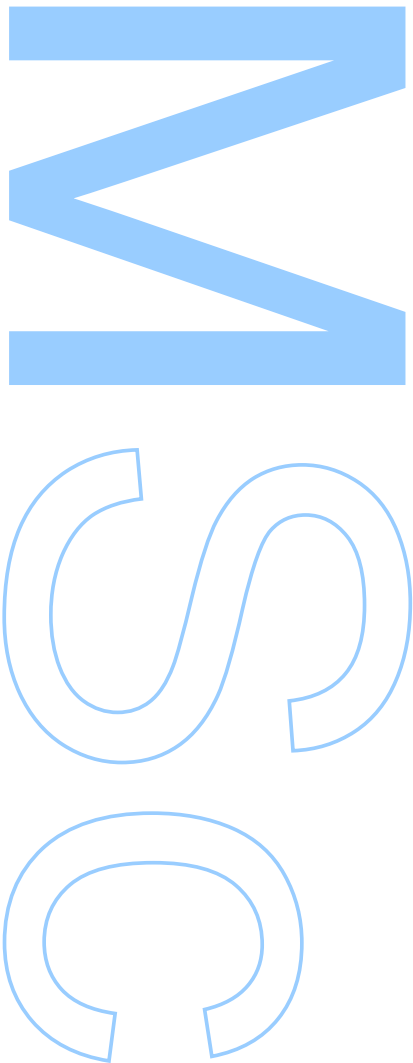
System For Pratical Evaluations of Network Administration Course

Diogo Nunes
Mestrado em Engenharia de Redes e Sistemas Informáticos
Departamento de Ciência de Computadores
2025

Orientador
Prof. Rui Prior, Faculdade de Ciências

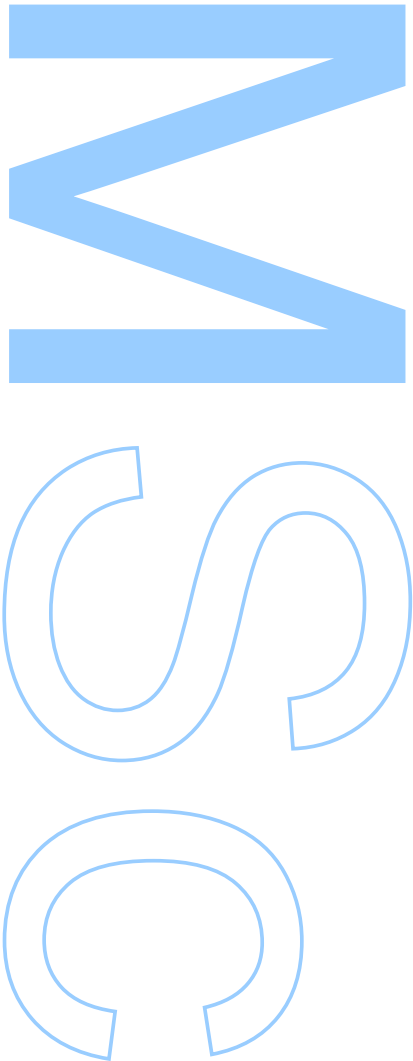


U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_____



Acknowledgements

Acknowledge ALL the people!

Resumo

Este tese é sobre alguma coisa

Palavras-chave: física (keywords em português)

Abstract

This thesis is about something, I guess.

Keywords: Computer Science

Table of Contents

List of Figures

List of Abbreviations

ANN Artificial Neural Network. 6

CS Computer Science. 1

1. Introduction

In today's digital age the need for qualified Computer Science (CS) professionals is growing. The CS field is vast and has many areas of expertise, one of which is network administration. It is a crucial part of any organization, as it is responsible for the maintenance and management of the organization's network infrastructure. Proper training for network administrators is crucial for preparing them for real-world situations. One way to provide this training is through practical evaluations, allowing students to apply the knowledge they have acquired in a real-world scenario, helping them to develop the skills they will need in their future careers.

Creating a physical network environment for practical evaluations may be costly and challenging to scale for large student populations. Emulation and virtualization technologies can help to simplify and cost-effectively create practice environments for students. These technologies alone do not address the issue of manually reviewing a network topology's setup. Manually reviewing each student's network configuration can be time-consuming and prone to human error, rendering it challenging for their instructors. Automating the evaluation process may substantially alleviate the burden on educators and guarantee uniform and fair assessments.

1.1. Aims and Objectives

This dissertation continues the work of a previous student, who carried out research and first steps of development of a system for automated evaluation system for network topologies. The main goal is to design and implement a scalable system capable of automatically evaluating evaluating network topologies that make use of different vendors and device types. The support for different vendors and device types is crucial, as it allows students to practice with a variety of networking equipment, preparing them for the real-world scenarios they will face in their future careers. Automating the evaluation process will help educators dedicate more time to other tasks such as supporting students, and would also provide a more consistent and fair evaluation, eliminating the possibility of human error.

Talk about the end goal

The main steps of this project are as follows:

- Study the bases for the system already developed

- Requirements gathering
- Identification of the main problems that need to be solved
- Proposal of solutions for these problems
- System design
- Implementation of a prototype
- Testing with volunteers to validate the system and identify possible limitations.

2. Background

This chapters main focus is to provide the reader with the necessary background information to understand the context of this project. The main goal of this project is to provide a system capable of automatically evaluating network topologies by validating configurations and running tests on different devices in the network. Analogue systems exist in the market, primarily focused in programming. These systems receive code from students and subsequently run tests on it against multiples test cases and are already widely deployed in educational environments. Shifting from programming to network topologies appears simple at first glance but comes with a particular set of challenges not present in programming evaluations. Each student will require an individual working environment, which can be addressed by using virtualization platforms. There is also the matter of communicating with the devices in the network, which can be addressed by using network automation tools. Finally there is the matter of combining these technologies to create a system capable of automatically evaluating network topologies.

2.1. Programming Evaluation Systems

While not directly related, they are the main inspiration for this project. Programming evaluation systems are widely deployed in universities and other educational institutions. These systems receive code from students and subsequently run tests on it, outputting a score and even being configurable to provide students the first test case that they failed in, guiding students to the solution without handing it out.

These tools typically provide a structured approach to test coding and problem solving skills. They begin by offering a problem statement coupled with an optional image and an example test case, normally in the form of input and expected output. Users can interact with the system by use of an online code editor, where they can write their solution and submit it for evaluation, or by uploading a file with their solution. The system then evaluates the provided solution against multiple pre-defined test cases, and validating the output agaisnt the know-good output, outputting a score based on the number of test cases passed. The system may also be configured to have time and/or memory constraints, to ensure that temporal and spatial complexity are also taken into account.

All of these, serve to provide a thorough evaluation of the student's solution, which can help guide a student to better their coding and problem solving skills.

In the context of the Department of Computer Science (DCC), Mooshak and Codex are commonly deployed to be used in the context of classes and even exams and programming contests.

The main differentiator between these systems and the one proposed in this project is the ability to solve a network exercise using multiple configurations across multiple devices, while programming evaluation systems will expect the same output every time, given the same input. Another key difference is the fact that programming evaluation systems dont always provide a working environment for the students to test their code, owing to the fact that students might prefer to user their own development environment for initial development and testing. This project aims to provide a working environment for students to work on for a few reasons that will be discussed later on

DISCUSS REASONS WHY LATER

2.1.1 Mooshak

Mooshak is a web-based system for managing programming contests and also to act as an automatic judge of programming contests [1]. It supports a variety of programming languages like Java, C, etc. Under each contest students will find one more problem definitions each containing varying sets of test cases in input-output pairs. After submiting their solution, the system will compile and run the code against the test cases giving a score based on the the amount of test cases passed. The system can also differentiate between differing types of errors, such as not giving the expected output, poorly formatted output, failure to compile or even exceeding the time limits. Mooshak also includes some features designed to drive competition between students, like a real time leaderboard and the ability to have more than 100% of the score for a given contest.

The system however is not without its limitations as it uses plain text files for its test cases and validates the output of student's code character by character, which can lead to false negatives if the output is not formatted exactly as expected.

2.2. GNS3

Graphical Network Simulator-3 (GNS3) is an open-source graphical network emulator software that allows the user to create complex network topologies and interact with the various devices in it. It is widely used for educational purposes and is often used in preparation for professionals network certifications like the Cisco Certified Network Associate (CCNA).

GNS3 employs a simple drag and drop interface to allow users to add new devices, make links between them and even add textual annotations. The software allows users to interact with the devices by way of a console or even a GUI if the device supports it. The software also allows users to export their topologies to be shared with others, which can be useful for teachers to provide students with a pre-configured topology to work on.

Additionally, the software supports packet capturing which is essential for students to develop their debugging and troubleshoting skills. Finally the software can also be interacted with via a Representational State Transfer (REST) Application Programming Interface (API) which can be of particular interest for this project.

2.2.1 Architecture

The software can be employed in a variety of ways due to its architecture [2] that separates the user interfaces that it offers, namely the locally installed gns3-gui as well as the browser accessible gns3-web, from the gns3-server that runs the emulations and the controller who orchestrates everything.

add image about gns3 architecture

2.2.1.1 Controller

The controller is integrated in the gns3-server project and is responsible for communicating with all the other components of the software. The controller is a singleton, meaning there should only be one instance of it running at any given time, and it does not support concurrent requests. It is able to control multiple compute instances if so desired, each capable of hosting one or more emulator instances, varying depending on their complexity. The controller also exposes the REST API allowing the ability to interact with the software programatically. All communication is done over Hypertext Transfer Protocol (HTTP) in JavaScript Object Notation (JSON) format and there is support for basic HTTP authentication as well as notifications via websockets.

2.2.1.2 Compute

The compute is also integrated in the gns3-server project and controls the various emulators required to run the nodes in the topology. The list of currently supported emulators is:

- Dynamips - Used to emulate Cisco routers and basic switching.

- IOS on Unix (IOU) - Used to emulate Cisco Internetworking Operating System (IOS) devices.
- Quick Emulator (QEMU) - Used to emulate a wide variety of devices.
- Virtual PC Simulator (VPCS) - A basic program meant to simulate a basic PC
- VMware/VirtualBox - Used to run virtual machines with nested virtualization support
- Docker - Used to run containers

2.2.1.3 GUI

The GUI is composed of two separate but with mostly identical functionality, namely the gns3-gui and the gns3-web projects. The gns3-gui project is a desktop application that is used to to interact with a local or remote gns3-server instance. It is written in Python and uses the Qt framework for the graphical interface. The gns3-web is a web application that is accessed via a web browser it is still in a beta stage but is already capable enough to be used as a substitute for the gns3-gui.

add image of gns3-web

2.3. ProxmoxVE
2.4. Nornir
2.5. Python?

Main technologies used to talk about Python Nornir GNS3 ProxmoxVE Flask Requests -> celery -> HTTPX WSGI -> Gunicorn Linux NGINX? Gunicorn?

References

[1] J. Leal and F. Silva, “Mooshak: a web-based multi-site programming contest system,” *SOFTWARE-PRACTICE & EXPERIENCE*, vol. 33, no. 6, pp. 567–581, MAY 2003. [Cited on page 4.]

[2] GNS3 Documentation, “Architecture,” n.d., accessed: March 15, 2025. [Online]. Available: https://docs.gns3.com/docs/using-gns3/design/architecture/ [Cited on page 5.]

Appendix Title Here

Write your Appendix content here.