FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Configuration and Study of a Computer Network and Development of a File Transfer Protocol Client

Alexandre Ferreira
Paulo Saavedra
Class nº7

Licenciatura em Engenharia Informática e Computação

November 9, 2025

© FEUP - Faculty of Engineering, University of Porto, 2025

Configuration and Study of a Computer Network and Development of a File Transfer Protocol Client

Alexandre Ferreira

Paulo Saavedra
Class nº7

Licenciatura em Engenharia Informática e Computação

November 9, 2025

Resumo

Este relatório descreve o estudo, configuração e análise de uma rede de computadores, bem como o desenvolvimento de um cliente File Transfer Protocol (FTP). A configuração da rede foi realizada através de scripts automatizados, abrangendo a atribuição de endereços Internet Protocol (IP), configuração de routing, Domain Name System (DNS) e ligação de múltiplos dispositivos (máquinas, switch e router). O cliente FTP, desenvolvido em C, implementa autenticação de utilizador, modo passivo, transferências binárias e download de ficheiros, seguindo as normas RFC959 e RFC1738. O projeto demonstra competências práticas tanto na configuração de redes como na implementação de protocolos de aplicação.

Abstract

This report describes the study, configuration, and analysis of a computer network, as well as the development of an File Transfer Protocol (FTP) client. The network setup was automated through scripts, covering Internet Protocol (IP) address assignment, routing configuration, Domain Name System (DNS), and the interconnection of multiple devices (machines, switch, and router). The FTP client, developed in C, implements user authentication, passive mode, binary transfers, and file downloads, following RFC959 and RFC1738 standards. The project demonstrates practical skills in both network configuration and application-layer protocol implementation.

Contents

List of Figures

List of Tables

List of Acronyms

FTP File Transfer Protocol

IP Internet Protocol

DNS Domain Name System

Chapter 1

Introduction

Briefly introduce the project objectives: network configuration and FTP client development.

Chapter 2

Part 1 – Download Application

2.1 Architecture of the Download Application

Describe the FTP client architecture, main modules, and protocol features.

2.2 Successful Download Report

Describe a successful file download, including a Wireshark screenshot of FTP packets

Chapter 3

Part 2 – Network Configuration and Analysis

3.1 Experiment 1 - Configure an IP Network

3.1.1 Network Architecture

For this experiment, we connected TUX3 and TUX4 through the switch and configured their IP addresses as requested in the project description.

3.1.2 Objectives

State the learning objectives.

3.1.3 Main Configuration Commands

List the main commands/scripts used.

3.1.4 Relevant Logs

Show relevant logs and outputs.

3.1.5 Analysis

Discuss the results and learning points.

3.2 Experiment 2 - Implement two bridges in a switch

3.2.1 Network Architecture

Describe the network setup for this experiment.

3.2.2 Objectives

State the learning objectives.

3.2.3 Main Configuration Commands

List the main commands/scripts used.

3.2.4 Relevant Logs

Show relevant logs and outputs.

3.2.5 Analysis

Discuss the results and learning points.

3.3 Experiment 3 - Configure a Router in Linux

3.3.1 Network Architecture

Describe the network setup for this experiment.

3.3.2 Objectives

State the learning objectives.

3.3.3 Main Configuration Commands

List the main commands/scripts used.

3.3.4 Relevant Logs

Show relevant logs and outputs.

3.3.5 Analysis

Discuss the results and learning points.

3.4 Experiment 4 - Configure a Commercial Router and Implement NAT

3.4.1 Network Architecture

For this experiment, the commercial router was connected to the lab network through ether1 and to the Y1 bridge through ether2. The ip addresses and routes were configured as specified in the project description.

3.4.2 Configuring a Static Route in a Commercial Router

To configure a static route in the commercial router, we accessed its console and entered the necessary commands. We added a new static route with the destination network, subnet mask, and gateway as per the project requirements. After saving the configuration, we verified the route was correctly added by checking the routing table in the router's interface.

3.4.3 ICMP Redirection

3.4.4 Analysis

Discuss the results and learning points.

3.5 Experiment 5 - DNS

3.5.1 Network Architecture

Describe the network setup for this experiment.

3.5.2 Objectives

State the learning objectives.

3.5.3 Main Configuration Commands

List the main commands/scripts used.

3.5.4 Relevant Logs

Show relevant logs and outputs.

3.5.5 Analysis

Discuss the results and learning points.

3.6 Experiment 6 - TCP Connections

3.6.1 Network Architecture

Describe the network setup for this experiment.

3.6.2 Objectives

State the learning objectives.

3.6.3 Main Configuration Commands

List the main commands/scripts used.

3.6.4 Relevant Logs

Show relevant logs and outputs.

3.6.5 Analysis

Discuss the results and learning points.

Chapter 4

Conclusions

Summarize findings, challenges, and skills acquired.

Chapter 5

References

Chapter 6

Annexes

6.1 Download Application Code

Include the source code

6.2 Configuration Commands

List scripts and manual commands used.

6.3 Captured Logs

6.3.1 Experiment 4 - Configure a Commercial Router and Implement NAT

No.	Time	Source	Destination	Protocol	Length	Info
7	11.458149940	172.16.100.1	172.16.101.254	ICMP	98	Echo (ping) request id=0x1669, seq=1/256, ttl=64 (reply in 8)
8	11.458578848	172.16.101.254	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1669, seq=1/256, ttl=63 (request in 7)
10	12.474141815	172.16.100.1	172.16.101.254	ICMP	98	Echo (ping) request id=0x1669, seq=2/512, ttl=64 (reply in 11)
11	12.474520515	172.16.101.254	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1669, seq=2/512, ttl=63 (request in 10)
12	13.498125438	172.16.100.1	172.16.101.254	ICMP	98	Echo (ping) request id=0x1669, seq=3/768, ttl=64 (reply in 13)
13	13.498499705	172.16.101.254	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1669, seq=3/768, ttl=63 (request in 12)

Figure 6.1: TUX3 pinging RC's ether2 interface

21	19.466225787	172.16.100.1	172.16.101.1	ICMP	98	Echo (ping) request id=0x166a, seq=1/256, ttl=64 (reply in 22)
22	19.466638778	172.16.101.1	172.16.100.1	ICMP	98	Echo (ping) reply id=0x166a, seq=1/256, ttl=63 (request in 21)
24	20.474131789	172.16.100.1	172.16.101.1	ICMP	98	Echo (ping) request id=0x166a, seq=2/512, ttl=64 (reply in 25)
25	20.474580273	172.16.101.1	172.16.100.1	ICMP	98	Echo (ping) reply id=0x166a, seq=2/512, ttl=63 (request in 24)
26	21.498136126	172.16.100.1	172.16.101.1	ICMP	98	Echo (ping) request id=0x166a, seq=3/768, ttl=64 (reply in 27)
27	21.498583530	172.16.101.1	172.16.100.1	ICMP	98	Echo (ping) reply id=0x166a, seq=3/768, ttl=63 (request in 26)
29	22.522123877	172.16.100.1	172.16.101.1	ICMP	98	Echo (ping) request id=0x166a, seq=4/1024, ttl=64 (reply in 30)
30	22.522574250	172.16.101.1	172.16.100.1	ICMP	98	Echo (ping) reply id=0x166a, seq=4/1024, ttl=63 (request in 29)

Figure 6.2: TUX3 pinging TUX2

47	39.618235357	172.16.100.1	172.16.101.253	ICMP	98	Echo (ping) request id=0x1672, seq=1/256, ttl=64 (reply in 48)
48	39.618428060	172.16.101.253	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1672, seq=1/256, ttl=64 (request in 47)
50	40.634117824	172.16.100.1	172.16.101.253	ICMP	98	Echo (ping) request id=0x1672, seq=2/512, ttl=64 (reply in 51)
51	40.634354731	172.16.101.253	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1672, seq=2/512, ttl=64 (request in 50)
52	41.658122432	172.16.100.1	172.16.101.253	ICMP	98	Echo (ping) request id=0x1672, seq=3/768, ttl=64 (reply in 53)
53	41.658318495	172.16.101.253	172.16.100.1	ICMP	98	Echo (ping) reply id=0x1672, seq=3/768, ttl=64 (request in 52)

Figure 6.3: TUX3 pinging TUX4's ether2 interface

```
    59 47.810271669  172.16.100.1       172.16.100.254     ICMP      98 Echo (ping) request  id=0x1673, seq=1/256, ttl=64 (reply in 60)
    60 47.810495585  172.16.100.254     172.16.100.1       ICMP      98 Echo (ping) reply    id=0x1673, seq=1/256, ttl=64 (request in 59)
    62 48.826123774  172.16.100.1       172.16.100.254     ICMP      98 Echo (ping) request  id=0x1673, seq=2/512, ttl=64 (reply in 63)
    63 48.826345881  172.16.100.254     172.16.100.1       ICMP      98 Echo (ping) reply    id=0x1673, seq=2/512, ttl=64 (request in 62)
    64 49.850133334  172.16.100.1       172.16.100.254     ICMP      98 Echo (ping) request  id=0x1673, seq=3/768, ttl=64 (reply in 65)
    65 49.850350150  172.16.100.254     172.16.100.1       ICMP      98 Echo (ping) reply    id=0x1673, seq=3/768, ttl=64 (request in 64)
```

Figure 6.4: TUX3 pinging TUX4's ether1 interface