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

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

1 Introduction **1**

2 Part 1 – Download Application **2**
- 2.1 Architecture of the Download Application . . . 2
- 2.2 Successful Download Report . . . 2

3 Part 2 – Network Configuration and Analysis **3**
- 3.1 Experiment 1 - Configure an IP Network . . . 3
  - 3.1.1 Network Architecture . . . 3
  - 3.1.2 Objectives . . . 3
  - 3.1.3 Main Configuration Commands . . . 3
  - 3.1.4 Relevant Logs . . . 3
  - 3.1.5 Analysis . . . 3
- 3.2 Experiment 2 - Implement two bridges in a switch . . . 3
  - 3.2.1 Network Architecture . . . 3
  - 3.2.2 Objectives . . . 4
  - 3.2.3 Main Configuration Commands . . . 4
  - 3.2.4 Relevant Logs . . . 4
  - 3.2.5 Analysis . . . 4
- 3.3 Experiment 3 - Configure a Router in Linux . . . 4
  - 3.3.1 Network Architecture . . . 4
  - 3.3.2 Objectives . . . 4
  - 3.3.3 Main Configuration Commands . . . 4
  - 3.3.4 Relevant Logs . . . 4
  - 3.3.5 Analysis . . . 4
- 3.4 Experiment 4 - Configure a Commercial Router and Implement NAT . . . 4
  - 3.4.1 Configuring a Static Route in a Commercial Router . . . 4
  - 3.4.2 ICMP Redirection . . . 5
  - 3.4.3 Configuring Network Address Translation . . . 5
  - 3.4.4 Network Address Translation . . . 5
- 3.5 Experiment 5 - Domain Name System (DNS) . . . 5
  - 3.5.1 Configuring a DNS Service . . . 5
  - 3.5.2 DNS Packets . . . 5
- 3.6 Experiment 6 - TCP Connections . . . 6
  - 3.6.1 TCP Connections in FTP . . . 6
  - 3.6.2 Automatic Repeat reQuest in TCP . . . 6
  - 3.6.3 TCP Congestion Control . . . 7

4 Conclusions **8**

5 References **9**

6 Annexes **10**
6.1 Download Application Code 10
6.2 Configuration Commands 10
6.3 Captured Logs 10
6.3.1 Experiment 4 - Configure a Commercial Router and Implement NAT . . 10
6.3.2 Experiment 6 - TCP Connections 11

List of Figures

6.1 TUX3 pinging RC's ether2 interface 10
6.2 TUX3 pinging TUX2 10
6.3 TUX3 pinging TUX4's ether2 interface 10
6.4 TUX3 pinging TUX4's ether1 interface 11
6.5 Changes in Throughput during concurrent FTP download 11

List of Tables

3.1 TCP Header Fields and Their Role in ARQ 6
3.2 TCP Header Fields and Their Role in Congestion Control 7

List of Acronyms

FTP File Transfer Protocol

IP Internet Protocol

DNS Domain Name System

TCP Transmission Control Protocol

ICMP Internet Control Message Protocol

NAT Network Address Translation

ARQ Automatic Repeat reQuest

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 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.

3.4.2 ICMP Redirection

To test Internet Control Message Protocol (ICMP) redirection, we initiated ping requests from TUX2 to TUX3. Initially, the packets were routed through TUX4 as it was the shortest path to subnet 172.16.Y0.0/24. After that, as requested in the project description, we disabled redirection acceptance on TUX2, and changed the routes to force the packets to go through the commercial router. This way, we observed that TUX2 continued to send packets through TUX4, ignoring the ICMP redirect messages from the commercial router. With ICMP redirection disabled, after the first ICMP redirect, TUX2 switched to its original routing path, demonstrating the effect of disabling ICMP redirect acceptance on the routing behavior of the host.

3.4.3 Configuring Network Address Translation

As Network Address Translation (NAT) was already enabled by default on the commercial router, to understand its functionality, we disabled it through the router's console and observed its effects on the network communication. We then re-enabled NAT to restore normal operation.

3.4.4 Network Address Translation

To understand NAT functionality, we performed ping tests from TUX3 to the File Transfer Protocol (FTP) server before and after disabling NAT on the commercial router. With NAT enabled, TUX3 was able to successfully ping the server, as the router correctly translated the private Internet Protocol (IP) address of TUX3 to a public IP address for communication. With NAT disabled, the pings failed, indicating that the server could not reach TUX3's private IP address directly. This demonstrated the importance of NAT in allowing devices within a private network to communicate with external networks.

3.5 Experiment 5 - Domain Name System (DNS)

3.5.1 Configuring a DNS Service

To configure the DNS service on TUX2, TUX3, and TUX4, we modified the `/etc/resolv.conf` file on each machine to include the IP address of the FTP server `services.netlab.fe.up.pt`. This allowed the machines to resolve the domain name to its corresponding IP address when attempting to connect to the FTP server.

3.5.2 DNS Packets

Domain Name System (DNS) packets were captured using Wireshark while performing a domain name resolution for `google.com` from TUX3. The captured packets showed the standard DNS query and response process, including the query sent by TUX3 to the DNS service and the corresponding response containing the resolved IP address of the server.

3.6 Experiment 6 - TCP Connections

3.6.1 TCP Connections in FTP

In order to successfully download a file using the FTP client, two Transmission Control Protocol (TCP) connections are established between the client and the server. The first connection is the control connection, which is used for sending commands and receiving responses. This connection is established on port 21 of the server. The second connection is the data connection, which is used for transferring files. Depending on whether active or passive mode is used, this connection can be initiated by either the client or the server on a dynamically assigned port. Every TCP connection involves a three-way handshake process to establish the connection, followed by data transfer, and finally a four-way handshake to terminate the connection.

3.6.2 Automatic Repeat reQuest in TCP

Automatic Repeat reQuest (ARQ) is a fundamental mechanism used in TCP to ensure reliable data transmission. It works by requiring the receiver to send acknowledgments (ACKs) back to the sender for the data packets received. If the sender does not receive an acknowledgment within a specified timeout period, it assumes that the packet was lost or corrupted and retransmits the packet. This process continues until the sender receives an acknowledgment for all sent packets, ensuring that all data is correctly received by the receiver. On duplicate ACKs, TCP can also perform fast retransmissions to improve efficiency.

The following table summarizes the key TCP header fields involved in the ARQ mechanism and their roles:

Field	**Role in ARQ**
Sequence Number	Identifies the **first byte** in this segment. Used for ordering and retransmission.
Acknowledgment Number	Indicates **next byte expected** by the receiver. Confirms receipt of previous bytes.
Flags	Especially **ACK**, **SYN**, **FIN**. ACK is critical for ARQ.
Window Size	Flow control; tells sender how many bytes can be sent before receiving an ACK.
Checksum	Detects corrupted segments; segments failing the checksum are discarded and retransmitted.

Table 3.1: TCP Header Fields and Their Role in ARQ

3.6.3 TCP Congestion Control

TCP employs several congestion control mechanisms to manage network congestion and ensure efficient data transmission. The primary algorithms used in TCP congestion control include Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery.

The following table highlights the main TCP header fields relevant to congestion control and their roles:

Field	**Role**
Sequence Number	Tracks the byte position for retransmission and ordering.
Acknowledgment Number	Confirms receipt of bytes; triggers congestion window (cwnd) increase.
Window Size (rwnd)	Receiver's advertised window (flow control).
Flags (ACK)	ACK signals successful receipt.
Optional TCP Options	e.g., SACK (Selective ACK) can improve fast retransmit.

Table 3.2: TCP Header Fields and Their Role in Congestion Control

To measure the effects of the TCP congestion control mechanism, we initiated a file download using our FTP client while capturing the TCP packets with Wireshark. By analyzing the captured packets, we observed the changes in the throughput during the concurrent download process, as can be seen in Figure 6.5.

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

6.3.2 Experiment 6 - TCP Connections

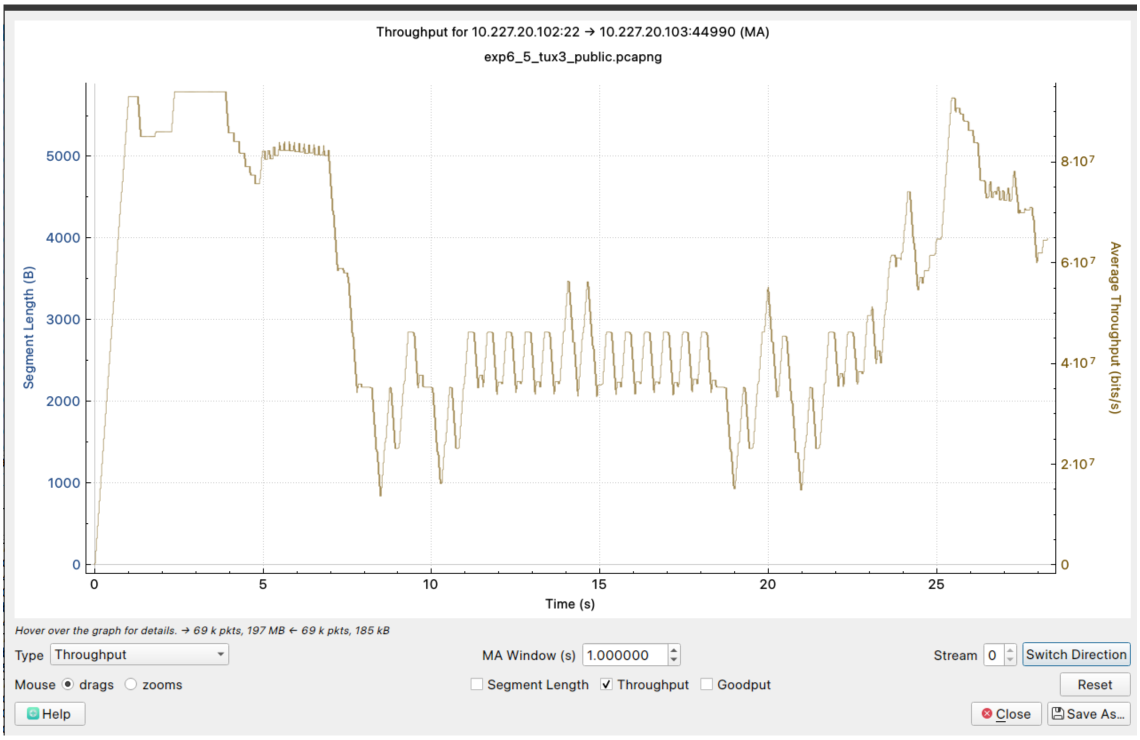

Figure 6.5: Changes in Throughput during concurrent FTP download