Trabalho Prático Nº.1 – Protocolos da Camada de Transporte

Duração: 1 aula

Este trabalho deve ser realizado com recurso à máquina virtual **XubunCORE_7_5** que está disponibilizada em http://marco.uminho.pt/ferramentas/CORE/xubuncore.html (user: *core* password: *core*)

Nota: *Não siga o guião cegamente, com copy/paste dos comandos, sem se interrogar do que está a fazer! Tente perceber os comandos, e o que é suposto fazerem, antes de os fazer! Não perca os objetivos de vista!*

Relatório

O relatório final do TP1 deve incluir:

- Uma secção de "Questões e Respostas" que dê resposta adequada às questões enumeradas no enunciado, incluindo para cada questão: a questão, a resposta e a prova da realização da mesma (se aplicável);
- Uma secção de "Conclusões" que autoavalie os resultados da aprendizagem decorrentes das várias vertentes estudadas no trabalho.

Submissão

O relatório em formato livre deve ser submetido na plataforma de ensino https://elearning.uminho.pt, usando a funcionalidade de transferência de ficheiros do grupo, com o nome CC-TP1-PL*<Turno>*G*<Grupo>*.pdf (por exemplo, CC-TP1-PL1-G1.pdf para o grupo 1 do PL1) no final do dia da aula prevista para a conclusão do trabalho.

Parte I: Instalação, configuração e utilização de serviços de transferência de ficheiros

Para este exercício recomenda-se que utilize a seguinte topologia `CC-Topo-2022.imn` que se apresenta na Figura 1 - Topologia de Rede (backbone, acesso e local) e que pode descarregar da plataforma de *elearning (http://elearning.uminho.pt)* ou diretamente de http://marco.uminho.pt/disciplinas/CC-LEI/CC-Topo-2022.imn

$ *wget http://marco.uminho.pt/disciplinas/CC-LEI/CC-Topo-2022.imn*

Neste exercício pretende-se transferir o mesmo ficheiro usando 4 serviços diferentes: SFTP, FTP, TFTP e HTTP, capturando todos os pacotes trocados durante a transferência com o Wireshark. Para isso será necessário realizar os seguintes passos:

- **[Xubuncore Linux]:** verificar se o software servidor está instalado; instalar se necessário;
- **[Xubuncore Linux]:** preparar uma pasta com os ficheiros a transferir; um ficheiro de texto e um ficheiro binário;
- **[Xubuncore Linux]:** executar o core com a topologia virtual;
- **[Topologia virtual, Servidor1]:** configurar servidores de modo a darem acesso a essa pasta;
- **[Topologia virtual, Portatil1]:** usar o software cliente respectivo para transferir os ficheiros disponibilizados.

Uma vez concluídas as transferências, deve-se concluir sobre as várias aplicações e protocolos usados.

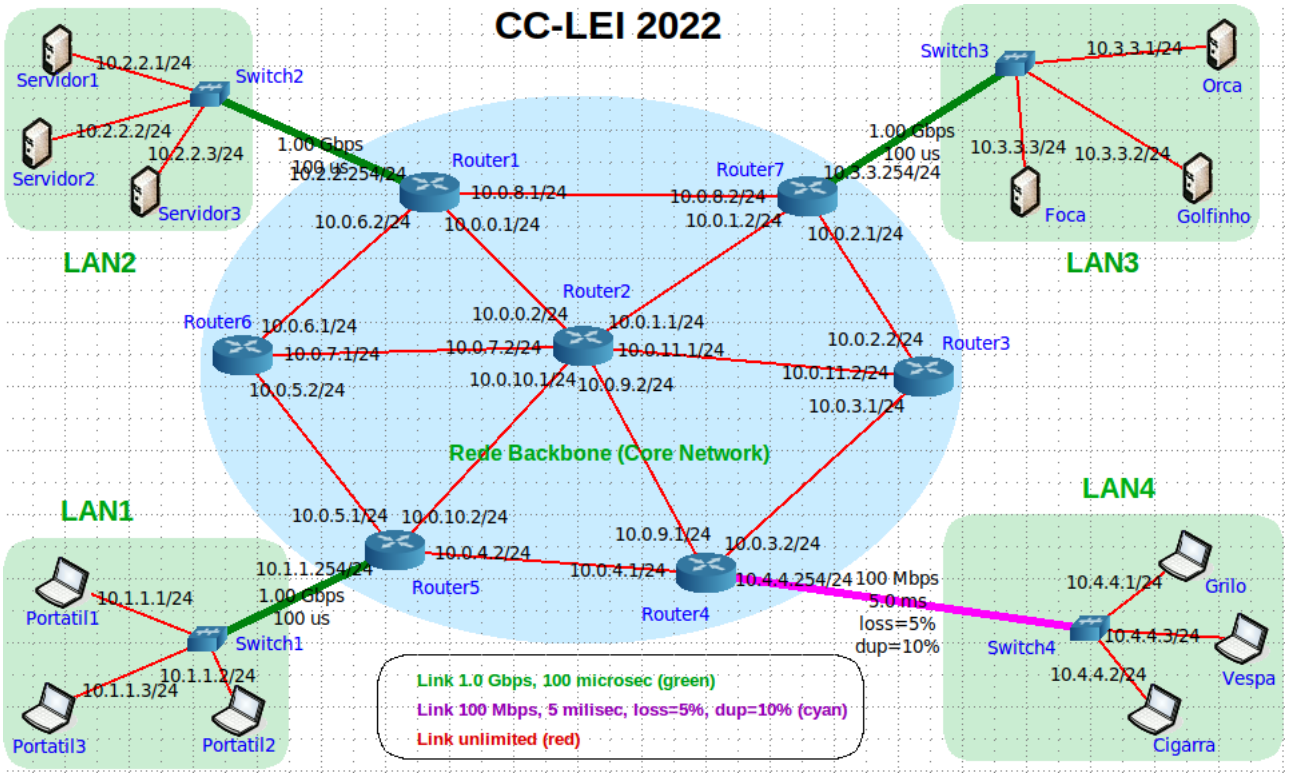

Figura 1 - Topologia de Rede (backbone, acesso e local)

ETAPA 1: verificar se o software (cliente e servidor) está instalado e instalar se necessário *[Máquina XubunCORE Linux (host principal), usando linha de comando (bash), user core, password core]*	
Comandos	Observações
`$ sudo apt install openssh-server` `$ sudo apt install openssh-client`	O software SSH já deve estar instalado de raiz no Linux e o serviço SSH/SFTP já está configurado e ativo por omissão em todas as topologias virtuais criadas pelo CORE; Não deverá ser necessário fazer nada de especial para usar SSH.
`$ sudo apt install vsftpd`	Existem vários packages de software FTP para Linux. Neste exercício sugere-se a utilização do "vsftpd". O cliente ftp já existe no Linux e não é preciso instalar.
`$ sudo apt install atftpd` `$ sudo apt install atftp`	Para software servidor TFTP propõe-se o uso do "atftpd", que é um servidor tftp avançado e também do respetivo cliente "atftp". Não existe nenhum servidor ou cliente pré-instalado.

`$ sudo apt install mini-httpd`
`$ sudo apt install wget`
`$ sudo apt install lynx`
	O software "mini-httpd" foi escolhido por ser um servidor web simples e que usa poucos recursos. Já o "lynx" e o "wget" são clientes Web, para consola, muito usados e poderosos!

`$ sudo dpkg --status openssh-server`
`$ sudo dpkg --status vsftpd`
`$ sudo dpkg --status atftpd`
`$ sudo dpkg --status atftp`
`$ sudo dpkg --status mini-httpd`
…
	Opcional. O comando "*dpkg*" é o gestor de pacotes Debian. Neste caso está a ser usado para verificar o estado dos pacotes que instalámos previamente.

Uma vez instalado o software necessário, e ainda antes de iniciar o core com a topologia virtual, recomenda-se que prepare uma única pasta com dois ficheiros (um ficheiro de texto e um ficheiro binário) para serem disponibilizados pelos vários servidores (SFTP, FTP, TFTP e HTTP). A pasta estará acessível e visível em todos os nós da topologia virtual, pois todos partilham exatamente o mesmo sistema de ficheiros!

ETAPA 2: preparar uma pasta com os ficheiros a transferir; um ficheiro de texto e um ficheiro binário; *[Máquina XubunCORELinux (host principal), usando linha de comando (bash), user core, password core]*	
Comandos	Observações

`$ sudo mkdir -p /srv/ftp`
 	
`$ sudo usermod -d /srv/ftp ftp`
 	
`$ sudo cp /etc/hosts /srv/ftp/file1`
 	
`$ sudo cp /bin/ls /srv/ftp/file2`
	O servidor FTP instala um novo utilizador no sistema com *username* "ftp" sem password para poder servir ficheiros da *home* desse utilizador de forma anónima a qualquer cliente FTP. A pasta a criar chama-se *"/srv/ftp".* O comando *mkdir* criará a pasta se ela não existir (e todas as incluídas no path que forem necessárias – opção "-p"). O comando *usermod* faz dela a "*home*" do user "*ftp*".
	Depois são copiados para lá dois ficheiros: o *"/etc/hosts"* que é um ficheiro de texto pequeno e que vai ser o "*file1*" e o ficheiro executável *"/bin/ls"* que será o ficheiro binário (executável) "*file2*". Pode optar por colocar ou editar outros ficheiros nessa pasta. Tudo o que estiver lá ficará acessível.

Podemos agora emular a topologia do ficheiro `CC-Topo-2022.imn` no *core*.

ETAPA 3: executar o *core* com a topologia virtual `CC-Topo-2022.imn`;	
Comandos	Observações
`$ wget http://marco.uminho.pt/disciplinas/CC-LEI/CC-Topo-2022.imn`	Obtem o ficheiro "`CC-Topo-2022.imn`" da página da disciplina com o comando de linha "*wget*".
`$ sudo systemctl start core-daemon.service` `$ sudo core-gui CC-Topo-2022.imn`	Executa o *core* com a referida topologia. O daemon já deve estar previamente ativado, no entanto o primeiro comando inicia o *daemon* do *core.* O segundo comando lança a interface gráfica (cliente *core-gui*) com poderes de *root*.
Topologia virtual, `Servidor1`, botão do rato do lado direito, *Shell Windows → bash*.	Obter uma *bash shell* no nó `Servidor1`
Topologia virtual, `Portatil1`, botão do rato do lado direito, *Shell Windows → bash*.	Obter uma *bash shell* no nó `Portatil1`
Topologia virtual, `Grilo`, botão do rato do lado direito, *Shell Windows → bash*.	Obter uma *bash shell* no nó `Grilo`
Topologia virtual, `Router1`, botão do rato do lado direito, *Wireshark → Eth2*.	Lançar um processo *wireshark* no `Router1` para capturar todos os pacotes que passam pelo seu interface `eth2` (certifique-se de que esse é o local certo para a captura!).

São três os objetivos fundamentais:

i) testar a conectividade e analisar as características gerais dos *links* (ligações com diferentes larguras de banda e diferentes atrasos) utilizando o comando "*ping"* e/ou "*traceroute*";

ii) depois transferir os ficheiros `file1` e/ou `file2` que colocámos na pasta `/srv/ftp` (partilhada em todos os nós da topologia), inicialmente para o cliente `Portatil1`, capturando a transferência com o *wireshark* no router `Router1`;

iii) comparar os tempos de transferência do ficheiro `file2` para o cliente `Portatil1` e para o cliente `Grilo`.

2.1 PING

O objetivo é testar a conectividade (mais do que 90 segundos após o *run* inicial da topologia) através do comando *ping* ao `Servidor1`. O comando `bash` *"ping –c 20 10.2.2.1 | tee file-ping-output"* realizado no `Portatil1` e também no cliente `Grilo`. Os resultados da execução são armazenados em ficheiros com o nome *"file-ping-output"* nos nós `Portatil1` e `Grilo`.

Topologia Virtual, Portatil1 (ping)

```
root@Portatil1$ ping –c 20 10.2.2.1 | tee file-ping-output
```
Nota: … o resultado irá ficar também armazenado em "file-ping-output" …

```
root@Portatil1$ less file-ping-output
```

REPETIR o *ping,* na Topologia Virtual, a partir do nó *Grilo* na LAN4. Observe as perdas e duplicações de pacotes.

2.2 SFTP

O servidor SSH já deve estar em execução no servidor `Servidor1`, e podemos verificar isso com os comandos "ps" e "netstat".

Topologia Virtual, Servidor1 (servidor SFTP)

```
root@Servidor1$ ps –ef | grep ssh
```
… verificar se existe um processo sshd em execução…

```
root@Servidor1$ netstat –n –a
```
… verificar se está alguém à escuta na porta 22…

NOTA: Estes comandos servem apenas para termos a certeza de que o servidor SSH está ativo (ver porta 22)!

Desta forma podemos transferir o ficheiro por **sftp**, a partir da *bash* do `Portatil1`

Topologia Virtual, Portatil1 (cliente SFTP)

```
root@Portatil1$ rm /root/.ssh/known_hosts
```
Nota: … a identidade dos nós da topologia virtual está sempre a mudar… pelo que é preciso apagar a lista de hosts bem conhecidos do SSH

```
root@Portatil1$ sftp core@10.2.2.1
```
Nota: … não se esqueça que a password do utiliazdor "core" usado no exemplo é "core" … se usar outro user terá que usar outra password…

```
sftp> pwd
sftp> cd /srv/ftp
sftp> dir
sftp> get file1
sftp> quit
```

REPETIR a transferência por **sftp** a partir do nó *Grilo* na LAN4 da Topologia Virtual, observando as diferenças.

2.3 FTP

Para transferir por **ftp** é necessário executar o servidor manualmente na *bash* do `Servidor1`.

Topologia Virtual, Servidor1 (servidor FTP)

```
root@Servidor1$ chmod a-w /srv/ftp
```
... directoria não pode ter acesso para escrita... por questões de segurança...

```
root@Servidor1$ vsftpd /etc/vsftpd.conf -osecure_chroot_dir=/srv/ftp -oanonymous_enable=YES
```
... este comando vai manter-se em execução no terminal sem passar para background...
... pode ser enviado para background com um "control-Z", que suspende o processo, seguido do comando "bg", que o envia para background
... para mais informações sobre os parâmetros escrever "man vsftpd" ou "man vsftpd.conf" num terminal

NOTA: Estes comandos servem para configurar e ativar manualmente o servidor FTP na linha de comando!

E depois transferir o ficheiro a partir do `Portatil1`.

Topologia Virtual, Portatil1 (cliente FTP)

```
root@Portatil1$ ftp 10.2.2.1
```
... entrar com username anonymous e qualquer password (aconselha-se o e-mail)
```
ftp> status
ftp> pwd
ftp> dir
ftp> get file1
ftp> quit
```

REPETIR a transferência por **ftp** a partir do nó *Grilo* na LAN4 da Topologia Virtual, observando as diferenças.

2.4 TFTP

Para transferir por **tftp** é necessário preparar a directoria e executar o servidor manualmente na *bash* do `Servidor1`.

Topologia Virtual, Servidor1 (servidor TFTP)

```
root@Servidor1$ chmod –R 777 /srv/ftp
```
... directoria tem de ter acesso para escrita para todos...
```
root@Servidor1$ touch atftpd.log
```
... se houver problemas podemos ver neste ficheiro de log o que se passou...
```
root@Servidor1$ atftpd --verbose=3 --user root.ftp --logfile atftpd.log \
                       --bind-address 10.2.2.1 --daemon --no-fork /srv/ftp/
```
... atenção que a barra \ serve para continuar o comando noutra linha e não é necessária se escrever tudo na mesma linha ...
... este comando vai manter-se em execução no terminal sem passar para background...
... pode ser enviado para background com um "control-Z", que suspende o processo, seguido do comando "bg", que o envia para background
... para mais informações sobre os parâmetros escrever "man atftpd" num terminal

NOTA: Estes **comandos** servem para configurar e ativar manualmente o servidor TFTP na linha de comando!

E depois transferir o ficheiro a partir do `Portatil1`:

Topologia Virtual, Portatil1 (cliente TFTP)

`root@Portatil1$ atftp 10.2.2.1`
`ftp> status`
`ftp> get file1`
`ftp> quit`

REPETIR a transferência por **tftp** a partir do nó *Grilo* na LAN4 da Topologia Virtual, observando as diferenças.

2.5 HTTP

Para transferir por **http** é necessário preparar a diretoria e executar o servidor manualmente na *bash* do `Servidor1`.

Topologia Virtual, Servidor1 (servidor HTTP)

`root@Servidor1$ mini_httpd -d /srv/ftp/`
…
`root@Servidor1$ ps –ef`
… para verificar se o daemon ficou em execução…

NOTA: Estes **comandos** servem configurar e ativar manualmente o servidor HTTP, confirmando que ficou em execução.

E depois transferir o ficheiro a partir do `Portatil1`.

Topologia Virtual, Portatil1 (cliente HTTP)

`root@Portatil1$ wget http://10.2.2.1/file1`
`root@Portatil1$ wget http://10.2.2.1/file2`
…
Ou com o comando lynx:
`root@Portatil1$ lynx http://10.2.2.1/file1`
`root@Portatil1$ lynx http://10.2.2.1/file2`

REPETIR a transferência por **http** (wget) a partir do nó *Grilo* na LAN4 da Topologia Virtual, observando as diferenças.

QUESTÕES (Parte I)

1. De que forma as perdas e duplicações de pacotes afetaram o desempenho das aplicações? Que camada lidou com as perdas e duplicações: transporte ou aplicação? Responda com base nas experiências feitas e nos resultados observados.
2. Obtenha a partir do *wireshark*, ou desenhe manualmente, um diagrama temporal para a transferência de *file1* por FTP. Foque-se apenas na transferência de dados [ftp-data] e não na conexão de controlo, pois o FTP usa mais que uma conexão em simultâneo. Identifique, se aplicável, as fases de início de conexão, transferência de dados e fim de conexão. Identifique também os tipos de segmentos trocados e os números de sequência usados quer nos dados como nas confirmações.
3. Obtenha a partir do *wireshark*, ou desenhe manualmente, um diagrama temporal para a transferência de *file1* por TFTP. Identifique, se aplicável, as fases de início de conexão, transferência de dados e fim de conexão. Identifique também os tipos de segmentos trocados e os números de sequência usados quer nos dados como nas confirmações.
4. Compare sucintamente as quatro aplicações de transferência de ficheiros que usou nos seguintes pontos (i) uso da camada de transporte; (ii) eficiência; (iii) complexidade; (iv) segurança;

Parte II: Uso da camada de transporte por parte das aplicações

Numa linha de comando da máquina virtual **XubunCORE**, mas fora do emulador, sem ativar o **core-gui**, execute:

```
$ sudo wireshark
```

Capture o tráfego em determinados instantes que considere adequados, observe atentamente como as várias aplicações utilizam os serviços da camada inferior:

- Acesso via browser ao URL: http://marco.uminho.pt/disciplinas/CC-LEI/ (nota: pode usar comandos *wget* ou *lynx*)
- Acesso em *ftp* para cc2022.ddns.net (Username: *cc* Password: *cc2022*)
- Acesso em *tftp* para cc2022.ddns.net (usando *tftp* ou *curl*; por exemplo: *curl tftp://cc2022.ddns.net/file1*)
- Acesso via *telnet* para cc2022.ddns.net (Username: *cc* Password: *cc2022*) ou para *router-ext* (193.136.9.33)
- Acesso *ssh* para cc2022.ddns.net (Username: *cc* Password: *cc2022, ssh cc@cc2022.ddns.net*)
- Resolução de nomes usando *nslookup* www.uminho.pt
- *ping* www.google.pt
- *traceroute* cisco.di.uminho.pt ou www.fccn.pt

... e outras aplicações Internet bem conhecidas que considere importantes e que possa explorar!

QUESTÕES (Parte II)

1. Com base na captura de pacotes feita, preencha a seguinte tabela, identificando para cada aplicação executada, qual o protocolo de aplicação, o protocolo de transporte, porta de atendimento e *overhead* de transporte.

Comando usado (aplicação)	**Protocolo de Aplicação (se aplicável)**	**Protocolo de transporte (se aplicável)**	**Porta de atendimento (se aplicável)**	***Overhead* de transporte em bytes (se aplicável)**
`ping`				
`traceroute`				
`telnet`				
`ftp`				
`tftp`				
`http(browser)`				
`nslookup`				
`ssh`				
Outras:				