A Web Driven SDN Orchestrator For The Provisioning of ACI Fabric and Lab Infrastructure

Matthew Gaynor

922830

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Bachelor of Science
of the
University of Portsmouth.

School of Computing
Engineering Project

2023

Declaration

No portion of the work contained in this document has been submitted in support of an application for a degree or qualification of this or any other university or other institution of learning. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Date: 2023

Abbreviations

ACI	Application Centric Infrastructure
CLI	Command Line Interface
CX	Customer Experience
DMZ	Demilitarised Zone
DNS	Domain Name System
DPG	Distrubited Port Group
DVS	Distributed Virtual Switch
FEX	Fabric Extender
NOS	Network Operating System
NTP	Network Time Protocol
RADIUS	Remote Authentication Dial-In User Service
SVS	Solution Validation Services
ToR	Top of Rack
VPN	Virtual Private Network
WAN	Wide Area Network

Abstract

The purpose of this project is to provide a solution to CX Labs UK within Cisco Systems LTD that will streamline the operation of DMZ lab-space. The application shall allow lab staff to prepare network infrastructure for an incoming project with little to no CLI interaction and it shall be web-based. ACI, VMware ESXi and vCenter shall be used to manage the networking and virtual machines respectively. The lab space will be represented by a small testbed containing the minimum quantity of networking equipment and compute resources required to accurately simulate the environment.

Acknowledgements

I would like to thank Cisco, my manager and good friend, Dave Smith, for his close help in allowing me to use equipment necessary for testing and developing the solution.

Consent to Share

I consent / do not consent for this project to be archived by the University Library and potentially used as an example project for future students.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Client

The client is CX Labs UK within Cisco Systems. CX Labs provides lab space for use by business units internal to the company. Most of the space is used for the testing of customer networks by SVS (Solution Validation Services). SVS provide bespoke testing services to customers wishing to use Cisco's expertise to test a range of situations, from regression and firmware testing to full upgrade and migration plans.

In order to match the customers environment as close as possible, a scaled down version of the customers network is usually recreated in the lab space managed by CX Labs. CX Labs hold many devices that cover most of the Cisco portfolio, which allows for the recreation of most networks. Most of this lab space is hosted within the internal Cisco corporate network, which requires any users to be employees of Cisco in order to access testbeds. More and more customers however are requesting remote access to their testbeds. To facilitate this, a fully isolated DMZ environment is provided, which allows direct WAN connectivity to a testbed, allowing for a VPN tunnel to be established and hence remote access granted to a testbed from any location to any permitted person.

1.2 The Problem

Currently the solution for the DMZ network infrastructure consists of 4 Nexus 9K devices, with FEXs for RJ45 connectivity and 2960S ToR switches. Whilst this solution is functional and works, the major downside is that there is no automation or configuration management solution deployed. This means over time, configuration drift occurs as manual changes are made, but not made the same to all switches. Unused VLANs are also not removed from the switches which leads to bloat and a larger configuration than is required. The same situation also occurs on vCenter with the configuration of the DPGs within the DVS where unused DPGs are never removed or are labelled incorrectly.

The manual configuration of the required routing and VPN termination as well as other lab requirements such as a NTP, DNS and RADIUS all take a lot of time to configure. This time could be better utilised, such as preparing the physical rack space for the racking and stacking of new equipment.

1.3 Aims and Objectives

The aim of this project is to provide an easy to use dashboard that allows CX Labs to easily onboard and manage projects. This will be achieved by providing a web based interface that allows the user to create a new project, which will then automatically create the required configuration within ACI, vCenter, and other associated network infrastructure. Management of existing projects will also be supported, operations such as expanding a projects rackspace utilisation, or removing a project from the environment will be supported. The dashboard will also provide a view of the current projects utilisation of the lab space, as well as the current utilisation of the lab space as a whole.

1.3.1 Deliverables

- A web based dashboard that allows the user to manage and provision projects
- User guides
  - User guide for the dashboard
  - User guide for how to provision ACI, vCenter and other associated network infrastructure so that it will be compatible with the automation solution
- A report detailing the design and implementation of the solution

1.4 Constraints

Due to the fact that the testbed utilised for testing and development is in a remote datacenter, physical access to the testbed will be limited. This may result in delays if a problem with the physical infrastructure occurs which could lead to the project running over time. The solution will be designed to be as resilient as possible, but there is a chance that the solution may not be able to recover from a failure. There may also be a chance that access to the testbed is revoked, which would result in the project being delayed until access is restored.

External factors such as the availability of the testbeds internet uplink and power supply may also have an impact on the projects delivery within the required timescale.

1.5 Evaluation

To ensure that the project delivers on the required functionality, the following evaluation criteria will be used to evaluate the project:

- Does the dashboard allow for the easy creation, management and deletion of lab projects?

- Does the dashboard allow for a logical view of the entire lab space as a whole?
- Does the dashboard show project status and health of the associated devices?
- Does the automation platform create a schema in ACI that is easy to follow and troubleshoot should a problem arise?

Chapter 2

Literature Review

The aim of this literature review is to research and analyse existing solutions, documentation and research on the automation of networking infrastructure. To ensure this review is of maximum usefulness, it will also involve analysing best practises and standards when developing software and automation solutions. This will allow for the optimisation of the planning and implementation stages of the project that will subsequently follow.

Sources for this review will be from relevant books, online websites, professional publications and Request for Comments. Multiple services will be used to identify and obtain sources that can be used for the purposes of this review.

- Google Scholar
- IEEE Explore
- ResearchGate
- University of Portsmouth Library

Research was conducted that was related to the following set of topics:

- What is software defined networking?
- What types of software defined networking exist?
- What are the advantages and disadvantages?

2.1 Definition of Software Defined Networks

All industry experts and academics define software defined networking similarly, that is providing automation and intelligence to networks via the means of software and APIs. Kreutz et al. (2015) state that “SDN was originally coined to represent the ideas and work around OpenFlow at Stanford University”.
SDN is often defined using four pillars, Kreutz et al. (2015) define these as the following:

1. “The control and data planes are decoupled. Control functionality is removed from network devices that will become simple (packet) forwarding elements”
2. “Forwarding decisions are flow based, instead of destination based. A flow is broadly defined by a set of packet field values acting as a match (filter) criterion and a set of actions (instructions)”
3. “Control logic is moved to an external entity, theso-called SDN controller”
4. “The network is programmable through software applications running on top of the NOS that in-teracts with the underlying data plane devices.”

2.2 Overview of Software Defined Networks

Since this project is centred around software defined networking and the automation of these networks, it is critical to ensure that the principles and their method of operations are understood.
An official survey paper from the IEEE that analysed the state of SDN provides a good explanation as to why the need for network programmability arose in the first place. Conventionally, “Computer networks are typically built from a large number of network devices such as routers, switches and numerous types of middleboxes” Nunes et al., 2014. Nunes et al. (2014) go on to state that due to the large amount of manual configuration required to achieve the desired traffic flow, “network management and performance tuning is quite challenging and thus error-prone”. This leads to one of the solutions, Software Defined Networking. “Software Defined Networking (SDN) is a new networking paradigm which the forwarding hardware is decoupled from control decisions. It promises to dramatically simplify network management and enable innovation and evolution”. Software defined networking “is designed to use standardized application programming interfaces (APIs) to quickly allow network programmers to define and reconfigure the way data or resources are handled within a network.” Kirkpatrick, 2013

2.3 Types of Software Defined Networks

Software Defined Networking is a blanket term for any solution that tackles conventional networking with a programmatic view. There are “the two main delivery models: Imperative and declarative” CDW (2015). CDW (2015) go on to state that Imperative SDN is where “A centralized controller (typically a clustered set of controllers) functions as the network’s ‘brain’” and that declarative SDN is where “the intelligence is distributed out to the network fabric. While policy is centralized, policy enforcement isn’t”. CDW (2015) give “a protocol such as OpenFlow explicitly telling network switches precisely what to do and how to do it” to be an example of imperative SDN.

Chapter 3

Methodology

Here the methodology used to develop and test the software will be detailed. This will include the tools used, the development process and the testing process. Using the appropriate methodology is key to the success of a project and will ensure that the required features are implemented within the required timescale.

3.1 Development

Whilst there are many software development approaches, Kanban was chosen as the best methodology because only one person will be working on the development of the software, not a team of developers. Kanban is a simple and effective way to manage a single person's workload and is a good fit for this project. A solution such as Trello, Atlassian, 2023, can be utilised as it is free for small teams and is quick and easy to use. The Trello board that will be used for development is shown in figure 3.1.

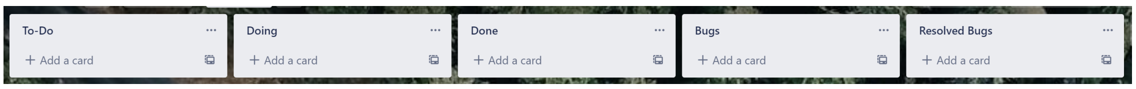

Figure 3.1: Trello Kanban board used to manage the development process

The project requirements will be broken down into smaller tasks, and each task will be added to the Trello board. The tasks will be added to the "To Do" column and then moved to the "Doing" column when the task is being worked on. Once the task is complete, it will be moved to the "Done" column. This will allow the project to be broken down into smaller tasks and will allow the project to be managed effectively. The tasks will be added to the board in the order that they are required to be completed, and will be worked on in that order. This will ensure that the project is completed in the correct order and will ensure that the project is completed in the required timescale. There are also 2 other lists, namely "Bugs" and "Resolved Bugs" which will allow for the ease of tracking bugs and ensuring that they are resolved during the development cycle.

3.2 Time Management

To ensure that the project keeps to time, a Gantt chart is to be used which will outline the key milestones of the project and when they should be met. To generate this chart, teamgantt, 2023, was used as it has a free tier and meets all of the requirements for this project. Figure 3.2 shows the Gantt chart for the project.

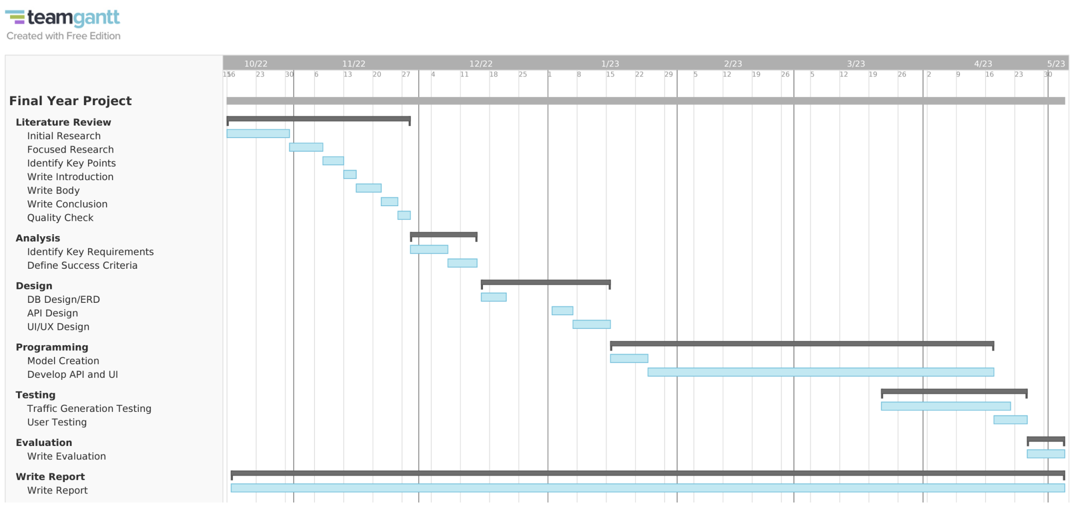

Figure 3.2: Gantt chart used to outline the key project milestones

As the project will follow the Kanban development methodology, some stages overlap and will occur simultaneously. This is because testing will be critical to influencing the development process and ensuring that features work correctly as they are developed. The tight timescale of this project also means that tight features be tested as they are written to prevent issues later in the project.

3.3 Project Management

To ensure secure storage, accessibility and history of code as it is written, Git will be used to manage the project. Git is a version control system that allows for the tracking of changes to code and the ability to revert to previous versions of the code. GitHub will be used to host the code and will allow for the code to be accessed from anywhere, GitHub also has a free tier which will be used for this project. The GitHub repository for this project can be found at https://github.com/mattg66/fyp.

Chapter 4

Requirements

This chapter will provide in more detail the requirements of the automation platform. The requirements will be split into 2 sections, namely the functional requirements and the non-functional requirements. The functional requirements will outline the features that the platform must have, and the non-functional requirements will outline the requirements that the platform must meet in order to be successful. In order to prioritise the requirements, the MoSCoW method will be used. This method will allow for the requirements to be prioritised and will ensure that the most important requirements are met first. The MoSCoW method is a prioritisation method that splits requirements into 4 categories, namely Must Have, Should Have, Could Have and Won't Have.

4.1 Functional Requirements

ID	Details	Priority
FR1	Visual representation of rack space	Must Have
FR2	Add and remove racks from the space	Must Have
FR3	Add and remove Terminal Servers from racks	Must Have
FR4	Add and remove Fabric Nodes from racks	Must Have
FR5	Add, remove and update projects	Must Have
FR6	Expand or contract a projects consumption of rack space	Must Have
FR7	Automate configuration of ACI fabric	Must Have
FR8	Deploy virtual router using vCenter API	Must Have
FR9	Deploy virtual services stack to provide remote access VPN	Could Have
FR10	Continuous monitoring of ACI and vCenter health	Won't Have
FR11	Terminal server automated management	Must Have
FR12	Login system to restrict access	Could Have

Table 4.1: Functional Requirements

4.2 Non-Functional Requirements

ID	Details	Priority
NFR1	Must be easy to use for staff with less technical knowledge	Must Have
NFR2	The system status should be easily visible to staff (e.g. errors, project status)	Must Have
NFR3		

Table 4.2: Non-Functional Requirements

References

Atlassian. (2023). *Trello brings all your tasks, teammates, and tools together*. Retrieved January 3, 2023, from https://trello.com/

CDW. (2015). The future of networking arrives. *Commun. ACM*, 16–19. https://webobjects.cdw.com/webobjects/media/pdf/CDWCA/White-Paper-The-Future-of-Networking-Arrives-MKT2862CA.pdf.

Kirkpatrick, K. (2013). Software-defined networking. *Commun. ACM*, *56*(9), 16–19. https://doi.org/10.1145/2500468.2500473

Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, *103*(1), 14–76. https://doi.org/10.1109/JPROC.2014.2371999

Nunes, B. A. A., Mendonca, M., Nguyen, X.-N., Obraczka, K., & Turletti, T. (2014). A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials*, *16*(3), 1617–1634. https://doi.org/10.1109/SURV.2014.012214.00180

teamgantt. (2023). *Teamgantt is the refreshing solution that brings project scheduling software online*. Retrieved January 3, 2023, from https://trello.com/