RC: Project 2

Project 2 – FTP Download and Network Configuration

António Oliveira Santos
up202008004

José Luís Nunes Osório
up202004653

December 2022

1 Abstract

The final evaluation component of the project saw us downloading a file from the Internet through a computer in a computer network configured throughout the semester. Both the application and the network were developed according to the requisites in the guide and the goal was, henceforth, met.

2 Introduction

This project was divided in 2 parts, that were developed simultaneously throughout the semester. One part consisted on the development of a download application implementing the FTP protocol according to RFC959, to be written in C and using Berkeley sockets. The configuration of a computer network was the accompanying part of this project, this network required access to the Internet, amongst other features that will be discussed ahead.

3 Download Application

The first part of this project consists of the development of an FTP download application that retrieves a specific file using the FTP protocol specified in the RFC-959, taking in an argument consisting of an URL following the syntax described in the RFC-1738. Examples:

```
main ftp.up.pt/pub/kodi/timestamp.txt
main rcom:rcom@netlab1.fe.up.pt/pipe.txt
```

The URLs sent in the arguments follow the structure of:

```
ftp://[user:password]@<host>/<url-path>
```

Through the development of this application we were able to gain an understanding of:

- The specifics of the TCP/IP protocol
- The specifics of the FTP protocol
- Locating and using RFC protocols
- Utilizing Berkeley sockets and TCP in C
- Using the hostname resolving DNS service in C
- Implementing an FTP client in C

3.1 Architecture of the download application

The application follows a predetermined path on downloading a given file:

1. Parsing the URL to extract username and password as well as the hostname, the path to the file and the filename.
2. Establishing a connection to the domain provided.
3. Logging in with the user info
4. Activating passive mode and creating a connection to the provided port.
5. Retrieving the file from the server.
6. Closing the connection and leaving.

Communication to remote servers is done through the use of TCP sockets, using the `gethostbyname()` function to resolve domain name and the standard FTP control port of **21**. The login process uses the provided login info, defaulting to an anonymous connection when it is not given. When parsing the URL, login info and URL info are stored in the **user_info** and **url_info** structs, respectively. The FTP server responses are read to a buffer of **1024 bytes** until reaching the end of a response, interpreting response codes in the necessary steps to check for any errors that might have occurred. When activating passive mode, the response sent from the server is handled differently, parsing the 6 received bytes *(IP1, IP2, IP3, IP4, PORT1, PORT2)* correctly to establish a connection to the data socket and retrieve the file. Upon downloading the file, the connections are closed on both sockets and any allocated memory is freed before exiting the program.

3.2 Report of a successful download

Report of a successful download is attached.

4 Experiments

4.1 General Concerns

In all of the further descriptions of the experiments realized along this project the following terminology will be used:

1. Computers are named tuxY[2, 3, 4], where Y is the number of the table in which the group realized the experiment.

2. The "switch" is a network switch used to connect a computer in a network, which receives and forwards packets to their correct destination, using MAC addresses. In this project Cisco brand switches were used.

3. The "ruler" is a similar hardware device to a switch that allowed easy access to the different ports of the tuxes, eth0, eth1, eth2, and serial. Ports in this device were commonly connected to the switch.

4. The "router" is another network device, physical or virtual, that is used to pass information between several computer networks. In this project, as will be explored further ahead, we implemented our own virtual router and also used a physical MikroTik router, as such, in order to distinguish between these two, we will be refering to the virtual router as simply router and the physical MikroTik router as MikroTik router.

5. In order to access the switch terminal the console port of this device must be connected to the T4 in the external ruler, which will not be discussed further as it does not play a big enough role in these experiments. Along side this, a connection between T3 and the serial port of one of the tuxes, must be established.

6. In a similar fashion, in order to access the MikroTik router's console, a connection from T4 to "Router MT" in the ruler is required, whilst keeping T3 and the serial port connected.

7. In order to access these terminals, GTKTerminal is used with the Baud Rate set to 115200.

4.2 Experiment 1

4.2.1 Description

The first experiment of this project is an introductory setup of the larger network that must be built. In this first contact we must configure an IP Network, for that we will use 2 machines and connect them through the switch. For that tux23 and tux24 were used.

4.2.2 Physical Configuration

This experiment required a very simple configuration of the swtich and ruler. We simply had to connect ports eth0 of tux23 and tux24 to two different ports of the switch.

4.2.3 Commands Configuration

Configuring the network was also a simple task requiring only a few commands on each tux.

Tux23:

```
ifconfig eth0 172.16.20.1/24
```

Tux24:

```
ifconfig eth0 172.16.20.254/24
```

4.2.4 Logs and Analysis

Using wireshark to capture packages, we were able to take a closer look at the communications that were happening underneath the ping commands between the two IPs of the newly created network. It is important to note that before collecting these logs, the ARP tables of the machines were cleared.

The wireshark logs show us the computers attempting to establish a communication, at first an ARP message is broadcasted into the network, to the ff:ff:ff:ff:ff:ff, asking "Who has 172.16.1.254? Tell 172.16.1.22". Essentially as there are no ARP entries mapping to each computer, the IP of the destination must be resolved, with this goal in mind the computer pinging, in this case tux23 with IP address 172.16.20.1 and MAC address 00:21:5a:61:2b:72, broadcasts a message using the ARP protocol asking the network for the computer with an IP address of its destination, once tux24 listens to this message it sends a response and the ping request and reply are exchanged. The ping packets are sent with the MAC addresses of the destination and source already resolved and are therefore exchanged correctly arriving at their correct destinations.

4.3 Experiment 2

4.3.1 Description

The second experiment saw the introduction of bridges and a third computer, tux 22. The goal of this step was to correctly configure 2 bridges in the switch and analyze the behaviour of communications.

4.3.2 Physical Configuration

The configuration was similar to that of the previous experiment with the only difference being that the there is an extra connection now from tux22 to the switch.

4.3.3 Commands Configuration

For tux23 and tux24 the commands were similar to that of the previous stage.

Tux22:

```
ifconfig eth0 172.16.21.1/24
```

Switch:

```
interface bridge add name=
bridge20
interface bridge add name=
bridge21
interface bridge port remove [
REMOVE CONNECTED PORTS IN THE
SWITCH]
interface bridge port add bridge
=bridge20 interface
=172.16.20.1/24
interface bridge port add bridge
=bridge20 interface
=172.16.20.254/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.1/24
```

4.3.4 Logs and Analysis

After the setup is concluded we are able to start capturing packets being transmited between the machines.

It is clear that connection between tux23 and tux24 remains the same, as there has not been any changes to this "system", however, communications between them and tux22, both "from" and "to", are impossible. The impossibility of this communication is expected as there is no connection between. In fact the fact that there are two different broadcast addresses, one in bridge20 and another in bridge21, reinforces this notion and helps us draw the conclusion that if we want to analyze the communication between these two bridges something must establish a link between them.

4.4 Experiment 3

4.4.1 Description

From the previous experiment we were able to figure out that connection between the bridges was impossible, meaning it required something or someone to finish the link, with this in mind, the goal for this experiment was to implement a router in tux24. In theory, by placing this machine in both networks we could be able to transmit messages between the two subnets.

4.4.2 Physical Configuration

Just like before, the configuration was sim-

ilar with a new cable between tux24's eth1 and a port on the switch.

4.4.3 Commands Configuration

Resetting the configuration of all of the computers, the necessary commands are as follows.

Tux23:

```
ifconfig eth0 172.16.20.1/24
route add -net 172.16.21.0/24 gw
172.16.20.254
```

Tux24:

```
ifconfig eth0 172.16.20.254/24
ifconfig eth1 172.16.21.253/24
echo 1 > /proc/sys/net/ipv4/
ip_forward
echo 0 > /proc/sys/net/ipv4/
icmp_echo_ignore_broadcasts
```

Tux22:

```
ifconfig eth0 172.16.21.1/24
route add -net 172.16.20.0/24 gw
172.16.21.253
```

Switch:

```
interface bridge add name=
bridge20
interface bridge add name=
bridge21
interface bridge port remove [
REMOVE CONNECTED PORTS IN THE
SWITCH]
interface bridge port add bridge
=bridge20 interface
=172.16.20.1/24
interface bridge port add bridge
=bridge20 interface
=172.16.20.254/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.1/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.253/24
```

4.4.4 Logs and Analysis

This experiment has expanded oon the previous by allowing communications between the two bridges. As expected, the introduction of a router has allowed packets to be shared between the two bridges and hence computers in them. Analyzing the routes in each of the tuxes we are able to see that tux23 and tux22 both have routes that lead packages with the destination of the other subnet through the known port of the router in their subnet, that is to say that, per example, packages in tux23 that have a destination in the subnet 172.16.21.0 are directed through the gateway in 172.16.21.254 to find their destination, meaning that when a attempting to communicate with tux22, tux23 sends packages through tux24 who then, acting as a router between the two, directs packages to their correct final destination. This system is made possible by enabling IP forwarding and disabling ICMP echo-ignore-broadcast in tux24, these changes quarantee that broadcasts in each network to the other network are heard in both and that computers in bridge20 are able to ask for the MAC addresses of computers in bridge21, which then allows for the transfer of packets as IP is forwarding is enabled and packets can be forwarded through the router.

In the wireshark file captured in tux23 by the group we are able to see the ARP protocol resolving requests for IP addresses in both subnets. The logs show ARP communications requesting information about the addresses in 172.16.20.254 which is the address of tux24, meaning that tux23 is looking to know were it should sent its packets, and right after we are able to notice tux24 looking for 172.16.20.1, which leads us to believe that tux22 has broadcasted a request for the address of tux23 and tux24 is passing it forward to the, all of this happens when a ping is attempted to an address in bridge21.

4.5 Experiment 4 5

4.5.1 Description

With a local network of 3 computers implemented it was then possible to attempt to establish a connection with an outside network, the Internet. With the previous setup leading up to this point in the project, all that was needed to do was add a comercial router to one of the bridges. In addition, experiment 5 requested the DNS to be configured in all of the machines, since this is a small step and one closely related to experiment 4, we will also explore its conclusions in this part of the report.

4.5.2 Physical Configuration

The physical configuration is in all similar to the previous steps, im addition a connec-

tion between P2.1 and eth1 in the MikroTik router and another one between eth2 and a port in the switch was necessary. The first allows the router to be connected to the room's Internet and the second connects the router to one of the bridges in the switch, and therefore to the local network that we created thus far.

4.5.3 Commands Configuration

Resetting the configuration of all of the computers, the necessary commands are as follows.

Tux23:

```
ifconfig eth0 172.16.20.1/24
route add -net 172.16.21.0/24 gw
172.16.20.254
```

Tux24:

```
ifconfig eth0 172.16.20.254/24
ifconfig eth1 172.16.21.253/24
echo 1 > /proc/sys/net/ipv4/
ip_forward
echo 0 > /proc/sys/net/ipv4/
icmp_echo_ignore_broadcasts
```

Tux22:

```
ifconfig eth0 172.16.21.1/24
route add -net 172.16.20.0/24 gw
172.16.21.253
```

Switch:

```
interface bridge add name=
bridge20
interface bridge add name=
bridge21
interface bridge port remove [
REMOVE CONNECTED PORTS IN THE
SWITCH]
interface bridge port add bridge
=bridge20 interface
=172.16.20.1/24
interface bridge port add bridge
=bridge20 interface
=172.16.20.254/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.1/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.253/24
interface bridge port add bridge
=bridge21 interface
=172.16.21.254/24
```

MikroTik Router:

```
ip address add address
=172.16.21.254/24 interface=eth2
ip address add address
=172.16.1.29/24 interface=eth1
ip route add dst-address
=0.0.0.0/0 gateway=172.16.1.254
ip route add dst-address
=172.16.20.0/24 gateway
=172.16.21.253
```

In order to correctly setup DNS in each computer the file **/etc/resolv.conf** was edited and the line **nameserver 172.16.2.254** was added.

4.5.4 Logs and Analysis

Regarding experiment 4, in order to study the influence of redirects on communications these were disabled in tux23 and through the analysis of wireshark captures of a ping command from this machine to tux22 we were able to conclude that the paths taken by packages in these different instances differed. When redirects were enabled, since tux23 does not, by default, know a way to directly reach tux22, it sends packages to the MikroTik router and these are then delivered to tux24, upon this exchange, further communications are no longer done through this path as it now acknowledges tux24 as a more direct route to its destination. When this is not the case, every single package triggers a redirect, this behaviour makes these exchanges take longer and therefore the whole process is slower.

The implementation of DNS, in experiment 5, simply added a few more packets as the DNS server attempts to resolve the URL that is sent in the ping request into its IP address in the Internet.

4.6 Experiment 6

4.6.1 Description

In the final experiment of the project, the group was asked to run the developed FTP download application in a machine in the configured network and capture the packages exchanged in its execution.

4.6.2 Physical Configuration

The configuration is exactly the same as in the previous experiences, as no additional configuration was required for this one.

4.6.3 Commands Configuration

The configuration is exactly the same as

in the previous experiences, as no additional configuration was required for this one.

4.6.4 Logs and Analysis

The wireshark files that resulted from the execution of the application in tux23 confirmed and reassured us that the developed software was fulfilling its intended purpose, we saw both TCP and FTP packets being exchanged, with TCP packets acting as mediators of the communication and FTP packets containing the data of the file being downloaded.

5 Conclusions

Throughout this project the pratical approach that was taken allowed us to more deeply and intimately explore how computer networks function and their design. The download application was designed according to the requested functionalities and is a solid and well structured code base, furthermore all of the experiments were completed without any major problems and they accurately complement the theoretical notions discussed, which allows us to conclude that the goals of this project have been met.