ERSC ENGENHARIA DE REDES E SISTEMAS DE COMPUTADORES ESTG-IPVC

Network Automation Using Python and Ansible

Submitted on 14/07/2025
Project 2, authored by
Eduardo Miguel Moreira Junqueira (eduardo.j@ipvc.pt),
supervised by
Prof. Nuno Torres (nunotorres@ipvc.pt).

***Abstract*—The context of this project is to develop an automated system for network configuration management using Python and Ansible. In order to reduce manual configuration time, as well as minimize the occurrence of errors and attacks in the configuration of devices. The initial stage includes investigating, testing, and evaluating. A software company network is later created and documented according to an established detailed structure. Using Python and Ansible scripts to automate network configuration is part of the second phase. To manage configuration changes across several hardware devices simultaneously, new Ansible playbooks will also be generated by Python scripts. The proposal is to compare the manual configuration with the automated configuration network with the same topology but with different use cases to show the aim of automating the network. A comparison of automated and manual software company network processes will be implemented, and its results will be presented in different test cases. The Network and Information Security Directive 2 (NIS2), represents a higher level of cybersecurity and continues today to emphasize the provision of a flexible and adaptable framework for enterprises to manage security risks in The European Union (EU).By supporting collaboration, integration regulations, and prioritizing responsibility, it creates a strong digital ecosystem and international collaborations. Comparing the manual and automated network to prove some regulations about this directive and explaining the differences in a table overview is an additional task to prove. Finally, by verifying compliance with its safety guidelines, the project aims to show the resulting manual documentation, configuration, and automated deployment tests.**

***Index Terms*—software company network, manual network, network automation, Python scripting, ansible, NIS2.**

I. Introduction

This paper contains different sections that explain the whole structure of the project. This paper aligns the template of the course Computer Systems and Network Engineering (ERSC) of the school Institute Polytechnic Viana do Castelo (IPVC)[1].The view share for this article is the following reference [2], and the documentation that will be necessary to do the following article is the following reference [3].All data about this project is on the following GitHub repository [4].

Initial Section **Abstract:** A brief summary of the introduction. Section I **Introduction:** Analysis of the project core definitions. Section II **Related Works:** Explains all the software explored, programming language and directive documentation. Section III **Methodology:** Theoretical and practical methodology implemented in this project . Section IV **Implementation:** Creation of Software Company Network structure and manual implementation. Python scripts will enable the configuration of manual for automated approaches. Section V **Validation and Results:** Generate the automated network, final overview of the automated structure, demonstration and execution with the final overview of the creation network with different use cases. Section VI **Discussion:** Considerations and feedback about the previous section V, with some verification about the NIS2 directive checklist impacts. Section VII **Conclusion:** Final summary of article, database and acronyms. Section VII **References:** Bibliography of the entire project.

A. The project proposal

This project aims to automate the initial setup and ongoing management of network devices using Python scripts and Ansible playbooks.

Currently, 95% of network operations are completed manually, which costs a lot of money, effort, and time consuming [5].

Automation will reduce manual effort, minimize configuration errors, and ensure a consistent application of security and operational policies throughout the network.

Using industry best practices and testing in simulated environments, the project will demonstrate significant gains in efficiency, scalability, and compliance [5].

This paper presents the differences between manual vs. automated networks in a previously planned structure with different tests.

These differences are crucial to obtain good results to prove some regulation of the NIS2 directive. It is also crucial to understand how NIS2 impacts network security practices.

Python scripts and Ansible will explore how scripts, and playbooks simplify network configuration, ensuring automation, scalability, and reducing human error.

Automation is the key to the future of the network. For that, a proposal is the motivation to do this project.

B. The project overview and objectives

The objective is to create Python scripts and Ansible playbooks that automate and configure the network devices (such as routers, switches, PCs, and firewalls). This automation targets:

- Reducing manual labor and error rates during device configuration [5].
- Ensuring consistent application of security and operational policies [5].
- Enabling scalable and repeatable network deployments [5].
- Demonstrate efficiency gains by comparing manual and automated processes [5].

C. Problem Statement and Motivation

An automatize network it's a problem because all scripts behind in "backend", represent an automated network that needs some software available to adapt these scripts and implement them in different cases of use. For big network structures can be very hard to implement, because they consume a lot of memory, space, code and time to automatize all of this. For a comparative process is require also a manual network configuration that spends more time and more cost than automatize network.

II. Related Works

A. Manual Network Configuration

Small software organizations independently financed and organized companies with fewer than 50 employees are fundamental to many national economies' growth. In the US, Brazil, Canada, China, India, Finland, Ireland, Hungary, and many other countries, small companies represent up to 85% of all software organizations. However, to persist and grow, small software companies need efficient, effective software engineering solutions[6].

The International Organization for Standardization set up a working group on life-cycle profiles, which refers to the gradual development and evolution of a software system for Very Small Enterprises (VSEs) companies with fewer than 25 employees. This group is establishing a common framework for describing assessable life-cycle profiles used in VSEs.

It aims to let VSEs be recognized as producing quality software systems without the initial expense of implementing and maintaining an entire suite of systems and software engineering standards or performing comprehensive assessments [6].

B. Cisco Packet Tracer (CPT)

There is an evident difference between virtualization of network devices and simulation.

In this project the simulation process will be implemented, the virtualization isn't the aim of the project.

While simulators can be a helping hand in the initial learning stages, they become limited when the network topologies get more complex, both in size and in the protocols involved, since only a subset of features are usually available, in this case the CPT [7].

The principal manual software network for this project is CPT which is a comprehensive teaching and learning tool to build simple and complex networks in different contexts like academic, studying, and Cisco certification[8].

But these software is a simulator of manual network where all configurations are manual by the user. There

would be possible automation, but for that the network would have to be created first and then devices would have to be connected to each other and enable Secure Shell (SSH) communication .

This tool allows to use only Cisco equipment and not other equipment, thus making the network limited to the Cisco system. These devices are configured with Cisco Configuration Commands Command Line Interface (CLI).

To use other hardware equipment would have to create Virtual Machines or containers in Docker, but it is not possible, since there is no support yet for automated scripts configuration.

So,CPT will serve as a comparison for the manual process of creating a prototype network. At the end of this project this will serve as a response time comparator for the configuration and communication network.

The Networking Academic by Cisco is for all Cisco Networking Academic instructors, students, and alumni, in the figure 1, is a example of simulator.

Fig. 1. This is a figure about the Networking academy that supports the CPT, open-source software for Schools and Cisco workers [9].

C. Automation Network

Traditional network management relies heavily on manual device configuration by input command by command. While this approach provides low-level control, it becomes increasingly inefficient for complex networks, because is very difficult to replicate consistently across multiple devices, which increases the risk of a huge failure network and operational failures[10].

An example will be such if the network is down and the services are in the cloud also for a reason like cyber attack or another reason, manual configuration for a huge network with a lot hosts and there's a big network more extensive than VSEs can take hours or days to recover again.

To resolve this limitation, network automation has emerged as a critical component in modern network operations. To validate the automation strategy, scripts in Python and playbooks in Ansible are essential to demonstrate the efficiency of automation by comparing manual and automated configuration processes in terms of execution time and error rate.

Therefore, automation would be much more efficient and faster with the process of realization.

Automation of network configuration has been used more in processes that spend a lot of time. It is a method that helps with scripts and AI can be used for a big and complex network database.

For the creation of automated network, the first step is better understand a schematic of the overall summary configuration automated network in figure: 2.

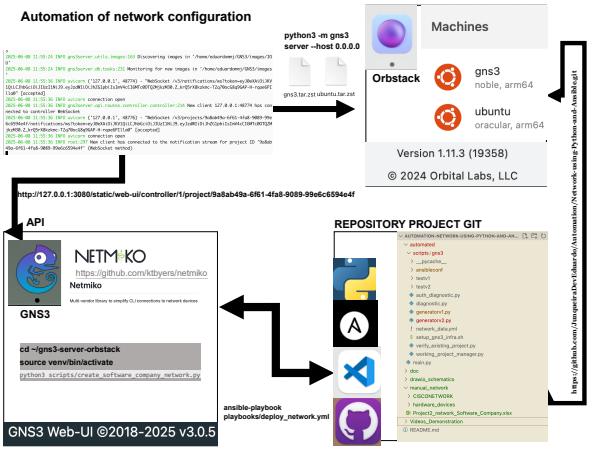

Fig. 2. Schematic automated tree network knowledge[9].

Automation of a network is a stand-alone process that allows to create, configure, and manage network devices such as (router, switch, and host) without continuous manual intervention. Generally, this process is a repetitive process that, with scripting, will be autonomous[10].

As the volume of business and the number of servers increase, these issues become progressively more pronounced. Python scripts and batch management tools have been integrated into the server operation and maintenance process, significantly improving the efficiency of maintenance personnel [10].

D. Graphical Network Simulator-3 (GNS3)

The second simulator for this project is GNS3 which now allows the automated emulation, configuration, test-

ing, and troubleshooting of virtual and real automated networks and can simplify the network by operations and automation [11].

The creation and management of network nodes is done via GNS3, while the automation of configurations such as Internet Protocol (IP) assignment and SSH services is carried out with Ansible playbooks in Python scripts [12].

GNS3 is in an Open-source cloud computing platform (OpenStack) environment, this project adapts the automation logic to a simpler and more accessible environment with GNS3.

One of the most used applications to automate Cisco networks[7]. Cloud-based platform (GitHub) repository[11] documentation will be important for all the releases of GNS3.

In this project GNS3 on Apple Silicon Chip ARM M2 (macOSM2), is just possible download software and containers with Advanced RISC Machines (ARM). So GNS3 *v3.0.4*, will be tested for the first month deadline project, and then a new version *v3.0.5*, that came out on 14/05/25 with Graphical User Interface (GUI) for ARM64 will be used until the final delivery of project deadline.

GNS3 on macOSM2 will be a problem, but buy a new computer windows for this project isn't a solution. So the idea was try to realize this project with ARM64 architecture.

The Logo below is GNS3,free software and is supported by 800.000 members in the figure: 3.

Fig. 3. This is a figure about the logo of GNS3 that will be used in automated network[9].

A shorter explanation of most differences between GNS3 and CPT are in a Table I II-D.

TABLE I
DETAILED COMPARISON BETWEEN GNS3 AND CPT

Feature	GNS3	CPT
Operating System Support	**Yes** — GNS3 can run real network operating systems, including Linux Virtual Machine (VM), Docker containers, and official Cisco images.	**No** — CPT is limited to simulated Cisco IOS environments and does not support running real operating systems or third-party VM.
SSH	**Yes** — GNS3 supports full SSH connectivity, enabling the use of automation tools like Ansible and Python scripts.	**No** — CPT does not provide real SSH access to devices, so it cannot be used with Ansible or other external automation tools.
Ability to Run Real Services	**Yes** — Users can install and run real network services (e.g., SSH, web servers, monitoring agents) on devices within GNS3, providing a highly realistic lab environment.	**No** — CPT is limited simulated services and cannot run real software or external applications on devices.
Support for Ansible/Python Scripting	**Yes** — GNS3 allows execution of Ansible playbooks and Python scripts against real or emulated devices, enabling advanced automation and testing scenarios.	**No** — CPT does not support remote execution of Ansible or Python scripts.
Integration with Linux Tools	**Yes** — Full compatibility with Linux command-line tools and utilities. Users can install any required packages, making GNS3 suitable for end-to-end workflow testing.	**No** — CPT cannot integrate with external Linux tools or external packages, limiting its use to predefined functionalities.

E. Virtual Machine Software (VMware)

For ”Simplify Network Operations and Automation” by ”Broadcom” [13], was the perfect open-source software to use in AMR architecture in the macOSM2 that can be configured for network operations and automation in the figure 4.

Fig. 4. This is the logo of software was choose to implement in change of VMware [9].

F. Virtual Machine/Container Orb Stack

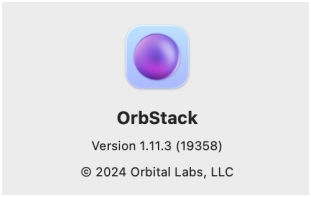

Fig. 5. This software it’s supported by GNS3 and Ubuntu *v3.0.4*remote Virtual Machines [9].

Two months to finished the project a new tool OrbStack in the figure 5 was the best approach to use and finalize the practical part. This is a platform for building Linux containers and virtual machines on macOSM2 , with native support for Apple Silicon chips M1, M2(my case), M3, and M4. It replaces tools like Docker Desktop or traditional VM, but with much more efficiency, speed, and simplicity. OrbStack is like a lightweight VMware that allows to create and manage several instances of Ubuntu (or other Linux systems among others), as if they were small servers[14].

The Orbstack allowed to: **I:** The creation of Linux containers Ubuntu in seconds [15]. **II:** The installation and execution of the GNS3 server in a stable manner[15]. **III:** Real network connectivity with assigned IPs, essential for communication with the GNS3 GUI[15]. **IV:** The use of tools such as Ansible and Python within containers and Simulation of Cisco devices connected virtually to the Linux container with some limitations of ARM versions[15]. **V:** The simultaneous execution of different services in separate environments (e.g., GNS3, Ansible, and others).Run Cisco images such as ”IOSv, vIOS, IOU” (via GNS3)[15]. **VI:** OrbStack was instrumental in avoiding the use of heavy VM like VMware, enabling higher performance and resource savings. For a device that are used macOSM2 with 8GB RAM can be a problem when the memory is full and the computer needs to swap memory and turns slow [15].

G. Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics[16]. In this project will be important first install the proper libraries to run the scripts Python with the correct libraries. So, the following commands are specific for each type of library that will be needed to run the script.

python3 -m pip install pyyaml : read files *.yml*[16];

Python3, Virtual Python Module (venv), Python Package Index (pip), and Netmiko GitHub. [16];

[16];

The scripts in Python have the aim to:

- **Verification;**
- **Implementation;**
- **Generation;**
- **Validation;**

H. Ansible

Ansible is an open-source automation framework for a wide range of applications. The most important feature of this automation approach is works without an agent, so there isn’t require to install additional software [17].

This means there does not have to be a service or process in the background. The main building block of the Ansible architecture, the Ansible Automation Engine, consists of the inventory, the modules, the application programming interface Application Programming Interface (API) and plugins. The inventory is used to prepare and build a network. The API provides the inter-operation of end-to-end modules. The modules are the automated executable units that are controlled by means of the playbooks. The users create the so-called playbook and controls the automated process of the Automation Engine. Playbooks are created in the YAML Ain’t Markup Language (YAML) markup language to also. In order to run the automation from a cloud, there is a direct connection of the cloud with the Automation Engine.[18].

Ansible typically comprises six key components, namely: Ansible, Ansible Playbook, Inventory, Modules, Plugins, and API Ansible [18].

Ansible: At its core, Ansible serves as the primary execution tool responsible for issuing commands to remote hosts[17]. **Ansible Playbook:** This YAML-formatted playbook, often referred to as the "task playbook," defines the commands that hosts should execute based on the commands issued by Ansible[17]. **Inventory:** This is responsible for managing information about the hosts, including host ports, IP addresses, and other relevant details[17]. **Modules:** Modules play a pivotal role in executing commands on the hosts[17]. **Plugins:** These plugins serve as supplementary components enhancing the functionality of the modules when executing various tasks[17]. **API:** Acts as the physical interface for the program to invoke Ansible capabilities[17].

I. NIS2 checklist directive

On January 16, 2023, the Directive (EU) 2022/2055, which relates to measures aimed at ensuring a high level of cybersecurity[19]. The new cyber security strategy aims to maintain a worldwide and secure internet safer for everyone.

The checklist of NIS2 Directive marks an important milestone in strengthening the EU's cybersecurity framework. By requiring enhanced national readiness, promoting collaboration across borders, and implementing stricter safeguards in key sectors, it aims to create a robust and secure digital environment. The network is a important aspect to consider because nowadays with the exponential number of Internet of Things (IoT) Embedded systems have become an integral part of our everyday lives. Devices are increasingly connected, especially in the Internet of Things (IoT) environment products are more and more huge and complex[18] [20].

Principal Responsibilities Checklist

1) **Risk Assessment:** Systematic evaluation of networks and information systems to detect vulnerabilities and implement corrective measures [19].
2) **Incident Reporting:** Prompt notification of substantial incidents to national authorities or CSIRTs (Computer Security Incident Response Teams) within designated deadlines [19].
3) **Cybersecurity Policies:** Formulation and upkeep of extensive policies customized to address emerging risks [19].
4) **Supply Chain Security:** Evaluation and incorporation of cybersecurity stipulations in third-party contracts [19].
5) **Employee Training:** Consistent awareness initiatives and simulations to guarantee that personnel comprehend cybersecurity threats and optimal procedures [19].
6) **Information Sharing:** Cooperation with national authorities to bolster collective cybersecurity resilience [19].
7) **Documentation:** Comprehensive records of incidents, risk evaluations, and compliance actions for examination by regulatory bodies [19].

III. METHODOLOGY

There are two types , the theoretical, and the practical methodology. They are inter-ligated. The theoretical data are composed for all bibliography and state of art. The practical parts are composed for all demonstration of implementation and validation part that are interpreted by theoretical methodology.

For a best practice, this types of methodology are integrated in articles like this to recognize that was a smart organization between the theoretical and practical part.

- In the first phase, an investigation and exploration of the theoretical documentation and practical component. The tool "Rayyan" was essential for references in this paper. This tool is AI-powered to be designed for efficient systematic literature review management, that enhances collaborative research with powerful review automation and streamlined data management [21].

- After this brief, initial investigation, it will be necessary to develop a network and document it according to an idealized structure [22],[23],[9].

- Then, move on to the development of network configuration automation, using Python and Ansible scripts for it [4].

- Next, deployment/enhancement of different playbooks in Ansible to manage configuration changes on multiple devices simultaneously, in CPT or other for testing purposes [4].

- After the previous steps are completed, a comparison between the standalone and non-standalone process will be made graphically and documented to extract important data from the last phase that follows[4].

- Last phase, the verification of some legislation, ”checklists”, of the NIS-2 standard in the context of Project 1 (make the playbooks comply with the same directive)[19].

For a better comprehensive explanation of the methodology, in a figure 6, we explain all the processes for this paper.

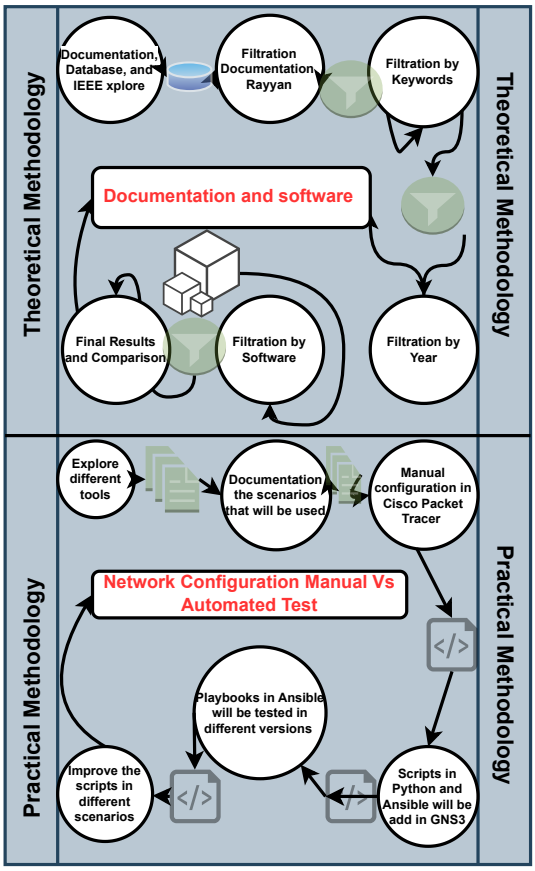

Fig. 6. This is an overview schematic structure about methodology of this project[9].

IV. Implementation

The implementation will be structure by :

IV-A Automated Network: Structure of manual documentation and the prototype manual configuration in CPT and the implementation for the automated by following the manual structure.

IV-B Python Scripts and Ansible : All generator and diagnostic scripts in python and the results for ansible playbooks.

VI-A Comparator: Manual vs Automation Overview.

A. Automated Network:

For Manual and Automated methods it is important to compare with one defined and completed network structure. In this case, initial Manual Network needs a structure that is about of a Software Company Network that has connectivity between the network that are directly connected to a router. There are 10 different networks that represents 10 different Virtual Local Area Network (VLAN) that are directly connected to a switch [24]. All departments are created and they are a real example of the actual division of a network in a Software Company now a days [24]. There is a file in excel that shows all of the structure with all important configuration aspects **Project2˙Network˙Software˙Company.xlsx**.

A small software company network are explicit in the figure 7:**VLANS** Departments with VLANs 10 to 100, so 10 in total different VLANS. **Switches:** (10 + 1 core) in total, with management IPs. **Routers:** 11 in total, serving as department gateways. **PCs/Laptops:** 13 in total, across all departments. **Servers:** 8 in total, Development, IT, Finance, and Infrastructure. **Printers:** 2 in total, Sales and HR departments. **Total hardware:** 64 network devices ready for automation.

This structure was the base for all of this project. Of course that for each test will be implement just a short number of departments because are 10 different.

Another type of example can be the user can create your own network with the number of devices, configurations, and another options. So in this case the user can automate also the network that is not created in a manual way before. This example will be implemented also but the focus where is compare the final results with the both manual and automated configurations and for that the data needs to be the same.

In the schematic structure below 7 is possible see the differences of the methodology behind the documentation of the Manual network that represents a prototype VSEs.

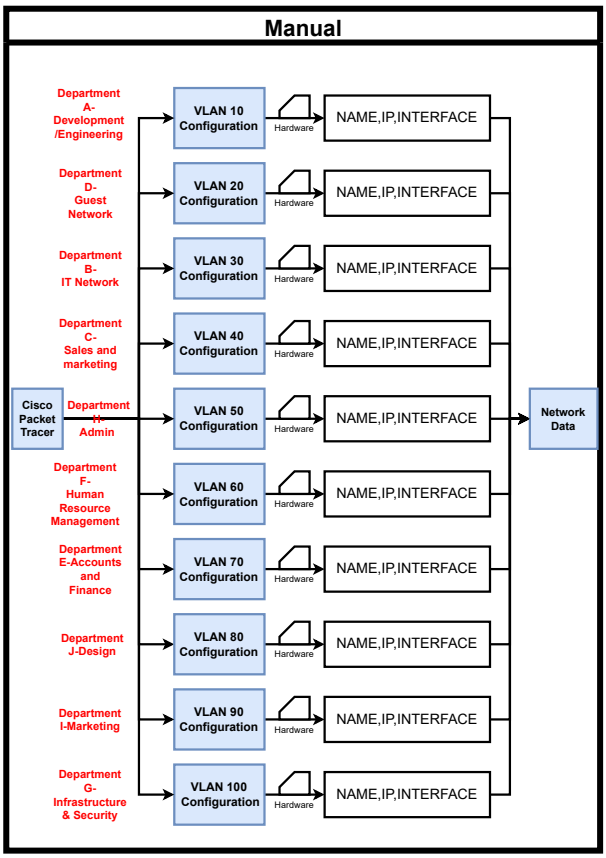

Fig. 7. This is an overview schematic structure about network methodology strategy for manual network[9].

Is important verify in the figure 7 is just an example and can be or not implemented in a real situation but in this case is just for this project and the configuration is quite simple of course can be implemented in a different way better in the future for a specific purpose. In this case there is no specific purpose just a example of a Small company network that represent a VSEs.

This software network prototype is important for the first phase. Basically the prototype created for VSEs in the section 7 will be explained in the flow chart 8:

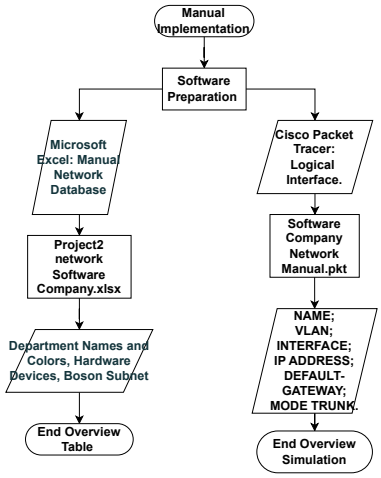

Fig. 8. This is a flow chart that represents the interpretation of design in a general overview of the manual prototype network in Cisco Packet Tracer[9].

Is important verify in the figure 8 this structure represent all data about manual configuration and summarize all content explained before here in this figure. The focus is now implement this in Cisco packet Tracer manual and check the difficulty, time, security, response, memory, to compare with the automated network.

B. Python Scripts and Ansible:

The automated Network needs also a flow chart to explain how the manual implementation will be changed for the automated implementation. So is important to understand before programming in Python or in Ansible the structure for the GNS3. The GNS3 is important for the automated implementation of the automated schematic below 9.

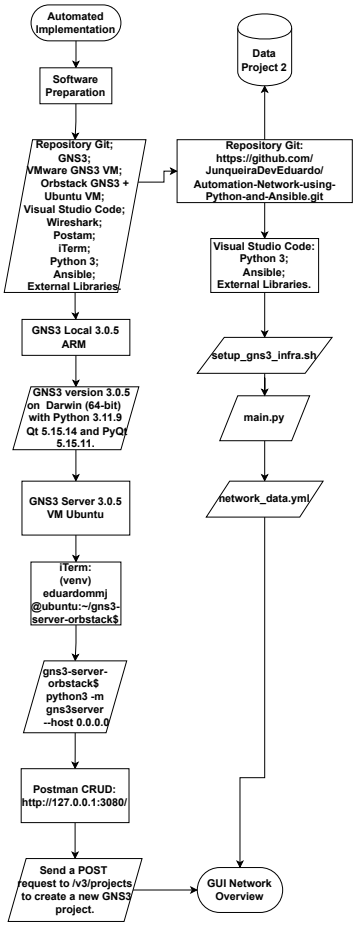

Fig. 9. This is a flow chart that supports the GNS3, open-source software for the simulation of the automated network [9].

The ansible generate scripts that have the main inventory structure from the validated network network_data.*yml*file that will be generate each device in department file *.yml*. Have also the important playbooks for the deploy *.yml*.

V. Validation and Results

There are some videos that explain better on the repository GitHub: [4].

CPT, after some search and read documentation it's possible automatize network, but the problem is before the automation it's required creates all network and configure communication between all network and after with ssh method can be automatized.

This is not the aim of the project, so after a long search and reading documentation, the software GNS3 can automatize a network by default with some exceptions but it's more powerful than CPT.

In the table explain exactly these differences and what is the best tool for automation I:

An, the implementation of scripts in Python and the Ansible playbooks was essential for the validation and results 10.

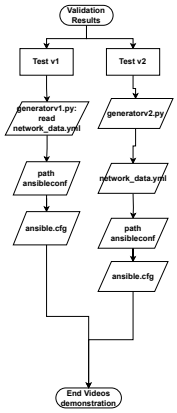

Fig. 10. This is a flow chart represents the test cases of scripts Python and Ansible playbook [9].

A. GNS3:

Interface Virtual Machine GNS3 in VMware in the figure below 11: Terminal of the Virtual Machine GNS3

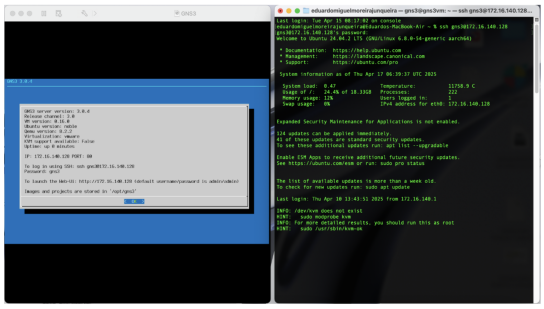

Fig. 11. Schematic about GNS3 GUI VM in VMware [9].

in VMware Workstation in the figure below :12

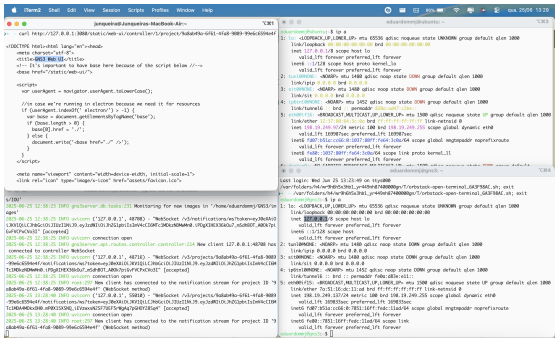

Fig. 12. Schematic about GNS3 GUI VM in VMware [9].

VI. Discussion

Key best practices and methodology include the following.

- **Workflow Documentation:** Documenting all manual and automated workflows to optimize for automation.
- **Version Control:** Maintaining scripts and playbooks in a Git repository for collaboration and rollback.
- **Modular and Reusable Code:** Structuring scripts and playbooks into reusable modules and roles for scalability.
- **Testing and Validation:** Testing automation in simulated environments (GNS3, CPT) before production deployment.
- **Continuous Improvement:** Regularly reviewing and updating automation to adapt to network changes and feedback.

A. Comparator Manual with Automated Network Overview

TABLE II
Comparison of Manual vs Automated Network Implementation: NIS2 Compliance Checklist

NIS2 Requirement	Manual Implementation	Automated Implementation
Risk Assessment	Configurations, scans, checklists, human overview.	Script-generated configurations and reports, generated scans, Artificial intelligence (AI) overview .
Incident Reporting	User checks logs, fills incident forms.	Automatic alerts, log monitoring, instant notifications, generate new network.
Cybersecurity Policies	Policies applied by admins, updated manually passwords, layers and hardware.	Policies enforced by playbooks, consistent deployment, version control patches.
Supply Chain Security	Manual verification of suppliers, contract administrators of network.	Automated patching, integrity checks, dependency monitoring,AI overview.
Employee Training	In-person sessions, laboratories with hardware devices.	Online explanations, online hardware,cloud network with AI support.
Information Sharing	Local software, email or file share.	Automated reporting, dashboard sharing, API integration, cloud network interface.
Documentation	Handwritten or Word logs, manual updates.	Code generators, Auto-generated logs, backups, versioned records.

VII. Conclusions

This project successfully demonstrates the transformation from manual network configuration to fully auto-

mated network deployment, providing a comprehensive comparison between traditional and modern network management approaches. The integration of all tools related in II are important for the conclusion of this project. Through overcoming significant technical challenges, particularly in Apple Silicon compatibility, this project demonstrates that network automation is achievable across diverse development environments, making these approaches accessible to a broader range systems. By following this structured and best-practice approach, the project will deliver a robust, scalable, and secure automated network configuration management system, maximizing efficiency and minimizing risks and manual effort. The data base of this project was represented in a repository of GitHub :[4], the overleaf article: [2], the prototype network: [22], the schematics [9],the intermediate word report [23], and finally the software to filter all references below [21]. Acronyms list:

IPVC	Institute Polytechnic Viana do Castelo
ERSC	Computer Systems and Network Engineering
GNS3	Graphical Network Simulator-3
CPT	Cisco Packet Tracer
macOSM2	Apple Silicon Chip ARM M2
ARM	Advanced RISC Machines
VM	Virtual Machine
VMware	Virtual Machine Software
IP	Internet Protocol
SSH	Secure Shell
VLAN	Virtual Local Area Network
API	Application Programming Interface
YAML	YAML Ain't Markup Language
CLI	Command Line Interface
GUI	Graphical User Interface
NIS2	Network and Information Security Directive 2
IoT	Internet of Things
OpenStack	Open-source cloud computing platform
AI	Artificial intelligence
EU	The European Union
Ubuntu	Linux Operator System
VSEs	Very Small Enterprises
pip	Python Package Index
venv	Virtual Python Module
GitHub	Cloud-based platform

REFERENCES

[1] pt. [Online]. Available: https://pt.overleaf.com/latex/templates/paper-template-ersc/jfqthsbhfydd.

[2] E. Junqueira. "Project 2 ersc ipvc automation of network configuration - online latex editor overleaf." (May 2025), [Online]. Available: https://pt.overleaf.com/project/67ef7baa14a19d89f59358cc (visited on 05/17/2025).

[3] en. [Online]. Available: https://www.overleaf.com/learn.

[4] JunqueiraDevEduardo, *Github - junqueiradeveduardo/automation-network-using-python-and-ansible: Manual vs automated network with python and ansible arm architecture, gns3, orbstack, cisco packet traccer.* en. [Online]. Available: https://github.com/JunqueiraDevEduardo/Automation-Network-using-Python-and-Ansible.git.

[5] A. Mazin, H. Z. Abidin, L. Mazalan, and A. Mazin, "Network automation using python programming to interact with multiple third-party network devices," in *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, 2023, pp. 59–64. DOI: 10.1109/ICITACEE58587.2023.10277400.

[6] I. Richardson and C. G. Von Wangenheim, "Guest editors' introduction: Why are small software organizations different?" *IEEE Software*, vol. 24, no. 1, pp. 18–22, 2007. DOI: 10.1109/MS.2007.12.

[7] R. Emiliano and M. Antunes, "Automatic network configuration in virtualized environment using gns3," in *2015 10th International Conference on Computer Science Education (ICCSE)*, 2015, pp. 25–30. DOI: 10.1109/ICCSE.2015.7250212.

[8] Cisco Networking Academy, *Cisco packet tracer*, Accessed on April 14, 2025, 2024. [Online]. Available: https://www.netacad.com/cisco-packet-tracer.

[9] *Diagramasprojeto2.drawio*, https://drive.google.com/file/d/1VrTOyrEzW7ied956Rqg7bXCpUOMQAXA1/view?usp=sharing, Accessed: 2025-5-23.

[10] W. Sha, M. Wu, and Y. Qiao, "Research on automated operation and maintenance methods for

power management networks based on ansible," in *2023 5th International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, 2023, pp. 591–595. DOI: 10.1109/ICAICA58456.2023.10405593.

[11] GNS3 Community, *Gns3 documentation*, No publication date. Accessed on April 14, 2025, 2024. [Online]. Available: https://docs.gns3.com/docs/.

[12] A.-F. Sicoe, R. Botez, I.-A. Ivanciu, and V. Dobrota, "Fully automated testbed of cisco virtual routers in cloud based environments," in *2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, 2022, pp. 49–53. DOI: 10.1109/BlackSeaCom54372.2022.9858288.

[13] VMware, *Network operations automation*, No publication date. Accessed on April 14, 2025, 2024. [Online]. Available: https://www.vmware.com/solutions/cloud-infrastructure/network-operations-automation.

[14] *What is OrbStack? · OrbStack docs*, en, https://docs.orbstack.dev/, Accessed: 2025-5-23.

[15] M. Weisel, *Gns3-server-orbstack: This project offers an automated deployment procedure for the GNS3 server on an OrbStack linux machine*, en.

[16] en. [Online]. Available: https://docs.python.org/3/library/index.html.

[17] A. Community, *Ansible documentation*, en. [Online]. Available: https://docs.ansible.com/.

[18] S. Arthofer, S. Wilker, and T. Sauter, "Development of a test automation framework based on a comparison of different approaches for test automation in the embedded systems area," in *2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS)*, 2024, pp. 1–6. DOI: 10.1109/ICPS59941.2024.10640021.

[19] T. E. PARLIAMENT and T. C. O. T. E. UNION, "Directive (eu) 2022/2555 of the european parliament and of the council - official journal of the european union," 2022.

[20] T. Katulić, "Transposition of eu network and information security directive into national law - 2018 41st international convention on information and communication technology, electronics and microelectronics (mipro)," pp. 1143–1148, 2018.

[21] Rayyan Systems, *Rayyan — ai-powered systematic review tool*, No publication date. Accessed on April 14, 2025, 2024. [Online]. Available: https://new.rayyan.ai/.

[22] Eduardo Junqueira, $Project2_network_software_company.xlsx$, en-us, 2025. [Online]. Available: https://ipvcpt-my.sharepoint.com/:x:/g/personal/eduardo_j_ipvc_pt/EeVkxh0SPNxGp39co22LiCYB8jyGscTHdPRm2UFI2k8--A?e=0Zpo1u.

[23] en-us. [Online]. Available: https://ipvcpt-my.sharepoint.com/:w:/g/personal/eduardo_j_ipvc_pt/EYnnC1SOO1JPmUoCgOVWWoEBEkzjH3qRPAJp9O5H-iw0Og?e=9J7NRV.

[24] en. [Online]. Available: https://www.quora.com/What-are-the-different-departments-in-a-software-company.