Computer Networks

Relatório Intercalar

Mestrado Integrado em Engenharia Informática e Computação

Redes de Computadores

Grupo:
José Carlos Alves Vieira - up201404446
Renato Sampaio de Abreu - up201403377

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

16 de Dezembro de 2016

Índice

1. Sumário

Este projeto, desenvolvido no contexto da cadeira Redes de Computadores do Mestrado Integrado de Engenharia Informática, tem como objetivo o desenvolvimento de uma aplicação de download ftp e a configuração e estudo de uma rede. Posto isto, o projeto incide maioritariamente em conteúdos que tínhamos abordado durante as aulas teóricas, principalmente a Mac Sublayer, Network Layer e Transport Layer.

O relatório permitiu-nos aprofundar os aspectos mais teóricos dos conteúdos anteriormente referidos, de forma a relacionar a parte teórica com a parte prática, possibilitando uma melhor aprendizagem por parte do grupo e ao mesmo tempo facilitar o processo de correção do projeto por parte do docente.

2. Introdução

O objectivo proposto para o segundo trabalho de Redes de Computadores era, numa primeira parte, implementar uma aplicação de download de um ficheiro único, seguindo o protocolo de transferência de ficheiros descrito no RFC 959 e aplicando como sintaxe de URL a descrita no RFC 1738.

Na segunda parte do trabalho, o objetivo foi configurar e estudar minuciosamente uma rede de computadores. Assim, ao longo das aulas laboratoriais realizaram-se várias experiências centrando-se à volta dos aspetos fundamentais de uma rede (IP, NAT, DNS, TCP, etc).

Este relatório subdivide-se nas seguintes secções:

- **Introdução**, onde é realizada uma introdução aos objectivos do projecto.
- **Aplicação de Download**, referindo a arquitetura e protocolo implementados e os resultados de uma transferência com sucesso.
- **Análise e configuração da rede**, referindo os objetivos principais e conceitos de cada experiência realizada bem como a arquitetura de rede e análise de logs.
- **Conclusões**, elaborando uma síntese das secções apresentadas e uma breve reflexão sobre os objectivos académicos alcançados.

3. Aplicação de Download

3.1. Arquitetura

A aplicação desenvolvida está estruturada em duas componentes: o download do ficheiro, especificado nos ficheiros ftp.h e ftp.c, e o parser do URL, presente nos ficheiros urlParser.h e urlParser.h.

Existem 2 estruturas de dados principais: **ftp_data** e **parsed_url** (Ver anexo I).

Os URL seguem a sintaxe descrita no RFC1738: ftp://[<user>:<password>@]<host>/<url-path portanto cada estrutura **parsed_url** armazena o url original, scheme, host, port, path, username, password, ip e o filename.

```
struct parsed_url {
    const char *url;
    char *scheme;
    char *host;
    int port;
    char *path;
    char *username;
    char *password;
    char *ip;
    char *filename;
};
```

De seguida são apresentadas as funções relativas ao parsing do url:

```
struct parsed_url * parse_url(const char *);
int hostToIP(struct parsed_url *);
void printParsedUrl(struct parsed_url *);
void freeUrlStruct(struct parsed_url *);
```

- **parse_url**, que recebe um URL e realiza o seu parser obtendo as variáveis existentes na estrutura **parsed_url.** No URL, o user e password podem estar omitidos e como tal, é atribuído por default o valor de "anonymous" ao username e posteriormente é pedido o email universitário ao utilizador que será associado à password.
- **hostToIP,** responsável pela obtenção do endereço IP associado ao host obtido no parse_url.
- **printParsedUrl**, escreve na consola a informação guardada em cada campo da estrutura.
- **freeUrlStruct**, liberta a memória alocada a variáveis que necessitaram de alocação de memória.

O protocolo de transferência de ficheiros (FTP), estabelecido no RCF959, recorre a uma conexão TCP de dados e de controlo. A conexão de controlo é utilizada para abrir e fechar sessões FTP tal como enviar comandos do cliente para o servidor. Por outro lado, a conexão de dados é utilizada durante a transferência de ficheiros (download ou upload) entre o cliente e o servidor. Posto isto, a estrutura **ftp_data** possui um descriptor para o socket de controlo e outro para o socket de dados.

```
struct ftp_data{
    int controlSocketFd;
    int dataSocketFd;
};
```

De seguida são apresentadas as funções relativas à conexão e transferência:

```
int connectSocket(struct ftp_data *ftp, const char *ip, int port);
int ftpConnect(struct ftp_data *ftp, const char* ip, int port);
int ftpLogin(struct ftp_data *ftp, const char *username, const char *password);
int ftpSetPassiveMode(struct ftp_data *ftp);
int ftpDownload(struct ftp_data *ftp, const char *path, const char *filename);
int ftpLogout(struct ftp_data *ftp);
int ftpRead(struct ftp_data *ftp, char *str, size_t size, int expectedCode);
int ftpSendMessage(struct ftp_data * ftp, char *str);
```

- **connectSocket**, responsável por abrir uma socket TCP e pelo estabelecimento da conexão com o servidor.
- **ftpConnet**, função que invoca connectSocket para associar o socket de controlo e lê a resposta do servidor (serviço pronto a ser usado).
- **ftpLogin**, responsável pelo login do utilizador (username e password) e pela verificação dos códigos recebidos do servidor.
- **ftpSetPassiveMode**, responsável por impôr o modo passivo. Calcula a porta através da mensagem recebida e associa o socket de dados.
- **ftpDownload**, responsável pelo download do ficheiro em questão.
- **ftpLogout**, responsável pelo logout do utilizador e pela verificação do código enviado pelo servidor.
- **ftpRead**, responsável pela leitura de todas as respostas enviadas pelo servidor e verificação do código respectivo.
- **ftpSendMessage**, responsável por escrever para o servidor.

Uma sessão de transferência de dados segue uma sequência comum de passos, e desta forma, a componente ftp está dividida em funções que de maneira geral representam esta sequência.

3.2. Transferência

Assim, em primeiro lugar é estabelecida a conexão de controlo ao servidor, através da função **ftpConnect**. Neste passo, é utilizada a função **connectSocket** que abre uma conexão de controlo recorrendo ao endereço IP (obtido através do URL) e à porta (por default, 21), sendo que o "controlSocket" resultante é utilizado no envio de comandos FTP (Cliente -> Servidor) e respostas FTP (Servidor -> Cliente).

Posto isto, é necessário realizar a autenticação do utilizador no servidor e portanto a função **ftpLogin** envia primeiro o comando USER juntamente com o username e posteriormente envia o comando PASS com a password. Entre cada comando enviado, a função valida a resposta recebida.

O modo FTP usado é o passivo e como tal é necessário que o cliente abra uma ligação de dados TCP existindo assim apenas ligações TCP de saída (dados e controlo). Podemos então considerar o server como passivo já que o socket TCP é aberto em "listening mode".

A função **ftpSetPassiveMode** é responsável por ativar o modo passivo: Primeiro é enviado o comando PASV e é recebida uma resposta por parte do servidor com o endereço de IP e a porta TCP. Posteriormente é analisada a resposta recebida e estabelece-se uma conexão de dados, utilizando a função **connectSocket** e o descriptor "dataSocket".

De seguida, na função **ftpDownload,** o cliente inicia a transferência do ficheiro usando o comando RETR, começando de seguida o download e respectivo armazenamento do ficheiro especificado através do socket de dados. Uma vez realizado o download, o server encarrega-se de fechar a conexão de dados TCP.

Por último, o cliente procede ao fecho da sessão FTP usando a função **ftpLogout**, a qual envia o comando QUIT e procede ao fecho da conexão de controlo.

4. Análise e Configuração de Rede

4.1. Experiência 1 - IP Network

Esta experiência teve objetivo a compreensão do funcionamento de uma simples rede de computadores ligados entre si bem como o tipo de dados enviados entre ambos.

Assim, ao longo da experiência verificamos o envio de pacotes ARP (Protocolo de resolução de endereços), sendo que estes são necessários para obter o endereço MAC e associá-lo a um endereço IP.

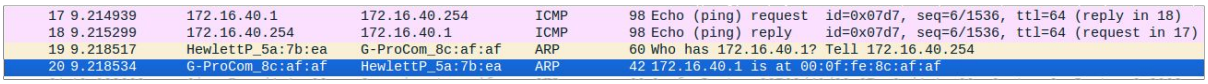

Figura 1 - Pacotes ARP e ICMP

Desta forma, existem dois pacotes ARP: Primeiro, é necessário associar o IP que se pretende alcançar a um determinado endereço MAC, daí que este pacote é enviado a toda a rede do computador de origem. O segundo é utilizado quando o computador que possua o IP em questão, envia um pacote ARP contendo o seu endereço MAC.

Após o estabelecimento dos endereços, inicia-se o envio de pacotes de protocolo ICMP, podendo ser do tipo reply ou request. Na experiência foi realizado o ping da máquina 4 a partir da máquina 1, e como tal cada pacote do tipo "request" tem como IP e MAC de origem a máquina 1 e de destino os valores IP e MAC da máquina 2. Por outro lado, os pacotes de tipo "reply" têm os valores contrários.

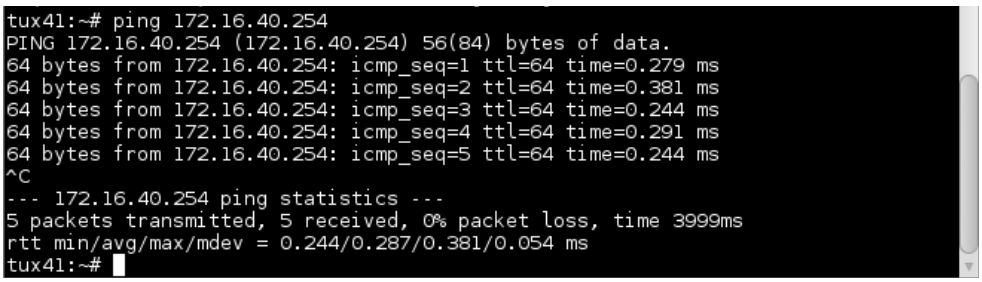

Figura 2 - Ping da máquina 172.16.40.254

É possível identificar o tipo de trama ethernet recebida analisando os bytes 12-13 de cada trama, visto que estes correspondem ao valor "eth.type".

A interface loopback é uma interface existente nos routers (neste caso da Cisco). É uma interface virtual que o computador usa para comunicar consigo mesmo, estando sempre disponível, disponibilizando uma forma de as aplicações locais que correm no computador se possam ligar a servidores na mesma máquina, caso contrário, se portas ethernet fossem desconectadas ou o Wi-Fi desligado, isso seria impossível.

4.2. Experiência 2 - Virtual LANs

A experiência 2 consiste em implementar duas Virtual LANs (Local Area Network). Assim, tal como especificado no guião e de acordo com a nossa área de trabalho (bancada 4), foram criadas a VLAN40, contendo as máquinas 1 e 4 com os IPs 172.16.40.1 e 172.16.40.254, respetivamente, e a VLAN41 apenas com a máquina 2 com o IP 172.16.41.1.

Desta forma, a configuração de cada vlan4**x**, sendo **x** o ou **1**, é realizada no switch da Cisco: Em primeiro lugar é necessário criar a vlan usando-se os comandos 'configure terminal', 'vlan4**x**' e 'end'. Posteriormente, de modo a configurar a rede de cada vlan, são adicionadas as portas correspondentes, recorrendo para isso a uma série de comandos: 'configure terminal', 'interface fastethernet 0/i' (onde i é o número da porta), 'switchport mode access', 'switchport access vlan 4**x**' (onde **x** é 0 ou 1, dependendo da vlan a que se quer adicionar a porta i).

Conforme as configurações exigidas no guião, o grupo adicionou a porta 1 e 4 à vlan40 e a porta 2 à vlan41, seguindo os passos anteriormente referidos, resultando na configuração apresentada na figura 2, a qual corresponde ao output do comando 'show vlan brief'.

Um domínio de broadcast é uma divisão lógica da rede em que qualquer node consegue alcançar qualquer outro node pertencente a esse broadcast (i.e ping). Como tal, a criação e respetiva configuração de uma vlan essencialmente cria o seu próprio domínio de broadcast. Então, devido à arquitetura da rede podemos concluir que existem dois domínios broadcast.

A análise dos logs obtidos ao longo da experiência permite confirmar a correta implementação da arquitetura de rede especificada no guião uma vez que o ping do tux44 a partir do tux41 (e vice versa) funciona e como tal pertencem ao mesmo domínio (vlan40). Por outro lado, tanto o tux41 como o tux44 não conseguem alcançar o tux42 podendo-se concluir que não conseguem alcançar a sub-rede pertencente à vlan41.

4.3. Experiência 3 - Router Configuration

A experiência 3 consiste em transformar a máquina tux44 num router. Para tal, foi necessário configurar a interface ethernet 1 (eth1) da máquina em questão através dos seguintes comandos: 'ifconfig eth1 up'; 'ifconfig eth1 172.16.**41**.253/24'; 'echo 1 > /proc/sys/net/ipv4/ip_forward'; 'echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts'. Por fim, adicionou-se a interface eth1 (através da porta 5) do tux44 à vlan41 de forma a que a máquina 2 e 4 pertençam à mesma sub-rede e possam comunicar entre si. A máquina 4 vai possuir então duas interfaces, eth0 e eth1, e duas rotas para cada tux.

A comunicação entre o tux41 e o tux42 é alcançada adicionando as portas necessárias para a respetiva subrede.

Na máquina 1 é adicionada a rota "route add -net 172.16.41.0/24 gw 172.16.40.254", que por sua vez possibilita a comunicação com as máquinas da subrede 172.16.41.0/24 utilizando como gateway 172.16.40.254, ou seja, os pacotes enviados para essa subrede são reencaminhados para a máquina com o IP identificado na gateway, que no nosso caso corresponde à máquina 4.

Após isto foi introduzido na máquina 2 a rota "route add -net 172.16.40.0/24 gw 172.16.41.253", que permite o envio de pacotes a partir da máquina para a subrede com endereço 172.16.40.0/24 passando pelo gateway.

Assim a máquina 1 pode comunicar com a máquina 2, sendo que as interfaces existentes na máquina 4 servem de intermediador entre esta ligação, ou seja, esta funciona como router.

As tabelas de forwarding são obtidas através do comando "route -n" e estas possuem a informação necessária para enviar um pacote através do melhor caminho até ao seu destino. As tabelas são constituídas pelo IP de destino do pacote, o IP para o qual o pacote é enviado, a máscara e a interface utilizada no "forwarding" do pacote.

Ao pingar a máquina 2 a partir da 1 é possível observar a sequência de reencaminhamento existente. O pedido efetuado para a máquina com IP 172.16.41.1 é reencaminhado para o "router", i.e máquina 4, através da interface ethernet 0, sendo que a máquina obtém o endereço MAC da interface em questão (172.16.40.254). Os dados são então enviados para a máquina 2, já que está conectada à máquina 4 através da interface ethernet 1. Neste último caso, a máquina 4 obtém o endereço MAC da máquina 2. Em ambos os casos em que é necessário obter os endereços MAC são utilizados pacotes ARP.

Os pacotes ICMP observados no wireshark nas requests correspondem aos valores IP e MAC da máquina 1 e 2, sendo respectivamente a origem e o destino, enquanto que nas replies é exatamente o contrário. Uma vez que seja possível alcançar a máquina de destino e o endereço MAC seja mapeado, são atribuídos como IPs aos pacotes ICMP o valor das respetivas máquinas de origem e destino, sendo que a rota efectuada e os respectivos reencaminhamentos são da responsabilidade do router.

4.4. Experiência 4 - Comercial Router and NAT

A experiência 4 consiste em configurar o router comercial e implementar o NAT.

Para tal, foi necessário configurar o gigabitethernet 0/0 e o gigabitethernet 0/1. Na consola do switch, usam-se os comandos 'configure terminal', 'interface gigabitethernet 0/i' (onde i é 0 ou 1, dependendo do gigabitethernet a configurar), 'ip address customIP 255.255.255.0' (onde customIP é 172.16.41.254 para o gigabitethernet 0/0 e 172.16.1.49 para o gigabitethernet 0/1), 'no shutdown', 'exit'. Estes comandos são executados tanto para o gigabitethernet 0/0 como para o 0/1. De seguida configura-se a rota, através dos comandos 'conf t', 'ip route 0.0.0.0 0.0.0.0. 172.16.1.254', 'ip route 172.16.40.0 255.255.255.0 172.16.41.253', 'end'.

Para a configuração do NAT, executaram-se os comandos 'conf t', 'interface gigabitethernet 0/0', 'ip address 172.16.41.254 255.255.255.0', 'no shutdown' (assegura-se que as configurações são mantidas, mesmo que o router seja desligado), 'ip nat inside', 'exit', 'interface gigabitethernet 0/1', 'ip adress 172.16.1.49 255.255.255.0', 'no shutdown', 'ip nat inside', 'exit', 'ip nat pool ovrld 172.16.1.49 172.16.1.49 prefix 24', 'ip nat inside source list 1 pool ovrld overload' (desta forma é mantida a gama de endereços), 'access-list 1 permit 172.16.40.0 0.0.0.255', 'access-list 1 permit 172.16.41.0 0.0.0.255' (criou-se uma lista de permissões e acessos aos pacotes que iriam ser enviados/recebidos, isto para todas as sub-redes. Foi utilizado o máximo permitido (255) para deixar que todas as subredes tivessem acesso à internet), 'ip route 0.0.0.0 0.0.0.0 172.16.1.254', 'ip route 172.16.40.0 255.255.255.0 172.16.41.253' (adicionaram-se rotas que basicamente redireciona os pacotes que têm como destino o ip 172.16.40.0 para o ip ip 172.16.41.253.), 'end'.

Testando o que foi implementado fez-se, pela primeira vez, um ping do tux42 para o tux41. Os pacotes eram enviados para o router, que posteriormente enviava para a rede 172.16.40.0. Realizando o mesmo processo novamente, o caminho seguido pelos pacotes é diferente, indo do tux42 directamente para a rede 172.16.40.0.

Ao fazer um ping do tux41 para o router (sem NAT) não há resposta, mas ao adicionar NAT já a obtemos.

A NAT (Network address translation) permite que um dispositivo (router Cisco) aja como um intermediador entre a Internet e uma rede local (lab network). Recorrendo a isto, todos os computadores pertencentes à rede local são representados apenas com um endereço de IP único. A NAT oferece então segurança adicional na medida em que toda a rede de computadores é representada por um único endereço.

4.5. Experiência 5 - DNS

A experiência 5 consiste em configurar o DNS de forma a conseguir aceder a redes externas, fazendo uso da internet através da rede interna que foi criada nas experiências anteriores.

Assim, fez-se a configuração através do ficheiro resolv.conf. Neste caso, o nameserver seria configurado com o IP 172.16.1.1

Pacotes de DNS são enviados e recebidos, contendo informação com queries e respostas. As queries possuem o "name", "type" e "class", enquanto que as respostas, para além dos campos que existem nas queries, também possuem "time to live", "data length" e "address". Em termos de exemplo, apresentamos os dados de um dos pacotes:

Queries: Name: google.com, Type: A (Host Address) (1), Class: IN (0x0001).

Answers: Name: google.com, Type: A (Host Address) (1), Class: IN (0x0001), Time to live: 155, Data length: 4, Address: 194.210.238.163.

4.6. Experiência 6 - TCP connections

A experiência 6 consiste em compilar a aplicação de download, correr a aplicação e fazer download de um ficheiro que esteja alojado na internet, mais especificamente num domínio introduzido aquando da execução da aplicação.

Para testar foi usado um servidor ftp para fazer o download, e depois de se ter corrido a aplicação, verificou-se que o ficheiro foi descarregado, demonstrando que as redes estavam bem configuradas.

A aplicação ftp realiza 2 conexões TCP, a conexão de controlo e a conexão de dados.

A informação de controlo FTP, como a identificação do utilizador, password ou comandos para mudar diretórios remotos é transportada pela conexão de controlo.

O estabelecimento da conexão TCP é realizado por fases. Em primeiro lugar, o cliente envia a mensagem SYN com informação relativa ao número de sequência, de seguida recebe a resposta do servidor, resposta essa que inclui SYN e ACK (acknowledgment) da mensagem anteriormente enviada. Para finalizar, o cliente envia uma mensagem de ACK (acknowledgment).

Relativamente ao mecanismo de ARQ do TCP, este usa o mecanismo de *Sliding Window*, que é uma variante do *Go-Back-N*. Este mecanismo permite ao receptor receber n bytes sem ter de esperar pela confirmação do ACK. O receptor envia um ACK que inclui o número de sequência esperado do próximo byte, ou seja, o receptor pode receber n bytes, começando com o byte com o número de sequência esperado. Tanto o receptor como o servidor possui uma "janela", janela esse com um tamanho menor ou igual ao tamanho do buffer. O servidor envia todos os bytes permitidos pelo tamanho da janela e fica à espera de um ACK. Então o receptor desloca-se na

janela para o número de sequência correspondente, indicando os bytes que podem ser enviados pelo servidor que o buffer da nova posição da janela permite. O servidor recebe o ACK com essa informação, voltando a enviar os bytes.

Como se pode observar pelo funcionamento da *sliding window*, um dos elementos mais importantes neste mecanismo é o número de sequência.

O mecanismo de controlo de congestão do TCP funciona com um comportamento em modo "serra". Basicamente, o número de bytes transmitidos por segmento vão sendo cada vez maiores, até que é detectado que um segmento foi perdido, fazendo com que o número de bytes transmitidos diminua e se entre no modo "congestion avoidance". Através dos logs pode-se verificar o número de bytes enviados por segmento (ver próxima imagem).

No.	Time	Source	Destination	Protocol	Length	Info
3872	6.265302	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5621113 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3873	6.265530	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes
3874	6.265548	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5623849 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3875	6.265781	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes
3876	6.265800	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5626585 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3877	6.266032	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes
3878	6.266049	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5629321 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3879	6.266281	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes
3880	6.266299	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5632057 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3881	6.266531	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes
3882	6.266550	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5634793 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3883	6.266783	193.136.37.8	172.16.40.1	FTP-DATA	4170	FTP Data: 4104 bytes
3884	6.266801	172.16.40.1	193.136.37.8	TCP	66	41743→46806 [ACK] Seq=1 Ack=5638897 Win=65535 Len=0 TSval=2384969 TSecr=722493402
3885	6.267032	193.136.37.8	172.16.40.1	FTP-DATA	2802	FTP Data: 2736 bytes

Figura 3 -Bytes transmitidos por segmento

Os resultados estão de acordo com o mecanismo, uma vez que começam com um valor baixo, sobem e voltam a descer (quando é detectado que um segmento foi perdido), criando o efeito "serra".

Analisando os logs recolhidos, podemos observar, usando o *I/O Graph*, que a conexão de dados foi afectada por termos começado a transferir outro ficheiro noutra máquina, ou seja, pelo aparecimento de uma segunda conexão de dados. Inicialmente começou-se um download no tux41, que atingiu uma velocidade de aproximadamente 8000 pacotes por segundo. No entanto, devido ao aparecimento de uma nova conexão de dados (na máquina tux42), a velocidade diminuiu para aproximadamente metade (4000 pacotes por segundo), sendo esta também a velocidade a que a segunda transferência começou. Quando o download no tux41 acabou, a velocidade no tux42 subiu para 8000 pacotes por segundo, até que ao fim do download.

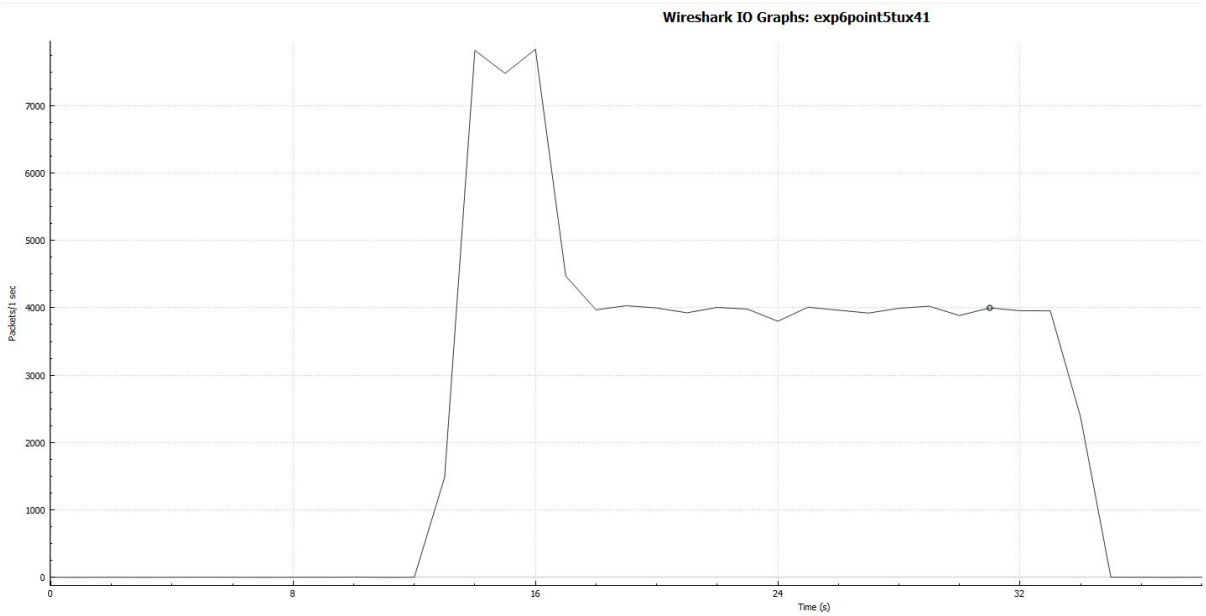

Figura 4 - I/O Graph registado no tux41

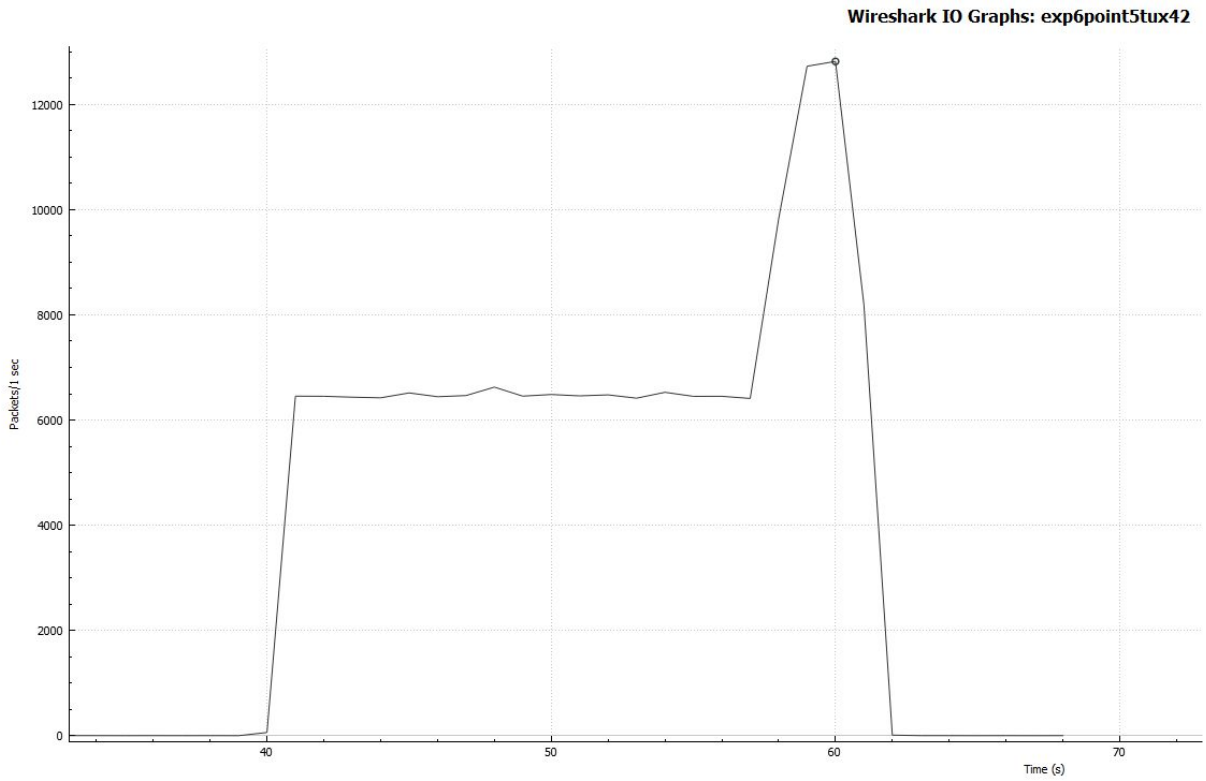

Figura 5 - I/O Graph registado no tux42

Pode-se então concluir que a velocidade é afetada pelo aparecimento de outra conexão de dados.

5. Conclusões

Com o término das experiências e do relatório em si, pode-se afirmar que o grupo conseguiu interiorizar e perceber os conceitos necessários inerentes a este segundo trabalho da cadeira de Redes de Computadores.

A implementação de uma aplicação de download não foi difícil, no entanto foi necessário estudar conceitos e protocolos relativos ao RFC959 sobre FTP.

As redes ficaram bem configuradas e as experiências foram bem sucedidas, logo pode-se concluir que os objectivos do projecto foram atingidos com sucesso.

De notar que o grupo inicialmente era constituído por 4 elementos. 2 deles abandonaram a cadeira, logo o trabalho recaiu sobre os 2 elementos restantes.

6. Contribuição

José Carlos Alves Vieira - 50%
Renato Sampaio de Abreu - 50%

7. Anexo I

Arquivo .zip contendo o código fonte e arquivo .zip com os logs capturados nas 6 experiências realizadas.

8. Anexo II

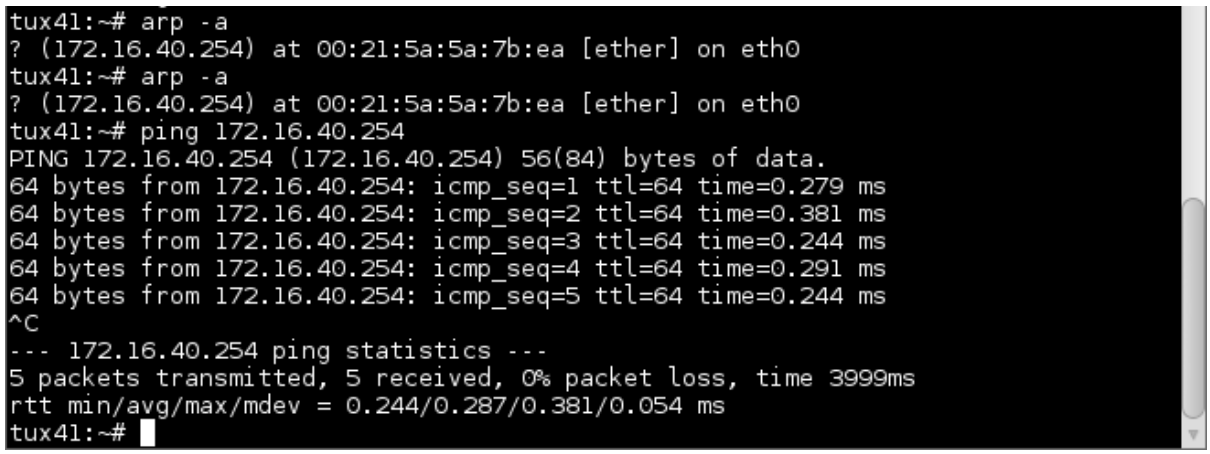

Figura 6 - Endereços MAC e IP

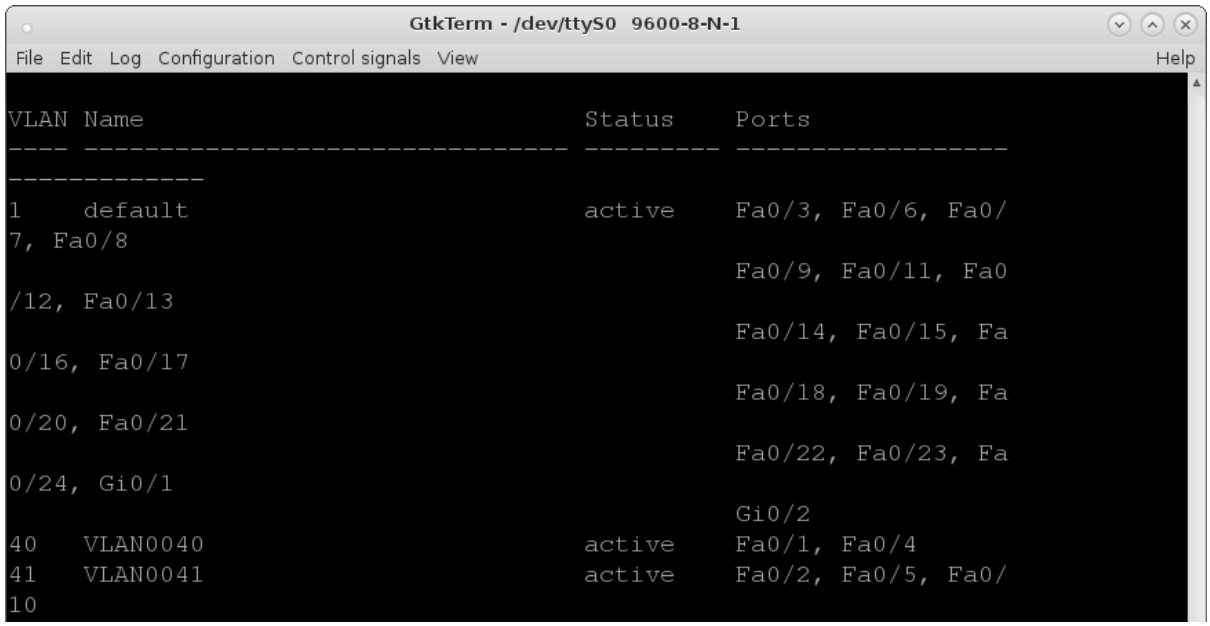

Figura 7 - VLAN 40 e 41

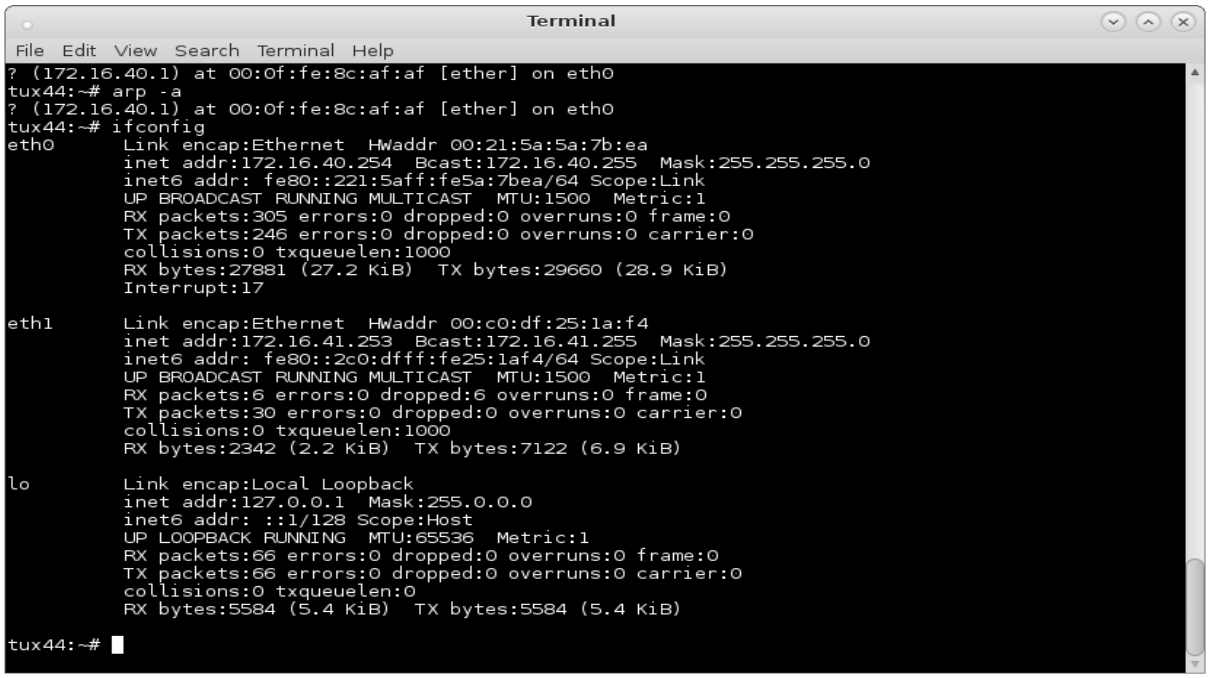

Figura 8 - Interfaces Ethernet 0 e 1

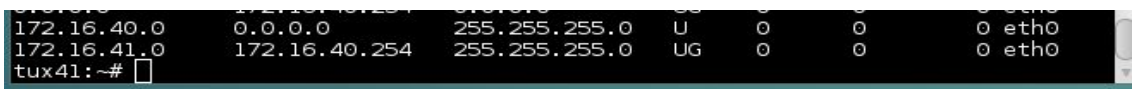

Figura 9 - Rotas Tux41, experiência 3

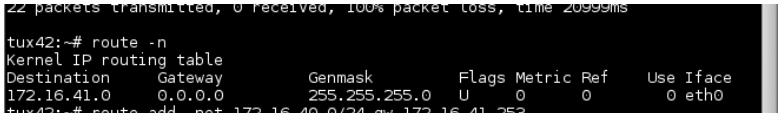

Figura 10 - Rotas Tux42, experiência 3

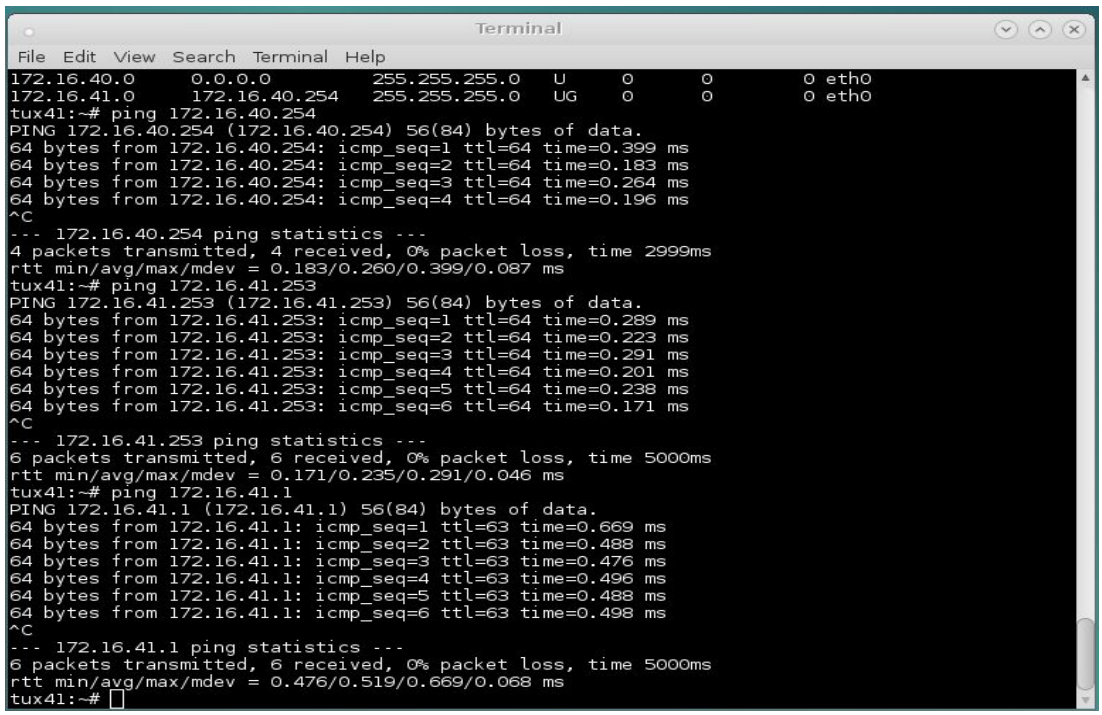

Figura 11 - Experiência 3, ping a todos as subredes