U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Redes de computadores

2º trabalho prático

Licenciatura em Engenharia Informática e Computação

Diogo Fernandes (202108752)
José Sereno (202108729)

Dezembro 2023

Contents

1 Introdução

Objetivos Este projeto teve dois objetivos:

1. Desenvolver uma aplicação em C que implemente o protocolo FTP - descrito no RFC959 - para fazer o download de um ficheiro através de um URL - a sintaxe deste URL deveria seguir o RFC1738.

2. Configurar e analisar o funcionamento de uma rede de computadores.

2 Desenvolvimento da aplicação

O programa download foi desenvolvido em C e tem como objetivo fazer o download de um ficheiro através de um URL que segue a sintaxe do RFC1738. Através da realização deste programa aprofundamos os nossos conhecimentos sobre os seguintes temas:

Client-Server & TCP/IP Aprendemos como se dá a comunicação entre um cliente e um servidor através do protocolo TCP/IP.

RFCs RFCs são documentos que descrevem os padrões da internet. Consultámos, como referido anteriormente, o RFC959 e o RFC1738, para além de outras documentações, para entender o funcionamento do protocolo FTP e da sintaxe do URL.

Sockets Aprendemos a usar sockets em C como forma de comunicação entre o cliente e o servidor.

DNS (Domain Name System) Procurámos entender o funcionamento do DNS e como é que este é usado para traduzir um URL num endereço IP.

UNIX Aprendemos alguns comandos de UNIX que usamos no nosso programa (ex.: getaddrinfo, socket, connect, recv, send)

2.1 Estrutura do código

O código do programa download está dividido em 3 ficheiros: download.c, server.c e url.h. É no ficheiro download.c onde se encontra a função main - onde se percebe o fluxo do programa. O ficheiro server.c contém as funções que implementam o protocolo FTP e o ficheiro url.h contém as funções que permitem fazer o parse do URL. O programa download é compilado através do comando make e é executado da seguinte forma:

2.2 Fluxo do programa

O programa é executado do seguinte comando:

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

O fluxo do programa será o seguinte:

1. Parse do URL para obter os seguintes campos: user (opcional), password (opcional), host, port (opcional, é usado o valor 21 por default no protocolo FTP) e url-path.
2. Criação de uma socket que, através de uma ligação TCP/IP, inicia uma conexão com o host na porta port.
3. Envio dos comandos USER e PASS para o servidor juntos dos valores user e password, respetivamente. Caso estes valores não tenham sido especificados, usa-se o valor anonymous.
4. Envio do comando PASV para o servidor para que este abra uma porta à qual nos conectaremos para futuramente recebermos o ficheiro.
5. Criação de uma nova socket que, através de uma ligação TCP/IP, inicia uma conexão com a nova porta aberta pelo servidor.
6. Envio do comando RETR através da porta inicial para o servidor junto do valor url-path para que o servidor nos envie o recurso especificado.
7. Leitura do ficheiro através da socket criada no ponto 5 e escrita do mesmo para um ficheiro local.
8. Fecho das sockets criadas.

O parse do URL é feito a partir da função parse_url que recebe uma string e devolve uma struct URL com os campos especificados no ponto 1. Esta função usa uma máquina de estados e expressões regulares.
Todo o processo de comunicação entre o cliente e o servidor é feito através de sockets e o protocolo usado é o TCP/IP. A criação das sockets é feita através da função getaddrinfo que recebe o hostname e a porta e devolve uma struct addrinfo com os campos necessários para a criação da socket.
As sockets são criadas através da função socket e são feitas as respetivas ligações através da função connect.
O envio de comandos pelas sockets é feito através da função send e a leitura das respostas do servidor é feita através da função recv.
Antes de ser enviado qualquer comando para o servidor, primeiro é feita a leitura do código de status do servidor através da função recv, de modo a termos uma noção do estado do servidor.
A escrita do ficheiro no disco é feita através da função write.

3 Configurar e analisar o funcionamento de uma rede

O objetivo deste conjunto de experiências é configurar uma rede de computadores de modo a que estes tenham acesso à internet para instalar ficheiros a partir de um servidor remoto usando o protocolo FTP desenvolvido ou seja, a aplicação.

3.1 Experiência 1

3.1.1 Arquitetura da rede

No fim desta experiência, a configuração da rede deverá consistir em 2 computadores (TUX63 e

TUX64) conectados pelo Switch.

3.1.2 Objetivo

O propósito desta experiência foi a configuração de dois computadores na mesma rede de modo a permiti-los comunicar.

3.1.3 Quais são os comandos necessários para esta experiência?

```
# TUX63:
ifconfig eth0 up
ifconfig eth0 172.16.60.1/24

# TUX64:
ifconfig eth0 up
ifconfig eth0 172.16.60.254/24
```

Estando conectados ao mesmo Switch, os computadores comunicam entre sí usando a bridge default. Podemos testar a comunicação entre os dois computadores usando o comando ping passando o endereço IP do outro computador como argumento:

```
ping 172.16.60.254 # No TUX63
```

3.1.4 O que são pacotes ARP e para que são usados?

ARP (Address Resolution Protocol) é um protocolo que traduz endereços IPv4 em endereços MAC numa LAN (Local Area Network). Esta tradução é importante porque apesar de os IPs serem usados para identificar os computadores numa rede, podem mudar ao longo do tempo e devido ao ambiente. Já os MACs são usados para identificar o hardware de um computador e são únicos e imutáveis.
Numa situação em que um computador 1 quer enviar uma mensagem para um computador 2, este começa por verificar se o endereço IP do computador 2 está na sua cache de vizinhos de rede. Caso contrário, o computador 1 terá de fazer uma tradução do endereço IP do computador 2 para o seu endereço MAC. Para fazer esta tradução, o ARP fará um broadcast - envia para todos os computadores da rede - de um pedido ARP que apenas será respondido pelo computador 2. O computador 2 responde ao pedido ARP com o seu endereço MAC e o computador 1 guarda-o na sua cache de vizinhos de rede.
O comando de UNIX: arp; é usado para manipular ou exibir a cache de vizinhos de rede IPv4 do kernel.

3.1.5 O que são os os endereços MAC e IP dos pacotes ARP e porquê?

O MAC Media Access Control é um endereço físico que identifica um dispositivo numa rede. É usado na data link layer para assegurar o endereço físico do computador - isto significa que está relacionado com o hardware. Os endereços MAC são únicos e não podem ser alterados. É composto por 6 bytes (48 bits) e é representado em hexadecimal. O IP Internet Protocol é um endereço lógico que identifica uma conexão entre um computador e uma rede. Pode mudar ao longo do tempo e devido ao ambiente. É usado na network layer para assegurar o endereço lógico do computador - isto significa que está relacionado com o software. Os endereços IP podem ser facilmente encontrados por terceiros, pois são transmitidos pela internet. É composto por 4 bytes (32 bits) e é representado em decimal.

3.1.6 Que pacotes é que o comando ping gera?

O comando ping gera pacotes ARP e pacotes ICMP. ICMP (Internet Control Message Protocol) é um protocolo da camada de rede que reporta erros e fornece outras informações relevantes para o processamento de pacotes IP. Neste contexto, o ICMP é usado pelo comando ping para testar uma conexão de rede IP.

3.1.7 Quais são os MAC e IP dos pacotes ping?

Ver o ponto 3.1.11.

3.1.8 Como determinar se um pacote é ARP, IP ou ICMP?

É possível determinar se um frame Ethernet recebido é ARP, IP, ICMP verificando a captura do WireShark, na coluna Portocol. O WireShark faz esta distinção através do campo Type do cabeçalho Ethernet. O valor 0x0800 indica que o pacote é IP ou ICMP já que este se encontra guardado no pacote do IPv4, o valor 0x0806 indica que o pacote é ARP.

3.1.9 Como determinar o tamanho de um pacote recebido?

O tamanho de um pacote recebido pode ser determinado através da captura do WireShark, na coluna Length. Além disso, o tamanho dos pacotes IPv4 pode ser determinado a partir de 2 bytes que

se encontram no pacote. Quanto aos pacotes ARP, estes possuem um tamanho fixo de 28 bytes.

3.1.10 O que é a interface loopback e porque é que é importante?

A interface loopback é uma interface de rede virtual que permite que um computador comunique consigo mesmo. É importante pois permite testar a stack de protocolos TCP/IP sem a necessidade de uma rede física. Essencialmente, a interface loopback, muitas vezes identificada pelo endereço IP 127.0.0.1, cria um ambiente isolado no próprio dispositivo, onde os dados enviados são retornados para a sí.

3.1.11 Análise dos logs

Inicialmente, o TUX63 não sabe o endereço MAC do TUX64 e vice-versa. Assim, o TUX63 envia um pacote ARP para o broadcast (MAC FF:FF:FF:FF:FF:FF) com o endereço IP do TUX64. O TUX64 recebe o pacote ARP vindo do broadcast, que contém o IP do TUX63, e responde ao TUX63 de modo a indicar que guardou o seu endereço IP no pedido ARP, enviando o seu endereço MAC. O TUX63 ao receber este pacote, guarda o endereço MAC do TUX64 na sua tabela ARP. Permitindo assim obter um PING request e PING response. Os endereços IP e Mac obtidos são os seguintes:

PC	IP	MAC
TUX63	172.16.60.1	00:21:5a:5a:75:bb
TUX64	172.16.60.254	00:21:5a:61:2d:df

3.2 Experiência 2

3.2.1 Arquitetura da rede

No final da experiência, a configuração da rede deverá consistir em 2 computadores (TUX63 e TUX64) conectados à bridge (bridge0) e um computador (TUX62) conectado à bridge (bridge1).

3.2.2 Objetivo

Esta experiência teve como objetivo ensinar-nos a configurar 2 domínios de rede diferentes no mesmo switch e verificar que por defeito não ocorre comunicação entre estes domínios, sendo necessária a sua configuração.

3.2.3 Quais são os comandos necessários para esta experiência?

```
# Continuando a partir da experiencia
    anterior:

# No tux2
ifconfig eth0 up
ifconfig eth0 172.16.61.1/24

# Switch
/system reset-configuration

/interface bridge add name=bridge60
/interface bridge add name=bridge61

# No tux3, para a sua porta eth0 X, no
    tux4 para a sua porta eth0 Y e no
    tux2 para a sua porta eth0 Z:
/interface bridge port remove [find
    interface=etherX]
/interface bridge port remove [find
    interface=etherY]
/interface bridge port remove [find
    interface=etherZ]
/interface bridge port add bridge=
    bridge60 interface=etherX
/interface bridge port add bridge=
    bridge60 interface=etherY
/interface bridge port add bridge=
    bridge61 interface=etherZ
```

3.2.4 Quantos domínios de broadcast existem? O que podemos concluir a partir dos logs? / Análise dos logs

Como configuramos 2 bridges, podemos concluir que existem 2 domínios de broadcast. Isto porque cada bridge é um domínio de broadcast. Podemos concluir isto a partir dos logs pois o TUX3 obteve uma resposta do TUX4, mas não do TUX2. Isto significa que o TUX3 está no mesmo domínio de broadcast que o TUX4, mas não no mesmo domínio de broadcast que o TUX2. Além disso, é possível concluir que passaram a existir 2 sub-redes, uma para cada bridge. Isto porque, apesar de ser possível fazer ping do TUX3 para o TUX4, não é possível fazer ping do TUX3 para o TUX2 já que, nos logs, não existem pacotes ICMP.

3.3 Experiência 3

3.3.1 Arquitetura da rede

No final desta experiência, é esperado que tenhamos uma arquitetura semelhante à da experiência anterior, com uma nova da conexão TUX64 à bridge (bridge61) que mediará a comunicação entre as duas bridges.

3.3.2 Objetivo

O objetivo desta experiência foi ensinar-nos a transformar o TUX64 num router e a configurar

o mesmo para que este possa comunicar com os restantes computadores e permitir que estes comuniquem entre sí.

3.3.3 Quais são os comandos necessários para esta experiência?

```
# Continuando a partir da
    experiencia anterior:

# Switch - seja agora a conexao do
    eth1 do tux4 a bridge61 pela
    porta W

/interface bridge port remove [
    find interface=etherW]
/interface bridge port add bridge=
    bridge61 interface=etherW

# No tux4
ifconfig eth1 up
ifconfig eth1 172.16.61.253/24

sysctlnet.ipv4.ip_forward=1
sysctlnet.ipv4.
    icmp_echo_ignore_broadcasts=0

# No tux3
route add -net 172.16.61.0/24 gw
    172.16.60.254

# No tux2
route add -net 172.16.60.0/24 gw
    172.16.61.253
```

Nota: Devido a não conseguirmos obter os logs na nossa bancada, foi necessário utilizar outros ips, mas o procedimento é o mesmo.

3.3.4 Que rotas existem nos computadores? Qual o seu significado?

Existem 2 rotas, no TUX62 e no TUX63. Como a rota 64 é um gateway de ambos, a rota 62 é para chegar ao 63 e a rota 63 é para chegar ao 62, passando pelo 64.

3.3.5 Que informação contém uma entrada da tabela de encaminhamento?

Uma entrada da tabela de encaminhamento contém o endereço de destino/origem, o endereço de gateway e a máscara de rede.

3.3.6 Que mensagens ARP, e os endereços MAC, são observadas e porquê? / Análise dos logs

No caso do ping do TUX63 para o TUX62. As mensagens ARP trocadas contêm apenas os endereços MAC do TUX63 e do TUX64 e não do destino final (TUX62). Isto ocorre devido à existência da rota. O TUX63 não conhece o endereço do TUX62, apenas conhece o endereço do gateway (TUX64) que leva ao TUX62.

Quando se apaga as tabelas ARP no TUX64 e se corre o mesmo ping novamente, os 3 computadores não se conhecem, pois não sabem os endereços MAC uns dos outros. Ao realizar o ping, é lançado um pedido ARP para o broadcast e para a sub-net da bridge60 a pedir o endereço MAC do TUX64, default gateway do TUX62. É gerada a resposta ARP e esta é enviada de volta para o TUX63, que guarda o endereço MAC do TUX64 na sua tabela ARP e vice-versa. De seguida, o ping passa pelo TUX64 e alcança o TUX62, sendo realizado o mesmo processo de troca de mensagens ARP entre o TUX64 e o TUX62.

É importante notar que o TUX63 não tem informação sobre o endereço MAC do TUX62 e vice-versa. Cada um destes computadores apenas conhece o endereço MAC do seu gateway, que é o TUX64.

3.4 Experiência 4

3.4.1 Configuração de um Router Comercial e Implementação de NAT

Nesta experiência foi-nos pedido que configurássemos um router comercial na nossa bridge (bridge61).

3.4.2 Quais são os comandos necessários para esta experiência?

```
# Router Serial Console
/interface bridge port remove [find
    interface=ether5]
/interface bridge port add bridge=
    bridge61 interface=ether5
/ip address add address=172.16.2.69/24
     interface=ether1
/ip address add address
    =172.16.61.254/24 interface=ether2
/ip route add dst-address
    =172.16.60.0/24 gateway
    =172.16.61.253
/ip route add dst-address=0.0.0.0/0
    gateway=172.16.2.254

# Terminais dos TUXs
route add default gw 172.16.61.254 #
    No tux62
route add default gw 172.16.60.254 #
    No tux63
route add default gw 172.16.61.254 #
    No tux64
sysctl net.ipv4.conf.eth0.
    accept_redirects=0
```

```
sysctl net.ipv4.conf.all.
    accept_redirects=0
route del -net 172.16.60.0 gw
    172.16.61.253 netmask
    255.255.255.0
traceroute -n
route add -net 172.16.60.0/24 gw
    172.16.61.253
sysctl net.ipv4.conf.eth0.
    accept_redirects=0
sysctl net.ipv4.conf.all.
    accept_redirects=0
/ip firewall nat disable 0
/ip firewall nat enable 0
```

3.4.3 Como configurar uma rota estática num router comercial?

Para configurarmos uma rota estática no router comercial começamos por ligar o router ao TUX63 através de um cabo de série. De seguida, acedemos à consola de comandos do router através do GTKTerm de modo a podermos configuar o router. Adicioná-mo-lo à bridge pretendida (bridge61) e atribuimos-lhe um IP na LAN. Temos também de lhe atribuir um IP para a rede exterior. Definimos também as rotas necessárias para chegar às redes da bridge (bridge60) e para chegar à internet, indicando os gateways necessários.

3.5 Experiência 5

3.5.1 Arquitetura da rede

A arquitetura desta experiência é a mesma da experiência anterior.

3.5.2 Objetivo

O objetivo desta experiência foi ensinar-nos a dar ping a hosts com a utilização do DNS.

3.5.3 Quais são os comandos necessários para esta experiência?

```
# Continuando a experiencia anterior:

# No tux2
echo 'nameserver 172.16.2.1' > /etc/
    resolv.conf

# No tux3
echo 'nameserver 172.16.2.1' > /etc/
    resolv.conf
```

3.5.4 Como configurar o DNS num host?

No terminal do TUX2 e do TUX3, precisamos de correr o comando hlsudo nano /etc/resolv.conf e adicionar a seguinte linha:

```
nameserver <DNS IP address>
```

O DNS (Domain Name System) mapeia um nome de um host/dominio para endereços de IP. Portanto, ao utilizar este comando, estamos a permitir dar pings a hosts e domínios.

3.5.5 Que pacotes são trocados pelo DNS e que informação é transportada?

Os pacotes DNS trocados são o DNS query e o DNS response. O DNS query contém o nome de domínio e o DNS response contém o endereço IP do nome de domínio. Tornando possível traduzir o nome de domínio num endereço IP no router.

3.5.6 Análise dos logs

Nesta experiência, a partir do TUX62, damos ping ao google.com. Como google.com não é um IP, é necessário utilizar o DNS. Inicialmente, o DNS procura os nameservers que estão definidos no ficheiro /etc/resolv.conf. De seguida, o DNS envia um pacote DNS query para o nameserver. O nameserver responde com um pacote DNS response que contém o endereço IP do google.com, permitindo com que o ping seja bem sucedido.