FEUP – Faculdade de Engenharia da Universidade do Porto

Redes de Computadores

2º Trabalho laboratorial

Rodrigo Campos Rodrigues – up202108847@fe.up.pt

Miguel Martins Leitão - up202108851@fe.up.pt

Porto, 22 de Dezembro de 2023

Sumário

Este projeto foi realizado no âmbito da Unidade Curricular Redes de Computadores (RCOM). O objetivo é a implementação de um protocolo FTP de download de ficheiros e a configuração e utilização de uma rede de computadores.

O desenvolvimento deste projeto permitiu fazer uso de e consolidar vários tópicos tratados nas aulas teóricas.

Introdução

Neste trabalho, o nosso foco foi criar e testar um programa para fazer download de ficheiros usando o protocolo FTP e configurar uma rede de computadores conforme as instruções fornecidas. O relatório está organizado em várias seções, que incluem:

- **Introdução:** Descrição do projeto e os seus objetivos.
- **Aplicação de download FTP:**
  - Funcionamento da aplicação FTP;
  - Resultados;
- **Configuração e análise de redes:**
  - Configurar uma rede IP;
  - Implementar duas bridges num switch;
  - Configurar um router em Linux;
  - Configurar um router comercial com NAT;
  - DNS;
  - Ligações TCP;
- **Conclusões:** Resumo das informações previamente tratadas e principais objetivos alcançados.

Aplicação de download FTP

Funcionamento da aplicação FTP

A aplicação **download** utiliza o protocolo FTP para fazer o download de ficheiros. Tem em consideração as normas RFC959 e RFC1738, que especificam o envio e tratamento de respostas, e o tratamento das diferentes partes do URL.

Inicialmente, a aplicação processa o URL passado em argumento. Este processo é feito através de aritmética de apontadores. Cada campo do url possui um apontador de início e fim, através dos métodos strchr() e strrchr() que retornam a posição da primeira e última ocorrência de um caractere, respetivamente. Após obter todos os componentes estes são armazenados numa estrutura **struct URL**, que incluí todos os parâmetros necessários para a transferência do ficheiro: *user, password, host, path, filename, ip* (este último é traduzido do host, na função getIpAddress, através do sistema DNS).

É de seguida criada uma socket para fazer o envio de comandos e receção de respostas. Faz-se a sua conexão ao endereço ip da struct e à porta 21 (porta reservada para comunicações TCP/IP). Faz-se a autenticação através do user e pass da struct e em seguida é ativado o modo passivo para fazer a receção de pacotes de dados do servidor.

Para fazer a receção de respostas do servidor, foi creada uma função handleResponses(), que processa byte a byte as respostas recebidas do servidor (como uma máquina de estados), dado a socket como argumento. É capaz de lidar com comandos de linha única ou multi-linha.

É criada uma segunda socket (de dados) que será responsável por receber os dados e os transmitir para um buffer que irá fazer a escrita desses mesmos dados para um novo ficheiro.

De seguida, e se todo o processo tiver ocorrido sem erros, faz-se então o término da ligação, e fecha-se as conexões das duas sockets.

Resultados

Todos os ficheiros presentes no servidor foram transferidos com sucesso, independentemente do seu tamanho. A aplicação, em caso de URL inválido retorna uma mensagem e aborta imediatamente. A aplicação mostrou-se semelhantemente robusta em caso de ficheiros não existentes ou tentativas de autenticação falhadas.

Este é o output em caso de uma transferência bem sucedida.

```
curlmike@curlmike:~/feup-rcom/proj2$ ./ftp ftp://rcom:rcom@netlab1.fe.up.pt/pipe.txt
------URL parsed-------
User: rcom
Password: rcom
Host: netlab1.fe.up.pt
Path: /pipe.txt
Filename: pipe.txt
IP: 192.168.109.136
-----------------------
220 Welcome to netlab-FTP server
[SERVER RESPONSE] 220 -> Command socket created.
331 Please specify the password.
[SERVER RESPONSE] 331 -> Username accepted.
230 Login successful.
[SERVER RESPONSE] 230 -> Authenticated.
227 Entering Passive Mode (192,168,109,136,192,16).
[SERVER RESPONSE] 227 -> Passive mode enabled.
[CLIENT] -> Data socket created.
[CLIENT] -> File command sent.
150 Opening BINARY mode data connection for /pipe.txt (1863 bytes).
[SERVER RESPONSE] 150 -> File transfer started.
226 Transfer complete.
[SERVER RESPONSE] 226 -> File 'pipe.txt' downloaded.
221 Goodbye.
[SERVER RESPONSE] 221 -> Connection closed.
curlmike@curlmike:~/feup-rcom/proj2$
```

Na imagem anterior é possível ver todos os campos do url separadamente, assim como os vários comandos enviados do client e respostas recebidas do servidor, com o respetivo código de resposta dado a norma RFC959.

O ficheiro obtido pelo download da aplicação ftp é válido tal como esperado.

Configuração e análise de redes

Experiência 1 - Configurar uma rede IP

O objetivo desta experiência baseia-se na configuração de dois endereços IP de dois computadores diferentes, o TUX43 e o TUX 44, ligados a um switch. Em seguida, analisou-se o funcionamento do protocolo ARP, bem como o comportamento da ligação após a eliminação das entradas dos IPs nas tabelas ARP.

Tendo isso em conta, ligamos o eth0 de cada um dos TUX ao switch e configuramos o IP de cada um utilizando o comando *ifconfig*. Configuramos o TUX43 com o endereço 172.16.40.1/24 e o TUX44 com o endereço 172.16.40.254/24. Posteriormente, observamos o endereço MAC de cada um dos computadores nas tabelas ARP que foram criadas após correrrmos o comando *ping* no terminal, verificando a conexão entre os dois computadores. O endereço MAC do TUX43 correspondia ao 00:21:5a:61:2f:13 e o do TUX44 correspondia ao 00:21::5a::c3:78::76. Nas tabelas ARP notou-se a existência de pacotes ARP e ICMP.

Protocolo ARP: realiza a ligação entre a *network layer* e a *link layer*. Na prática, mapeia o endereço de IP de um computador ao seu respetivo endereço MAC. No primeiro pacote são enviados

dois endereços de IP, o do computador de destino e o do computador de origem. O computador de destino responde ao outro computador, enviando um pacote do mesmo protocolo com o seu endereço MAC. Depois desta troca de informação, estes registos são guardados numa tabela nos dois computadores, para serem utilizados em futuras comunicações. Quando eliminados as entradas nas tabelas ARP, verificamos que existe uma troca de pacotes no início da ligação com o objetivo de voltar a preencher as tabelas. Esta troca de pacotes foi evidenciada nas X linhas.

Protocolo ICMP: envia mensagens de controlo, com o objetivo de detetar sucesso ou erros ao longo da comunicação entre os dois computadores. Nesta experiência, observamos mensagens de reply e request.

Comandos e Logs Wireshark Exp.1

Experiência 2 - Implementar duas *bridges* num switch

O objetivo desta experiência foi criar duas bridges no switch, uma com os TUX43 e TUX44 e a outra com o TUX42.

Numa primeira fase, configuramos a rede da mesma de acordo com as configurações da primeira experiência e utilizamos o comando *ifconfig* para configurar o eth0 do TUX42 com o endereço de IP 172.16.41.1/24. Em seguida, ligamos a consola do switch à porta série do TUX42. Com isto, mudamos a configuração do switch, criando duas bridges (bridge40 e bridge41), utilizando o comando /interface bridge add para adicionar uma bridge. Para colocar cada um dos TUXs nas respectivas bridges, tivemos primeiro de remover as portas às quais as interfaces de cada um dos TUXs estavam ligadas por defeito, utilizando o comando /interface bridge port remove, e, finalmente, adicioná-los às bridges com o comando /interface bridge port add.

Numa segunda fase, fizemos broadcast a partir do TUX43 e verificamos que este conseguiu estabelecer comunicação com o TUX44, mas não com o TUX42, isto porque o TUX44 encontra-se na mesma bridge que o TUX43 e o TUX42 não. Quando fizemos broadcast a partir do TUX42 reparamos que não conseguiu estabelecer comunicação com nenhum dos TUXs, isto porque se encontra numa bridge diferente que os outros dois. Estes comportamentos foram observados através dos pacotes ICMP capturados nos dois computadores (TUX43 e TUX42). No TUX43 observamos pacotes reply e request, enquanto que no TUX42 apenas observamos pacotes reply sem resposta.

Comandos e Logs Wireshark Exp.2

Experiência 3 - Configurar um router em Linux

O objetivo desta experiência foi transformar o TUX44 num router de forma a que o TUX43 e o TUX42 pudessem comunicar-se entre si mesmo estando em bridges diferentes.

Numa primeira fase, configuramos a rede de modo a obter uma rede igual à rede resultante da experiência 2. Posteriormente, conectamos o eth1 do TUX44 ao switch e configuramos o seu endereço de IP 172.16.41.253/24, utilizando o comando *ifconfig*. Em seguida, adicionamos a interface do switch à qual o TUX44 foi ligado à bridge41, seguindo o procedimento que foi utilizado na segunda experiência. Ativamos o IP forwarding e desativamos o ICMP echo-ignore-broadcast. Por fim, criamos routes nos TUXs

42 e 43, utilizando como gateway o endereço de IP do TUX44 que estava disponível a cada um deles (172.16.41.253/24 para o TUX42 e 172.16.40.254/24 para o TUX43).

Finalmente, utilizamos o comando ping para testarmos a comunicação entre o TUX42 e o TUX43, observando que esta ocorreu com sucesso. Assim, podemos confirmar que configuramos com sucesso o TUX44 como um router. Os pacotes resultantes da comunicação continham o endereço de IP da máquina de destino, mas o endereço MAC não era o da máquina de destino mas sim o endereço MAC do TUX44, uma vez que, sendo um router, trata do redirecionamento da informação entre as duas redes criadas.

Comandos e Logs Wireshark Exp.3

Experiência 4 - Configurar um router comercial com NAT

Utilizando as rede resultante da experiência 3, uma com o TUX43 e outra com o TUX42 (que utilizam o TUX44 como router para fazer a ligação das duas sub-redes) como base, foi configurada a bridge41, adicionando-a a um router comercial que permite fazer a ligação à internet.

Ligou-se inicialmente, uma das entradas do router à régua (que permitiu o acesso à internet), e outra ao switch (que permitiu comunicar com a rede LAN construída anteriormente). Adicionou-se a interface à bridge41, tal como nas experiências anteriores. Fez-se a troca do cabo que permitia interagir com a consola do switch pelo cabo que permitia interagir com a consola do router, de forma a poder fazer a sua configuração.

Através da consola do router, adicionaram-se os IPs correspondentes às outras interfaces. Foram adicionadas rotas default aos outros TUXes, ligando-os ao router, e uma rota que permitia ao router ligar-se aos outros TUXes através do TUX44. Após a configuração da rede foram tiradas as seguintes conclusões:

Na primeira situação, sem a conexão do TUX42 ao TUX44 e com ICMP redirects desativados, o envio de pacotes do TUX42 para o TUX43 foi feito pelo router (utilizando a *default* route) até chegarem ao endereço de destino (Anexo 4.1).

Ativando novamente ICMP redirects e ativando a conexão do TUX42 ao TUX44, o envio de pacotes do TUX42 para o TUX43 foi feito pelo TUX44, e a rota *default* do router não foi tomada, uma vez que a conexão através do TUX44 é mais direta. Daqui conclui-se que os pacotes ICMP optam por ligações mais rápidas na rede quando estas existem, utilizando redirects quando necessário.

O NAT é responsável pela tradução de endereços públicos em endereços locais e vice-versa. Quando um pacote é enviado para uma rede externa, é enviado com um endereço público como origem. Quando o computador de destino envia a sua resposta, esta é também enviada sob a forma de um endereço público. Estes endereços são traduzidos de volta para endereços locais.

Assim, é esperado que após se desativar o NAT seja perdido o acesso à internet, uma vez que não há forma de o router fazer a tradução de endereços públicos para endereços privados, ou seja não existe forma de o emissor receber uma resposta aos pedidos solicitados.

Comandos e Logs Wireshark Exp.4

Experiência 5 - DNS

O objetivo desta experiência foi fazer a configuração do DNS de modo a permitir o acesso a websites através do seu nome de domínio, utilizando a rede configurada nas experiências anteriores.

Para tal, apenas foi preciso mudar o conteúdo do ficheiro /etc/resolv.conf em cada TUX para: “nameserver 172.16.1.1” (endereço IP do router do laboratório, que comunica externamente e permite o acesso à internet). Para verificar que a configuração foi bem sucedida fez-se *ping google.com*.

Através desta experiência concluiu-se que o protocolo DNS é o primeiro a ser executado, já que é este o responsável por fazer a tradução de um nome de domínio para um endereço IP, que posteriormente é utilizado por todos os outros protocolos.

Comandos e Logs Wireshark Exp.5

Experiência 6 - Ligações TCP

Nesta experiência foi utilizado a aplicação download (ftp.c) desenvolvida juntamente com a rede configurada através das outras experiências. O objetivo foi avaliar o envio e a receção de pacotes e os mecanismos utilizados nas ligações TCP.

Inicialmente foi corrida a aplicação donwload no TUX43. Observou-se inicialmente a tradução do host para endereço IP, feito por DNS. De seguida observou-se a troca de pacotes FTP SYN-ACK que é um 3-way handshake feito entre o client e o server (o client envia uma trama SYN, o servidor responde com uma trama SYN-ACK e finalmente o client envia uma trama ACK). Finalmente fez-se a receção de pacotes de dados, seguidos de uma trama FIN-ACK (trama que sinaliza o término da ligação).

Uma vez que a aplicação download abre duas sockets (uma para envio/receção de comandos e outra para receção de pacotes de dados), ela estabelece duas ligações TCP, que utilizam o mecanismo ARQ.

O mecanismo ARQ é utilizado nas ligações TCP para controlo de erros (através de tramas ACK), e utiliza timeouts para garantir fluidez na comunicação. Caso um pacote passe do tempo limite é considerado como perdido e é retransmitido.

As ligações TCP incorporam dois outros mecanismos: controlo de congestionamento e controlo de fluxo. O controlo de fluxo permite que o envio de pacotes seja limitado a um número máximo de bytes, de forma a que não haja perda de informação (uma vez que os buffers do lado do recetor/emissor são finitos). Assim é juntamente enviado com o pacote de informação, uma *window size* em bytes, que especifica o número de bytes de podem ser enviados (do lado do emissor) de modo a que não haja perda de dados ou quebra no fluxo de informação.

O controlo de congestionamento consiste na avaliação da capacidade atual da rede, de modo a enviar o número ideal de pacotes sem que haja perdas significativas. Para tal são utilizados mecanismos como o *additive increase*, que consiste no aumento de um pacote em cada transferência, até que haja congestionamento da rede. Uma rede diz-se congestionada quando existem perdas de pacotes. Por sua vez, uma perda de um pacote pode ser determinada através do envio de 3 tramas ACK seguidas ou caso ocorra um *timeout*.

É possível observar o controlo de congestionamento analisando a velocidade de envio de pacotes na rede durante o decorrer da aplicação download. Nota-se que a velocidade vai aumentando significativamente, até que haja perda de pacotes significativa. A partir deste momento há um declínio na velocidade até que esta alcance valores estáveis.

Correndo a aplicação download em dois computadores na rede simultaneamente (TUX43 e TUX42), é possível observar novamente este mecanismo. Após se iniciar o download no TUX43, ao correr a aplicação no TUX42, observa-se que a velocidade de transmissão de pacotes no TUX43 se reduz para metade. A rede automaticamente diminui a velocidade de transmissão de modo a que a transferência fique estável, sem perda de pacotes de dados.

Logs Wireshark Exp.6

Conclusões

Este trabalho permitiu consolidar o nosso conhecimento no que diz respeito à configuração de uma rede de computadores, assim como a ligações TCP/IP e os mecanismos ARQ envolvidos durante a transmissão de dados. Os objetivos deste trabalho foram concluídos com sucesso.