RCOM - 2º Trabalho Laboratorial

Relatório Final

Licenciatura em Engenharia Informática e Computação - Redes de Computadores

Turma 8 - Grupo 5

Fernando Barros - 201910223
Matilde Silva - 202007928

1. Introdução

No âmbito da Unidade Curricular de Redes de Computadores, foi proposto o desenvolvimento de um projeto em duas partes. A primeira centra-se no desenvolvimento de uma aplicação para download de ficheiros via FTP, e a segunda centra-se numa série de experiências em diferentes fases de configuração da rede.

No que diz respeito à primeira parte do projeto, o aplicativo é desenvolvido em C e permite que os arquivos sejam transferidos da forma esperada sem descuidar da robustez necessária contra ocorrências de erros e entradas inválidas.

Quanto à segunda parte, a experiência decorreu com sucesso, os objetivos foram alcançados e as principais questões associadas à mesma foram resolvidas.

2. Download Application

A primeira parte deste trabalho consistiu no desenvolvimento de uma aplicação de download (transferência) de ficheiros de acordo com o protocolo FTP (file transfer protocol) descrito no RFC959 e com ligações TCP (Transmission Control Protocol) a partir de sockets, sendo assim a aplicação é capaz de fazer o download de qualquer tipo de ficheiros de um servidor FTP.

A aplicação aceita como argumento um link com a seguinte configuração: **ftp://[<user>:<password>@]<host>/<url-path>**. Este link está de acordo com a sintaxe url descrita no RFC1738. Nos casos em que não é fornecido *user* e *password*, os valores dos mesmos são, respetivamente, 'anonymous' e 'qualquer-password'.

Foi necessário estudar com cuidado o RFC959 "Internet message protocol" para compreender a parte do FTP e o RFC1738 "Uniform Resource Locators (URL)" para compreender o tratamento de informação provenientes de URLs.

2.1 Arquitectura

Em primeiro lugar é necessário fazer o parsing (processamento) do url que é passado através da consola quando se corre a aplicação. Para esta funcionalidade temos uma função chamada **getDetails** que guarda os valores resultantes do parsing (username, password, nome do host, path do ficheiro) nos argumentos passados como argumentos da função.

Em seguida utiliza-se uma função chamada **getIP** (fornecida) para a qual passamos o nome do host e que retorna o endereço IP correspondente. Com este endereço IP utilizamos a função **createConnection** para conectar com o socket, sendo utilizada sempre a porta 21 para iniciar a conexão.

Dentro desta função os seguintes passos são tomados: construção dos comandos *user*, *pass* e *retr ;* criação da conexão à socket, usando a função auxiliar **createSocket**; execução de uma *state machine* que mediante o status code recebido no ínicio da linha toma uma determinada ação (entre as quais, enviar o comando *user, pass, retrv ou pasv* e iniciar, continuar ou terminar o download do ficheiro).

Durante a execução são utilizadas algumas funções auxiliares para facilitar a interpretação das respostas recebidas no socket, tal como **getLastLineStatusCode**, que recebe o input do socket e, se

este tiver múltiplas linhas, retorna o status code da última linha recebida; **getPortNumber** que recebe a resposta do comando *pasv* como argumento e calcula a porta da qual o ficheiro será lido; **getFilename** que recebe como argumento *<url-path>* e extrai apenas o nome do ficheiro.

```
> int getLastLineStatusCode(char *buf){…

> int getFilename(char *buf, char* filename){…

> int getPortNumber(char* buf){…

> int createSocket(char* SERVER_ADDR, int SERVER_PORT){…

> int createConnection(char* SERVER_ADDR, int SERVER_PORT, char* user, char* passwd, char* path) {…
```

3. Configuration and Study Of a Network

3.1 Exp 1 - Configure an IP network

➔ *What are the ARP packets and what are they used for?*

Todos os hosts numa uma rede são localizados pelos seus endereços IP, mas os NICs não possuem endereços IP, estes possuem endereços MAC. ARP (*Address Resolution Protocol*) é o protocolo usado para associar o endereço IP a um endereço MAC.

Numa rede local (LAN), antes de enviar um pacote para outro host, um host primeiro transmite (envia) um pacote ARP.

➔ *What are the MAC and IP addresses of ARP packets and why?*

MAC address é um identificador numérico atribuído a uma interface de rede (NIC), dispositivo Wi-Fi ou Bluetooth, usado como endereço físico numa rede local. Esse número é único, exclusivo e composto por doze dígitos, formado por seis pares hexadecimais separados.

IP address é um endereço exclusivo que identifica um dispositivo na Internet ou numa rede local. IP vem do inglês "Internet Protocol" (protocolo de rede) que consiste num conjunto de regras que regem o formato de dados enviados pela Internet ou por uma rede local.

Uma máquina não consegue enviar diretamente dados para outra usando apenas o endereço lógico (endereço IP). Na prática, o endereço lógico de destino serve para descobrir o endereço físico (endereço MAC, associado à placa de rede) e assim a camada 2 do modelo OSI consegue entregar Data Frames à máquina de destino.

Cada pacote ARP contém campos para os endereços MAC e IP da máquina que envia e da máquina que recebe.

➔ *What packets does the ping command generate?*

Ping é uma utilidade da linha de comandos, disponível em praticamente qualquer sistema operacional com conectividade de rede, que atua como um teste para verificar se um dispositivo em rede pode ser acessado. O comando ping envia uma solicitação pela rede para um dispositivo específico.

Este comando vai gerar primeiro *ARP packets* para conhecer qual o endereço MAC do destinatário, posteriormente gera *ICMP packets* (*Internet Control Message Protocol*).

➔ *What are the MAC and IP addresses of the ping packets?*

Pacote request (de tux3 para tux4):
– IP address da source: 172.16.20.1 (tux3)
– MAC address da source: 00:0d:de:a6:a4:f1(tux3)
– IP address do destinatário: 172.16.20.254 (tux4)
– MAC address do destinatário: 00:21:5a:5a:7d:12 (tux4)

Pacote reply (de tux4 para tux3):
– IP address da source: 172.16.20.254 (tux4)
– MAC address da source: 00:21:5a:5a:7d:12 (tux4)
– IP address do destinatário: 172.16.20.1 (tux3)
– MAC address do destinatário: 00:0d:de:a6:a4:f1 (tux3)

➔ *How to determine if a receiving Ethernet frame is ARP, IP, ICMP?*

As Frame Ethernet têm 2 bytes dedicados ao campo *Type* que indica qual o protocolo da Frame e determina qual o processo deve ser entregue a Frame. Especificamente, se o protocolo for ARP o valor de *Type* é 0x0806, se o protocolo for IP o valor será 0x0800.

No caso da Frame ser do tipo IP, podemos analisar o seu *header* para determinar se o tipo é ou não ICMP.

➔ *How to determine the length of a receiving frame?*

Frames Ethernet não têm nenhum campo de *Length* ou semelhante, portanto, a única forma de saber o tamanho da frame era acedendo ao campo *Data* e contar o número de bytes até atingir o campo *Pad*.

➔ *What is the loopback interface and why is it important?*

Uma *loopback interface* é uma interface virtual que está sempre a correr e alcançável, desde que uma das interfaces do switch esteja operacional. Como resultado, a *loopback interface* é útil para tarefas de debugging, já que os seus endereços IP podem ser sempre *pinged*.

Por definição, todos os routers têm uma *loopback interface* interna com os endereços IP 127.XX.YY.ZZ reservados para este propósito.

3.2 Exp 2 - Implement two bridges in a switch

➔ *How to configure bridgeY0?*

/interface bridge add name=bridgeY0
/interface bridge port add bridge=bridgeY0 interface=ether3

➔ *How many broadcast domains are there? How can you conclude it from the logs?*

Neste caso existem 2 *broadcast domains*. Um deles constituído pelo tuxY3.eth0 e tuxY4.eth0, o outro por tuxY2.eth0. Confirmamos tal a partir dos logs, visto que o *ping broadcast* feito a partir do tuxY3.eth0 apenas atinge tuxY4.eth0, por sua vez, o *ping broadcast* feito a partir de tuxY2.eth0 não recebeu resposta nenhuma.

3.3 Exp 3 - Configure a Router in Linux

➔ *What routes are there in the tuxes? What are their meaning?*

Cada tux tem uma rota gerada automaticamente para a rede em que o mesmo se encontra. A gateway, neste caso, é sempre 0.0.0.0.

Além da rota pré-definida, foram adicionadas as seguintes rotas:

No tuxY3: **route add -net 172.16.21.0/24 gw 172.16.20.254** (sempre que o tuxY3 quiser comunicar com a rede 21 vai utilizar como gateway o tuxY4.eth0, que tem endereço IP 172.16.20.254)

No tuxY2: **route add -net 172.16.20.0/24 gw 172.16.21.253** (sempre que o tuxY2 quiser comunicar com a rede 20 vai utilizar como gateway o tuxY4.eth1, que tem endereço IP 172.16.21.253)

➔ *What information does an entry of the forwarding table contain?*

Na tabela de forwarding cada entrada possui as seguintes entradas: *Destination, Gateway, Genmask, Flags, Metric, Ref, Use, Iface.*

Destination e o IP do computador/rede de destino.

Gateway é o IP do computador para o qual vai ser mandada a mensagem e este é que vai dar routing da mensagem para o destino.

Genmask é a netmask utilizada para a rede de destino.

Flags representam informações sobre a rota.

Metric é a distância ao destino, normalmente contada em *hops*.

Ref é o número de referências para esta rota.

Use representa o contador de pesquisas pela rota.

Iface é a placa de rede usada para enviar a mensagem (eth0, eth1, etc.).

➔ *What ARP messages, and associated MAC addresses, are observed and why?*

Uma troca de mensagens ARP ocorre sempre que uma mensagem é enviada de uma máquina para outra sendo que os endereços MAC não são conhecidos.

Por exemplo, ocorre uma troca de mensagens ARP quando um tux (tux1) envia um ping a outro tux (tux2), e o tux1 não conhece o MAC address de tux2. O tux1 envia uma mensagem ARP, na qual pede o MAC address de tux2, através do seu IP. Quando uma mensagem ARP é enviada, o tux1 associa o seu MAC address à mensagem, para que o receptor esperado da mensagem, tux2, saiba a que tux responder.

Esta mensagem será enviada no modo *broadcast* (MAC address do receptor tem o valor 00:00:00:00:00:00), porque o MAC address de tux2 ainda é desconhecido. Quando recebe este *broadcast*, tux2 responde com uma mensagem ARP, na qual fornece o seu MAC address. Esta mensagem é enviada apenas para o MAC address de tux1.

➔ *What ICMP packets are observed and why?*

Os pacotes ICMP que conseguimos observar são do tipo request e reply, uma vez que, todos os tuxes reconhecem a presença uns dos outros, estando todas as routes adicionadas. Se os tuxes não se reconhecessem, os pacotes ICMP enviados seriam do tipo Host Unreachable.

➔ *What are the IP and MAC addresses associated to ICMP packets and why?*

Os endereços MAC e IP das máquinas/interfaces que recebem/enviam os pacotes são os endereços IP e MAC de origem e destino associados com os pacotes ICMP. Por exemplo, quando um pacote ICMP é enviado do tuxy1 para a interface do tuxy4 conectada à mesma sub rede (no nosso caso foi utilizada a interface eth0), os endereços MAC e IP de origem serão os do tuxy1 e os endereços IP e MAC de destino serão os associados à interface eth0 do tuxy4.

3.4 Exp 4 - Configure a Commercial Router and Implement NAT

➔ *How to configure a static route in a commercial router?*

Para configurar uma rota estática, é necessário aceder à consola do router, no caso deste trabalho, através do GTKterm, baudrate 115200. Para tal finalidade é necessário ligar através do cabo de série S0 de um TUX da bancada à entrada de configuração do router. Para configurar as rotas temos que executar o seguinte comando:

/ip route add dst-address=<dst-address>/<mask> gateway=<gateway>

➔ *What are the paths followed by the packets in the experiments carried out and why?*

No caso de a rota existir, os pacotes seguem essa mesma rota. Caso contrário, os pacotes são dirigidos pela rota default, neste caso, para **Rc**.

Quando desativados os redirecionamentos, no passo 4, caso haja redirects na mesma interface de rede, o tux não guarda na sua lista de forwarding uma entrada resultante do redirect de um outro tux.

No tuxY2 foram desativados os redirects, além disso, foi definido tanto para o tuxY4 como para o tuxY2 como *default route* o router **Rc**. O tuxY4 é o único que tem comunicação com o tux Y3 através da a rede Y0. O tuxY4, o tuxY2 e o router estão todos ligados à rede Y1.

Quando apagamos a route de tuxY2 para a a rede Y0, através de tuxY4, tuxY2 não conehce nenhuma rota para chegar a tuxY3, que se encontra na a rede Y0. Como tal, tuxY2 encaminha os pacotes para o router Rc, já que este é a default route de tuxY2. Rc, por sua vez, tem uma rota

definida para a rede Y0, com o tuxY4 como gateway, um pacote de tuxY2 consegue ainda chegar a tuxY3, mas com mais hops. O comando traceroute confirma esta rota.

Quando ativa a conexão de tuxY4 para a rede Y0, traceroute de tuxY2 para tuxY3 mostra que os pacotes com origem em tuxY2, primeiro passam por tuxY4 e depois, como tuxY4 contém uma gateway para a rede Y0, os pacotes são reencaminados de tuxY4 para o destino final, tuxY3. Como existe uma rota especificada para a rede Y0, tuxY2 não envia os pacotes para esta rede através de Rc.

➔ *How to configure NAT in a commercial router?*

De forma a configurar o router, foi configurada a interface interna do processo de NAT. A partir do GTKTerm, os seguintes comandos são inseridos:

- Para adicionar RC à bridgeY1:

  / interface bridge port remove [find interface = ether24]
  / interface bridge port add bridge = bridge21 interface = ether24

- Para configurar o ip address do RC e colocar o NAT ligado em default:

  / ip address add address =172.16.2. Y9 /24 interface = ether1
  / ip address add address =172.16. Y1 .254/24 interface = ether2
  / ip firewall nat add chain = srcnat action = masquerade out - interface = ether1

Com este último comando o NAT no router está ativo. Tal implica que é possível que dispositivos com endereços IP privados acessem a Internet através de um único endereço IP público compartilhado, economizando endereços IP públicos e aumentando a segurança da rede.

➔ *What does NAT do?*

O NAT (Network Address Translation) é um protocolo que tem como função a associação e transformação de um IP Address noutro IP Address, de forma a "mascarar" o remetente/destinatário de pacotes enviados, podendo ter vários fins como assegurar a privacidade e segurança de máquinas numa subrede privada local que estão a comunicar com máquinas "externas" (é implementado mais frequentemente em ambientes de acesso remoto), conservando os seus IPs, mas também pode permitir que essa comunicação seja possível permite que redes IP privadas que usem endereços não registados se conectem e comuniquem com a Internet ou redes públicas. As máquinas dessa rede, para as máquinas exteriores, são reconhecidas através de um IP único, que representa todos os dispositivos da mesma.

3.5 Exp 5 - DNS

➔ *How to configure the DNS service at a host?*

O serviço DNS é configurado no ficheiro conf, localizado no diretório /etc/ do anfitrião tux em questão. A configuração é feita através de dois comandos, um que representa o nome do servidor DNS, e um com o respetivo endereço IP:

– search services.netlab.fe.up.pt
– nameserver 172.16.1.1

➔ *What packets are exchanged by DNS and what information is transported?*

O host envia para o server um pacote com o hostname, esperando que seja retornado o seu endereço IP. O servidor responde com um pacote que contém o endereço IP do hostname em causa.

3.6 Exp 6 - TCP connections

➔ *How many TCP connections are opened by your ftp application?*

São abertas duas conexões FTP. A primeira, para comunicar com o servidor na porta pré-definida, 21. A segunda, após receber a resposta ao comando *pasv* para determinar qual a porta que usar para abrir a conexão, da qual se vai ler o ficheiro de transferência.

➔ *In what connection is transported the FTP control information?*

Na primeira conexão efetuada com o servidor, pela porta 21.

➔ *What are the phases of a TCP connection?*

TCP fornece um serviço *connection-oriented*. Tal implica que exista uma conexão virtual entre os dois *endpoints*. Existem, portanto, três fases em qualquer conexão virtual: estabelecimento de conexão, transferência de dados e término de conexão.

➔ *How does the ARQ TCP mechanism work? What are the relevant TCP fields? What relevant information can be observed in the logs?*

O mecanismo TCP ARQ funciona de acordo com um método de janela deslizante que consiste na verificação de erros de transmissão de dados. Para este efeito, são utilizados acknowledgement numbers, que indicam o recebimento correto da trama, um tamanho de janela, que indica o tamanho do pacote recebido, e sequence number, que é o número de pacotes enviados.

➔ *How does the TCP congestion control mechanism work? What are the relevant fields. How did the throughput of the data connection evolve along the time? Is it according the TCP congestion control mechanism?*

O mecanismo de controlo de congestão do TCP tem como base os ACKs recebidos na transmissão de pacotes. Estes são o source clock da transmissão. É utilizada uma nova variável/valor por conexão, denominada Congestion Window, de modo a regular o tamanho da janela deslizante de transmissão de pacotes tendo em conta a congestão da conexão. Este valor é regulado, incrementando se a congestão da rede diminui e decrementado se a congestão da rede aumenta. Isto é normalmente feito incrementando Congestion Window por 1, a cada RTT (round trip time). Quando se detecta que um pacote é perdido, através de um timeout ou quando se recebe 3 ACKs duplicados, o valor de Congestion Window passa para metade. O bitrate da conexão será aproximadamente igual a Congestion Window/RTT. No início da conexão, pode também haver uma fase de slow start, que serve para, de modo a delimitar um threshold que é depois utilizado numa fase posterior de congestion avoidance. Foi registado que, quando o primeiro download, no tuxy3, começou, a taxa de transmissão aumentou rapidamente. Após o início do segundo download, no tuxy2, a taxa de transmissão no tuxy3 diminuiu rapidamente e a do tuxy2 aumentou também rapidamente, e passado alguns "altos e baixos", o throughput estabilizou relativamente num níıvel mais baixo do que era anteriormente, antes do segundo download ter começado. Estas mudanças fazem sentido e estão de acordo com o mecanismo de controlo de congestão do TCP, uma vez que quando apenas um download está a ser feito, o bitrate para a conexão do tuxy3 é mais alto do que quando os dois downloads estavam a ser feitos ao mesmo tempo, resultado do congestionamento da rede.

➔ *Is the throughput of a TCP data connections disturbed by the appearance of a second TCP connection? How?*

Sim, é. A taxa de transmissão de pacotes da conexão TCP que já estava iniciada diminui, uma vez que à outra conexão foi atribuída uma capacidade para a transmissão de pacotes na mesma, de modo que a taxa de transmissão de cada conexão é distribuída uniformemente.

4. Conclusão

O segundo trabalho de RCOM visa desenvolver uma aplicação que permita o download de ficheiros através de conexões utilizando os protocolos FTP e TCP, bem como a configuração de uma rede IP, para conhecer o real funcionamento de diversas máquinas e dispositivos, como o switch e o router, e também de modo a perceber diversas técnicas, (NAT, DNS, etc), protocolos (ICMP, etc) e estruturas de dados (forwarding tables, tabelas ARP, etc) utilizadas na comunicação entre essas máquinas.

Acreditamos que todos os objetivos deste trabalho foram alcançados e que o nosso conhecimento sobre os temas abordados no mesmo cresceu e foi aprofundado.