**Final Year Project**

**COMP60011**

**Security of network devices using Automation**

Name: Conrad Guy

Number: 21020093

Supervised by: Thomasz Bosakowski

Date : 02/05/2025

Contents

[1 Introduction 5](#_Toc197088828)

[1.1 Background 5](#_Toc197088829)

[1.2 Project Description 5](#_Toc197088830)

[1.3 Project Goals and objectives 6](#_Toc197088831)

[1.4 Project aim 6](#_Toc197088832)

[1.5 Hypothesis 6](#_Toc197088833)

[1.6 Deliverables 6](#_Toc197088834)

[1.7 Risks 6](#_Toc197088835)

[1.8 Ethics form 6](#_Toc197088836)

[2 Methodologies and methods 6](#_Toc197088837)

[2.1 Methods vs Methodologies what makes them different? 7](#_Toc197088838)

[2.1.1 Methodologies 7](#_Toc197088839)

[2.1.2 Agile 7](#_Toc197088840)

[2.1.3 Waterfall 8](#_Toc197088841)

[3 Research methodology 9](#_Toc197088842)

[3.1 Data collection techniques 9](#_Toc197088843)

[3.1.1 Experimental research 9](#_Toc197088844)

[3.2 Research onion 10](#_Toc197088845)

[3.2.1 Layer 1 – Philosophy 10](#_Toc197088846)

[3.2.2 Layer 2 – Approach 10](#_Toc197088847)

[3.2.3 Layer 3 - strategy 10](#_Toc197088848)

[3.2.4 Layer 4 – Choices 11](#_Toc197088849)

[3.2.5 Layer 5 – Time Horizon 11](#_Toc197088850)

[3.2.6 Layer 6 – Techniques and procedures 11](#_Toc197088851)

[4 Literature review 12](#_Toc197088852)

[4.1 Network Management Types 12](#_Toc197088853)

[4.1.1 In-Band Network Management 12](#_Toc197088854)

[4.1.2 Out-of-band Network Management 14](#_Toc197088855)

[4.2 Communication Protocols 15](#_Toc197088856)

[4.2.1 SSH 15](#_Toc197088857)

[4.2.2 Telnet 16](#_Toc197088858)

[4.2.3 HTTP 16](#_Toc197088859)

[4.2.4 NETCONF 17](#_Toc197088860)

[4.2.5 RESTCONF 17](#_Toc197088861)

[4.3 Security Configurations 17](#_Toc197088862)

[4.4 Automation Tools 23](#_Toc197088863)

[4.4.1 Ansible 23](#_Toc197088864)

[4.4.2 NAPALM 25](#_Toc197088865)

[4.4.3 Netmiko 27](#_Toc197088866)

[5 Artifacts Design 28](#_Toc197088867)

[5.1 Netmiko 29](#_Toc197088868)

[5.2 Script 1 – Management plane hardening 29](#_Toc197088869)

[5.3 Script 2 – control and management plane hardening 30](#_Toc197088870)

[5.4 Script 3 – backup file and configuration file 30](#_Toc197088871)

[5.5 Script 4 – output file configurations and update the startup configuration 30](#_Toc197088872)

[5.6 Script 5 – Unit test script for management plane 30](#_Toc197088873)

[5.7 Why use multiple scripts? 30](#_Toc197088874)

[6 Implementation 31](#_Toc197088875)

[6.1 Script 1 31](#_Toc197088876)

[6.1.1 Connect to a device 31](#_Toc197088877)

[6.1.2 Device configurations 32](#_Toc197088878)

[6.2 Script 2 35](#_Toc197088879)

[6.3 Script 3 35](#_Toc197088880)

[6.4 Script 4 37](#_Toc197088881)

[6.5 Script 5 38](#_Toc197088882)

[7 Testing 39](#_Toc197088883)

[7.1 Test 1 - Connecting the Devasc VM and the Cisco router VM together 39](#_Toc197088884)

[7.2 Test 2 – running Script 1 40](#_Toc197088885)

[7.2.1 Testing the enable and username passwords 40](#_Toc197088886)

[7.2.2 Testing script speed 41](#_Toc197088887)

[7.2.3 Testing AAA 42](#_Toc197088888)

[7.2.4 Testing SSH 42](#_Toc197088889)

[7.3 Test 3 – running script 2 43](#_Toc197088890)

[7.3.1 Testing ACLs 43](#_Toc197088891)

[7.4 Test 4 – running script 3 43](#_Toc197088892)

[7.4.1 Backup file test 44](#_Toc197088893)

[7.4.2 Configuration file test 45](#_Toc197088894)

[7.5 Test 5 – running script 4 46](#_Toc197088895)

[7.5.1 Using the configuration to output configuration commands 46](#_Toc197088896)

[7.5.2 Updating the startup configuration 46](#_Toc197088897)

[7.6 Test 6 – running script 5 47](#_Toc197088898)

[7.7 Hypothesis vs test results 47](#_Toc197088899)

[8 Conclusion 47](#_Toc197088900)

[9 References 48](#_Toc197088901)

[10 Appendix 52](#_Toc197088902)

[10.1.1 Ethics form 52](#_Toc197088903)

[10.1.2 Gannt charts 54](#_Toc197088904)

[10.1.3 Risk Assessment table 55](#_Toc197088905)

[10.1.4 Log books 56](#_Toc197088906)

[10.1.5 Appendix - Show run comparison 56](#_Toc197088907)

[Figure 1 - Agile sprint source - Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017) 9](#_Toc197094107)

[Figure 2 - waterfall process source - Senarath, U.S. (2021) 10](#_Toc197094108)

[Figure 3 - Flow of traffic for a client login source - Nevis (2016) 13](#_Toc197094109)

[Figure 4 - In-Band Network diagram source - Nevis (2016) 13](#_Toc197094110)

[Figure 5 - In-Band Network Telemetry source AlSabeh a et al. (2022) 14](#_Toc197094111)

[Figure 6 - Out-Of-Band Network Management source - Nevis (2016) 15](#_Toc197094112)

[Figure 7- Network Telemetry source - Lizhuang, T. et al. (2020) 16](#_Toc197094113)

[Figure 8 - Ansible Automation source - Ratan, A. (2018 24](#_Toc197094114)

[Figure 9 - Inventory example source - Ratan, A. (2018 25](#_Toc197094115)

[Figure 10 - Napalm vs ansible source - Okasha, K. (2020) 27](#_Toc197094116)

[Figure 11 - network Design 29](#_Toc197094117)

[Figure 12 – connecting to device 32](#_Toc197094118)

[Figure 13 - password and send\_config\_set "def" statements 33](#_Toc197094119)

[Figure 14 - AAA command list for configuration 34](#_Toc197094120)

[Figure 15 - main() function calling all other function in script 35](#_Toc197094121)

[Figure 16 - main() function performance checking code 35](#_Toc197094122)

[Figure 17 - connects to the router 36](#_Toc197094123)

[Figure 18 - backup script 36](#_Toc197094124)

[Figure 19 - clean configuration file from backup file 37](#_Toc197094125)

[Figure 20 - running code to change the startup configuration 38](#_Toc197094126)

[Figure 22 - unit test imports and class 39](#_Toc197094127)

[Figure 23 - test functions 39](#_Toc197094128)

[Figure 24 - ping test to check the router is reachable 40](#_Toc197094129)

[Figure 25 - SSH connection issues 40](#_Toc197094130)

[Figure 26 - SSH successful login 41](#_Toc197094131)

[Figure 27 - the enable password testing 41](#_Toc197094132)

[Figure 28 - testing username bob configuration 42](#_Toc197094133)

[Figure 29 - Admin101 configuration test 42](#_Toc197094134)

[Figure 30 - script 1 execution time average 42](#_Toc197094135)

[Figure 31 - show aaa method-list all (authentication) 43](#_Toc197094136)

[Figure 32 - show aaa method-list all (authorization) 43](#_Toc197094137)

[Figure 33 - show aaa method-list all (accounting) 43](#_Toc197094138)

[Figure 34 - show ssh 43](#_Toc197094139)

[Figure 35 - script 2 speed test 44](#_Toc197094140)

[Figure 36 - show access-list 44](#_Toc197094141)

[Figure 37 - shows the creation of the two files 45](#_Toc197094142)

[Figure 38 - shows the backup file and its contents 45](#_Toc197094143)

[Figure 39 clean configuration file test 46](#_Toc197094144)

[Figure 40 - configuration from file script test 47](#_Toc197094145)

[Figure 41 - update the startup config test 47](#_Toc197094146)

[Figure 42 - Ethics form 55](#_Toc197094147)

[Figure 43 - original Gannt Chart 55](#_Toc197094148)

[Figure 44 - updated Gannt Chart 56](#_Toc197094149)

[Figure 45 - risk assessment part A 56](#_Toc197094150)

[Figure 46 - Risk Assessment part B 57](#_Toc197094151)

[Figure 47 - log book 1 57](#_Toc197094152)

[Figure 48 - log book 2 58](#_Toc197094153)

[Figure 49 - log book 3 58](#_Toc197094154)

[Figure 50 - week 6 59](#_Toc197094155)

[Figure 51 - week 6 59](#_Toc197094156)

[Figure 52 - week 8 60](#_Toc197094157)

[Figure 53 - week 21 61](#_Toc197094158)

[Figure 54 - week 22 61](#_Toc197094159)

[Figure 55 - week 23 62](#_Toc197094160)

[Figure 56 - week 24 62](#_Toc197094161)

[Table 1 - management plane basic configurations source - Cisco (2024), Cisco (2007), Cisco (2015) 19](#_Toc197089443)

[Table 2 - control plane basic configurations source - cisco(2024) 21](#_Toc197089444)

[Table 3 - data plane basic configurations source - cisco(2024) 22](#_Toc197089445)

[Table 4 - Ansible Components 25](#_Toc197089446)

[Table 5 - Network table 29](#_Toc197089447)

# Introduction

## Background

Leiva, N. and Kashin, M. (2023) explains that network automation for network management is a relatively new field with traditional network management including directly running Command Line Interface (CLI) configurations monitoring commands.

The data used is unstructured, this traditional method is still widely used however with its unstructured nature of CLI driven operations, it creates issues with scalability as commands will have to be manually edited on each device and are error prone due to user error with manual configurations. Leiva, N. and Kashin, M. (2023) explains these two problems lead to a lack of standardization with network engineers having majority of their time taken up with configurations due to inefficiency.

Leiva, N. and Kashin, M. (2023) discusses the effects the entire network providing difficulty in scalability, support and security. Because of this network automation has been developed to improve the efficiency and reliability issues within traditional network management. With repeatable processes and operations being automated which can fix the productivity discussed before giving network engineers more time for other tasks. with less down time and more consistency.

## Project Description

This project will be investigating network security with network automation, this will include creating a network artifact with scripts used from an automation tool. The report will include a literature review which will go over the different network management types ranging from in-band and out-of-band network management.

And the traditional and automated methods to create ether types of network managements. Because these management types require communication to create the configurations of network security. In this document it will discuss what the different automatic networking tools and how they communicate to devices, to send automated scripts/playbooks to the device for configuration.

A variety of protocols will be discussed with their advantages and disadvantages for this project. To decide what networking automation tool to be used in the artifact different networking tools will be discussed with the fundamental differences and what they can bring to the artifact.

In order to ensure that the project successfully creates a secure network the configurations used in the scripts will have to be optimal and ensures security across a network. which will need to address during the implementation and testing phases will have to be researched and implemented.

The artifact will include a network design and implantation as well as scripts from a network automation tool that can secure a network device through configurations.

To ensure that the network is configured correctly to ensure the security of a network testing will need to be used on the network device to ensure the configurations are in place with no missing commands that the network automation script has not implemented which will ensure an efficient security set up.

## Project Goals and objectives

* To understand the effects network automation has on a network
* Investigate how network automation improves a network
* Investigate the pros and cons of network automation when applied to a network
* Investigate how to improve security through automation
* To create a working a secure network through the aid of network automation
* Design and create scripts that can be passed to the network for automation

## Project aim

The aim pf the project is to use network automation for network security and to investigate the practical uses that network automation can bring to network security, the project will create scripts to showcase network automation.

## Hypothesis

For this project the hypothesis will be that network automation will reduce the time for network configurations and aid in activities, such as backups through a backup file and updating the running configuration and the overall increase network security.

## Deliverables

The expected deliverables that this project will bring:

* Logbook – meeting logs for the project
* Gannt chart – detailed plan
* Scripts – scripts used in the creation of the automation

## Risks

A risk assessment will be found in the appendix section.

## Ethics form

A fully completed ethics form will be found in the appendix section

# Methodologies and methods

This part will explain the methodologies and methods for the project to consider. As well as the difference between methods and methodologies.

## Methods vs Methodologies what makes them different?

### Methodologies

It is important to make a clear distinction between the methodologies and the methods as they are both often confused with one another. Methodologies is a framework for the research often outlining the approach of the project. The approach for the project is decided on the researcher’s beliefs, principles and perspectives these can all be grouped in the individual’s values. Which will in turn support the approach of the project, Aguiar, G. (2024).

#### Methods

Methods differ with methodologies as the method makes up the specific techniques, tools and actions that the researcher will take within the project. For example, if a researcher has to gather data how will the data be gathered? Ether through techniques like surveys or interviews etc. methods are used inside of methodologies as the means to reach the projects goal., Aguiar, G. (2024)

### Agile

The agile methodologies are using the principle of constant improvement and needing to be enhanced and changed, often leading to the deadlines being anywhere from one day to one year, both of these differ to the waterfall methodology. Agile has more iterations used and have and less dates for the phases these iterations are constantly changing. Using agile often means the project is constantly being worked on, Merkow, M.S. (2019).

The agile methodologies were created after the waterfall methodology in the 1990s it is important to note that agile was designed to be lightweight compared to the waterfall counterpart and therefore more flexible the emphasis on documentation is also lowered with the agile method compared to the waterfall method, this is done to prioritise the speed to deliver the program required, McCormick, M. (2012).

the most common iterations for the agile methodology are mainly kanban and scrum methods, these use sprints to determine the time period of the project this is used to target specific requirements of the project, Merkow, M.S. (2019).

#### Scrum

The agile method is typically implemented in a scrum team with a scrum master at the head of the team then the product owner and the scrum team. Each role within the team is also important as the scrum master as the leader is supposed to make sure that no barriers are in the way of the team’s productivity. The team itself is made up of different roles like developers, testers and others needed for the project, Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017).

A diagram of a project

AI-generated content may be incorrect.

Figure - Agile sprint source - Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017)

Figure 1 shows the sprints found in a project each containing their own requirements and developments. Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017) explains that new features can be added to a project after each sprint even if not stated earlier on.

The scrum teams are set to work within the sprints that can last from anywhere between 1 to 3 weeks with certain tasks being focused on in the sprint, which will come from the sprint backlog which contains all the documentation of the sprint from the product owner, Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017)

The main advantages found within a scrum are the productivity through teamwork and communication mentality this will help optimise the teams and encourage them to be innovative and find the best possible solutions for a said task.

### Waterfall

The waterfall methodology differs from the agile methodology with the principles coming from hardware manufacturing approach to software development. This led to the waterfall approach being more structured, McCormick, M. (2012). Moreover, the waterfall approach was created in the 1970s.

This methodology ordered process for software development this follows the way a waterfall, with water moving downwards in a chain that can’t be reversed, McCormick, M. (2012). It is also important to consider that the next phase can only start after the current phase is completed entirely. Which means each stage takes up large amounts of time, waterfall has the following stages:

* Specification
* Analysis
* Project design
* Coding
* Testing

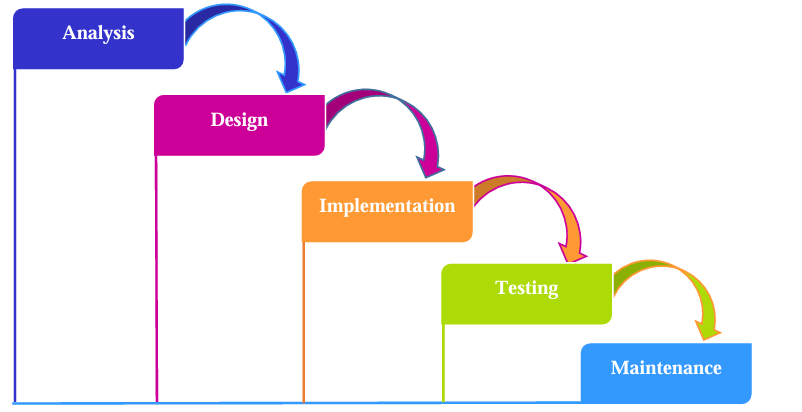


Figure - waterfall process source - Senarath, U.S. (2021)

Figure 2 shows the process of the waterfall methodology. It shows clearly that none of the phases overlap each other at one time. This is because the findings of each phase directly input into the next phase of the project for example the design cannot be achieved without the analysis part of the project. Each phase is also specified to fit one task for example the analysis phase goes over the software requirements of the project, while the design creates a plan for the meeting said requirements found in the analysis phase.

The waterfall method is typically best suited for projects with a very well defined scope and requirements already set out, this is because once a phase is completed it cannot be revisited. It also helps that the product is also very well understood with no unknown areas found in the project. It is also proffered for smaller projects with the waterfall methodology compared to the agile methodology.

# Research methodology

For this project the research methodology will be focused only on quantitative results of the project this is because the tests of the project focus on the feedback of the network devices, with no qualitative methods used. Because of this will help assess the impact and effectiveness of automation. Because of this the project will be using experimentation with tests against the independent variable being the scripts, and automation tools on the dependent variables being the network device.

Scripts will be created to perform certain tasks within the network security spectrum and then be tested against the hypothesis and any potential errors that may occur during the process.

## Data collection techniques

### Experimental research

Experimental research aims to test a hypothesis this is done by investigating the cause and effect between the variables within the project, this done using independent variables to discover the outcomes of dependent variables. Because of this there are a few core aspects of experimental research.

* Variable manipulation for independent variables to find the outcome on the dependent variables to assess the overall impact of the changes.
* Controlling the conditions of the project this may involve ensuring external factors are not affecting the outcomes of the project.
* There are also different types of experimental research found.
* one experiment type example is found below:

#### Quasi-experimental

Resemble true experimentation however does not have random assignment of groups with variables being assigned to specific groups for the project. This is particularly useful for making casual relationships between the independent variables. This can involve checking the dependent variables before and after the independent variable is changed for example checking the running configuration before and after a script has run.

## Research onion

The research onion is a tool as Jansen, D. (2025), describes is used to explain the different pathways that can be taken in the creation of a research methodology. The research onion is made up of six layers these layers will be discussed below for the overall methodology of the project.

### Layer 1 – Philosophy

Layer 1 is where the set of beliefs that the project will be built on. This is broken down into ontology what covers what we already known and how we can understand it. And secondly epistemology which covers how the knowledge is collected in the first place, Jansen, D. (2025). For this project positivism will be used this is because the project will focus on objective results of a project without personal viewpoints.

### Layer 2 – Approach

The research approach includes a general method in which the project will use which are ether inductive or deductive approaches. With inductive being qualitative research and deductive being quantitative research ,Jansen, D. (2025).

For this project a deductive approach is used this is because a deductive focuses on testing the aim of the project and focusing more on quantitative research, Jansen, D. (2025). These tests will be focused on the scripts that will be created in the implementation phase with the purpose of observing the outcomes of the network automation on a network device.

### Layer 3 - strategy

The research strategy focuses on how the actual research will be executed with multiple different research designs being available from case study research to archival research this is explained in ,Jansen, D. (2025). However, for this project the experimental research will be used with the manipulation of independent variables, to observe the effects on the dependent variables.

For this the network automation will be used to obverse the outcomes the scripts will have on the network device used. This strategy also matches with the above chosen strategies as a quantitative strategy.

### Layer 4 – Choices

Layer 4 simply focuses on whether different types of data are used or not for instance there are three types of data that are available these are mono, multi and lastly the mixed method. The multi method uses more than one qualitative and quantitative technique for research. While on the other hand the mixed method uses both quantitative and qualitative techniques, Jansen, D. (2025).

In this project the mono method is used as this project is using experimentation with a deductive approach which is quantitative in nature. Therefore, only quantitative techniques will be used with the network automation of a device being tested.

### Layer 5 – Time Horizon

The time horizon focuses on the time scale for the project there are only two options for research, these are longitudinal and cross sectional, Jansen, D. (2025). Longitudinal conducts research over a large timeframe typically to study affects over time, on the other hand cross sectional is for research at a certain point of time.

Because this project is conducted over a small timescale and will not be conducting research over a longer period, cross sectional will be used with network automation being researched within one point of time.

### Layer 6 – Techniques and procedures

The last layer of the research onion will focus on the specific techniques and procedures for the project to proceed. Techniques and procedures can only be used if they fit within the first 5 layers, Jansen, D. (2025). For this project all techniques and procedures will be quantitative and fall within the cross sectional timeframe:

* Log data – log data will be collected for use in a log analysis this will be compared with the changes made in network device by the scripts used.
* File data – the contents of the files created through automation will be collected for analysis.
* Log analysis – will be used to assess the changes in the configuration of a network device to see if the expected configurations have actually been transferred to the device.
* File analysis – will check for the necessary creation of specific files, as well as the content found within each file is the expected content that the automation script was intended for.
* Experimentation – experiments are used to test if the automation has met the aim of the project. Tests will be done on each script used, results for the tests will be included in the log and file analysis.
* Tools required – an automation tool, a virtual machine and a virtual router for configuration changes.

# Literature review

## Network Management Types

### In-Band Network Management

As Nevis (2016) discusses in band management occurs in flow of the live network traffic this can be found in the network layer switches directly, or next to the endpoints that access the network layer. This lets all client server side traffic transfer through the in-band management, this is useful for providing pre and post connect security services.

Post connect services are useful for access controls, visualisation and traffic control. While on the other hand pre connect services allow for compliance checks during for example the login phase, which will help identify what access polices are being used. In industry it is often used for the co-location of the authenticator PEP and PDP functions as seen in figure 3.

This acts as passive live network analysis, which helps defend against malware/malicious attacks as all network traffic is captured in key areas.

A diagram of a process flow

Description automatically generated

Figure - Flow of traffic for a client login source - Nevis (2016)

A diagram of a policy enforcement point

Description automatically generated

Figure - In-Band Network diagram source - Nevis (2016)

In band management systems as seen in figure 4 as Nevis (2016) also confirms it allows for stateful firewalls to be included in the access control and data inspection throughout the user session length.

#### Traditional Network Management

Traditional network management is split into 3 categories this includes active, passive and hybrid methods of measurement according to Lizhuang, T. *et al.* (2020).

Active traditional network management involves, concluding the network performance by using tools like ping and traceroute. However active management uses the out-of-band nodes forwarding path which does not copy the service flow of the network, so results don’t fully conclude the networks performance.

Passive network management focuses on analysing the status traffic operations through the use of traffic mirroring and proxy reporting, examples of tools that use this method are NetFlow, Sflow and IPFIX. This however is limited to the performance of the switches and the methods used, which are compression and sampling these inevitably affect the accuracy.

Hybrid network management allows for more optimal measurements as it uses both the passive and active methods of measurement, to use the advantages of both.

Traditional network management is widely used, being simple to set up. however has issues with the accuracy of the data received back.

#### Automation of In-Band

Lizhuang, T. *et al.* (2020) discusses the automation of in-band network management starts with a software-defined network (SDN) and the use of a programmable data plane (PDP). This allows for the use of in-band network telemetry. However telemetry is used for measuring the performance of a network for better accuracy, scalability and coverage.

This process is automated and only preformed remotely or at inaccessible points. The key feature being that the user plane and data plane are shared on the same link or packet.

A diagram of a computer

Description automatically generated

Figure - In-Band Network Telemetry source AlSabeh a et al. (2022)

As seen in figure 5 AlSabeh a *et al.* (2022) discusses, how network telemetry works with in-band management through adding meta data to packets through nodes, nodes are also known as switches, packets are then transmitted to a server or a host.

The INT source switch also signals which packet is used for telemetry collection, while the INT sink removes the INT header. This is forwarded to the Telemetry sever, while sending the starting packet to the receiver host.

### Out-of-band Network Management

Nevis (2016) notes out-of-band management is in-line but only during the login phase of the session of the user, this provides the benefits of compliance checks and enforcement of polices after the checks are complete and the polices made. The user will then opt out of the user traffic path for the session, this can be achieved by giving components to the authenticator and PDP for the session. The switches however will then be used for the PEP, this means enforcement options are isolating VLANs and ACLs through the switches, unlike in-band which can view block traffic.

Diagram of a computer system

Description automatically generated

Figure - Out-Of-Band Network Management source - Nevis (2016)

the dedicated node placed outside the line of traffic, and operating through the switch to pass off instructions and view the network performance.

#### Traditional network management

Felix Emmert, F. (2015) suggests out-of-band management involves logging in remotely to the node. This requires the use of protocols like SSH, Telnet, HTTP or IP ad example of a remote login can be done with Keyboard, Video and Mouse (KVM). Which allows the user to view the output through the video, which forwards the input through the mouse and keyboard. this method for example uses TCP/IP to allow for simple remote entry.

Out-Of-band network management has the ability to perform hardware resets by accessing the hardware remotely, through one of the protocols mentioned above. With this ability comes the option to configure power restore settings.

Out-Of-Band also allows for bootable media to be inserted into the server, which allows for system updates or the restoring of data by adding backups which helps with redundancy.

Out-Of-Band allows for a web interface to be ran which checks the systems performance, allowing for configuring and modifying the Baseboard Management Controller (BMC). The BMC sits in the hardware with its own OS and storage which has access to I/O ports.

Serial over LAN allows for remote access of the main system using a serial interface, letting the user access the BIOS and bootloaders. Lastly the CLI can also be used remotely to access the system which allows for commands like ping, trace route and all other BMC commands, these are accessed over IP on the UDP port 623. However, ssh using TCP port 22 can also be used for better security.

#### Automation of Out-Of-Band management

Presser, S. (2022) discuses Out-Of-Band network management can be automated in similar ways to In-Band, with the use of network telemetry, this is gathered from the device itself this has the benefit of not requiring any software on the device. Because Out-Of-Band is performance heavy the use of telemetry adds no extra performance cost. Specifically, the telemetry is collected through the BMC as stated before the BMC is a part of the hardware and monitors the performance, which can be accessed through remote access. Figure 7 shows in-band and out-of-band telemetry.

A diagram of a software measurement

Description automatically generated with medium confidence

Figure - Network Telemetry source - Lizhuang, T. et al. (2020)

Presser, S. (2022) explains that tools like RedFish which is an industry standard interface, allow for the BMC to integrate into the system of the hosts, giving the user access to all the telemetry gathered. This includes processor metrics, memory access statistics, sensor readings, network packet counts and disk access statistics, this all has a very low impact on the performance thanks to the BMC .

## Communication Protocols

### SSH

La Lau, R. (2021) explains the Secure Shell (SSH) protocol creates remote encrypted connections allowing for the opening of remote terminals and executing commands, and other features like file transfers (SCP SFTP). SSH is a client that uses a SSH daemon which is called SSHD, the connection used by SSH daemon is port 22. The client or child SSH needs the daemon to start and create the child processes.

SSH has a client and a server, the server listens for incoming ssh requests and then authenticates these requests. Afterwards a command prompt is given, these servers will be displayed as daemons as seen above with SSHD. Meanwhile, the client is used to connect to the server, an example of a client can be PUTTY for windows and ssh1 for Unix.

Lucas, M. (2018) explains that there are 2 versions of the ssh protocol SSH-1 and SSH-2 all modern devices will use SSH-2 by default. This is due to SSH-1 being vulnerable to attacks making it outdated. Unlike SSH-1, SSH-2 is constantly updated when new threats are found.

Encryption which is used by SSH turns readable writing AKA plaintext into unreadable ciphertext, while decryption does the exact opposite. An encryption algorithm is used for this conversion, these algorithms use a key which is a list of letters, numbers, symbols and data which all can be used to encrypt plaintext these keys are often randomly generated by OpenSSH. This makes it harder to decrypt plaintext. In order to decrypt the plaintext a secure encryption key is needed.

### Telnet

Lerner, K.L. and Lerner, B.W. eds. (2013) explains that telnet is a protocol used for communication between devices and servers through an interface. Telnet is useful because it can transfer commands that are written on the host computer through a network link in plaintext format, with the response from the receiver device also be transmitted through the same link.

telnet uses plaintext and a simple link to helps minimise the network performance, while giving the flexibility of not being at the location of said network allowing commands to be written remotely.

Telnet suffers from a well know issue which is the security factor, this is due to the data being transmitted in plaintext, meaning intercepted data can be easily seen as no encryption has been applied to the data. This also applies to passwords and configuration commands, this means while telnet is in use the management of a network will have to ensure that no unencrypted data can be seen outside of the company network, as well as employees who have no permission to see said data. This security factor often concludes with telnet being blocked from use in the network itself, with other protocols being preferred like SSH for example.

### HTTP

Grigorik, I. (2017) explains that the Hypertext Transfer Protocol is found on the application level, being used to distribute and collaborate with hypermedia information systems. For this task it is a stateless protocol which also allows it to branch into other tasks, including name servers and object management systems. This is achieved through headers, request methods and error codes.

there are two main versions of HTTP which are HTTP/1 and HTTP/2:

HTTP/1 – allows for the object body to not be ASCII coded, but could be any other type including HTML, image, text and more.

HTTP/2 – the modern version focuses on improving the performance through lower latency and better throughput.

HTTP/2 - can be used securely between a client and the server through the use of encryption, authentication and integrity checking. This is achieved by using TLS with ALPN negotiation to start HTTP/2, unlike HTTPS which naturally comes with encryption and authentication.

### NETCONF

Nevis (2016) explains NETCONF is a management layer protocol which can be compared to SNMP, both have the capacity for making configuration changes as well as the retrieval of data from devices. NETCONF also uses SSH for the connection between devices and uses XML to transport the data across SSH to the devices, this is embedded inside the NETCONF protocol stack to transport with SSH. The XML encoded data contains pre prepared operations for the devices often called a third party data store.

Nevis (2016) says each device that does support NETCONF may not support the same NETCONF features as a different device. This is important to consider before pushing an XML to the device, even if the tools are the same for multiple devices the structure of the data is often different which depends on the device’s vendor.

Nevis (2016) explains NETCONF provides benefits for creating multiple changes to a device, in one NETCONF session. However, when a change fails then any changes applied will not occur, unlike the command line configuration where commands are pushed through. It is also important that NETCONF provides the authentication, data integrity and confidentiality to access the remote devices.

### RESTCONF

Nevis (2016) describes RESTCONF as a combination of Representational State Transfer (REST APIs) and NETCONF. Because of this it uses HTTP combined with standard requests like “GET”, along with HTTP. RESTCONF also has support for JSON and XML which is used in particular by NETCONF, this provides flexibility for the user to choose the desired encoder out of the three choices.

Nevis (2016) explains configurations can also be changed with RESTCONF, this is achieved with operators which are “PATCH” and “PUT” as well as execution operations. Similarly to NETCONF the data structure is also very important here, as it will determine what RESTCONF can actually understand.

Nevis (2016) mentions RESTCONF has drawbacks with HTTP, because HTTP should not be allowed to carry the state between the devices. Meaning any RESTCONF transactions will be limited to one single HTTP call. HTTP is also limited in the ability to push only one HTTP request operation, for example a “create”. However, this is mitigated with a YANG patch which can combine operations.

## Security Configurations

It is important to preform device hardening on network devices to increase the overall network security, the network is split into three planes these include:

* Management Plane = manages the traffic that is sent to the IOS device, this plane is made of applications and protocols like SSH and SNMP, Netflow.
* Control Plane = processes the traffic of a network device which maintains the functionality of the network infrastructure, which contains applications and protocols in between the network devices like BGP, IGP, EIGRP, OSPF,
* Data Plane = data is forwarded through a network device. This won’t include traffic to the local IOS device.

#### Management plane

Cisco (2024) explains that the management plane covers a large number of different topics including, Password management, ACL Limiting, Accounting-Authentication-Authorization (AAA), Securing SNMP and secure management sessions. For this project the focus will mainly be on the management plane for security configurations, as it has more standard security configurations that all devices need. This will provide clear testing to ensure that the artifact successfully implements the code. The other planes will be implemented as well these are mentioned below, providing a full understanding on how each plane interacts with each other. Table 1 includes the basic management security configurations:

Table - management plane basic configurations source - Cisco (2024), Cisco (2007), Cisco (2015)

|  |  |  |
| --- | --- | --- |
| Command | Description | Testing |
| enable secret  username <username> privilege 15 algorithm-type scrypt secret <secret> | Creates a stronger password using “srypt”  ‘enable password’ configures the password for enable mode.  Use type 9 if possible for a more secure password. | “Show run | include username”  Show running config will show only usernames due to the “include” |
| username secret | Encrypts the user’s password with a MD5 hash. | “Show run | include username” |
| aaa new-model  aaa local authentication attempts max-fail <max-attempts>  aaa authentication login default  TACACS | Creates an aaa model, then  Locks a user out after failing the maximum attempts for the correct password.  After the aaa authentication method will use the local user database. | “Show aaa method-list all” shows the methods for all authorization, accounting and authentication. It should show different queues and when they are applied.  “Show run | include aaa”  To show only aaa configurations |
| no service password-recovery | Denies console access to insecurely open the device configuration to clear passwords, (unfortunately not supported). | “show run | include password” |
| ip ssh version 2 | Allows for the configuration of SSHv2. | Show run  Show ssh |
| line console 0  password cisco123!  login local',  exec-timeout 5 0  logging synchronous  line vty 0 4  exec-timeout 5 0  password cisco123!  login local  transport input ssh  exit | Transport input Acts as access controls for the vty and tty lines.  Login local is used for logging in with local users.  Logging synchronous shows the output of unsolicited messages. | Show run |
| exec-timeout | Kicks out sessions that have been sitting idle for a specified time. | Sh run |
| aaa authentication login default group tacacs+ | Creates a remote AAA server that manages remote authentications, users access the server with HTTP, SSH, HTTPS and telnet.  (TACACS preferred over RADIUS as it encrypts the entire TCP payload). | Sh run  “Show aaa method-list all” |
| control-plane host  management-interface FastEthernet 0/0 allow ssh snmp | The Management plane protection allows restrictions for interfaces traffic management, and how a device is accessed | Sh run |
| ip domain-name example.com  crypto key generate rsa modulus 2048  ip ssh time-out 60  ip ssh authentication-retries 3  ip ssh source-interface GigabitEthernet 0/1  ip ssh version 2 | Allows for a secure encrypted remote access management connection to a device using ssh. | Show run  Or  Show run | include ssh |
| aaa new-model aaa authentication login default group tacacs+ enable (if available for a hash)  tacacs-server host <ip-address-of-tacacs-server> tacacs-server key <key> | Creates a secret password for local users as a backup for authentication. | Show run  “Show aaa method-list all” |
| aaa authorization exec default group tacacs none  aaa authorization commands 0 default group tacacs none  aaa authorization commands 1 default group tacacs none  aaa authorization commands 15 default group tacacs none | Authorization of TACAS+ and AAA lets the commands by the user be denied or permitted. | Show run  “Show aaa method-list all” |
| aaa accounting exec default start-stop group tacacs  aaa accounting commands 0 default start-stop group tacacs  aaa accounting commands 1 default start-stop group tacacs  aaa accounting commands 15 default start-stop group tacacs | Sends information about every EXEC command used to the TACACS configured servers, as well as the date the command was used. | Show run  “Show aaa method-list all” |
| no ip bootp server  no ip finger | Disables unused boot strap protocol BOOTP.  Disables the finger service. | Show run |

#### Control Plane

Cisco (2024) explains that the control plane of a network device needs to be secure because this plane maintains the data and management plane. So, if this plane is compromised the network recoverability and stability is affected. See table 2 for control plane configurations:

Table - control plane basic configurations source - cisco(2024)

|  |  |  |
| --- | --- | --- |
| Command | Description | testing |
| no ip redirects | Prevents the router responding to each ICMP redirect, which would limit the CPUs performance. | Show run |
| no ip unreachables | Disables the ICMP unreachable methods. | Show run |
| no ip proxy-arp | Disables the proxy ARP which helps mitigate Man In The Middle Attacks. | Show run |
| ip icmp rate-limit unreachable | Allows you to change the limit of ICMP messages. | Show run |
| ttl-security | It is a command option for the neighbour command, and uses the Time-To-Live function to confirm that the BGP packets received from a connected direct peer router. | Show run |
| neighbor <ip-address> password <secret> | Allows for peer authentication with a MD5 digest of each packet. | Show run |
| neighbor maximum-prefix | Adds a limit of the max number of prefixes before the peer shuts down, useful because the more prefixes a router has the more memory BGP takes up. | Show run |
| distribute-list | With keyword ‘out’ = limits the amount of information advertised on the network.  With Keyword ‘in’ = limits the number of updates processed. | Show run |
| key chain <key-name>  key <key-identifier>  key-string <password>  interface <interface>  ip authentication mode eigrp <as-number> md5  ip authentication key-chain eigrp <as-number> <key-name> | Adds MD5 encryption to the passwords during the authentication of EIGRP. | Show run |
| router eigrp <as-number>   passive-interface default  no passive-interface <interface> | Helps manage the advertisement of the routing information, which can lead to information leaks or adding false information. | Show run |
| ip prefix-list <list-name> seq 10 permit <prefix>  router eigrp <as-number>  passive-interface default  no passive-interface <interface>  distribute-list prefix <list-name> out (or in) <interface> | Occurs on the interface with EIGRP, after the router filtration is enabled the information advertised becomes limited. | Show run |

#### Data Plane

Cisco (2024) explains that the network routing configuration directly determines a large portion of the data plane traffic going through a network. However, the IP network functionality allows path altering of packets across a network which provide security risks. See table 3 for data plane configurations:

Table - data plane basic configurations source - cisco(2024)

|  |  |  |
| --- | --- | --- |
| Command | Description | test |
| ip options {drop | ignore} | Keyword ‘drop’ = all of the IP options received by the IOS device are discarded, which helps lower the CPU load or a changing of security controls. | Show run |
| no ip source-route | Disables the IP source routing that is enabled by default, it can be used to route the traffic around security controls in place. | Show run |
| ip directed-broadcast | limits broadcasts in use of a ACL, important because IP broadcast packets can be sent to a remote IP subnet. | Show run |
| port-security | Mitigates any MAC address spoofing at the access interface, port security can dynamically learn addresses which also provide 4 different violation modes. | Show run |
| ip dhcp snooping  ip dhcp snooping vlan <vlan-range>  ip arp inspection vlan <vlan-range> | Used to reduce ARP poisoning attacks on any local segments, by intercepting/validating the IP-to-MAC pairing on any untrusted port.  The last command enables dynamic ARP Inspection. | Show run |
| private-vlan isolated | Creates a secondary vlan that has zero communication with devices. | Show run |
| interface <interface>   switchport  switchport mode access   switchport port-security  switchport port-security mac-address sticky  switchport port-security maximum <number>  switchport port-security violation <violation-mode> | Port security for protection against MAC address spoofing at the interface, this is dynamically learned with MAC address Sticky. | Show run |
| ip access-list extended ACL-TRANSIT-IN   permit icmp <trusted-networks> any    deny icmp any any | Only permits ICMP from the trusted network only all other traffic has ICMP denied.  This is required as ICMP should be rarely used from any external network. | Show access-list ACL-TRANSIT-IN  Shows all configurations for the access list  Show run |
| ip access-list extended ACL-TRANSIT-IN   deny tcp any any fragments  deny udp any any fragments  deny icmp any any fragments  deny ip any any fragments | Important to deny ip fragments with protocols, these fragments can be used to evade detection. | Show access-list ACL-TRANSIT-IN  Shows all configurations for the access list  Show run |
| deny ip any any option any-options | This can be put inside the same access list as above this is because packets from options can be used for bypassing security controls. | Show access-list ACL-TRANSIT-IN  Shows all configurations for the access list  Show run |

## Automation Tools

### Ansible

Ratan, A. (2018) discusses that Ansible is an open source framework and automation platform, that has found popularity through automation in an efficient and easy manner. This is helped by the agentless push-based architecture, which uses of a host managing the network through SSH, allowing for programs (ansible modules) to be ran. These programs are abstracted, meaning user input arguments only have to be given for the program to work, this allows for the automation to take place, giving the user more free time as the modules will take over and apply the commands required.

Ansible uses playbooks which are highly useful for automation, these playbooks are YAML files with concise automation tasks. Each module is run from the code base or collections seen in Figure 8:

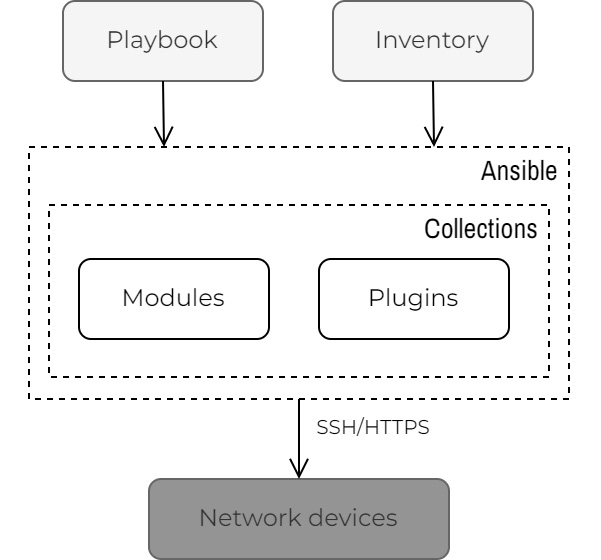


Figure - Ansible Automation source - Ratan, A. (2018

Meijer, B., Hochstein, L. and Moser, R. (2022) explain simply ansible is labelled as a configuration management tool. The playbooks used function similarly to other network management tools, in the way that ansible states the hosts that will be configured. Ansible executes the playbook in a certain order, it starts with the generation of the python script, then copying the script to the stated hosts, and executing the script on all hosts, ansible will wait for all executions to finish.

During the execution of the playbook ansible will execute the code equally across all hosts stated, ansible runs the code in the playbook in the same order it was created in.

An inventory is a list of statically known hosts ether defined in a text file or pulled from the external system, which are manged individually or collectively using groups. Inventories are then used in playbooks, an example of an inventory is shown in figure 9

A close-up of a computer screen

Description automatically generated

Figure - Inventory example source - Ratan, A. (2018

Ansible is written with python, which is a tool that is widely familiar to many network engineers.

Key components of ansible are found in table 4:

Table - Ansible Components

|  |  |
| --- | --- |
| Component | Description |
| Inventory | Configuration file defines the host information that needs accessing. |
| Plays | Defines the task executed on a certain number of nodes. |
| Playbook | A set of instructions created by users so ansible can deploy, manage and configure the nodes found in the inventory. Playbooks need at least two plays inside. |
| Tasks | Specific actions executed within the playbook. |
| Variables | Custom defined also store values depending on the execution of the playbook. |
| Roles | Define the order of how playbooks are executed. |

#### advantages of using ansible

Meijer, B., Hochstein, L. and Moser, R. (2022) explains due to ansible primarily using YAML and python it has an easy learning curb for new users, this also adds to the readable syntax. Moreover, the playbook has a separate readable format which furthermore helps users understand the code.

Meijer, B., Hochstein, L. and Moser, R. (2022) explains remote hosts require very little prior installation tools to use ansible which are only python, and SSH for the communication. Moreover, due to a large community ansible offers plugins and modules, which can be specific to certain tasks e.g. cloud libraries and playbooks, which can save time on some prior creation of playbooks.

Meijer, B., Hochstein, L. and Moser, R. (2022) explains playbooks allow for a push-back model which lowers potential mistakes in host configurations, with any mistakes in the playbook being corrected before running the playbook again, instead of manually entering the host.

This allows for an orchestration of playbooks which can manage different devices separately, due to this orchestration feature. Ansible has been used to configure large quantities of nodes within businesses making ansible highly scalable, these scripts are also idempotent being able to run multiple times against hosts.

#### Disadvantages using ansible

Sen, K. (2025) explains that the command line of ansible is better than the User interface, with more options being available on the command line compared to the interface. Moreover, the interface can desync with the command line which can create errors with the playbooks and commands.

Ansible does not keep track of any dependencies which means that ansible doesn’t understand the state of the host this is due to the high simplicity of ansible, meaning ansible only executes an ordered set of tasks ether successfully or unsuccessfully, because of the inability to maintain or create an extensive catalogue which would allow for a state to be noted.

Unlike Linux which ansible uses SSH, for Windows ansible switches to PowerShell instead of SSH. Ansible also still requires a Linux control host for Windows machines, meaning more support is found with Linux over Windows. Ansible is also newer compared to other network management tools, which means the community and support isn’t as large compared to other competitors like Puppet.

### NAPALM

Okasha, K. (2020) explains, Network Automation, Programmability Abstraction Layer with Multivendor support (NAPALM) is a python library found on multiple vendors. The goal is to provide a sustainable method that can be used regardless of the vendor equipment. Unlike ansible that uses different modules for each vendors OS, that is not required for NAPALM the output data is returned in a consistent/normalized state.

Each device is interacted with through the most available used API used on the node. As seen in figure 10:

A screen shot of a computer

Description automatically generated

Figure - Napalm vs ansible source - Okasha, K. (2020)

NAPALM supports specific vendor devices, and the common tasks used like configuration, interface operational state retrieval, Border Gate Protocol (BGP), Link Layer Discovery Protocol (LLDP).

Oswalt, M. et al. (2023) explains that NAPALMs core library already comes with the most popular APIs used. NAPALM is able to retrieve data from network hosts available to NAPLAM which is intern normalized by NAPLAM for all the hosts used, which is especially useful for vendors that have not any YANG models from ITEF/OpenConfig, which would normalise the data instead of NAPALM needing to.

Oswalt, M. et al. (2023) NAPALM provides two options for configuration which include:

* Configuration Merge option - uses an existing partial configuration or commands and then confirms they will exist on the remote target host. This can be achieved on all platforms.
* Configuration Replacement option – focuses on what the configuration is required for the host instead of what configurations are already existing. The replacement option needs to push a completed and ready configuration, this option doesn’t require any delete commands instead using commands like “config replace” as seen for cisco IOS.

#### Advantages of NAPALM

Oswalt, M. et al. (2023) explains the configuration replacement doesn’t require a delete option meaning the user only has to replace the current set with one command. While the configuration merge option, being the most often used does not remove any commands or command hierarchies. This is often safer and is used before replacement option until rendering is completed.

Oswalt, M. et al. (2023) because NAPALM uses Nornir for configuration management, this gives NAPALM the ability to access python libraries like Netmiko, Paramiko and Netconf. Allowing for the creation of more advanced scripts.

Oswalt, M. et al. (2023) enquires about the simplicity in terms of the language NAPALM uses. Python which has less overhead compared to Ansible, because of this the main benefit over playbooks includes easier debugging and having an easier development process compared to YAML.

Oswalt, M. et al. (2023) describes Programming languages having more sophisticated logic that need to be defined, compared to Nornir which offers, extensions that provide new functionality which cannot be achieved in YAML.

#### Disadvantages of NAPALM

Oswalt, M. et al. (2023) explains NAPALM is not a Domain-Specific Language (DSL); DSLs would include YAML these languages have a lower barrier of understanding compared to python, which is the predominant language that NAPALM uses is python.

Oswalt, M. et al. (2023) concludes NAPALM unlike other tools like ansible, which have community tools which is supported commercially for example by RedHat.

### Netmiko

Byers, K. (2024) describes Netmiko as a multi-vendor library created to simplify command line interface connections to network devices targeted. Netmiko in summary is tasked with gathering show commands as outputs, while implementing configuration changes. At the same time Netmiko tries to abstract the low-level state controls away from the user.

Ratan, A. (2018) explains python has a wide range of uses with one of them being for network management, this is seen with the python library Netmiko. Which takes advantage of the large support different network devices have for python, in general Netmiko is being used for automation with vendor support that includes Cisco IOS, NX-OS and more.

Oswalt, M. et al. (2023) explains through the use of the command line interface, SSH which is nether modern, or pragmatic has benefits which revolve around the fact that not all vendors have support for the pragmatic APIs, so SSH can be used to turn on other APIs automatically.

SSH has been primarily used as the API for transferring commands to remote hosts using a consistent SSH connection. Netmiko provides a transition using the SSH API in a more automated method, the SSH connection is handled in Netmiko through a device handler object. Because netmiko already understands how to talk with different network devices, netmiko has a low barrier of entry.

Oswalt, M. et al. (2023) explains Netmiko has multiple methods of sending configuration commands which can be done in a list via, “send\_config\_set()” or separately with “send\_command()”, with the list of command automatically putting the user into global configuration mode before executing the list. After the script has finished executing, then the SSH connection closer to avoid leaving the SSH connection open by accident.

#### Advantages of Netmiko

Oswalt, M. et al. (2023) explains Netmiko is heavily used for network automation which leads to Netmiko being highly developed, this also adds to the benefit of Netmiko being compatible with multiple different vendors. This is seen with the configuration mode on Netmiko which is covered by lots of vendors.

Oswalt, M. et al. (2023) explains netmiko Increases the security of the SSH connections by automatically ensuring that the SSH connection is closed. Netmiko can use templates called TextFSM which allow for any semi structured text usually from the command line to be restructured in a template form.

Oswalt, M. et al. (2023) explains Netmiko is used by other tools including NAPALM, where Netmiko acts as the primary SSH driver. While python is used in ansible it is not used within the playbooks which instead uses YAML.

#### Disadvantages of Netmiko

Oswalt, M. et al. (2023) says unlike other tools Netmiko requires templates to form a more structured output from the command line, this is because the basic output received is unstructured which can lead to misunderstanding in what the script has achieved.

# Artifacts Design

A diagram of a computer network

AI-generated content may be incorrect.

Figure - network Design

Table - Network table

|  |  |
| --- | --- |
| Network 192.168.30.0/24 | |
| IP address | device |
| 192.168.30.130 | Router 1 (csr1kv) interface g0/1 |
| 192.168.30.129 | PC1 (DEVASC VM) |
| 192.168.30.128 | PC2 |
| 192.168.30.127 | TACACS Server |
| 192.168.30.126 | Network 1 Switch |
| 192.168.30.125 | PC3 |

the network artifact will use two virtual machines this will be to create the scripts for this purpose a cisco router VM will be used alongside the DEVASC VM which will both connect together using SSH which will mimic the use of SSH with physical devices. Table 5 shows the network IP and the devices that are associated with them.

A star topology is used showcase a standard network as Gao, H. et al. (2022) explains star topologies are common for computer networks, also being convenient for the management and maintenance of a network. this topology has a central node which allows for easy fault detection and isolation of nodes, which create issues with redundancy. But for this report the artifact is focused on the network automation of a network device, meaning the topology security lays outside of the scope for this project.

It is important state that none of the tools discussed in this report have the capability to give a network device, that does not have an existing ip address a new IP address. This is because all these networking tools discussed use SSH, which needs a pre-existing IP address to connect to a network device to begin with.

## Netmiko

Netmiko will be used for the artifacts script this is due to the similarities with Paramiko, which both provide network automation and have the advantage of the python support with several vendors. However, as Ratan, A. (2018) explains Netmiko has support for cisco IOS while having the underlying library of the Paramiko libraries, which is used with ssh to be able to connect to network device. While Netmiko extends the features of Paramiko with sending and receiving outputs from the commands used.

Netmiko is a python library and is used by multiple other tools within the network management automation. Both examples used above use python most notably NAPALM, as well as Ansible but not in the playbooks. Python also being a common programming language led to Netmiko being chosen other tools may be used within future works.

Moreover, protocols like NETCONF or RESTCONF provide similar features to Netmiko, with RESTCONF using multiple different languages including JSON and HTTP. However, both struggle with the data structure which will change on the vendor, it is important understand all the tools discussed have no large advantages over Netmiko.

## Script 1 – Management plane hardening

The first script will perform some of the basic security configurations gathered from table 1 this will be performed only for the network management plane, including SSH, TACACS, VTY lines and more. This is due to the management planes responsibility of access, configuration and management of devices.

This script will automatically perform the security configurations and will use block configuration coding as seen in, Network Journey (2024) which connect to the device via SSH. The script will be tested to check if the configuration commands have been successfully implemented, this will include a testing script file for script 1 as well, for unit testing.

## Script 2 – control and management plane hardening

This script will behave similarly to the first script using lists as seen in, Network Journey (2024) to configure the commands found for network security within the control and data plane. This includes Access lists, distributed lists, EIGRP encryption, prefix lists and passive interfaces, this script will also be tested with configuration commands. These two planes are combined in this script as they provide less security options unlike the management plane.

## Script 3 – backup file and configuration file

The third script will create a backup of the current running configuration of the device which can be seen in, Network Journey (2024). This backup will include the commands used and any comments about this code, showing the complete contents of the running configuration.

The running configuration cannot be directly used as a configuration script, for this reason a second file which will contain a copy of the content from the backup file. This content will be altered taking out lines of the content that are, firstly not actual commands or comments, and secondly, any other content that will prevent the file from running.

The second file can be directly used to pass configurations to the device, useful for restoring any changes that are removed from the current configuration, this also helps network management.

## Script 4 – output file configurations and update the startup configuration

The fourth script will use the configuration file created from the third script, to update the running config by send commands from the file to the device. The script is also important to configure as the startup configuration as well, because they are the configurations used when the device restarted, Schrader, D. (2025).

The running configuration also can be changed separately from the startup configuration and will be used while the device is powered on, but it is not stored when restarted.

## Script 5 – Unit test script for management plane

Unit testing is a popular method of testing scripts to check if the python code itself works. The “unittest” framework is similar to other language testing frameworks, this framework can use automation for tests. A basic example from, Python (2025a) shows the usage of class which creates three individual tests that are then called by main. Moreover, this script will be used to test the actual python code of the management plane to check if works correctly, any errors will be showcased.

## Why use multiple scripts?

It is important as a security to separate the scripts to be used, this is because if all the code was stored in one script this can cause redundancy issues with the script, as the file may be corrupted or deleted. Moreover, there are security concerns with the code being only on one script, because if the script is stolen so will all the code used for the network management.

If the code is stored in 5 separate scripts it will minimize the odds of these issues from happening, as well as aiding the users by making the code easier to understand and more structured.

# Implementation

## Script 1

### Connect to a device

A screen shot of a computer

AI-generated content may be incorrect.

Figure – connecting to device

Figure 12 shows the connection of the script to the router with ssh by using the port number 22. For this code for three inputs are required for the connection to be established. These are the ip address, username and the password the rest are static

In figure 12 The Connect Handler from Netmiko is required as this command uses the information inside of “router1” for the initial connection to the device. As well as for entering enable mode on the device with the secret “cisco123!”.

The username is required from the user as it is used with ssh and the IP address to establish a connection. This username already has to be configured on the device and lastly the password for the username to gain access to the device.

### Device configurations

A computer screen with text

AI-generated content may be incorrect.

Figure - password and send\_config\_set "def" statements

Figure 13 shows the creation of the username with a secret. Moreover, these password will be hashed in the running configuration later all passwords will be “cisco123!”, for simplicity reasons because this implementation is to show the use of automation of network security.

Because this is a list called “commands” indicated by square brackets, to use the list in the configuration of a device the “def” called “sendManagment\_command”, which contains “send\_config\_set” is needed and will run the commands line by line. The last command in the list has no comma to signify the end, It is important to mention not all commands are available on this device as “no service password-recovery” does not work on this device so therefore isn’t included.

All “defs” are functions, which help increase readability and are called one at a time with “sendManagment\_command” function which is used to send the commands from the other functions to the network device.

A computer screen with text

AI-generated content may be incorrect.

Figure - AAA command list for configuration

Figure 14 shows the list found here contains all the triple AAA and TACACS commands used including the authentication authorization and accounting. The code here mirrors figure 13, however shows more commands inside each list has been organised to group related commands together.

Moreover, it is suggested by Cisco (2024) to use the type 9 scrypt, which doesn’t work on the network device used here. The TACACS server IP address also is not a viable IP address because the router it is configured on is not a network with a server available. However, this command is shown to showcase a secure network configuration.

The method of using a list and the “send\_config\_set” seen in figure 13 is used for multiple other configurations from the management plane, this includes unused services, line commands for vty and con 0 lines and encryption commands.

The first script uses the commands seen in the table 1, these commands are the general security hardening options found within the management plane. Multiple lists are used for this configuration.

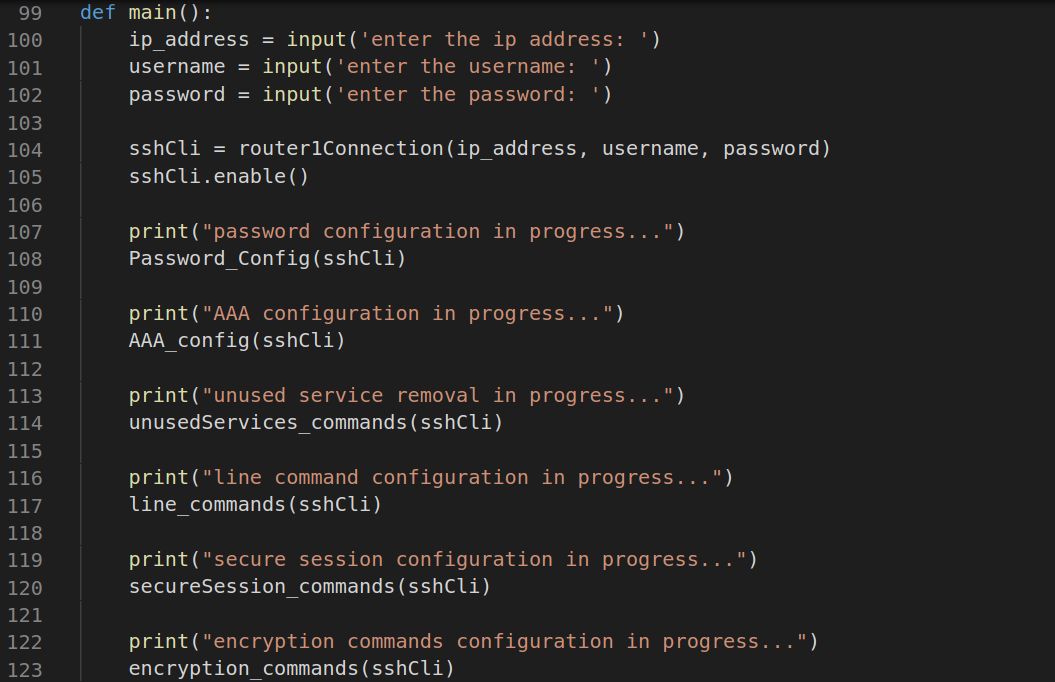


Figure - main() function calling all other function in script

Figure 15 shows the IP address, username and password prompts as well as the actual function used in figure 12 to connect to the router, with variables gathered from the prompts. Moreover, all the other functions are called in the main as well to configure the device this is required for unit testing and being concise.

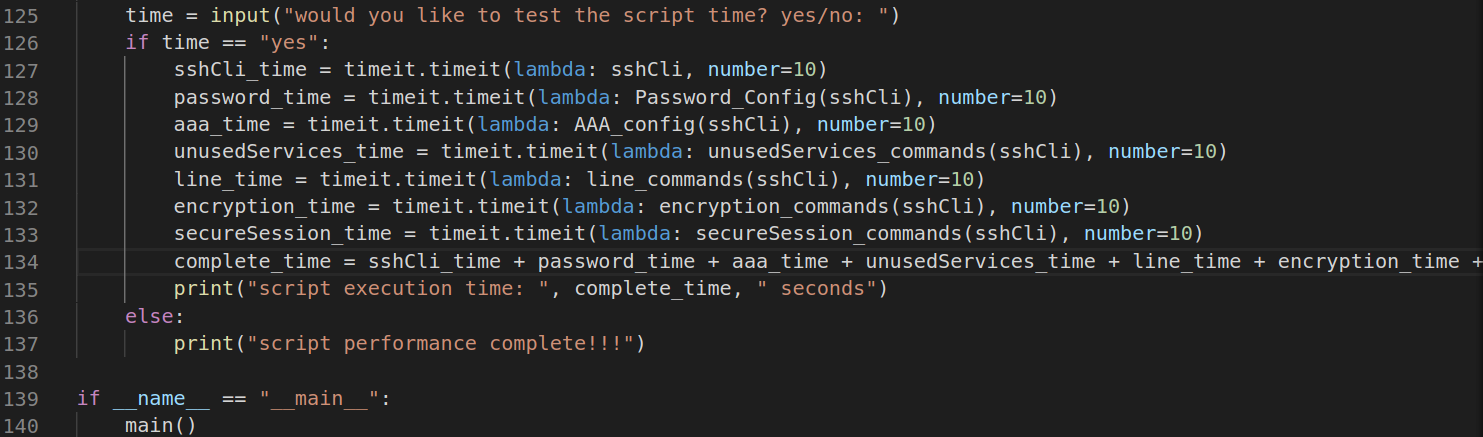


Figure - main() function performance checking code

Figure 16 shows the use of the “timeit” which was imported in figure 11 these variables are using “timeit” to test the functions to get an average performance these functions are ran by the number which is 10 meaning the functions are ran 10 times for the average speed and then added together. Lastly the last if statement is required to run the main() which will call other functions.

## Script 2

A computer screen with text and numbers

AI-generated content may be incorrect.

Figure - connects to the router

Similar to figure 12 but no functions are used with input and connect handler being all together, this is great for simplicity if no unit testing is used.

Moreover script 2 is similar to script 1 however, no functions are used the lists are used for the control and data plane configurations which are found in table 1 and 2. Script 2 also uses the same if statement for the speed testing of script 1 which seen in figure 16, which will be used in the testing phase to provide an average speed time of the script execution

## Script 3

Script 3 to shows the same initial start as the second script with figure 17, this is because all of the scripts connect to the device in the same manner, which helps make all scripts more uniform

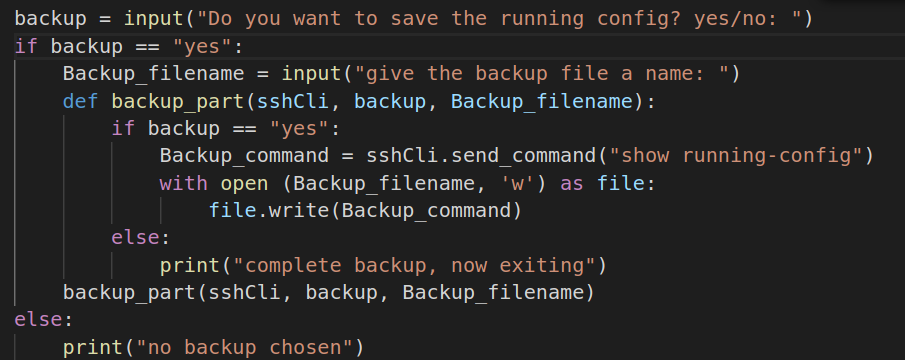


Figure - backup script

In figure 18 shows the script that creates the backup file for the running configuration, the new file new file is created after the user is prompted whether a backup is wanted in the first place. If yes is entered the next prompt asks the user for a new file name. An example of a file backup can be seen with Network Journey (2024).

the command “show running-config” in figure 18 is put within a variable to be written to a file later, which shows the output of the running configuration.

The output of “show running-config” is then put into the new config file, because the new file is created with write permissions, which allows the output to be copied over. By using a new file created earlier now inside the “with open loop” to with the write command.

It is important to have a backup file of the output to make sure all device configuration changes are recorded, encase the device has been reset/changed and needs to revert changes back. It can also help with a maintenance log of all changes made over time.

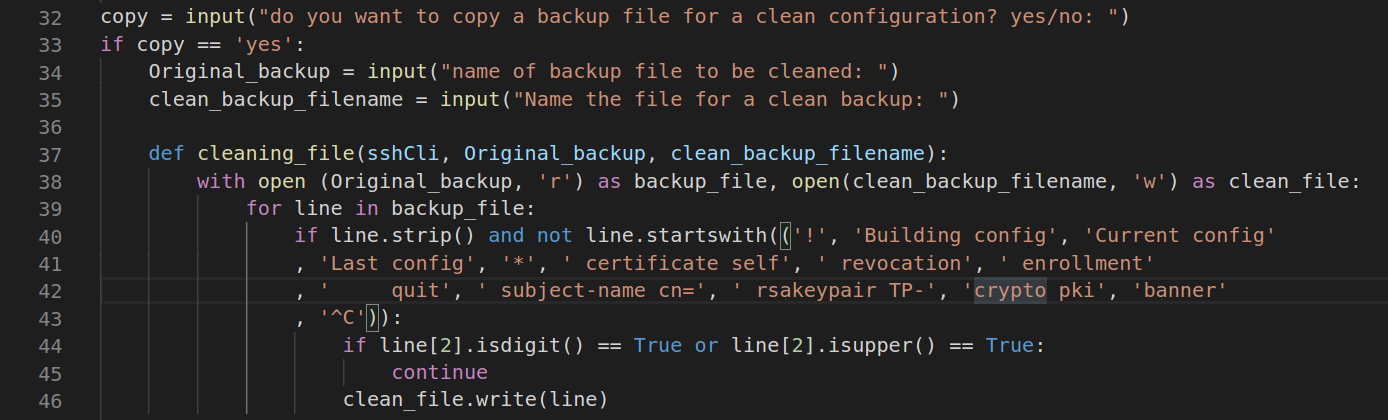


Figure - clean configuration file from backup file

Figure 19 shows, a prompt is sent to ask if a copy of the backup file is needed, another prompt will ask the user for the name of an existing backup file. The third prompt will ask for a name of a new file that will be created. Both the backup file and new file will be opened at the same time, the new file been given write permissions for copying the backup file, with a variable that the script can access similar to GeeksforGeeks (2024).

However, this new file will have an edited version of the backup file, as comments and lines that aren’t commands are removed from the second file. Including any commands that are not required, or cause issues for automation will be ignored in script, by reading every line of the backup file. The if loop in figure 19 shows commands, comments and text that needs to be ignored with the “continue”. Remember all lines that aren’t ignored are written into the second file.

Lastly for figure 19 the if loop containing the “isdigit()” and “isupper()” targets the hash for a crypto pki which does not run within the complete file for “send\_config\_from\_file”, so they are ignored for the full complete second. This why a backup file is also used.

This allows the file to contain the commands from the backup file in the correct format. This can be automatically sent to the device to change the running configuration, which acts as a further backup for specifically commands only.

The third file also uses the timeit from the previous two scripts created, however script 3uses a smaller testing number as there is more configurations pushed.

## Script 4

A screen shot of a computer program

AI-generated content may be incorrect.

Figure - running code to change the startup configuration

Script 4 is also important for the implementation containing two important parts. At the start of the script the file has the same start as the second script which is used to connect to the device this is seen in figure 17.

Figure 20 shows the next a prompt asks if the user wants to use the second file created within script 3, which contains the configuration commands in the correct format from the backup file. Next to update the running configuration of the device after the function is called by “change2”. the “send\_config\_from\_file” sends commands line by line directly from the file chosen by the user.

These changes are then returned from the function being printed to make sure that the command has ran each command successfully. Sending the config from a file is preferred over the list method when it comes to the running config as the formatting of the running configuration isn’t suited for list form.

The startup config part of the script starts with prompt to ask the user to update the startup configuration. This is important because any changes made to the running configuration are not saved because they are only saved on the RAM not the NVRAM, Schrader, D. (2025). if the device is restarted or powered off for whatever reasons, including maintenance or any hardware issues occurring .

Because of this figure 20 shows the variable which contains the “copy run start” command this command is the shorthand version for “copy running-config startup-config” also seen in, Schrader, D. (2025). this command will copy any changes of the running configuration to the startup configuration, which will be tested later, this is also printed to check if the command has run correctly. this command will copy any changes of the running configuration to the startup configuration, which will be tested later. This is also printed to check if the command has run correctly.

"send\_command\_timing” is used to enter the default startup configuration file when the option is available it is chosen with the newline “/n” command.

## Script 5

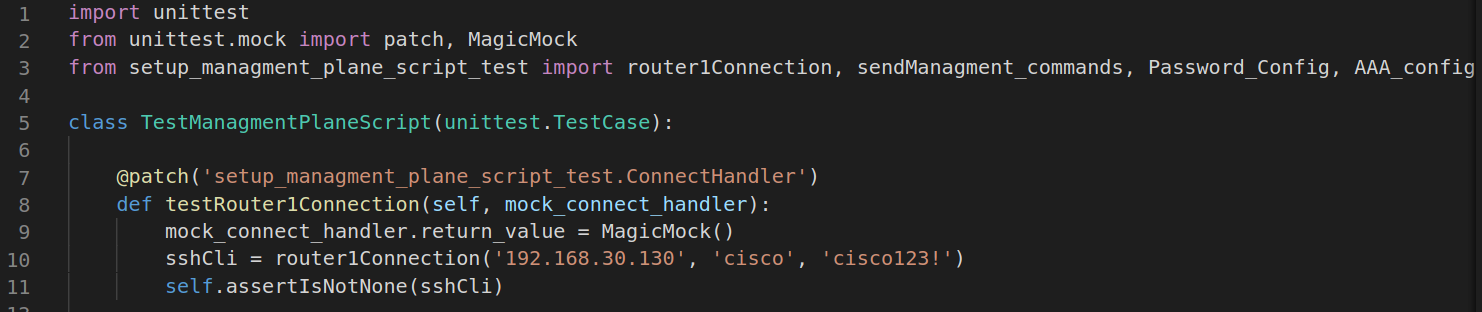


Figure - unit test imports and class

Figure 22 shows the necessary imports needed with script 1 also being imported, this is seen with the last import with all the functions from script , moreover the unit test and mock is imported. script 5 will use the unit test which can be seen in Python (2025a) this will use the class to create multiple test cases for each function which correspond to the functions in script 1 for testing purposes.

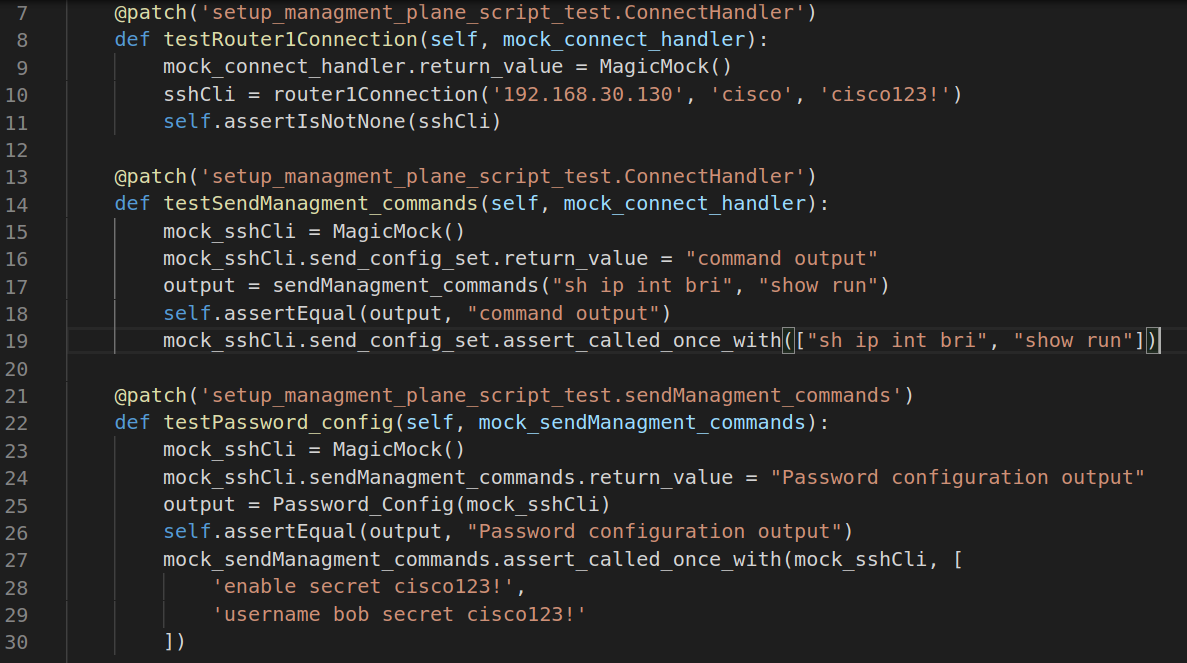


Figure - test functions

Moreover, in Python (2025b) this shows the use of a patch these are responsible for handling all the attributes on a class level. Afterwards the attributes and methods need to be created when accessed “Mock” and “MagicMock” allow for data stores and can be configured this therefore is used for assertions.

in figure 23 “MagicMock” creates the new variable with mock which is called “mock\_sshCli” which uses the “send\_Managment\_commands” variable to push commands to the network device instead of the variable from script 1, there are more functions for the testing of script 1 these are just example of the code used.

# Testing

The testing phase will use the scripts from the implementation, which will be testing against the hypothesis with the script execution time results, as well as testing to see if the lists created in the implementation of each script is successfully pushed to the configuration of the network device. For this timeit is needed to check the script speed this can be seen in, GeeksforGeeks (2025). Timeit allows for a script to be ran a number of times to collect an average script speed.

## Test 1 - Connecting the Devasc VM and the Cisco router VM together

A screen shot of a computer

AI-generated content may be incorrect.

Figure - ping test to check the router is reachable

Figure 24 tests to see that the router can be reached it is important to test this before the ssh connection because if IP is unreachable then SSH also won’t work.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

Figure - SSH connection issues

Figure 25 shows an initial failure of SSH however this is resolved later with the removal of the old RSA key.

A screenshot of a computer

AI-generated content may be incorrect.

Figure - SSH successful login

Figure 26 has to connect the two VMs together with SSH this is because it forms the connection for which Netmiko communicates with the device. This is done by using “ssh [cisco@192.168.30.\*](mailto:cisco@192.168.30.*)” the star is the last number of the ip address this uses the default account which is cisco with the devices IP address. This is done successfully as the prompt at the bottom shows the routers host name “CSR1kv”

## Test 2 – running Script 1

The next test uses the first script of the management plane to run and configure the router to change the running configuration from the base version these changes will be shown in running configuration so show run will be the first test to see if the configurations have been passed, “Appendix – Show run comparison” compares the original running config to the new running config this shows all commands added for the management plane have been added successfully these new commands are seen as blue and underlined.

As seen in “Appendix – Show run comparison” not all commands are found this can be seen with “login local” as this is seen earlier in the AAA configuration which uses the local authorization. Moreover “crypto key generate rsa modulus 2048” is also not saved in the running configuration but is saved in the NVRAM instead for the startup, Cisco. (2017).

### Testing the enable and username passwords

The passwords configurations need to be tested to ensure that the device has password hardening which will help prevent unauthorized access. this is done by using SSH to login into the device with the username and IP address afterwards the command “enable” is to check if a password is added . This can be seen in figure 27, 28 and 29:

A black rectangular object with white text

AI-generated content may be incorrect.

Figure - the enable password testing

A screenshot of a computer

AI-generated content may be incorrect.

Figure - testing username bob configuration

A screenshot of a computer

AI-generated content may be incorrect.

Figure - Admin101 configuration test

### Testing script speed

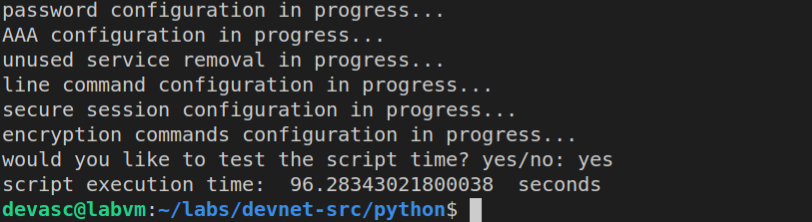


Figure - script 1 execution time average

Figure 30 shows the configurations in progress found in the main() of figure 15, this shows that the configurations have been pushed to an average of 85 seconds. This can help for further optimization however it also shows how fast automation is in network security.

### Testing AAA

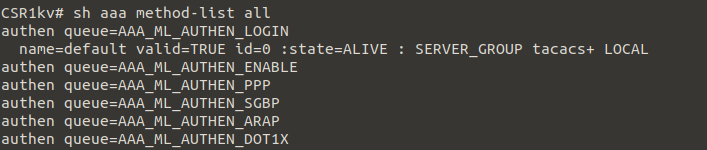


Figure - show aaa method-list all (authentication)

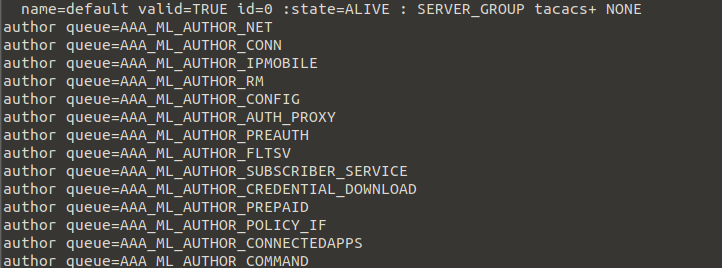


Figure - show aaa method-list all (authorization)

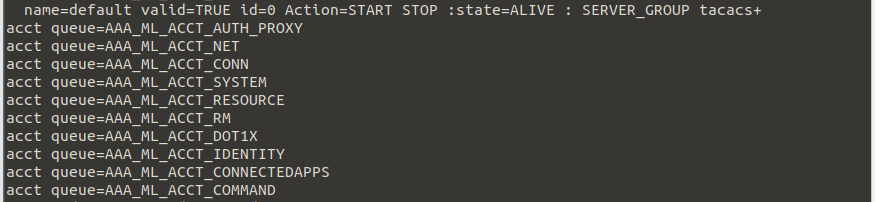


Figure - show aaa method-list all (accounting)

Figures 31, 32 and 33 show that the AAA tacacs server group has successfully been implemented being “ALIVE” moreover the queue show the authentication and authorization, and accounting have been added successfully.

### Testing SSH

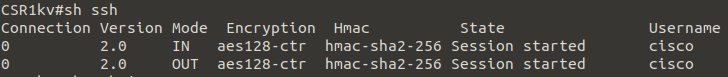


Figure - show ssh

Figure 34 shows that “show ssh” has run and is in use by the user cisco both in and out are on the same connection being zero figure 29 shows an established ssh connection.

## Test 3 – running script 2

In the “appendix -show run comparison” shows that the commands found in script two have been added to the original script this will include all configurations found in the script for the data and control plane like passive interfaces, prefix lists, access-lists, distributed list and key chain to name a few.

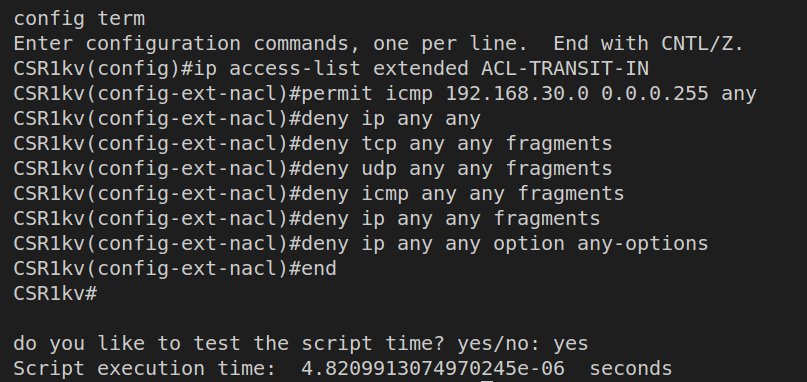


Figure - script 2 speed test

Figure 35 shows some of the config terminal output showing the commands have successfully been pushed with no errors as well as the low script execution time for the commands to be pushed.

### Testing ACLs

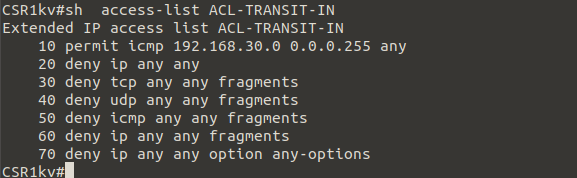


Figure - show access-list

Figure 36 shows the access-list with all the configurations have been successfully added for the correct network shown in the network table in the Artifact design this is also confirmed in the “Appendix – Show run comparison”

## Test 4 – running script 3

The second script will be tested based on the successful creation of two files and both files being populated with the intended content. For the backup file this will contain the unedited form of the command show run and the second file contains a cleaned edited version of the backup file automatic configuration figure 36 shows the two files being created both “full\_config.txt” and “clean\_full\_config.txt”. as well as the script execution time of only 8 seconds

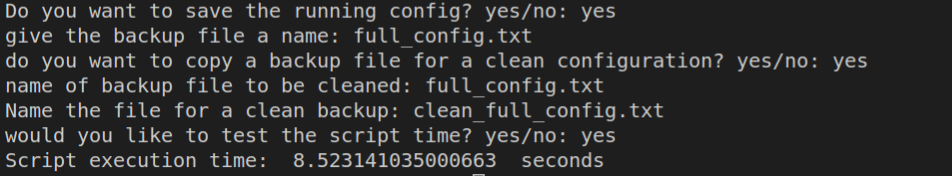


Figure - shows the creation of the two files

### Backup file test

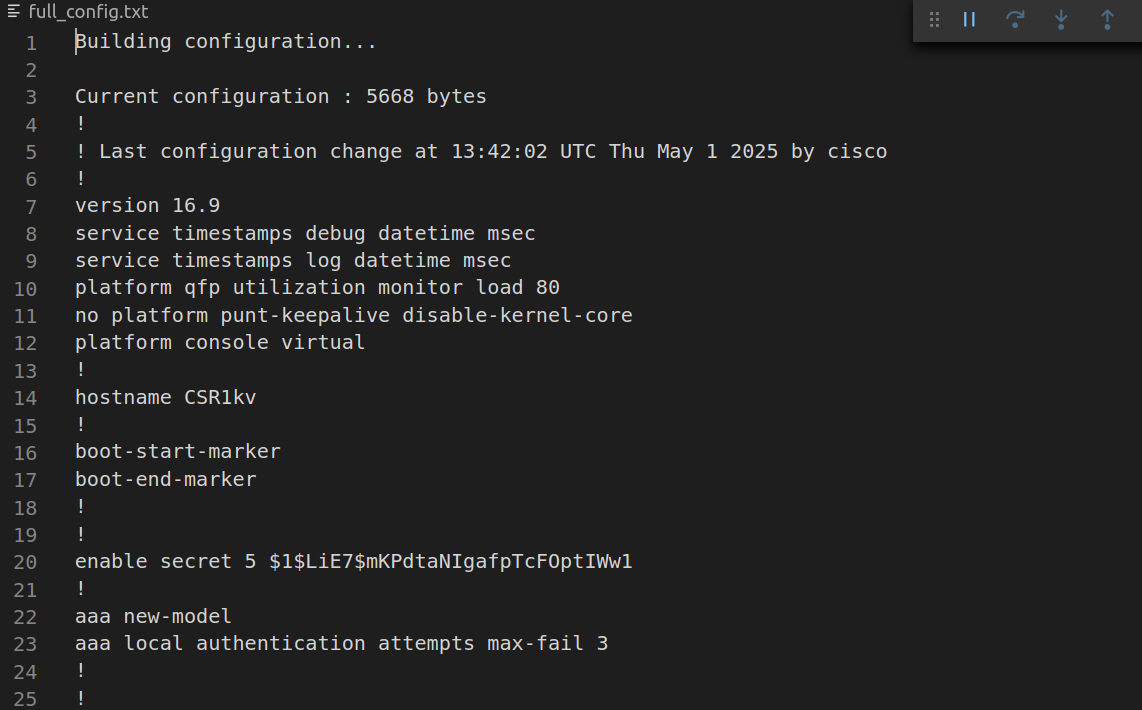


Figure - shows the backup file and its contents

The backup file in figure 37 has been successfully created using the name the user suggested showing the file creation has succeeded. Figure 38 shows that the running configuration has been successfully copied into the file allowing for “full\_config.txt” to be a backup file. This snippet is part of the whole running configuration this has been compared to the original running config in “Appendix – show run comparison”.

### Configuration file test

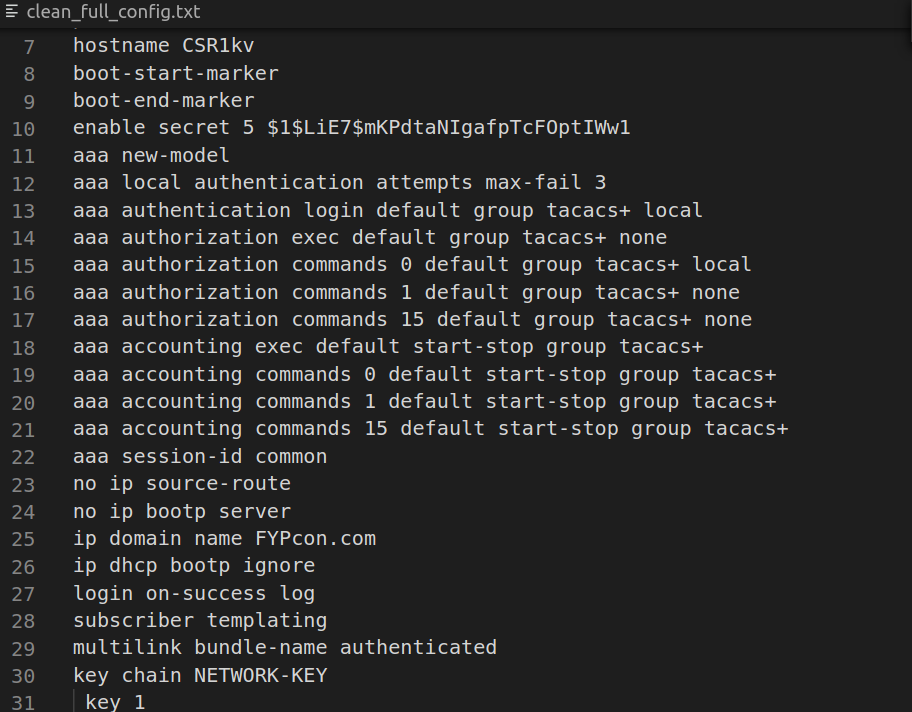


Figure - clean configuration file test

As seen in figure 39 the cleaned configuration file has the correct name from figure 37. this file has successfully removed the unwanted parts of the running configuration inside of the backup file with no commented lines. Moreover, the banner and crypto pki commands are also not found within this file due its effects on script number 3. Also, the absence of “login local” and “crypto key modulus 2048” commands can be added manually to the file if needed.

## Test 5 – running script 4

### Using the configuration to output configuration commands

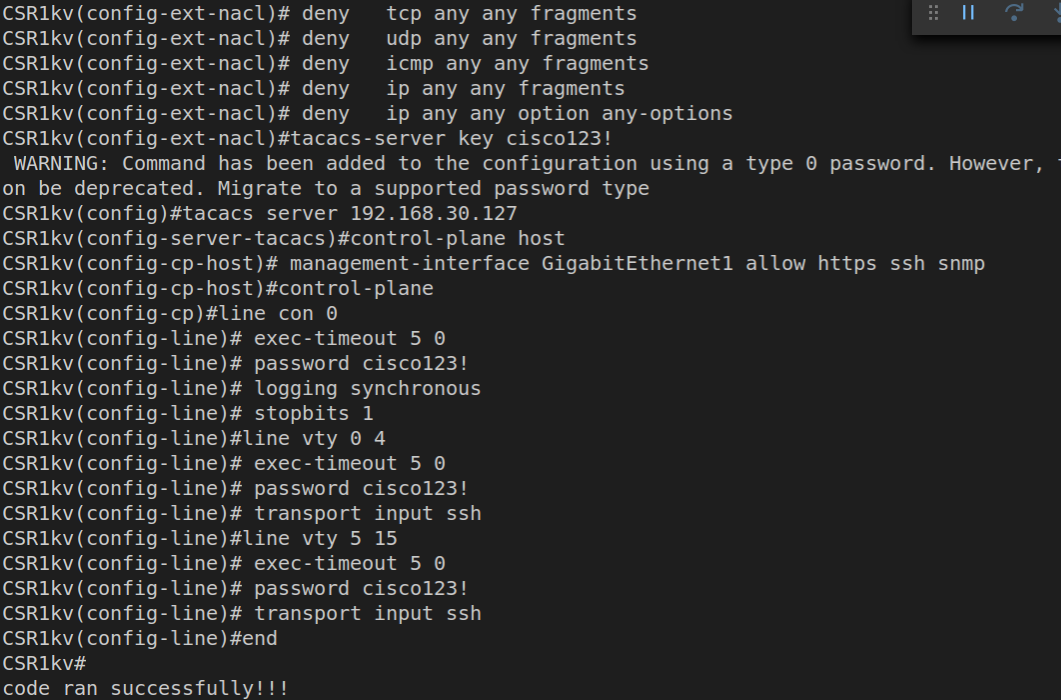


Figure - configuration from file script test

Figure 40 shows the successful testing of the third script the prompt asks which file to use the configuration from which is the same file from earlier this both tests script 1, 2 and 3 as the format of script 2 is correct with no errors found within the output. Moreover the “send\_config\_from\_file” did not timeout due to the lack of banners and crypto pki commands the second file can further be tested manually by restarting the router to remove the current configuration changes made by script 1 and 2. This will then allow for the testing of script 4 to see if the same configurations are added as seen in, table 1, 2 and 3 this can be seen in “Appendix - show run comparison”.

### Updating the startup configuration

A screenshot of a computer

AI-generated content may be incorrect.

Figure - update the startup config test

Figure 41 shows that default destination file name being chosen this means that the changes made in the appendix -show run comparison has been successfully added, moreover this can be tested manually with “show start”, which will show the startup configuration which should contain all the changes made.

As the script four equally passes multiple commands to the network device this also shows up in the section time for the script which is just under a minute.

## Test 6 – running script 5

This test is a simpler test as the unit test file role is to test the management plane script. Because of this when the file is running, if done so correctly no output will be shown after the script has run. Because the script has found no errors within the python code of script 1 meaning the test has been successful.

## Hypothesis vs test results

The hypothesis has been met when it comes to the test execution speed results which showcase how fast that the scripts can run with most scripts running under 60 seconds. It is to be expected that the larger the script however the time taken to run also increases as seen with script 1.The configurations have also been successfully implemented with all but 2 commands not showing up in the running configuration due to the TACACS authentication already being local and secondly the crypto key not being stored in the running configuration.

These commands implemented will harden the network device making it more secure against any threats. And lastly the addition of a clean configuration file has exceeded the hypothesis as it also aids in the maintenance of the network device helping revert changes if needed.

# Conclusion

For this project within network automation for network security the project has hit the aim of the project which was to provide a script which can successfully configure a network with the correct security configurations. Script 3 was made as well to provide an automatic backup of a script, while script 4 used commands in a configuration to configure the running configuration, or the startup configuration if required of a network device. Lastly a file was created to show the testing of script 1 which proved successfully with all lines of the code working as intended all of this will meet the automation goals of the network security.

As well as providing scripts which will help in the maintenance of the network with backups and configuration files. These scripts can be applied to multiple devices which helps with the scalability for example seen the artifact design. This meets project shows the advantages found with automation and network security.

This report however did not cover any live testing on the lab kit with the main focus being on the scripts used. This could be added in for the future works as well as using scripts to test other network devices like switches.

Moreover, as discovered during the testing and the literature review stage the automation tools used for the network security need the network devices to have an IP address already configured, and an open SSH port this means that network automation can only be achieved, if the network device has already been configured manually to have a IP address, which can take some time to do manually to start off with.

Finally, the project successfully showcased the advantages of network automation for security and can easily be scaled up or down to meet the unique requirements that a network can bring.

# References

McKay, M.C. (2023) *In band vs out of band: Understanding the difference - [updated December 2024 ]*, *DigitalGadgetWave.com*. Available at: [https://digitalgadgetwave.com](https://digitalgadgetwave.com/in-band-vs-out-of-band-understanding-the/)/in-band-vs-out-of-band-understanding-the/ (Accessed: 02 December 2024).

Lizhuang, T. *et al.* (2020) *In-band network telemetry: A survey*, *Computer Networks*. Available at: <https://www.sciencedirect.com/science/article/pii/S1389128620313396> (Accessed: 02 December 2024).

Janakaraj, P. *et al.* (2020) *Towards In-Band Telemetry for Self Driving Wireless Networks*. Available at: [https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9162923)=9162923 (Accessed: 02 December 2024).

AlSabeh a *et al.* (2022) *A survey on security applications of P4 programmable switches and a stride-based vulnerability assessment*, *Computer Networks*. Available at: [https://www.sciencedirect.com/science/article/pii/S1389128622000287](https://www.sciencedirect.com/science/article/pii/S1389128622000287#sec8)#sec8 (Accessed: 09 December 2024).

Felix Emmert, F. (2015) *Out-of-band network management*, *Out-of-Band Network Management*. Available at: [https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2015-](https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2015-03-1/NET-2015-03-1_10.pdf)03-1/NET-2015-03-1\_10.pdf (Accessed: 25 November 2024).

Presser, S. (2022) *Towards real- time classification of HPC workloads via out-of-band telemetry | IEEE conference publication | IEEE xplore*. Available at: [https://ieeexplore.ieee.org/document](https://ieeexplore.ieee.org/document/9912703)/9912703 (Accessed: 09 December 2024).

La Lau, R. (2021) ‘Secure Shell (SSH)’, in *Practical Internet Server Configuration*. United States: Apress L. P, pp. 159–174. Available at: [https://learning.oreilly.com/library/view/practical-internet-](https://learning.oreilly.com/library/view/practical-internet-server/9781484269602/html/506968_1_En_7_Chapter.xhtml)server/9781484269602/html/506968\_1\_En\_7\_Chapter.xhtml (Accessed: 09 December 2024)

Lucas, M. (2018) *SSH mastery*, *Google Books*. Available at: [https://www.google.co.uk/books/edition](https://www.google.co.uk/books/edition/SSH_Mastery/xEATEAAAQBAJ?hl=en&gbpv=1&dq=ssh%2Bhow%2Bdoes%2Bit%2Bwork&pg=PP7&printsec=frontcover)/SSH\_Mastery/xEATEAAAQBAJ?hl=en&gbpv=1&dq=ssh%2Bhow%2Bdoes%2Bit%2Bwork&pg=PP7&printsec=frontcover (Accessed: 09 December 2024).

Lerner, K.L. and Lerner, B.W. eds. (2013) 'TCP/IP' in *Computer Sciences*, 2nd ed., vol. 4, Detroit, MI: Macmillan Reference USA, 259-262, available: [https://link-gale-](https://link-gale-com.ezproxy.staffs.ac.uk/apps/doc/CX2761000308/GVRL?u=staffordshire&sid=bookmark-GVRL&xid=2cef2beb)com.ezproxy.staffs.ac.uk/apps/doc/CX2761000308/GVRL?u=staffordshire&sid=bookmark-GVRL&xid=2cef2beb [accessed 11 Dec 2024].

Grigorik, I. (2017) *HTTP protocols*, *O’Reilly Online Learning*. Available at: [https://learning.](https://learning.oreilly.com/library/view/http-protocols/9781492030478/ch04.html#HTTP2_UPGRADE)oreilly.com/library/view/http-protocols/9781492030478/ch04.html#HTTP2\_UPGRADE (Accessed: 16 December 2024).

Ratan, A. (2018) *Practical network automation : a beginner’s guide to automating and optimizing networks using python, ansible, and more*. Second edition. Birmingham ; Packt Publishing. Available at: <https://learning.oreilly.com/library/view/practical-network-automation/9781789955651/58efb295-5fcf-4902-a1c2-c0b8a5e41586.xhtml> (Accessed: 16th December 2024)

Nevis (2016) *An architectural view of LAN security: In-band versus out- ...*, *Nevis*. Available at: [https://nevisnetworks.](https://nevisnetworks.com/wp-content/uploads/sites/7/2016/09/InBand-vs-OutofBand.pdf)com/wp-content/uploads/sites/7/2016/09/InBand-vs-OutofBand.pdf (Accessed: 09 January 2025).

Okasha, K. (2020) *Network automation cookbook*, *O’Reilly Online Learning*. Available at: <https://learning.oreilly.com/library/view/network-automation-cookbook/9781789956481/d1288c15-7a4d-49d9-9b50-27e3078f30db.xhtml> (Accessed: 09 January 2025).

Meijer, B., Hochstein, L. and Moser, R. (2022) *Ansible*. 3rd edn. Sebastopol: O’Reilly Media, Incorporated. (Accessed: 04 April 2025)

Cisco (2024) *Harden IOS devices*, *Cisco*. Available at: <https://www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html> (Accessed: 20 January 2025).

Sen, K. (2025) The Pros and cons of Ansible: Upguard, RSS. Available at: <https://www.upguard.com/blog/top-5-best-and-worst-attributes-of-ansible> (Accessed: 04 April 2025).

Oswalt, M. et al. (2023) Network Programmability and Automation. 2nd edition. Sebastopol: O’Reilly Media, Incorporated. Available at: <https://learning.oreilly.com/library/view/network-programmability-and/9781098110826/ch10.html#id270> (Accessed: 4th April 2025)

Byers, K. (2024) Netmiko, PyPI. Available at: <https://pypi.org/project/netmiko/> (Accessed: 05 April 2025).

Gao, H. et al. (2022) Coherence analysis of symmetric star Topology Networks, Frontiers. Available at: <https://www.frontiersin.org/journals/physics/articles/10.3389/fphy.2022.876994/full> (Accessed: 07 April 2025).

cisco (2015) *Basic router configuration*, *Cisco*. Available at: <https://www.cisco.com/c/en/us/td/docs/routers/access/800M/software/800MSCG/routconf.html> (Accessed: 21 April 2025).

Cisco (2007) *Configuring System Message Logging*, *Cisco*. Available at: <https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst2960/software/release/12-2_40_se/configuration/guide/scg/swlog.pdf> (Accessed: 21 April 2025).

Network Journey (2024) *Cisco Netmiko scripting with examples: A comprehensive guide*, *Network Journey*. Available at: <https://networkjourney.com/cisco-netmiko-scripting-with-examples-a-comprehensive-guide/> (Accessed: 25 April 2025).

Schrader, D. (2025) *How to manage the running configuration file for Cisco Devices*, *How to copy a Cisco Running Config to startup config to preserve configuration changes*. Available at: <https://blog.netwrix.com/copy-running-config-to-startup-config-cisco/> (Accessed: 25 April 2025).

GeeksforGeeks (2024) *How to open multiple files using*, *GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/how-to-open-multiple-files-using-with-open-in-python/> (Accessed: 26 April 2025).

Cisco (2017) *crypto key generate rsa*, *Cisco*. Available at: <https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/security/a1/sec-a1-xe-3se-3850-cr-book/sec-a1-xe-3se-3850-cr-book_chapter_00.pdf> (Accessed: 26 April 2025).

Merkow, M.S. (2019) *Secure, Resilient, and Agile Software Development.* 1st ed. Milton: Auerbach Publishers, Incorporated. Available at: <https://doi.org/10.1201/9781003025399>. (Accessed: 27 April 2025)

McCormick, M. (2012) *Waterfall vs. Agile Methodology - McCormick PCS info*, *mccormickpcs*. Available at: <http://mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf> (Accessed: 27 April 2025).

Srivastava, A., Saraswat, S. and Bhardwaj, S. (2017) *SCRUM model for agile methodology*, *IEEE*. Available at: <https://ieeexplore.ieee.org/abstract/document/8229928?casa_token=Za-gNBOgbf0AAAAA:12CwG6LaBbR_4Jz9afZvz4-vAphJtNylVx7HPczaNhnKMKDf99p89dT8eASoj0QNXUJ3jbaj9Q> (Accessed: 27 April 2025).

Senarath, U.S. (2021) *Waterfall methodology, prototyping and agile development*, *ResearchGate*. Available at: <https://www.researchgate.net/profile/Udesh-S-Senarath/publication/353324450_Waterfall_Methodology_Prototyping_and_Agile_Development/links/60f41f71fb568a7098b9d035/Waterfall-Methodology-Prototyping-and-Agile-Development.pdf> (Accessed: 28 April 2025).

Aguiar, G. (2024) *(PDF) distinguishing between method and methodology in academic research*, *ResearchGate*. Available at: <https://www.researchgate.net/publication/373826835_Distinguishing_between_Method_and_Methodology_in_Academic_Research> (Accessed: 28 April 2025).

Em, S. (2024) *(PDF) exploring experimental research: Methodologies, designs, and applications across disciplines*, *ResearchGate*. Available at: <https://www.researchgate.net/publication/378795357_Exploring_Experimental_Research_Methodologies_Designs_and_Applications_Across_Disciplines> (Accessed: 28 April 2025).

Jansen, D. (2025) *Saunders’ research onion explained (+ examples)*, *Grad Coach*. Available at: <https://gradcoach.com/saunders-research-onion/> (Accessed: 29 April 2025).

Python (2025a) *Unittest - unit testing framework*, *Python documentation*. Available at: <https://docs.python.org/3/library/unittest.html#basic-example> (Accessed: 01 May 2025).

Python (2025b) *Unittest.mock - mock object library*, *Python documentation*. Available at: <https://docs.python.org/3/library/unittest.mock.html> (Accessed: 01 May 2025).

GeeksforGeeks (2025) *Timeit in python with examples*, *GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/timeit-python-examples/> (Accessed: 01 May 2025).

# Appendix

### Ethics form

A close-up of a form

Description automatically generated

Figure - Ethics form

### Gannt charts

A screenshot of a gantt chart

Description automatically generated

Figure - original Gannt Chart

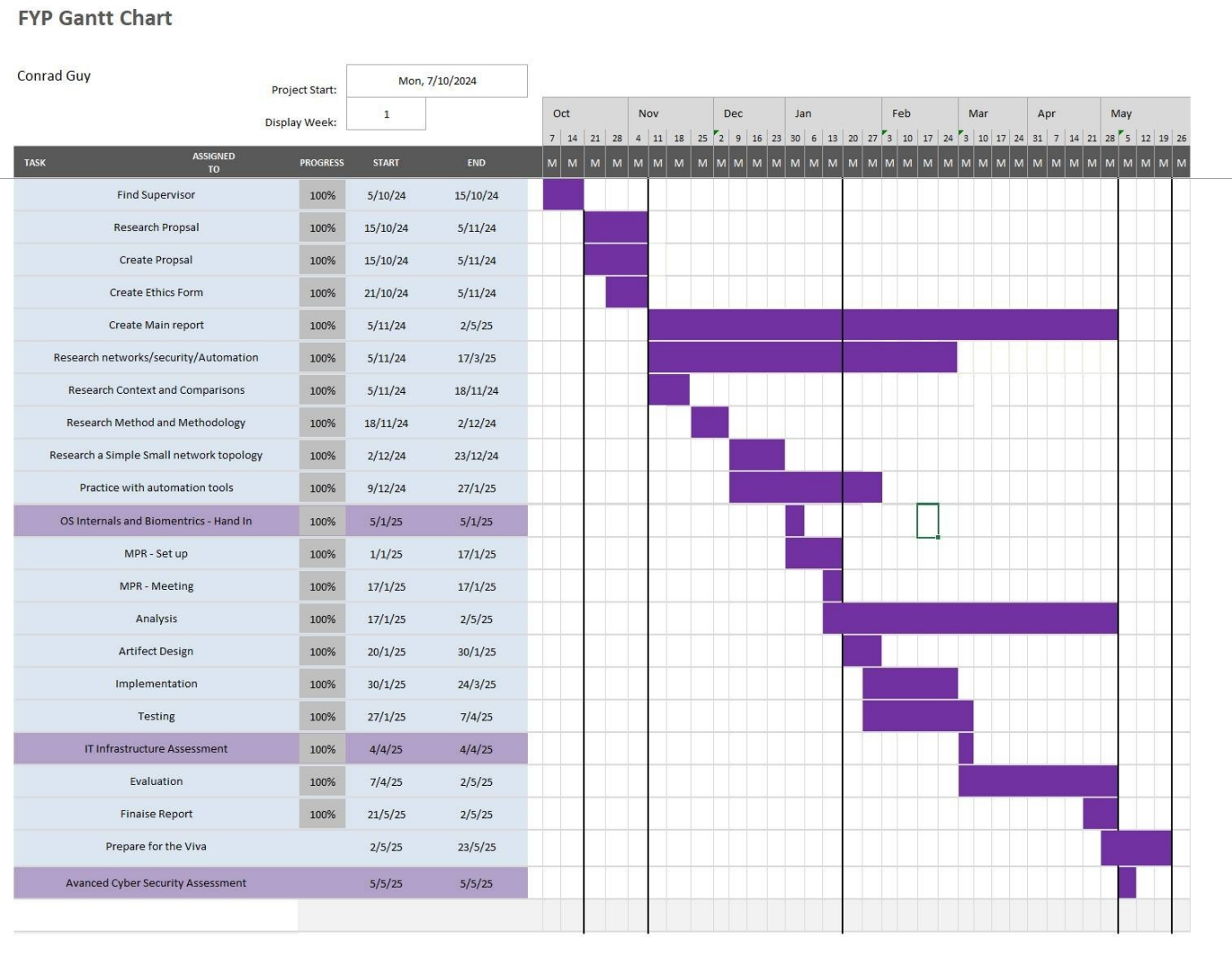


Figure - updated Gannt Chart

### Risk Assessment table

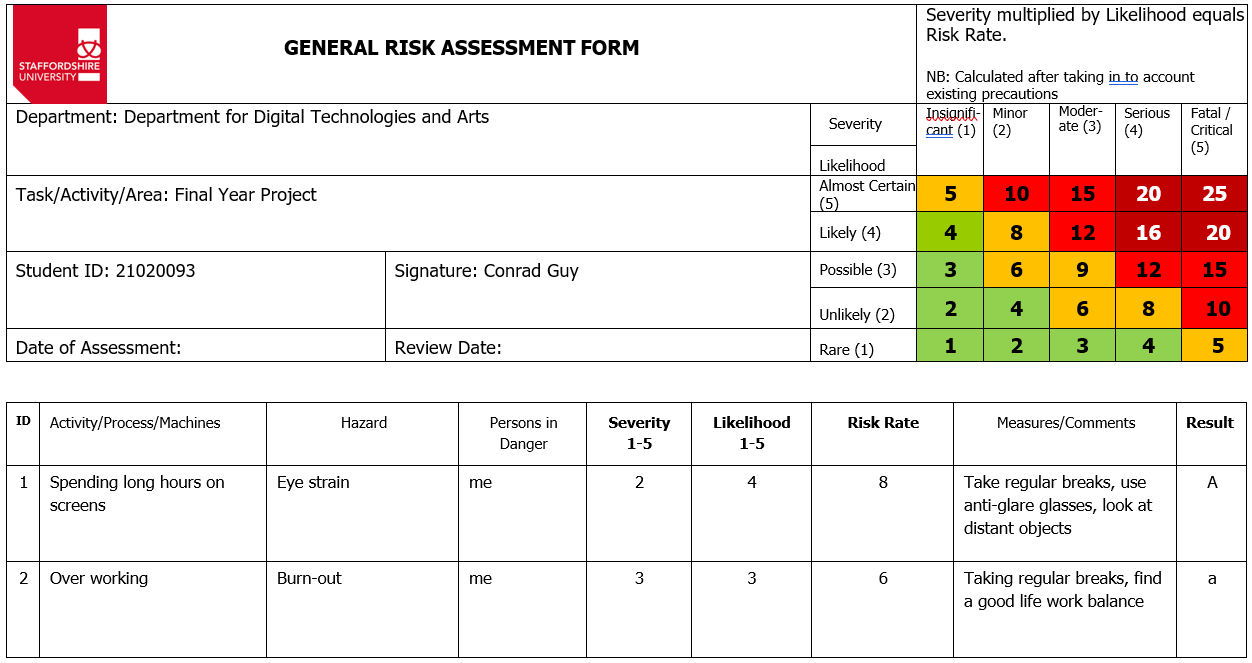


Figure - risk assessment part A

**A white sheet of paper with black text

AI-generated content may be incorrect.**

Figure - Risk Assessment part B

### Log books

A screenshot of a computer

AI-generated content may be incorrect.

Figure - log book 1

A screenshot of a computer

AI-generated content may be incorrect.

Figure - log book 2

A screenshot of a computer

AI-generated content may be incorrect.

Figure - log book 3

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 6

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 6

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 8

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 21

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 22

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 23

A screenshot of a computer

AI-generated content may be incorrect.

Figure - week 24

### Appendix - Show run comparison

Building configuration...

Current configuration : 5669 bytes

!

! Last configuration change at 00:14:19 UTC Wed Apr 30 2025 by cisco

!

version 16.9

service timestamps debug datetime msec

service timestamps log datetime msec

platform qfp utilization monitor load 80

no platform punt-keepalive disable-kernel-core

platform console virtual

!

hostname CSR1kv

!

boot-start-marker

boot-end-marker

!

!

enable secret 5 $1$nP4O$RzCjzwl69Am2RpZ32Gc7t/

!

aaa new-model

aaa local authentication attempts max-fail 3

!

!

aaa authentication login default group tacacs+ local

aaa authorization exec default group tacacs+ none

aaa authorization commands 0 default group tacacs+ local

aaa authorization commands 1 default group tacacs+ none

aaa authorization commands 15 default group tacacs+ none

aaa accounting exec default start-stop group tacacs+

aaa accounting commands 0 default start-stop group tacacs+

aaa accounting commands 1 default start-stop group tacacs+

aaa accounting commands 15 default start-stop group tacacs+

!

!

!

!

!

aaa session-id common

no ip source-route

!

!

!

!

!

!

!

no ip bootp server

ip domain name FYPcon.com

ip dhcp bootp ignore

!

!

!

login on-success log

!

!

!

!

!

!

!

subscriber templating

!

!

!

!

!

multilink bundle-name authenticated

!

!

!

!

!

key chain NETWORK-KEY

key 1

key-string cisco123!

!

crypto pki trustpoint TP-self-signed-2450887421

enrollment selfsigned

subject-name cn=IOS-Self-Signed-Certificate-2450887421

revocation-check none

rsakeypair TP-self-signed-2450887421

!

!

crypto pki certificate chain TP-self-signed-2450887421

certificate self-signed 01

30820330 30820218 A0030201 02020101 300D0609 2A864886 F70D0101 05050030

31312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274

69666963 6174652D 32343530 38383734 3231301E 170D3235 30343239 31353133

32355A17 0D333030 31303130 30303030 305A3031 312F302D 06035504 03132649

4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D32 34353038

38373432 31308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201

0A028201 0100CB95 79AB3B0A 6C1249BA 436E87D0 E434CEB1 6200CA97 A4D0BBF0

EAB6B3C6 79F6D250 2245A8DB B5BC8CBF C38CC52C 5172752B 94FA31BF B080C1C4

541334FC FD210F2B E26FAB77 2ACFA831 5231E14A 27823E39 67A6AB3F 64CE57FA

76A38705 198C58EA 8B62C816 E8E2D651 ABDC08CF 5EAD128B C30A188E 9EFC8715

E76E4A18 E4804890 980C5DF1 618AA90D A7644C36 2FAAD12F F5DFE1C5 F7452939

7727042B C6C9D9A9 5FC69CF0 DEB75DC6 2A4E8E56 65BCDD3D D7FFB742 E2923212

AD66B05B 42FDF16D AC61CE63 9874B65F F4C6557D FD3E107C 8A61CCCE 25622A54

E416E868 D1A4D36D EC297616 E5071C35 3CD7D126 3CF6AB18 63EFBAB3 A6247C04

2EFB49DF 119F0203 010001A3 53305130 0F060355 1D130101 FF040530 030101FF

301F0603 551D2304 18301680 14A3FBDB 19243D25 253D3BA2 9831EF98 48EDF99B

34301D06 03551D0E 04160414 A3FBDB19 243D2525 3D3BA298 31EF9848 EDF99B34

300D0609 2A864886 F70D0101 05050003 82010100 1ABCE267 3692C945 E065E341

DFD83AC8 706AE402 7662F7BE C68EE32C 92AD138A 2CE8A487 E9D80CD4 689273BE

D7B3D96B 691807C4 7680E624 12649C29 04BF551C 93DEFE2F 15DD6B8B C18929FB

E24E3B26 4CF3918E F1033F21 DB447885 8C43993F 433DB96F 9A5344AE 9DBFA049

765D6F10 C82A6F6A 0F611737 D9DC971C E6148DF4 4B9D4024 5FFF750D C0935D52

D728AB13 8311A07A ECD97547 DC066597 F675F55E EC5F19DD 6394D64E 88A6B5A3

2339E275 A19EFAEF D53066F1 BE3B3F5D B302C984 D6093119 67EEB2B5 E118CAD1

00EAA5F2 0EA21AF9 97BB02B5 7FDDEA69 5352A7BB 85707CF5 910748B7 5FC1523A

02FFB85E 44DCE87C 398A4801 0E31489C E4E709CC

quit

!

!

!

!

!

!

!

!

license udi pid CSR1000V sn 9AX31ZC3T3R

no license smart enable

diagnostic bootup level minimal

!

spanning-tree extend system-id

!

netconf-yang

!

restconf

!

username cisco privilege 15 password 0 cisco123!

username bob secret 5 $1$wcMR$UXTkZ3dcejQkO1Sr09gd31

username Admin101 privilege 15 password 0 cisco123!

!

redundancy

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

interface GigabitEthernet1

description VBox

ip address dhcp

ip authentication mode eigrp 1 md5

ip authentication key-chain eigrp 1 NETWORK-KEY

negotiation auto

no mop enabled

no mop sysid

!

!

router eigrp 1

distribute-list NETWORK-PREFIX out GigabitEthernet1

network 192.168.30.0

passive-interface default

no passive-interface GigabitEthernet1

!

ip forward-protocol nd

no ip http server

ip http authentication local

ip http secure-server

!

ip ssh time-out 90

ip ssh source-interface GigabitEthernet1

ip ssh version 2

ip scp server enable

!

!

ip prefix-list NETWORK-PREFIX seq 10 permit 192.168.30.0/24

!

ip access-list extended ACL-TRANSIT-IN

permit icmp 192.168.30.0 0.0.0.255 any

deny ip any any

deny tcp any any fragments

deny udp any any fragments

deny icmp any any fragments

deny ip any any fragments

deny ip any any option any-options

!

!

tacacs-server key cisco123!

tacacs server 192.168.30.127

!

!

!

control-plane host

management-interface GigabitEthernet1 allow https ssh snmp

!

!

control-plane

!

!

!

!

!

banner motd ^C

\* \*

\*\* \*\*

\*\*\* \*\*\*

\*\*\* Cisco Networking Academy \*\*\*

\*\*\* \*\*\*

\*\*\* This software is provided for \*\*\*

\*\*\* Educational Purposes \*\*\*

\*\*\* Only in Networking Academies \*\*\*

\*\*\* \*\*\*

\*\* \*\*

\* \*

^C

!

line con 0

exec-timeout 5 0

password cisco123!

logging synchronous

stopbits 1

line vty 0 4

exec-timeout 5 0

password cisco123!

transport input ssh

line vty 5 15

exec-timeout 5 0

password cisco123!

transport input ssh

!

!

!

!

!

!

end