Text

Description automatically generated

Projeto 2 - Continuação de Projeto 1 | Ano Letivo 2024/2025

Aluno: Eduardo Miguel Moreira Junqueira, [eduardo.j@ipvc.pt](mailto:eduardo.j@ipvc.pt). Nº MEC: 30241  
Supervisionado pelo professor: Nuno Torres, [nunotorres@ipvc.pt](mailto:nunotorres@ipvc.pt).

Automatizar a configuração de uma rede utilizando Python e Ansible.

Tabela de Conteúdo

[Introdução: 2](#_Toc199009341)

[Descrição de projeto: 2](#_Toc199009342)

[Objetivos do projeto: 3](#_Toc199009343)

[Proposta de projeto: 3](#_Toc199009344)

[Atividades a Desenvolver: 3](#_Toc199009345)

[Aplicações de configuração: 4](#_Toc199009346)

[Configuração manual de uma rede, benefícios e desafios: 5](#_Toc199009347)

[Benefícios: 5](#_Toc199009348)

[Desafios: 5](#_Toc199009349)

[Configuração automática de uma rede, benefícios e desafios: 6](#_Toc199009350)

[Benefícios: 6](#_Toc199009351)

[Desafios: 6](#_Toc199009352)

[Exploração de ferramentas neste projeto: 6](#_Toc199009353)

[1-Cisco Packet Tracer: 6](#_Toc199009354)

[2-GNS3 (Graphical Network Simulator): 7](#_Toc199009355)

[2-Docker Containers OrbStack: 7](#_Toc199009356)

[*3- Automatically produce the configuration of network equipment’s:* 8](#_Toc199009357)

[*Ansible* linguagem: 9](#_Toc199009358)

[Playbooks em *Python*: 9](#_Toc199009359)

[Desenvolvimento de scripts Python e Ansible para aplicação na rede: 9](#_Toc199009360)

[Exploração dos resultados e implementação (testes iniciais e melhoria nos scripts desenvolvidos): 9](#_Toc199009361)

[NIS 2 checklist 9](#_Toc199009362)

[Definições de termos e configuração: 10](#_Toc199009363)

[Ubuntu/GNS3 Virtual Machine: 10](#_Toc199009364)

[Ubuntu Virtual Machine: 10](#_Toc199009365)

[GNS3 Virtual Machine: 10](#_Toc199009366)

[Script para a instalação automática Git, Ansible, Python, Pip e configuração ambiente virtual : 10](#_Toc199009367)

[Comandos Configurações etapas: 11](#_Toc199009368)

[Ansible: 12](#_Toc199009369)

[Instalações GNS3: 13](#_Toc199009370)

[Referências GNS3: 13](#_Toc199009371)

[“Small Software Companies network departments”: 14](#_Toc199009372)

[Cisco Packet Tracer construção da rede manualmente: 1h de trabalho 14](#_Toc199009373)

[Cisco Packet Tracer configuração da rede manualmente: 2h de trabalho 18](#_Toc199009374)

[Processos realizados: 20](#_Toc199009375)

[Referências Bibliográficas: 21](#_Toc199009376)

[Figura 1 Esta figura pretende demonstrar as ferramentas e sua integração no sentido de automação (DiagramasProjeto2, n.d.). 5](#_Toc199009755)

[Figura 2 OrbStack imagens referentes a Machines, Container, GUI e suas respetivas versões 8](#_Toc199009756)

[Figura 3 Máquinas Virtuais. 10](#_Toc199009757)

[Figura 4 **Ambiente virtual criado com sucesso! Venv Python** 11](#_Toc199009758)

[Figura 5. Configuração host local GNS3 12](#_Toc199009759)

[Figura 6 Criação de um protótipo da Rede de uma pequena empresa de Software. 15](#_Toc199009760)

[Figura 7 Departamentos diferentes com diferentes Vlans 16](#_Toc199009761)

[Figura 8 Departamentos diferentes com diferentes Vlans 16](#_Toc199009762)

[Figura 9 Departamentos diferentes com diferentes Vlans 16](#_Toc199009763)

[Figura 10 Departamentos diferentes com diferentes Vlans 17](#_Toc199009764)

[Figura 11 Departamentos diferentes com diferentes Vlans 17](#_Toc199009765)

[Figura 12 Configura as interfaces como trunk, permitindo tráfego de múltiplas VLANs na mesma ligação física 19](#_Toc199009766)

# 

# Introdução:

Este projeto encontra-se no seguimento do Projeto 1 que se intitula como Automação de uma

A proposta do projeto tem como intuito criar um ou vários scripts em *Python* para automatizar o setup inicial dos dispositivos na rede. A configuração de rede é baseada em princípios fundamentais que visam garantir a conectividade e a comunicação eficiente entre dispositivos em uma rede. Esses princípios incluem: **I):** A criação da rede; **II):** a atribuição de endereços *IP*; **III):** a configuração de *Gateways*; **IV):** a atribuição de servidores *DNS;* **V):**  a definição de políticas de segurança; **VI):** utilizar *playbooks Ansible* para gerir alterações de configuração em vários dispositivos em simultâneo (router, switch, pc, firewall, etc.); **VII):** testar scripts de automação fazendo o deploy em um ambiente de rede simulado (Cisco Packet Tracer, GNS3, etc.); **VIII):** demonstrar a eficiência da automação comparando os tempos de configuração manual e automatizada, assim como as taxas de erro;

**IX):** desenvolver um sistema automatizado para a gestão de configuração de redes, utilizando *Python* e *Ansible*, de modo a reduzir o tempo do setup manual, assim como minimizar a ocorrência de erros na configuração de dispositivos na rede.

## Descrição de projeto:

- Na primeira fase, será realizada uma averiguação e exploração da componente prática deste Projeto 2. Assim, será importante explorar de uma forma mais aprofundada as ferramentas, que serão usadas, para a parte prática do projeto, *Cisco Packet Tracer, Python, Ansible* e outras que acrescentem para a automação de uma rede e testes da mesma.

- Após esta breve investigação, inicial, será necessário desenvolver uma rede e documentar a mesma de acordo com uma estrutura idealizada.

- Depois, avançar para o desenvolvimento da automação de configuração da rede, usando scripts de *Python* e *Ansible* para a mesma.

- De seguida, implementação/melhoria de diferentes playbooks em *Ansible* para gerir alterações de configuração em vários dispositivos em simultâneo, no *Cisco Packet Tracer* ou outro para efeitos de teste.

- Após as etapas anteriores estarem concluídas, uma comparação entre o processo autônomo e não autônomo será feita graficamente e documentada para retirar dados importantes da última fase que se segue.

- Última fase, a comprovação de algumas legislações, “checklists”, da norma *NIS-2* no contexto do Projeto 1 (fazer com que os playbooks obedeçam à mesma diretiva).

## Objetivos do projeto:

- Estudo das ferramentas Cisco Packet Tracer, GNS3, entre outras.

- Desenvolvimento de scripts para automação da rede.

- Comparação de Resultados (manual vs. automático).

- Aplicação da *NIS2* checklists (fazer com que os playbooks obedeçam à mesma diretiva).

## Proposta de projeto:

A principal proposta deste projeto é comparar a configuração manual com a configuração automática de rede(s) criada(s). A estruturação das redes será feita em Excel, sendo que vai existir uma rede menor para testes iniciais e outra maior para testes finais e comparações.

A estrutura para a Metodologia da parte prática está definida por um protótipo inicial que está sempre a ser alterado consoante a evolução do projeto.

O artigo acadêmico no *Overleaf* pretende resumir tudo o que está neste relatório e na parte prática.

Os links para os ficheiros são os seguintes:

Excel estrutura(s):

<https://ipvcpt-my.sharepoint.com/:x:/g/personal/eduardo_j_ipvc_pt/EeVkxh0SPNxGp39co22LiCYB8jyGscTHdPRm2UFI2k8--A?rtime=dVdwo-KV3Ug>

Estruturas Draw.io:

<https://drive.google.com/file/d/1LXcOSiEvW1qO3UI6qcEUB6ka3ZuGuA4/view?usp=sharing>

Overleaf Artigo:

<https://pt.overleaf.com/4594197176xybmbppyxxnr#8b55af>

Link relatório word:

<https://ipvcpt-my.sharepoint.com/:w:/g/personal/eduardo_j_ipvc_pt/EYnnC1SOO1JPmUoCgOVWWoEBEkzjH3qRPAJp9O5H-iw0Og?e=KmTygd>

Link rayyan overview all sources:

<https://new.rayyan.ai/reviews/1468412/overview>

O projeto enfatiza a automação na integração dos seguintes softwares:

I): *Cisco Packet Tracer*, para configuração manual de rede com topologia.

II): *GNS3* local, para configuração automatizada de rede com topologia.

III): Máquinas Virtuais para *Ubuntu e GNS3* e sua respetiva configuração.

III): *VMware*, para Máquina Virtual GNS3 remota no Ubuntu.

IV): OrbStack para Máquinas Virtuais em substituição ao VMware e containers para Containers *Docker* para *hosts* na rede e sua respetiva configuração.

VI): *Ansible Scripts*, para scripts automatizados em playbooks.

VII): *Scripts Python*, para scripts automatizados na configuração de rede.

Atividades a Desenvolver:

|  |  |  |
| --- | --- | --- |
| ID | Descrição | Duração (Semanas) |
| 1 | Exploração de ferramentas como: Cisco, *Ansible*, GNS3, entre outras. | 6 semanas |
| 2 | Desenvolvimento de scripts *Python* e Ansible para aplicação na rede. | 5 semanas |
| 3 | Exploração dos resultados e implementação, (testes iniciais e melhoria nos scripts desenvolvidos). | 2 semanas |
| 4 | Comparação, documentação e vídeos práticos. | 2 semanas |

## Aplicações de configuração:

A configuração de rede é aplicada em uma ampla variedade de cenários, desde redes domésticas, empresas, espaços públicos, até redes complexas de diferentes organizações, como por exemplo a rede universitária do campus cidade Viana do Castelo, IPVC.

No contexto nacional, a rede do Instituto Politécnico de Viana do Castelo (IPVC) representa várias infraestruturas WAN dependendo do polo universitário. Esta estrutura é complexa, distribuída por múltiplos polos, exigindo conectividade inter-campus segura e eficiente. Este tipo de rede educacional pode beneficiar significativamente de soluções automatizadas de configuração e gestão, como as implementadas neste trabalho.

A configuração de rede desempenha um papel fundamental na garantia do funcionamento adequado de uma rede. Uma configuração incorreta pode levar a problemas de conectividade, baixo desempenho, vulnerabilidade de segurança e dificuldades na administração da rede. A estabilidade será comprometida e a velocidade de upload e download de dados também.

Portanto, a automação seria um processo bem mais eficiente e rápido com a realização de scripts em Python. Para a introdução desta metodologia prática na figura em baixo *Fig1*, é importante que as ferramentas sejam suportadas para automatizar scripts e implementação dos mesmos.

A screenshot of a computer

AI-generated content may be incorrect.

Figura 1 Esta figura pretende demonstrar as ferramentas e sua integração no sentido de automação (DiagramasProjeto2, n.d.).

* ubuntu → contém Ansible, usado para scripts de automação e orquestração.
* gns3 → corre o GNS3 Server (gns3server) com interface web acessível pela porta 3080.
* Containers para os host de cada departamento da rede.

## Configuração manual de uma rede, benefícios e desafios:

### Benefícios:

- Para redes menores ou com especificações de configuração que exigem verificação constante de um administrador de rede.

- Para redes de baixo custo, locais (redes estáticas ponto a ponto), pouco significativa a nível de prioridade e segurança.

- Para redes sem flexibilidade e escalabilidade.

### Desafios:

- Erro humano, afetando a disponibilidade e integridade da rede, dependendo do fator humano como principal fator de risco.

-Complexidade, redes atuais podem ser extremamente complexas, com os dispositivos *IoT* a aumentar exponencialmente, é necessária uma maior escalabilidade e segurança dos mesmos.

- Segurança: garantir a segurança da rede é um desafio constante, pois cada vez mais o sistema pode estar vulnerável a possíveis intrusões de congestionamento ou interferência na rede.

## Configuração automática de uma rede, benefícios e desafios:

Automação de uma rede é um processo autônomo que permite criar, configurar e gerir dispositivos de rede como (router, switch e host) sem intervenção manual contínua. Geralmente, este processo é um processo repetitivo que, com a criação de scripts, será autônomo.

### Benefícios:

- Para redes *WAN* com elevado número de dispositivos *IoT,* redes com elevado número de *Vlans* onde a rede é escalável e complexa.

- Facilidade na administração da rede com a automação de tarefas repetitivas.

- Maior controle e acesso de toda a rede (visualização das mesmas em gráficos gerados pela mesma automação) e, em caso de falha ou perda de dados, é sempre possível reativar a rede no menor e mais seguro tempo possível.

### Desafios:

- A implementação de uma infraestrutura de rede automatizada requer maior controlo de configuração em scripts e processos de integração de ferramentas de software que permitem acessibilidade para processo de automação.

- Para configurações mais específicas de redes, como nível de camadas de segurança em diferentes Vlans, ou privilégios em diferentes *Hosts* ou dispositivos da rede.

- Como automatizar de uma maneira eficiente e segura um script para uma rede complexa não é o suficiente.

- O fator humano não está em causa, logo a automação precisa de corresponder aos requisitos do consumidor final e respeitar a confidencialidade e integridade dos dados.

# Exploração de ferramentas neste projeto:

Esta exploração de ferramentas demorou mais do que o previsto inicialmente de 3 semanas, pelo motivo de o Cisco Packet Tracer não corresponder ao que era pedido no projeto no que diz respeito à automação em Ansible com *Playbooks* em *Python.*

Logo, foi necessário aumentar o tempo de investigação de ferramentas e leitura de documentação.

Neste projeto foram testadas 3 diferentes ferramentas, das quais:

## 1-Cisco Packet Tracer:

O Cisco Packet Tracer, neste projeto, não permite a automação total de uma rede. Ou seja, o Cisco Packet Tracer é apenas um simulador de software, da *Cisco Network Academy*, que simula redes e permite a sua respetiva configuração.

Mas estas são criadas de forma manual. Haveria uma possível automação, mas para isso a rede teria de ser toda criada e os dispositivos da mesma teriam de estar conectados entre si. Para automatizar, teria de haver comunicação *SSH* e esta teria de ser configurada também com o respetivo endereço IP*.*

Esta ferramenta permite apenas usar equipamentos da *Cisco* e não outros equipamentos, tornando assim a rede limitada ao sistema *Cisco*. Estes dispositivos são configurados com comandos de configuração Cisco (CLI). Para a utilização de outros equipamentos, teria de criar *Virtual Machines* ou *containers* em Docker, mas não é possível, visto que não existe suporte, nem comandos para configuração.

Assim, podemos perceber que, para este projeto, o *Cisco Packet Tracer* vai servir de comparação, mas para o processo manual de criação de uma rede. No final deste projeto, este vai servir como um comparador de tempo de resposta à configuração e comunicação de uma rede.

“There is an evident difference between virtualization of network devices and simulation. 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. Some examples of simulators are Cisco Packet Tracer (R. Emiliano and M. Antunes, 2015).”

Neste sentido foi criada 2 redes uma que representa uma empresa real de software reengineering com 10 departamentos e em cada departamento está presente

## 2-GNS3 (Graphical Network Simulator):

Inspirado numa arquitetura publicada no *IEEE* , onde se utiliza *Terraform* para provisionamento e Ansible para configuração de routers Cisco CSR1000V em ambiente *OpenStack*, este projeto adapta a lógica de automação para um ambiente mais simples e acessível com *GNS3 e Docker*. Assim, a criação e gestão de nós de rede é feita via GNS3, enquanto a automação de configurações (como atribuição de IP e serviços SSH) é realizada com Ansible playbooks em *Python* (A. -F. Sicoe, R. Botez, I. -A. Ivanciu and V. Dobrota, 2022).

“A different approach is provided by virtualized applications that are able to manage a set of virtual machines running the same operating system kernel version of the real network device. GNS3 is one of the most used applications to virtualize Cisco networks (R. Emiliano and M. Antunes, 2015).”

GNS3 é um emulador de software de rede lançado pela primeira vez em 2008 e atualmente continua com atualizações,

Primeiramente foi feita uma instalação através de um repositório do GitHub <https://github.com/GNS3/gns3-gui/releases>, onde tenho todos os releases do GNS3.De verificar que para usar o GNS3 no macOS M2 com chip Apple Silicon ,eu só posso descarregar software e containers com (ARM64). Assim, instalei o GNS3 v 3.0.4 e funcionou durante 2 semanas , sendo que depois saiu uma nova versão v 3.0.5 no dia 14/05/25 com GUI para ARM64 mas a Virtual Machine para o VMware não deu suporte para ARM64 até ao dia de hoje 17/05/25.

Ou outra escolha, depois de muita pesquisa de instalação, foi dada por GNS3 na documentação e um comentário de um utilizador nos seguintes links:

<https://www.gns3.com/community/featured/gns3-on-apple-silicon>

<https://marcstech.blog/archives/gns3-apple-silicon/>

<https://orbstack.dev/download>

2 exemplos de estruturas de rede no GNS3:

<https://res.cloudinary.com/binarynature/image/upload/v1714344147/paper-street-soap-company-full-overlay_dfzizx.png>

<https://res.cloudinary.com/binarynature/image/upload/v1714593311/kickin-it-old-school-full_uisy65.png>

Concluindo, agora tenho a GUI v3.0.5 e a VM v3.0.5 do GNS3, mais recente atualizada a funcionar em localhost. As diferenças que foram feitas: passei a não utilizar mais a VMware e passei a usar uma nova ferramenta cujo nome é *OrbStack.*

### Referências GNS3 e instalações GNS3:

<https://marcstech.blog/archives/gns3-apple-silicon/> GNS3 with macOS on Apple Silicon

<https://github.com/mweisel/gns3-server-orbstack> --> gitclone no meu terminal.

<https://github.com/gns3/gns3-gui/releases> --> para a GUI local no meu PC versão mais recente 3.0.5

“Computer networking is a central topic in computer science courses curricula offered by higher education institutions. Network virtualization and simulation tools, like GNS3, allows students and practitioners to test real world networking configuration scenarios and to configure complex network scenarios by configuring virtualized equipment’s, such as routers and switches, through each one’s virtual console.”

## 2-Docker Containers OrbStack:

O OrbStack é uma plataforma para criar containers e máquinas virtuais Linux no macOS, com suporte nativo para chips Apple Silicon (M1/M2/M3/M4).Ele substitui ferramentas como Docker Desktop ou VMs tradicionais, mas com muito mais eficiência, velocidade e simplicidade.

OrbStack é como uma “máquina virtual leve” que permite criar e gerir várias instâncias de Ubuntu (ou outros sistemas Linux entre outros), como se fossem pequenos servidores.

Esta ferramenta permitiu:

- A criação de containers Linux (Ubuntu) em segundos.

- A instalação e execução do servidor GNS3 de forma estável.

- Conectividade em rede real com *IPs* atribuídos, essenciais para a comunicação com o GNS3 GUI.

- A utilização de ferramentas como Ansible e Python dentro dos containers.

- A simulação de dispositivos Cisco conectados virtualmente ao container Linux.

- A execução simultânea de diferentes serviços em ambientes separados (ex.: GNS3, Ansible).

-Executar imagens Cisco como IOSv, vIOS, IOU (via GNS3).

O OrbStack foi fundamental para evitar o uso de VMs pesadas como o VMware ou Virtual Box , permitindo maior desempenho e economia de recursos no MacBook Air com chip M2.

Esta ferramenta só tem 1 mês de versão Pro o que significa que até dia 14 de Junho eu poderei efetuar todas as alterações no que diz respeito á parte prática e realizar vídeos demonstrativos. Após esta data o Open source plano Pro é esgotado e não consigo ter mais acesso á ferramenta e ao meu projeto.

Como a apresentação deste trabalho vai ser realizada em junho/julho ainda terei tempo para me focar na parte teórica até dia 14 de junho e depois irei realizar o artigo no Overleaf e PowerPoint para apresentação com os Vídeos práticos.

A screenshot of a phone

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Figura 2 OrbStack imagens referentes a Machines, Container, GUI e suas respetivas versões

## *3- Automatically produce the configuration of network equipment’s:*

Estas aplicações ainda não foram usadas só verificadas por documentação.

-Netomata Config Generator:(<http://www.netomata.com>),

-Gen-IT (<http://gen-it.net>),

-AutoNetKit (<https://autonetkit.org>): relaciona-se com o *GNS3* mas não atribui *IPs* automaticamente.

-Solarwinds Network Config Generator (<http://www.solarwinds.com>).

-Estas ferramentas são para configurar automaticamente equipamentos de rede tal como está neste repositório: <https://github.com/sk2/autonetkit>

## *Ansible* linguagem:

## Playbooks em *Python*:

# Desenvolvimento de scripts Python e Ansible para aplicação na rede:

A ser realizado

# Exploração dos resultados e implementação (testes iniciais e melhoria nos scripts desenvolvidos):

Trabalho Futuro próximas 3 semanas

# NIS 2 checklist

Automation tools can effectively rectify the aforementioned concerns, streamlining the management and configuration of servers, especially within server clusters, for maintenance personnel. With the surge in the number of server clusters, the management of account passwords has become a focal point of server operation and maintenance work. Efficient password management constitutes one of the current research focal points. Furthermore, passwords and other login information for server clusters are stored within automated operation and maintenance tools, warranting enhancements to their security (W. Sha, M. Wu and Y. Qiao 2023).

Trabalho Futuro próximas 4 semanas.

# Definições de termos e configuração:

Nesta secção está explicado as definições e configuração por texto, esquemas, tabelas e figuras todos os processos realizados.para este trabalho existe ferramentas e termos que são importantes salientar:

## Ubuntu/GNS3 Virtual Machine:

## Ubuntu Virtual Machine:

Máquina Virtual para o Ansible.

hosts.ini:**[gns3]**198.19.249.237 ansible\_user=eduardommj ansible\_ssh\_pass=08062004 ansible\_python\_interpreter=/usr/b>

## GNS3 Virtual Machine:

Máquina Virtual para o GNS3.

Inventory.ini:**[gns3]**198.19.249.208 ansible\_user=eduardommj ansible\_ssh\_pass=08062004 ansible\_python\_interpre>

Máquinas virtuais sem interface gráfica. Ubuntu 22.04 LTS para ARM64.

VM Ubuntu foi criado dentro da ferramenta automaticamente.

A screenshot of a computer

AI-generated content may be incorrect.

Figura 3 Máquinas Virtuais.

VM GNS3 foi criada com o seguinte comando: “orb create --arch arm64 ubuntu:noble gns3”

## Script para a instalação automática Git, Ansible, Python, Pip e configuração ambiente virtual :

**#!/bin/bash**

**echo " Atualizar repositórios..."**

**sudo apt update**

**echo " Instalar dependências principais (git, ansible, etc)..."**

**sudo apt install -y git ansible python3 python3-pip**

**echo " Dependências instaladas."**

**echo " Clonar repositório GNS3 OrbStack..."**

**git clone https://github.com/mweisel/gns3-server-orbstack.git**

**echo " Entrar no diretório clonado..."**

**cd gns3-server-orbstack || exit**

**echo " Criar ambiente virtual Python..."**

**python3 -m venv venv**

**echo " Ativar ambiente virtual..."**

**source venv/bin/activate**

**echo " Instalar dependências Python no ambiente virtual..."**

**pip install -r requirements.txt**

**echo " Ambiente virtual criado com sucesso!"**

A screenshot of a computer screen

AI-generated content may be incorrect.

Figura 4 **Ambiente virtual criado com sucesso! Venv Python**

## Comandos Configurações etapas:

Todos os comandos usados foram procurados neste link: <https://docs.orbstack.dev/headless>

/bin/bash -c "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh>)#Instalação Brew.sh

brew install orbstack #Instalação OrbStack GUI

brew upgrade --greedy orbstack #Update OrbStack

orb shell gns3 #entrar diretamente na máquina Gns3

python3 -m gns3server --host 0.0.0.0 #Estás a correr o gns3server manualmente na máquina ubuntu, dentro de um ambiente virtual Python (venv)

A Figura seguinte Fig 3, mostra ....

Comandos para iniciar o GNS3 por HTTP para qualquer Host 0.0.0.0:

**cd ~/gns3-server-orbstack**

**source venv/bin/activate**

**python3 -m gns3server --host 0.0.0.0**

A screenshot of a computer

AI-generated content may be incorrect.

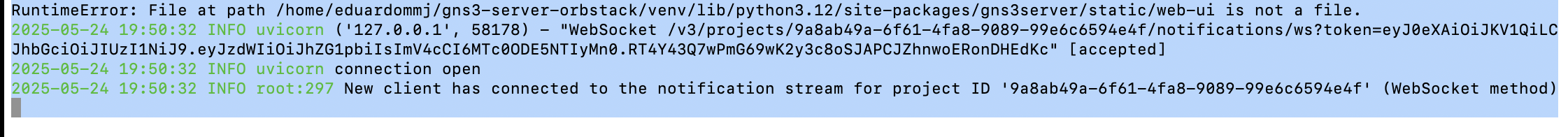


Figura 5. Configuração GNS3 Server via HTTP por WebSocket no domínio http://127.0.0.1:3080

## Ansible:

Ansible, is an open-source automation framework for a wide range of applications. The most important feature of this automation approach is that Ansible works without an agent, so there is no need to install additional software. 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 (API) and plugins. The inventory is used to prepare and build a network. The API provides the interoperation of end-to-end modules. The modules are the automated executable units that are controlled by means of the playbooks. To complete the Ansible Automation Engine, plugins can be used for example for caching or logging. The users create the so-called playbook and controls the automated process of the Automation Engine. Playbooks are created in the 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. Ansible is also able to automate different networks, this is represented with the block networks The hosts block represents the embedded systems that are to be automated (S. Arthofer, S. Wilker and T. Sauter,2024).

Due to Ansible’s architecture, scalability can only be achieved through multiple devices controlled by the Ansible Controller. This means parallel testing only through multiple devices on the network can be enabled. Scalability is therefore possible, but involves additional effort. Time saving: The simple creation and distribution of test cases can also save time with Ansible compared to manual tests. If scalability is sought through multiple devices in the network, additional time can be saved in the test runs. It should be noted that Ansible is a test distribution tool and additional effort is required for test execution and test logging. This circumstance requires valuable time in setting up a test automation. After successfully setting up a test automation with Ansible, this additional effort is no longer necessary. Functionalities: Ansible is an automation tool that is used to distribute software, specifically test cases, within a network. Test cases can be started and stopped from a central controller. In order to obtain data about the tests carried out, the controller must be expanded with this functionality via a so-called playbook. A clear presentation of successful and unsuccessful tests can also only be achieved with additional effort. Costs: Ansible is a free open-source automation tool. Maintenance: New test cases can also be added or maintained very easily with Ansible (S. Arthofer, S. Wilker and T. Sauter,2024).

Ansible typically comprises six key components, namely: Ansible, Ansible Playbook, Inventory, Modules, Plugins, and APIs Ansible (S. Arthofer, S. Wilker and T. Sauter,2024).

Ansible: At its core, Ansible serves as the primary execution tool responsible for issuing commands to remote hosts.

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.

Inventory: Inventory is responsible for managing information about the hosts, including host ports, IP addresses, and other relevant details.

Modules: Modules play a pivotal role in executing commands on the hosts.

Plugins: These plugins serve as supplementary components enhancing the functionality of the modules when executing various tasks.

API: The API acts as the physical interface for the program to invoke Ansible's capabilities.

## Scripts em Python:

#Exemplo Vlan 1:

{

"description": "Creates a VLAN 0 (Infrastructure and Security) network with a switch and 10 hosts.",

"steps": [

{

"step": 1,

"description": "Create the switch node using a built-in Ethernet switch template.",

"endpoint": "/v2/projects/{project\_id}/nodes",

"method": "POST",

"json\_payload": {

"name": "Switch-G",

"node\_type": "ethernet\_switch",

"compute\_id": "local",

"x": -200,

"y": 100

}

},

{

"step": 2,

"description": "Create 10 Docker-based host nodes to represent the department's servers.",

"repeat": 10,

"endpoint": "/v2/projects/{project\_id}/nodes",

"method": "POST",

"json\_payload": {

"name": "Server-{index}",

"node\_type": "docker",

"compute\_id": "local",

"x": "{x\_offset}",

"y": 300,

"docker\_image": "alpine:latest"

}

},

{

"step": 3,

"description": "Connect each host node to the switch via a link.",

"repeat": 10,

"endpoint": "/v2/projects/{project\_id}/links",

"method": "POST",

"json\_payload": {

"nodes": [

{

"node\_id": "switch\_node\_id",

"adapter\_number": 0,

"port\_number": "{index}"

},

{

"node\_id": "host\_node\_id\_{index}",

"adapter\_number": 0,

"port\_number": 0

}

]

}

}

],

"note": "Replace placeholders like {project\_id}, {index}, {x\_offset}, and node\_ids dynamically in your script logic."

}

# “Small Software Companies network departments”:

Para este projeto foi implementada redes para pequenas empresas de software que têm um menor numero de departamentos e um menor numero de host em cada departamento.

Assim para uma melhor perceção existem vários departamentos para estes tipos de rede dos quais se destacam os seguintes:

Source: https://www.quora.com/What-are-the-different-departments-in-a-software-company

-“Human resource Manager”;

- “IT services”;

-“ Accounts and Finance”;

-“ Sales and marketing (Business development)”;

-“Guest Network”

-“ IT services”;

-“ Infrastructures and Security (Network, Systems & Services)”;

-“Development Engineering”;

-“ Administration department”;

-“ Marketing”;

-“Design”;

“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 percent of all software organizations. However, to persist and grow, small software companies need efficient, effective software engineering solutions.” (I. Richardson and C. G. Von Wangenheim, Jan.-Feb. 2007)

“The International Organization for Standardization set up a working group (ISO/IEC JTC1/SC7/WG24) on SE Life-Cycle Profiles 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” (I. Richardson and C. G. Von Wangenheim, Jan.-Feb. 2007)

Foi Criado uma estrutura protótipo, onde a implementação desta rede é criada e implementada manualmente no Cisco Packet Tracer e automatizada nas ferramentas de Automatização.

## Cisco Packet Tracer construção da rede manualmente: 1h de trabalho

Link para ficheiro Excel com todos os detalhes da construção da rede:

<https://ipvcpt-my.sharepoint.com/:x:/g/personal/eduardo_j_ipvc_pt/EeVkxh0SPNxGp39co22LiCYB8jyGscTHdPRm2UFI2k8--A?e=M4QkFn>

Esta rede pretende ser uma versão mais pequena de uma empresa de software composta por 10 departamentos onde cada departamento tem a sua respetiva VLAN.

A screen shot of a computer

AI-generated content may be incorrect.

Figura 6 Criação de um protótipo da Rede de uma pequena empresa de Software.

A white background with black text

AI-generated content may be incorrect.

Figura 7 Departamentos diferentes com diferentes Vlans

A white background with black lines

AI-generated content may be incorrect.

Figura 8 Departamentos diferentes com diferentes Vlans

A diagram of a network

AI-generated content may be incorrect.

Figura 9 Departamentos diferentes com diferentes Vlans

A white background with black lines

AI-generated content may be incorrect.

Figura 10 Departamentos diferentes com diferentes Vlans

A white background with black text

AI-generated content may be incorrect.

Figura 11 Departamentos diferentes com diferentes Vlans

## Cisco Packet Tracer configuração da rede manualmente: 2h de trabalho

Link para ficheiro Excel com todos os detalhes da construção da rede:

<https://ipvcpt-my.sharepoint.com/:x:/g/personal/eduardo_j_ipvc_pt/EeVkxh0SPNxGp39co22LiCYB8jyGscTHdPRm2UFI2k8--A?e=M4QkFn>

Foi necessário configurar todos os dispositivos das diferentes Vlans para ter comunicação com todos:

enable

conf t

vlan 40

name VLAN\_40

exit

interface range fa0/2 - 8

switchport mode access

switchport access vlan 40

exit

interface fa0/1

switchport mode trunk

exit

A screenshot of a computer

AI-generated content may be incorrect.

Figura 12 Configura as interfaces como trunk, permitindo tráfego de múltiplas VLANs na mesma ligação física

enable

conf t

interface g0/0

encapsulation dot1Q 0

ip address 192.168.0.1 255.255.255.240

exit

!

enable

conf t

interface g0/0.10

encapsulation dot1Q 10

ip address 192.168.10.1 255.255.255.240

exit

!

enable

conf t

interface g0/0.20

encapsulation dot1Q 20

ip address 192.168.20.1 255.255.255.240

exit

!

enable

conf t

interface g0/0.30

encapsulation dot1Q 30

ip address 192.168.30.1 255.255.255.240

exit

!

enable

conf t

interface g0/0.40

encapsulation dot1Q 40

ip address 192.168.40.1 255.255.255.248

exit

!

enable

conf t

interface g0/0.50

encapsulation dot1Q 50

ip address 192.168.50.1 255.255.255.248

exit

!

enable

conf t

interface g0/0.60

encapsulation dot1Q 60

ip address 192.168.60.1 255.255.255.0

exit

!

enable

conf t

interface g0/0.70

encapsulation dot1Q 70

ip address 192.168.70.1 255.255.255.248

exit

!

enable

conf t

interface g0/0.80

encapsulation dot1Q 80

ip address 192.168.80.1 255.255.255.0

exit

!

enable

conf t

interface g0/0.90

encapsulation dot1Q 90

ip address 192.168.90.1 255.255.255.248

exit

!

# Processos realizados:

- Pesquisa aprofundada sobre documentação e trabalhos relacionados;

- Criação de estrutura de redes em Excel;

- Criação e sua respetiva configuração da estrutura da rede manualmente no Cisco Packet Tracer.

- Tentativas de automatização na Aplicação Cisco Packet Tracer;

- Exploração da ferramenta GNS3 e sua respetiva configuração e documentação;

- Exploração de Máquinas virtuais para criação de servidor GNS3 e containers;

- Exploração de Docker files;

- Esquema draw.io estilo “DevOps” integração de ferramentas para automatizar processos de configuração e conexão de dispositivos na rede;

- Artigo e Report com detalhes e bibliografia;

- Instalação da v3.0.4/5 GUI GNS3;

- Instalação da VMware GUI para máquinas virtuais (já não está mais em uso.);

- Instalação da v3.0.4/5 Server GNS3;

- Instalação Ansible e sua respetiva configuração;

- Instalação Python e PIP com a sua respetiva configuração;

- Criação de script inicial para configuração do servidor;

- Exploração da ferramenta *OrbStack.*

# Referências Bibliográficas:

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," 2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS), St. Louis, MO, USA, 2024, pp. 1-6, doi: 10.1109/ICPS59941.2024.10640021. keywords: {Automation;Embedded systems;Reviews;Manuals;Cyber-physical systems;Software;Hardware;test automation;embedded systems;robot framework;ansible;ranorex},

W. Sha, M. Wu and Y. Qiao, "Research on Automated Operation and Maintenance Methods for Power Management Networks Based on Ansible," 2023 5th International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 2023, pp. 591-595, doi: 10.1109/ICAICA58456.2023.10405593. keywords: {Automation;Power system management;Configuration management;Passwords;Maintenance engineering;Servers;Task analysis;network automation;server management;path optimization},

A. -F. Sicoe, R. Botez, I. -A. Ivanciu and V. Dobrota, "Fully Automated Testbed of Cisco Virtual Routers in Cloud Based Environments," 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Sofia, Bulgaria, 2022, pp. 49-53, doi: 10.1109/BlackSeaCom54372.2022.9858288. keywords: {Cloud computing;Codes;Network topology;Sea measurements;Configuration management;Manuals;Virtual machining;Ansible;Cloud Automation;Cloud Computing;Configuration Management;Infrastructure as Code},

R. Emiliano and M. Antunes, "Automatic network configuration in virtualized environment using GNS3," 2015 10th International Conference on Computer Science & Education (ICCSE), Cambridge, UK, 2015, pp. 25-30, doi: 10.1109/ICCSE.2015.7250212. keywords: {Routing protocols;Network topology;IP networks;Topology;Computer science;Operating systems;Generators;Automatic configuration;GNS3;virtualization;Cisco IOS;virtual lab classroom},

*DiagramasProjeto2* (n.d.). Google Docs. <https://drive.google.com/file/d/1VrTOyrEzW7ied956Rqg7bXCpUOMQAXA1/view?usp=sharing>

#Ainda não está no Overleaf em formato BiB.tex

I. Richardson and C. G. Von Wangenheim, "Guest Editors' Introduction: Why are Small Software Organizations Different?," in IEEE Software, vol. 24, no. 1, pp. 18-22, Jan.-Feb. 2007, doi: 10.1109/MS.2007.12.

keywords: {Companies;IEC standards;ISO standards;Software quality;Software engineering;Software standards;Software maintenance;Innovation management;Knowledge management;Environmental management;small software organizations;best practices;version control;software tools and environments},