defined Network (SDN) technology. The architecture of SDN and SDWAN are shown in the Figure. 1 and Figure. 2 respectively. SDN separates data plane from the control plane. The control plane is centralized to provide a global view of the network for provisioning, configuring, and troubleshooting. It offers a centralized approach to network management. SDWAN is the application of SDN principles for WAN.

To keep up with the networking demands innovation in SDWAN is important. Automation of key processes in deploying, configuring, or upgrading the networks can deliver uninterrupted and seamless performance to the users with less manual intervention by the IT team. Automation helps in reduction of capital and operational expenditures [2], [3]. Cisco offers its own SDWAN architecture with added advantages [4].

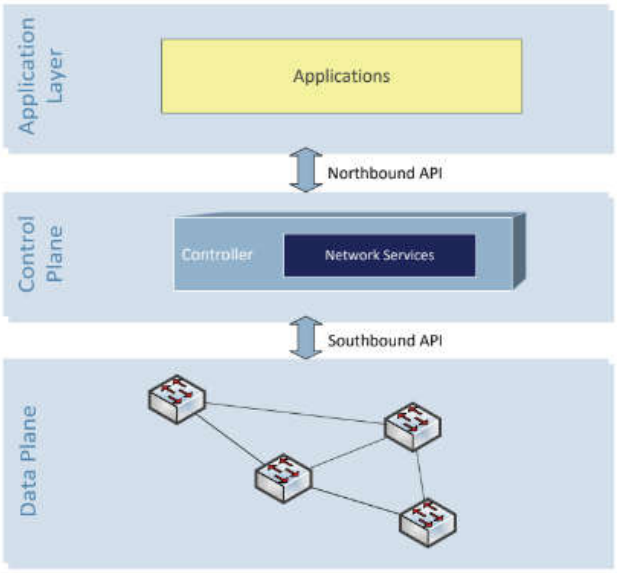
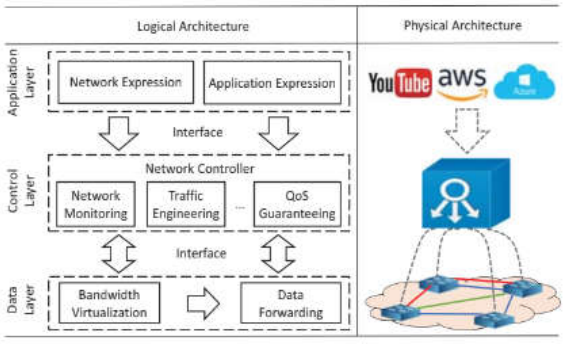


Fig. 1. SDN Architecture [1].

Fig. 2. SDWAN Architecture [10].

Upgrading a Cisco SDWAN overlay is a tedious manual process. The collection of logs pre and post upgrade, pre-check



analysis, post-check analysis and log comparison consumes a lot of time. The availability of APIs for collecting these data can reduce the time expenditure with the help of automation. The proposed work aims to design an automated process for the upgrade of controllers and the comparison of their log files. Python programming language is used to build the software system.

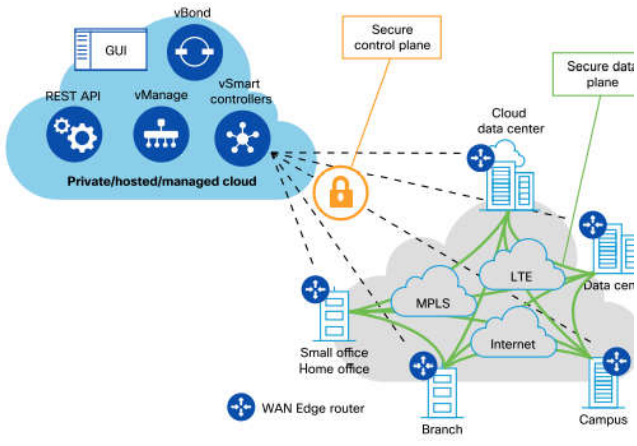
CISCO offers a user-friendly web interface for monitoring, troubleshooting, and deploying the SDWAN network. The deployed network from time to time needs software upgrade to enhance performance or release with bug fixes. It is important to maintain the state of a network pre and post upgrade. Due to the segregation of data and control plane, the control plane upgrade affects the data plane as well. During upgrade there is momentary disconnect between the planes. This may result in changes in the network once the connections are reestablished. Hence it is important to collect the logs and system state information pre and post upgrade and later compare them for discrepancies. But the current method of data collection involves manual process of taking screenshots of the display, running commands, and collecting outputs from Command Line Interface (CLI). This method is time consuming and prone to human error.

CISCO offers well developed REST Application Programming Interface (API)s to vManage [15]. The availability of APIs helps in collection of data through programming. Automating the entire process of pre and post upgrade log collection and comparison can bring down time consumption of the entire process from hours to minutes. This project aims at developing suitable software system architecture to automate the process end to end.

II. CISCO SDWAN ARCHITECTURE [15]

The Cisco SD-WAN architecture is mainly composed of four divisions. The architecture is shown in the Figure. 3.

Fig. 3. Cisco SDWAN Architecture [15].



A. Cisco vManage

Cisco vManage represents the user interface in the management plane. It offers both single tenant and multi-tenant dashboard to fulfill all customer and user requirements. It offers interface for performing provisioning, troubleshooting and monitoring activity for network administrators and operators.

B. Cisco vBond

The orchestration plane is represented by the Cisco vBond. It is responsible for Zero-Touch provisioning process. It is responsible for onboarding the device into the SD-WAN fabric when the device boots up for the first time in an unconfigured state. It facilitates authentication, control/management information distribution, and Network Address Translation (NAT) traversal.

C. Cisco vSmart

The Cisco vSmart resides in the control plane. It acts as the brain of the SD-WAN solution. Policies created on the vManage are enforced onto the routers by the vSmart. It forms Overlay Management Protocol (OMP) tunnels with the routers to share information. When routers come online, they share information with the vSmart. vSmart then decides the routing based on policies and informs the routers. This determines how traffic traverses the SD-WAN fabric.

D. Cisco WAN Egde Routers

These routers reside in the data/forwarding plane. They are responsible for actual forwarding of data across the fabric. The routers can be physical or virtual based on the requirements.

III. MANUAL UPGRADE PROCESS

Before the design of automation process, the manual process needs to be discussed. This is important as it gives the idea as to what processes need to be automated. The manual process also provides the information required to improve the tasks through automation.

In the vManage upgrade process data collection is very important. Data is collected pre and post upgrade. This is called pre-check logs and post-check logs, respectively. These data are used in analysis to determine the state of the system before and after upgrade. Any differences arising in the data are detected as errors and the network is debugged with necessary measures.

A. Pre check logs

These logs are collected before upgrade. Pre check logs contain the state of the system before upgrade. During the upgrade process the connection between the controllers and edge devices are terminated. After the upgrade once the system reboots the connections are reestablished. The pre check logs help us in detecting any changes in these connections. The information collected during in pre check logs are vManage Dashboard information, device and controllers count, control connections, software version, local history of vManage, template count, certificate count, control status of devices, templates and certificates assigned to each device, etc.,

Pre check logs are usen in pre check log analysis. This analysis is done to determine whether the system is fit for an upgrade or not. Certain conditions are evaluated on the data to discover any underlying errors. Some conditions include checking the last updated date of devices. The devices need to be up to date for proper functioning. If the last updated date does not match current date, then it means the device holds old information. If the system process with upgrade in this condition, the device may not reboot after the upgrade process or the device may show erroneous data after upgrade. Another condition to check for is the CPU and memory utilization of the controllers prior to upgrade. If the CPU utilization is higher than 80% it means some heavy lifting is being done by the control

plane. Proceeding for a software upgrade in this condition results in loss of data.

If the pre check analysis is success, then software upgrade of the controllers is performed. If it is a failure, then necessary debussing is done and then upgraded.

B. Post check logs

These logs are collected after upgrade. Post check logs contain the state of the system after upgrade. Data collected during the pre-check logs are also collected during post check logs. These logs are collected once the system reboots and all the controllers are upgraded without errors. These logs help in comparison with pre check logs.

C. Comparision

After the upgrade, pre and post check logs are compared. The comparison is done to detect any changes in the system state pre and post upgrade. Control connections, certificate and templates of the devices and some other parameters needs to be same. If any differences is found corresponding debugging steps are taken.

IV. AUTOMATED UPGRADE PROCESS

Automation is done with the help of APIs. REST APIs can be used to collect data by sending HTTP requests and getting response. Cisco vManage has well developed APIs with corresponding documentation. This can be used to automate data collection and with the help of Python a new comparator can be designed.

Python is a general-purpose scripting language. It has well developed libraries to handle communication, send http requests and process http response. The following libraries are used during this project:

1. requests: This is one of the HTTP libraries of Python. It hides the complexities and makes it easy for the user to send requests and receive responses.

2. json: This library helps in parsing JavaScript Object notation (JSON) strings. It helps in inter conversion of JSON to dict or list for easy usage in Python code. Since the API response is in JSON format, this library helps in parsing it to extract data.

3. difflib: This library helps user in comparison of different kinds of data format. It gives outputs in various formats for easy readability indicating differences in given input data.

Other libraries which help in some lower-level tasks are datetime, tabulate, etc.,

A. Specifications of the Design

The software system is designed to perform different tasks during upgrade process. The tasks are collection of pre and post upgrade data, pre-check log analysis and pre and post check logs comparison. The specifications for these tasks are as follows:

1. API list: The list of APIs affecting the data must be discovered. These APIs must be collected and used as source of data for the project. The Cisco documentation for APIs is shown in the Figure. 4.

2. API response: The software system must be able to securely login to the vManage. The system must be able to send API requests and get responses.

3. Pre-check log analysis: The software system must be capable of detecting any discrepancies in the pre-check log to indicate success/failure to proceed to software upgrade of controllers.

4. Comparator: A smart comparator needs to be implemented that can ignore unwanted information and compare only the important parameters. This comparator must be programmable to include new parameters as the data is scaled.

B. Automation of Data Collection

Data collection is automated with the help of vManage APIs. First step is to collect the list of APIs that carry necessary data in their response. These APIs can be collected using browser inspect tools or postman software. The working and response structure of the APIs are documented after collection. A documentation is maintained about the collected APIs along with the information to be extracted from their response. The APIs are sequentially fired to get the necessary response. These responses are collected and stored as a consolidated JSON file. This file can be filtered to extract required data. A script for filtering JSON file is developed. Based on the data needed, the corresponding key value pairs are extracted from JSON response. The filtered data is stored in a text format.

C. Comparator Design

The available comparators use difference method to compare two files. The differences are indicated in the result with the help of position markers, either in the form of special characters or color coding. But comparison of parameters that have no significant effect on the result are of no use. Due to the high volume of data stored in the log files, it becomes difficult to filter out unwanted comparison from the result file. This again results in manual work and is time and effort consuming.

A new comparator is designed for the automation process. This comparator is programmed to ignore certain keywords in the parameters that are unwanted. Difference of log files is generated using difflib using Python language. From the result, the markers are erased for parameters that are of less importance. Due to the removal of these markers the final output will have markers only on important parameters. The user will now only have to check for differences in these parameters if it is indicated by the result. This reduces significant time of the user in manual filtering of result data.

D. Working

The flow diagram is shown in the Figure. 4. The automation process begins with executing the script to collect API response. This script helps to login to vManage and send requests to it. The received responses are then consolidated and stored in a single JSON file. At first pre check logs are collected before upgrade. This pre check log is passed to pre check log analysis script for checking previously discussed conditions. If the log file passes all the conditions, then the software upgrade of controllers s performed. If it fails, then necessary debugging steps are taken out. After debugging again pre check analysis is performed for success or failure. Once the software upgrade of controllers is successful, the same script used for API response collection can be used for post check logs. Both pre and post check log files are passed through filtering script to extract required data. This data will be in pre and post check text format. These text files are given as input to the designed comparator. The parameters to be ignored are loaded into the comparator. Once the comparison is completed, a result file is generated with color markings to

indicate differences in pre and post check files. Based on the differences, either the upgrade is termed as success or debugging is done to fix and post upgrade errors.

Fig. 4. Flow diagram of the software system.

V. RESULTS

To test the working of the script, a SDWAN system is upgraded using automated process. After the pre and post check log collection, the collected data is compared with the data on vManage GUI. It is found that the automation script collects all these data within few seconds. This saves both effort and time as no longer screenshots of the GUI are needed. The collected JSON logs are filtered to successfully produce text file containing necessary information for comparison. The new comparator is tested on these text files. Figure. 5 shows the comparison result of available comparators. Left half is pre check log and right half is post check log. The parameter lastupdated is not important in comparison. The result generated from new comparator is shown in the Figure. 6. It is evident from the result that the comparator can ignore unwanted parameters.

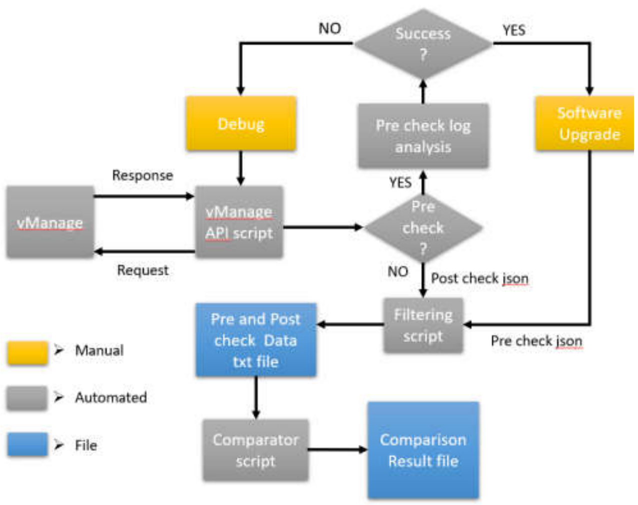


Fig. 5. Pre and post check log comparison in regular comparator.

```
lastupdated              lastupdated
-------------------      -------------------
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
2022-06-29 14:22:40      2022-06-29 17:44:54
```

Fig. 6. Pre and post check log comparison in designed comparator.

```
lastupdated               lastupdated
-------------------  -  -------------------
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
2022-06-29 14:22:40     2022-06-29 17:44:54
```

VI. CONCLUSION

SD-WAN is the next generation communication solution which provides solution to problems in traditional WAN. CISCO implements SD-WAN solution with its own architecture. This SD-WAN overlay architecture provides a GUI for performing network related activities. This GUI interface helps in easy deployment, troubleshooting and configuring the networks. But when it comes to upgrading the network controllers, the manual process is very tedious and time consuming. From the data collection to comparison this process is time expensive and prone to human error. Hence to mitigate these issues a solution is provided through automation. Automation of these tasks performed during upgrade can bring down time consumption from hours to mere minutes. The project is aimed at achieving automation of pre and post-check logs collection. These are used to analyze the state of the network to take decisions such as whether to upgrade or not. Due to the availability of vManage APIs the aim is to use those to provide automation solution. Using APIs to collect data through Python script can speed up the upgrade process. The project also aims at building a comparator for pre and post-check logs comparison. This comparator must be capable of intelligently comparing important parameters. Finally, the script needs to perform upgrade process of a SDWAN overlay to verify its validity. A software system is designed using Python

programming language to achieve the mentioned objectives. The software system is divided into four parts, each handling one task during upgrade process. At the beginning, with the help of browser inspect tools the web APIs required to extract data are discovered. These APIs are used to send request to vManage and get response. These responses are filtered using the filter script to extract the required data. This achieves the objective of automating pre and post-check log collection. The pre-check log analysis script analyzes the pre check logs to indicate whether to upgrade or not. A comparator is designed to perform intelligent comparison by ignoring unwanted parameters. To test the script an upgrade of a SD-WAN overlay is performed to validate different test cases. From filtered data the information hosted on the GUI is collected into the output file. This eliminates the manual task of taking screenshots of data. The pre-check log is analyzed and devices holding old data are recorded into output file. The CPU utilization details recorded helps in debugging. The designed comparator is seen to be effective in ignoring unwanted data and comparing only the important data.

ACKNOWLEDGMENT

This work is possible because of the technical support and expertise of Telstra. We also acknowledge the suggestions and support provided by RV College of Engineering.

REFERENCES

[1] P. Danielis, V. Altmann, J. Skodzik, E. B. Schweissguth, F. Golatowski, and D. Timmermann, "Emulation of sdn-supported automation networks," in 2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA), 2015, pp. 1–8. doi: 10.1109/ETFA.2015.7301517.

[2] J. Rossavik, "The Business Benefits of Automation and Orchestration White Paper," CISCO, Tech. Rep., Jan. 2022.

[3] D. M. H. Mortensen, "Economic Benefits of Network Automation," Cisco and ACG Research, Tech. Rep., Jan. 2020.

[4] S. Mosher and C. Hill, "Cisco Software-Defined WAN for Secure Networks," Cisco, Tech. Rep., Mar. 2020.

[5] C. Hill, "Site-to-Any-Cloud and Site-to-Site Use Cases with Cisco SD-WAN and Equinix," Cisco, Tech. Rep., Sep. 2020.

[6] J. Hodes, "SD-WAN Implementation Differentiation Layer Strategies," Juniper Networks, Tech. Rep., Feb. 2017.

[7] J Crawshaw, "The Integration Challenges of Software-Defined Virtualized Enterprise Networking," Ekinops, Tech. Rep., Apr. 2018.

[8] R. P M and D. .M, "A study of software defined networking with openflow," International Journal of Computer Applications, vol. 122, pp. 5–12, Jul. 2015. doi: 10.5120/21694-4798.

[9] P. B. Patil, K. S. Bhagat, D. K. Kirange, and S. D. Patil, "Software defined networks using mininet," 2020.

[10] Z. Yang, Y. Cui, B. Li, Y. Liu, and Y. Xu, "Software-defined wide area network (sd-wan): Architecture, advances and opportunities," in 2019 28th International Conference on Computer Communication and Networks (ICCCN), 2019, pp. 1–9. doi: 10.1109/ICCCN.2019.8847124.

[11] S. Troia, L. M. M. Zorello, A. J. Maralit, and G. Maier, "Sd-wan: An open source implementation for enterprise networking services," in 2020 22nd International Conference on Transparent Optical Networks (ICTON), 2020, pp. 1–4. doi: 10.1109/ICTON51198.2020.9203058.

[12] S. Troia, L. M. Moreira Zorello, and G. Maier, "Sd-wan: How the control of the network can be shifted from core to edge," in 2021 International Conference on Optical Network Design and Modeling (ONDM), 2021, pp. 1–3. doi: 10.23919/ ONDM51796.2021.9492375.

[13] A. Awasthi, "Sdwan (software defined-wan) network engineering and project management," Semiconductor Science and Information Devices, vol. 2, May 2020. doi: 10.30564/ssid.v2i1.1870.

[14] V. Lanka and H. K. Mohan, "Network in a box - automation of secure sd-wan with service chaining in juniper nfx," in 2019 11th International Conference on Communication Systems Networks (COMSNETS), 2019, pp. 527–529. doi: 10. 1109/COMSNETS.2019.8711426.

[15] A. Rohyans, A. Shaikh, C. B. Rajaram, and D. Klebanov, Cisco SD-WAN Cloud scale architecture. CISCO, 2019, isbn: 978-0072380323.