**Trabalho 2**

Configuração de uma rede e desenvolvimento de uma aplicação de download

**Relatório Final**



Mestrado Integrado em Engenharia Informática Computação

Redes de Computadores

Turma 4:

André Rodrigues Barros – up201303567

Leonardo Ferreira – up2013

Luís Miguel Azevedo Duarte – up201303585

Faculdade de Engenharia da Universidade do Porto Rua Roberto Frias, 4200-465 Porto, Portugal

19 Dezembro 2015

**Índice**

**Resumo…………………………………………………………………...3**

**Introdução………………………………………………………………..4**

**Aplicação de download**

**Arquitetura………………………………………………………..5**

**Exemplo dos resultados da aplicação download……………….8**

**Configuração de uma rede**

**Experiência 1……………………………………………………….9**

**Experiência 2……………………………………………………..10**

**Experiência 3……………………………………………………..10**

**Experiência 4……………………………………………………..12**

**Experiência 5……………………………………………………..13**

**Experiência 6……………………………………………………..14**

**Resumo**

Este relatório serve como complemento do segundo projeto da unidade curricular de Redes de Computadores, do MIEIC, na FEUP. O objectivo deste projecto consiste no desenvolvimento de uma aplicação de download de ficheiros e na configuração de uma rede de computadores, logo este relatório encontrar-se-á divido nessas duas respetivas metades, acompanhadas das suas respectivas subsecções.

A secção que refere a configuração de rede, foi abordada em seis experiências distintas propostas ao grupo. Tais experiências consistiam numa configuração de um IP de rede, de um router em Linux, de um router comercial e do DNS, e na implementação de duas VLAN’s(virtual Local Area Network)e do NAT (Network Address Translation), e de seguida, testar a aplicação de download desenvolvida anteriormente para verificação de um bom funcionamento nas ligações TCP(Transmission Control Protocol), sendo que todos estes conceitos e protocolos irão ser explicados com mais detalhe ao longo do relatório.

**Introdução**

Este segundo projeto divide-se em duas componentes:

- Desenvolvimento de uma aplicação de download.

- Configuração de uma rede.

Relativamente à a aplicação de download, o objetivo passava por ganhar um conhecimento mais profundo sobre o que é um cliente, um servidor e as suas especificidades em TCP/IP, a caracterização de protocolos em aplicações, como definir um URL e descrever o comportamento de um servidor FTP. De seguida, o grupo teria de implementar um cliente FTP e ligação TCP utilizando sockets.

O grande objetivo da configurar uma rede é permitir a execução de uma aplicação, neste caso a aplicação de download desenvolvida, num switch partilhado por duas *VLAN’s*, estando o *NAT* ativo numa destas. No entanto, para o correto funcionameto da aplicação a *VLAN* quer se encontra sem *NAT* terá de possuir uma ligação à Internet.

**Aplicação Download**

- Arquitetura

- Exemplo de resultados do download com a respetiva explicação

Uma das componentes a avaliar no segundo projeto de Redes de Computadores era o desenvolvimento de uma aplicação, em C, que permitisse fazer downloads de um servidor. Para o desenvolvimento desta aplicação, o grupo estudou vários links e documentos, entre os quais o RFC959 que explica detalhadamente o protocolo de transferência de dados, ou FTP.

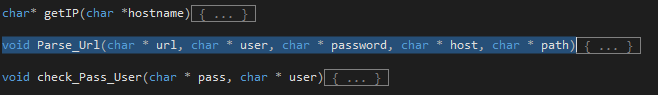
Iremos abordar detalhadamente todos os aspetos que envolvem esta aplicação assim como o método para a sua implementação.

**Arquitetura**

A aplicação download está dividida em 2 blocos: um dos blocos trata de extrair toda a informação necessária de um URL; outro contém todas as funções necessárias para o FTP assim como as responsarias pela ligação dos sockets aos servidores.

Estes dois blocos são conjugados no project.c permitindo ao utilizador descarregar um ficheiro de um servidor.

*Primeiro Bloco:*

 É composto pelos ficheiros URL.c e URL.h e permite á aplicação aceder á informação passada como parâmetro aquando na execução do download na linha de comandos.

**Figura 1**: Funções responsáveis pela extração da informação do URL assim como do IP do host.

*Funções:*

**getIP** -> permite á aplicação aceder ao ip do servidor. Este processo dá-se graças à função gethostbyname que retorna uma estrutura do tipo hostent, que é usada na função inet\_ntoa através de um cast para a estrutura do tipo in\_ddr e é devolvida um char\* no formato de pontos e números relativos ao IP. Recebe como parâmetro o host;

**Parse\_Url** -> Faz o parse de toda a informação exposta pelo link passado na linha de comandos. Guarda a informação do user, password, host e do path do ficheiro que pretende fazer download. Nesta função, também é considerado o facto de o link poder não ter paswords ou não ter user e password (neste caso o utilizador entraria em modo anonymous). Caso falte argumentos como o host ou path do file no link, então é lançada uma mensagem de erro na consola. Recebe como parâmetros o user, a password, o host, o path assim como o url;

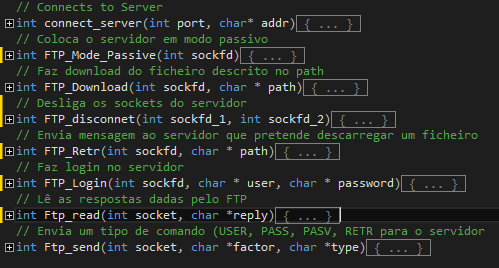
**Check\_Pass\_User** -> apenas faz print na consola para informar o utilizador se entrou em modo anonimo ou se não colocou a password no link passado na linha de comandos. Recebe como parâmetros o user e password do utilizador

*Segundo Bloco:*

Bloco que tem uma grande preponderância na aplicação do download e que é representado pelo clientFTP.c.

Após o parse do Url, é necessário ligar o cliente servidor FTP através de um socket TCP ao servidor FTP. Esta ligação é realizada pela função **connect\_server.**

De seguida, apresentaremos todas as funções necessárias assim como todos os passos a proceder para descarregar um ficheiro descrito no path inidicado pelo utilizador.



**Figura 2**: Funções que permitem aceder ao servidor e obter o ficheiro pretendido

*Funções:*

**connect\_server** -> responsável por estabelecer a ligação via sockets a um servidor. Recebe dois parâmetros: o endereço a que tem que se ligar assim como a porta (na primeira ligação que se estabelece com o servidor deve ser utilizada, por defeito, a porta 21 por se tratar de FTP). Se a ligação for bem-sucedida, é retornado o descritor da ligação que será necessária para os restantes processos;

**FTP\_Login** –> envia-se os comandos user e password ao servidor com a respetiva informação, através do Ftp\_Send, e recebe-se a resposta por parte do servidor, pela função Ftp\_read. Se após a transmissão do dados para autenticação, recebermos o código 530 (código que indica que indica que os dados de login estão errados) então o programa é terminado. Os parâmetros correspondem ao socket, ao user e á password;

**FTP\_Mode\_Passive** -> envia-se o comando PASV ao servidor. Uma vez feito este envio, recebemos uma resposta com o novo IP ao qual a aplicação download se deve associar para proceder á transferência do ficheiro assim como a nova porta a ligar. De seguida, repetimos o processo descrito na função connect\_server mas desta vez com o novo IP, com a nova porta associada a um novo socket (é calculada com base na resposta dada pelo servido após o envio do comando PASV). Recebe como parâmetro o socket;

**FTP\_Retr** -> envia-se o comando RETR ao servidor, através do primeiro primeiro socket com o path do ficheiro a descarrega. Toma como parâmetros o path do ficheiro e o socket;

**FTP\_Download** -> Cria-se um ficheiro com o mesmo nome que o ficheiro pretendido. De seguida, em cada iteração da leitura, é escrito para o novo ficheiro a informação contida no ficheiro original. Toma como parâmetros o path do ficheiro e o socket;

**FTP\_Disconnect** –> Após receber o o ficheiro corretamente (o utilizador recebe o código 226 que significa que recebeu o ficheiro corretamente), é enviado ao utlizador o comando QUIT que permite desconectar os sockets do servidor e fecha-se os sockets. Recebe os sockets envolvidos nas operações efetuadas pela aplicação.

**Exemplo dos resultados da aplicação download**

Testou-se a aplicação donwload com vários ficheiros de diferentes tipos e tamanhos e tudo correu como esperado.

Executou-se na consola -> ./proj <ftp://ftp:pass@ftp.up.pt/pub/robots.txt>

Output esperado na consola após execução da aplicação.

Info Aplication Download

User: ftp

Password: pass

Host: [ftp.up.pt](ftp://ftp.up.pt)

Path to file: pub/robots.txt

Start Connection…

IP Address: 193.136.37.8

Login…

Reply: 220 Bem-vindo à Universidade do Porto

Element sent: USER ftp

Reply: 331 Please specify the password.

Element sent: PASS pass

Reply: 230 Login successful.

Enter Pasv mode…

Element sent: PASV pasv

Reply: 227 Entering Passive Mode (193,136,37,8,180,67)

New IP Address: 193.136.37.8

Starts a new Connection…

Element sent: RETR pub/robots.txt

Reply: 10 Opening BINARY mode data connection for pb/robots.txt (23 bytes).

Filename described by path: robots.txt

New File received successfuly!!!

Reply: 226 File send OK.

Element sent: QUIT quit

Reply: 221 Goodbye

Disconnected from server

**Configuração de rede**

Segue-se agora a descrição das experiências realizadas que possibilitaram a configuração de uma rede. Os comandos utilizados como exemplo referem-se à estação de trabalho 2.

**Experiência 1**

Esta experiência consiste na configuração de um IP de Rede, ou seja, pretende-se que computadores diferentes sejam capazes de comunicar entre si através da configuração dos seus IP’s.

Inicialmente precede-se à configuração dos computadores através dos comandos:

* Ifconfig eth0 up
* Ifconfig eth0 172.16.20.1/24 (tux1) ou 172.16.20.254/24 (tux4)
* Route add default gw 172.16.20.254 (tux1)

Deste modo, as máquinas tux1 e tux4 são capazes de comunicar entre si através do switch sendo a gateway default o IP do tux4.

De seguida registou-se os IP’s e os endereços MAC dos dois computadores:

|  |  |  |
| --- | --- | --- |
|  | IP | MAC |
| tux1 | 172.16.20.1 | 00:0f:fe:8c:af:9d |
| tux4 | 172.16.20.254 | 00:22:64:a6:a4:f1 |

Posteriormente apagou-se as tabelas de ARP no tux1:

* arp –a
* arp –d 172.16.20.254

Finalmente, utilizou-se o comando ping de modo a analisar a ligação através do programa WireShark no qual foi possível verificar os requests e replys bem como os packets enviados e recebidos. (Registo log em anexo exp1.pcapng).

Da experiência realizada foi possível perceber como se configura uma rede IP, bem como verificar a importância de packets ARP, sendo que estes constituem um protocolo para encontrar um endereço da camada de ligações a partir do endereço da camada de rede, isto é, transforma endereços de IP em endereços MAC.

Na verdade, os endereços IP e MAC diferem sendo o primeiro o endereço do computador e o segundo o endereço físico do router.

Foi ainda possível perceber que é através do protocolo e do header do IP que se distingue uma trama de Ethernet de ARP, de uma IP e de uma ICMP.

Por fim, foi perceptível a importância de uma interface loopback pois esta constitui uma rede virtual que o computador utiliza para comunicar consigo mesmo. É utilizada principalmente para diagnostico e procura de erros e para conectar-se a servidores que se encontram a correr na máquina local.

**Experiência 2**

Esta segunda experiência tem como objectivo a implementação de duas VLAN’s no switch.

Inicialmente, procede-se à configuração do tux2 como exemplificado na experiência anterior.

|  |  |  |
| --- | --- | --- |
|  | IP | MAC |
| tux2 | 172.16.21.1 | 00:21:5ª:61:2b:72 |

De seguida procede-se à criação das VLAN’s 20 e 21:

* conf t
* vlan 20
* end
* conf t
* vlan 21
* end

Procede-se posteriormente à configuração das portas às respectivas VLAN’s (20 – tux1 e tux4, 21 – tux2):

* conf t
* interface fastethernet 0/Número da porta
* switchport mode access
* switchport access vlan 20/21
* end

Por fim procede-se à captura dos packets através do comando ping nos diferentes tux’s verificando-se que o tux1e tux4 são capazes de comunicar mas não conseguem comunicar com o tux2 que se encontra numa VLAN diferente (consultar os logs exp2\_log\_tux1.pcapng, exp2\_log2\_tux1.pcapng e exp2\_log3\_tux2.pcapng).

**Experiência 3**

Esta experiência teve como objectivo configurar a máquina tux4 como router entre as duas sub-redes criadas na experiência anterior. Para tal, foi necessário ligar a interface ethernet 1 do tux4 e configurá-la com um IP dentro da mesma gama que o tux2, posteriormente adicionando esta interface à sub-rede da máquina 2.

Após estes passos, foi adicionada uma rota ao tux1 com o comando:

* route add -net 172.16.y1.0/24 gw 172.16.y0.254

o primeiro endereço identifica a gama de endereços para a qual se quer adicionar a rota, sendo que o segundo é o IP para o qual se deve reencaminhar o pacote (tux4).

De seguida, repete-se o mesmo passo para a máquina 2, mas desta vez usando endereços distintos:

* route add -net 172.16.20.0/24 gw 172.16.21.253

sendo o segundo IP novamente o da máquina 4, mas na sub-rede comum ao tux2 e tux4.

Percorridos estes passos, foi possível efectuar *ping* da máquina 1 para a máquina 2, começando com um pedido para o IP do tux2, que é reencaminhado para o tux4, pois este encontra-se conectado a ambas as sub-redes, sendo, assim, possível aceder às duas máquinas (ao tux1 através da interface eth0 e ao tux2 através da eth1) e enviar os pacotes da primeira para a segunda máquina. O processo de resposta é exatamente o reverso, sendo o pacote reencaminhado da máquina 2, pela máquina 4 até à máquina 1. (consultar logs log\_exp3\_tux1.pcapng, log\_exp3\_tux2.pcapng, log\_exp3\_tux4.pcapng, log2\_exp3\_tux1.pcapng, log2\_exp3\_tux2.pcapng e log2\_exp3\_tux4.pcapng).