Comunicações por Computadores
Trabalho Prático 1

Trabalho realizado por:

Francisco Pinto Lameirão
Luís Miguel Moreira Ferreira
Pedro Dantas da Cunha Pereira

PL 3 - Grupo 8

A97504
Francisco Lameirão

A95111
Luís Ferreira

A97396
Pedro Dantas

Sistemas Operativos

Índice

1 Exercícios Parte 1

1.1 Exercício 1

Enunciado: 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.

Resposta: Neste caso de estudo, quando ocorrem perdas de pacotes e existe uma tentativa da recuperação dos mesmos, acaba por se verificar o envio de pacotes duplicados que, em torno, contribuem para um intervalo de tempo de envio mais elevado, visto que o próprio número total de pacotes também aumenta. Como consequência, o desempenho das aplicações acaba por piorar, visto que o seu tempo de execução total será mais elevado do que o normal.

```
PING 10.4.4.1 (10.4.4.1) 56(84) bytes of data.
64 bytes from 10.4.4.1: icmp_seq=1 ttl=61 time=1.43 ms
64 bytes from 10.4.4.1: icmp_seq=2 ttl=61 time=0.471 ms
64 bytes from 10.4.4.1: icmp_seq=3 ttl=61 time=2.98 ms
64 bytes from 10.4.4.1: icmp_seq=4 ttl=61 time=0.623 ms
64 bytes from 10.4.4.1: icmp_seq=5 ttl=61 time=3.41 ms
64 bytes from 10.4.4.1: icmp_seq=6 ttl=61 time=1.06 ms
64 bytes from 10.4.4.1: icmp_seq=7 ttl=61 time=0.810 ms
64 bytes from 10.4.4.1: icmp_seq=8 ttl=61 time=0.526 ms
64 bytes from 10.4.4.1: icmp_seq=9 ttl=61 time=3.46 ms
64 bytes from 10.4.4.1: icmp_seq=10 ttl=61 time=0.691 ms
64 bytes from 10.4.4.1: icmp_seq=11 ttl=61 time=2.83 ms
64 bytes from 10.4.4.1: icmp_seq=12 ttl=61 time=0.668 ms
64 bytes from 10.4.4.1: icmp_seq=13 ttl=61 time=2.70 ms
64 bytes from 10.4.4.1: icmp_seq=14 ttl=61 time=0.508 ms
64 bytes from 10.4.4.1: icmp_seq=15 ttl=61 time=3.19 ms
64 bytes from 10.4.4.1: icmp_seq=16 ttl=61 time=1.05 ms
64 bytes from 10.4.4.1: icmp_seq=17 ttl=61 time=0.811 ms
64 bytes from 10.4.4.1: icmp_seq=18 ttl=61 time=0.690 ms
64 bytes from 10.4.4.1: icmp_seq=19 ttl=61 time=3.19 ms
64 bytes from 10.4.4.1: icmp_seq=20 ttl=61 time=0.562 ms

--- 10.4.4.1 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19152ms
rtt min/avg/max/mdev = 0.471/1.583/3.463/1.150 ms
~
```

Figure 1: Ping Portatil1

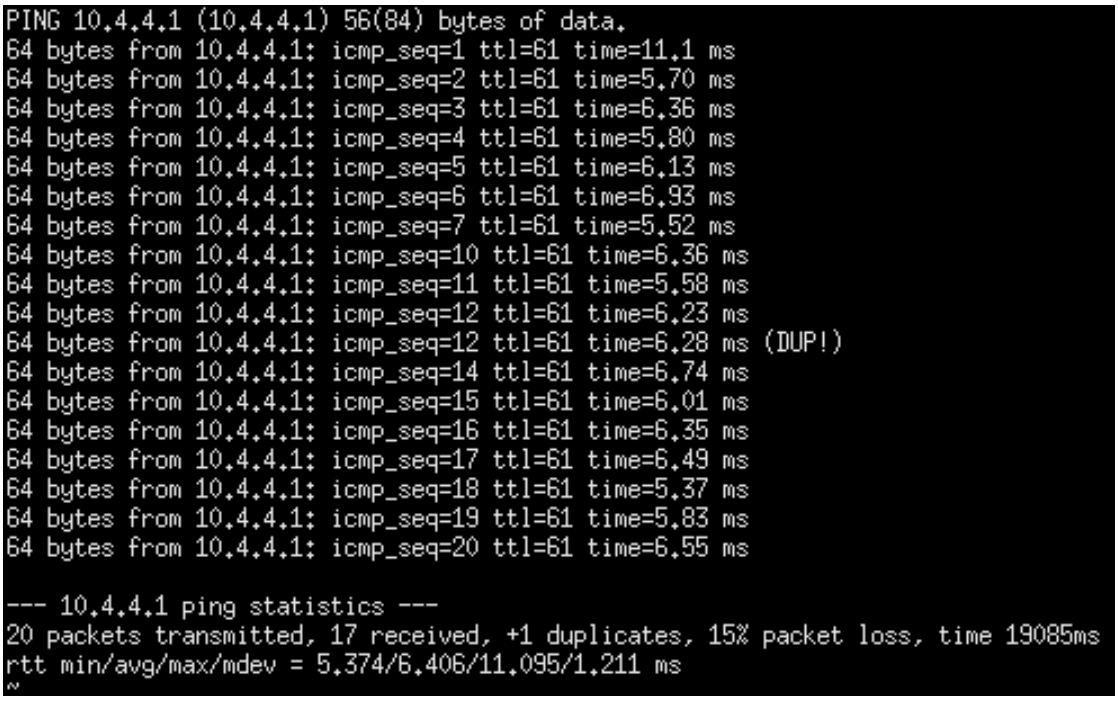


Figure 2: Ping PC1

Em relação à camada à qual será atribuída a tarefa de lidar com as perdas e duplicações dos pacotes, podemos dizer que esta estará dependente do tipo de protocolo de transporte a ser utilizado.

UDP: No caso de ser escolhido um protocolo no qual se utilize um formato UDP, será a camada de aplicação a responsável pelos problemas de perda e duplicação de pacotes. Isto porque o formato em questão considera como fator principal a velocidade à qual os pacotes são enviados, ignorando possíveis erros nos mesmos, sendo que, caso exista de facto algum erro, o pacote em questão é apenas descartado.

TCP: Caso seja utilizado o protocolo TCP, a camada responsável pelos problemas referidos será a camada de transporte, visto que, ao contrário do formato UDP, este dá prioridade à deteção e correção de erros em relação à velocidade à qual os pacotes são enviados. Quando é encontrado algum tipo de erro, dá-se uma tentativa de retransmissão dos pacotes.

1.2 Exercício 2

Enunciado: 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.

Resposta:

FTP: O protocolo FTP poderá ser executado em modo passivo ou modo ativo, sendo a escolha entre estes dois modos feita com base em quem inicia a conexão entre cliente e servidor. Neste protocolo, é utilizado um par de conexões entre servidor e cliente, sendo que a primeira conexão a ser criada é a conexão com a porta 21 do servidor e a segunda

com a porta 20 do servidor. Após a abertura da porta 21 dos servidores é iniciada a escuta das conexões de entrada do cliente. Os clientes realizam a conexão com a porta 21 dos servidores remotos, de modo a iniciar as transferências de ficheiros. A porta 20 é necessária para garantir que estas transferências são, de facto, realizadas.

Modo Passivo: Caso seja o cliente a iniciar a conexão de dados, o protocolo FTP será executado em modo passivo. Isto significa que apenas o servidor será necessário portas para o tráfego de dados. De modo a evitar problemas de segurança, a maioria dos servidores utilizam uma conexão FTP passiva.

Modo Ativo: No caso da conexão de dados ser iniciada pelo servidor, a conexão FTP será executada em modo ativo. Isto traduz-se na necessidade de tanto o servidor como o cliente abrirem portas de modo a receber tráfego.

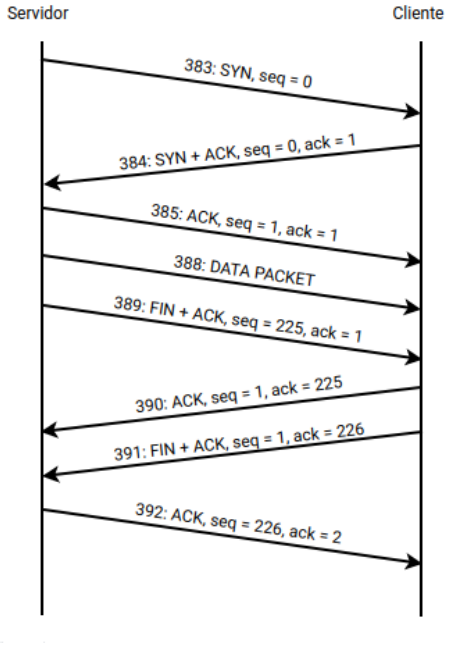


Figure 3: Diagrama Temporal FTP

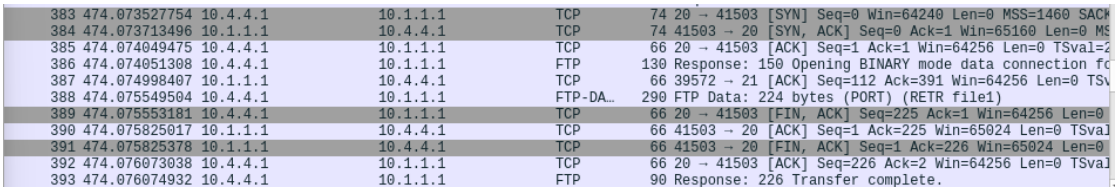


Figure 4:

Tal como podemos verificar nas imagens apresentadas anteriormente, a conexão inicia-se com o envio de um segmento TCP SYN por parte do servidor, indicando quais os números de sequência a ser sincronizados de modo a iniciar a conexão. Assim que recebido pelo cliente, é dada uma resposta pelo mesmo, através do envio de um segmento TCP SYN e uma trama ACK de forma a confirmar a receção do segmento anteriormente enviado pelo servidor. Seguidamente, o servidor envia também uma trama ACK de modo a confirmar a receção do SYN enviado pelo cliente.

Após o estabelecimento e a verificação desta conexão, é então inicializada a fase de transmissão, sendo que esta começa com o envio dos dados pretendidos por parte do servidor.

Finalmente, passamos para a fase de ... que começa com o envio de um segmento TCP FIN, de modo a informar o cliente que todos os dados foram enviados. Como resposta, o cliente envia uma trama ACK e um segmento TCP FIN com uma trama ACK adicional, referente ao TCP FIN enviado pelo servidor anteriormente. Finalmente, é enviada uma última trama ACK pelo servidor, indicando o sucesso na receção do segmento TCP FIN enviado pelo cliente.

1.3 Exercício 3

Enunciado: 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.

Resposta:

Como é possível verificar nas imagens abaixo apresentadas, inicialmente é enviado um Read Request ao servidor por parte do cliente. Após isto, o servidor envia um pacote no qual estão contidos os dados que pretende enviar. Finalmente, o cliente envia uma trama ACK de modo a confirmar o sucesso da receção dos dados.

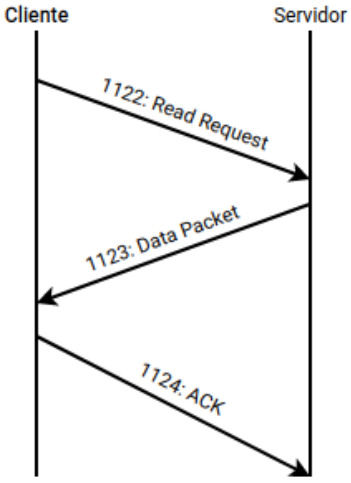


Figure 5: Diagrama Temporal TFTP

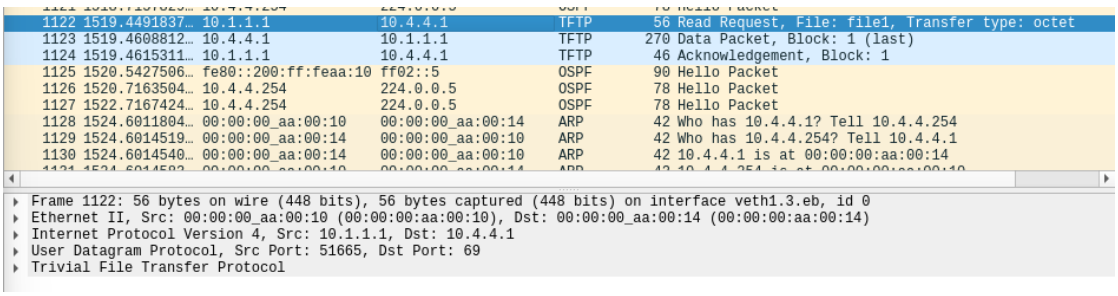


Figure 6: Captura TFTP

No protocolo TFTP, todas as transferências são iniciadas através de um pedido de leitura de um ficheiro (ou de escrita no mesmo), sendo que este mesmo pedido será também utilizado como pedido de conexão. Caso o pedido seja aceite por parte do servidor, é estabelecida a conexão entre servidor e cliente e o ficheiro será transmitido através de blocos com 512 bytes, sendo este um limite de tamanho fixo. Na eventualidade de o tamanho do pacote ser superior a 512 bytes, este sofrerá fragmentação, sendo então enviados vários pacotes com o mesmo limite de tamanho.

1.4 Exercício 4

Enunciado: 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;

Resposta:

(i) Uso da Camada de Transporte

- **SFTP** - Protocolo TCP;
- **FTP** - Protocolo TCP;
- **TFTP** - Protocolo UDP;
- **HTTP** - Protocolo TCP;

(ii) Eficiência

- **SFTP** - Semelhante ao FTP, mas como tem os dados encriptados é mais segura;
- **FTP** - O protocolo FTP utiliza *acknowldges* de modo a confirmar e garantir a transmissão dos segmentos. No entanto, o facto de ter de esperar pelos referidos *acknowledges* para continuar acaba por baixar o nível de eficiência deste protocolo.
- **TFTP** - Visto que o TFTP usa o protocolo UDP, é considerado menos viável, tendo em conta que, sem o uso de *acknowledges*, é impossível confirmar a receção dos segmentos enviados. Caso seja bem sucedido, é mais rápido que o protocolo FTP, no entanto, a maioria das vezes existe a necessidade de retransmissão dos pacotes múltiplas vezes.
- **HTTP** - Permite que vários HTTP requests sejam enviados numa única ligação TCP sem ser preciso esperar pelas respostas individuais de cada um.

(iii) Complexidade

- **SFTP** - Este protocolo permite gestão, acesso e transferência de dados. Estas funcionalidades têm todas custos de processamento elevados, pelo que o protocolo é considerado bastante complexo.
- **FTP** - O protocolo FTP suporta a possibilidade de existirem vários pedidos de transferência em paralelo, nos quais se realiza uma nova conexão de dados em cada transferência. Isto leva a que existam diferentes velocidades de transferência. Tendo em conta a frequência com a qual estas novas conexões são criadas, o protocolo FTP é considerado bastante complexo.

- **TFTP** - É uma alternativa do protocolo FTP, sendo uma versão mais "trivial" (tal como indicado no nome) e simples. Posto isto, é também um protocolo com menos funcionalidades que o FTP, para além de utilizar o protocolo UDP ao invés do protocolo TCP. Com todos estes fatores em mente, podemos afirmar que é um protocolo com um nível de complexidade baixo.

- **HTTP** - Implementa uma garantia de confiança, escalabilidade e encerramento de sistemas. Por ter todas estas capacidades assumimos que se trata de um protocolo bastante complexo.

(iv) Segurança

- **SFTP** - Sendo que este protocolo usa SSH (*Secure Shell*), podemos afirmar que se trata de um protocolo bastante seguro. Isto porque o SSH utiliza uma arquitetura em camadas, nas quais a camada de transporte (com auxílio do protocolo TCP/IP) fornece encriptação, autenticação do servidor e proteção da integridade dos dados, e a camada de autenticação tem como objetivo gerir a autenticação dos clientes.

- **FTP** - Conhecido por falhas na segurança, pois não fornece encriptação de dados. Assim sendo, qualquer pessoa é capaz de capturar pacotes na rede e obter assim nomes de utilizadores, palavras-passe, etc. Isto faz com que o protocolo FTP seja considerado muito inseguro.

- **TFTP** - Não fornece autenticação. Assim sendo, não protege dados que se pretendam transferir, fazendo com que este não seja um protocolo seguro.

- **HTTP** - É um protocolo da camada de aplicação e é frequentemente usado para transferências de dados na Internet. Apesar disto, atendendo ao facto de que a informação é representada em texto e não é encriptada, os dados podem ser alterados.

2 Exercícios Parte 2

2.1 Exercício 1

Enunciado: Com base no trabalho realizado, tanto na parte I como na parte II, identifique para cada aplicação executada, qual o protocolo de aplicação, o protocolo de transporte, porta de atendimento e *overhead* de transporte.

Resposta:

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)
wget, lynx ou via browser	HTTP	TCP	80	20
ssh, sftp	SSH	TCP	22	20
ftp	FTP	TCP	21	20
Tftp	TFTP	UDP	69	8
telnet	Telnet	TCP	23	20
nslookup ou dig	DNS	UDP	53	8
Ping	Ping	-	-	-
Traceroute	Traceroute	UDP	33437	8
Outras:				

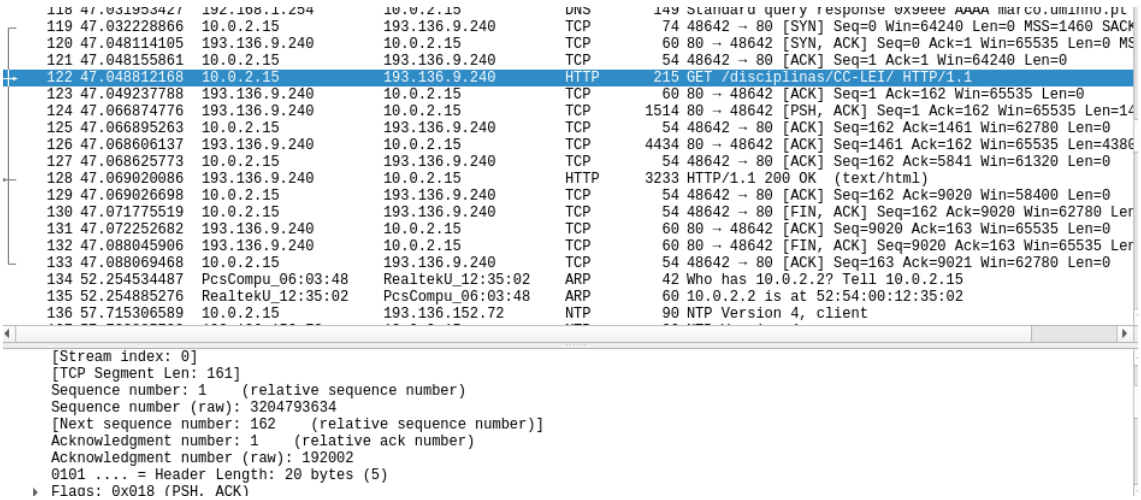


Figure 7: Captura Wget

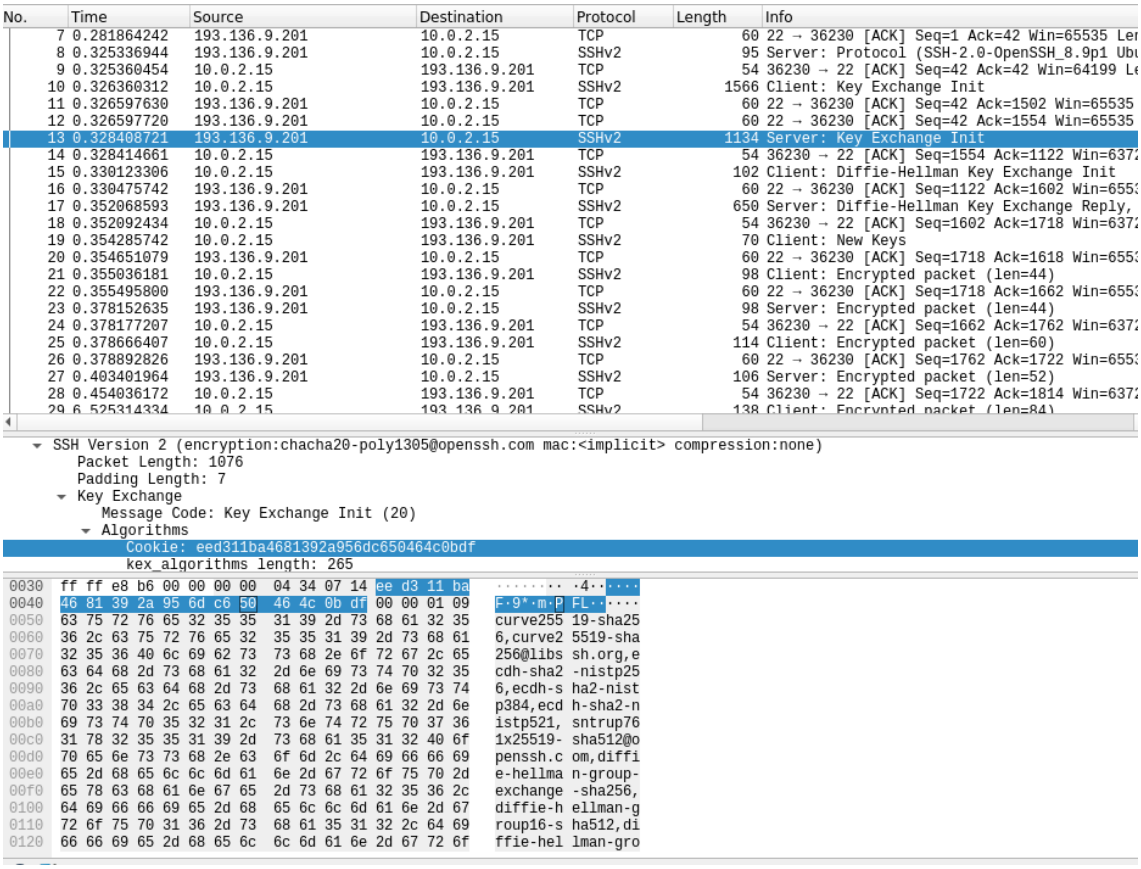


Figure 8: Captura SSH

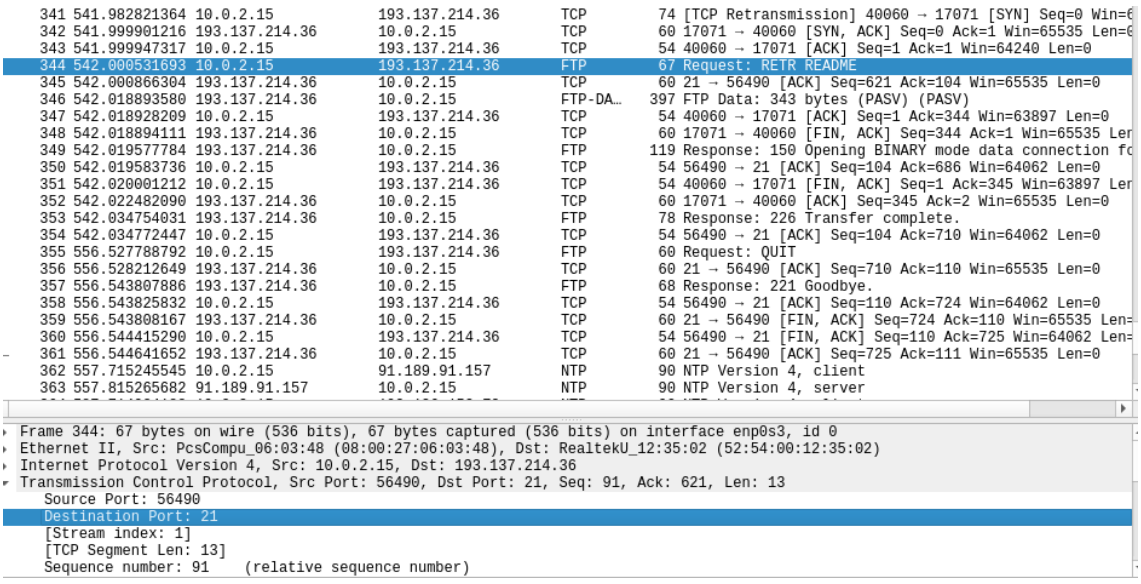
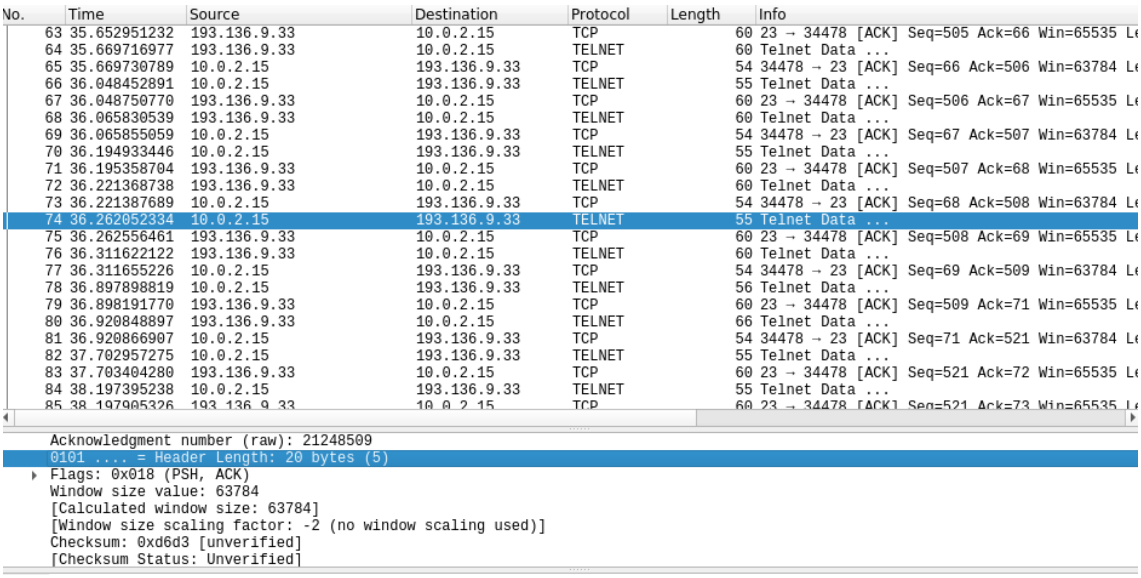


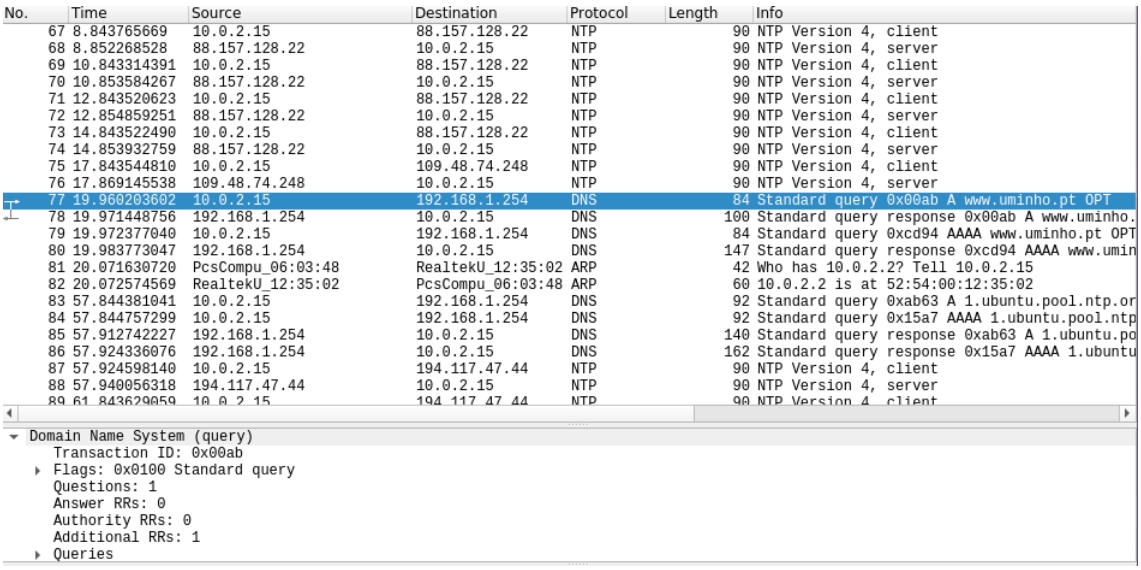
Figure 9: Captura FTP



No.	Time	Source	Destination	Protocol	Length	Info
63	35.652951232	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=505 Ack=66 Win=65535 Le
64	35.669716977	193.136.9.33	10.0.2.15	TELNET	60	Telnet Data ...
65	35.669730789	10.0.2.15	193.136.9.33	TCP	54	34478 → 23 [ACK] Seq=66 Ack=506 Win=63784 Le
66	36.048452891	10.0.2.15	193.136.9.33	TELNET	55	Telnet Data ...
67	36.048750770	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=506 Ack=67 Win=65535 Le
68	36.065830539	193.136.9.33	10.0.2.15	TELNET	60	Telnet Data ...
69	36.065855059	10.0.2.15	193.136.9.33	TCP	54	34478 → 23 [ACK] Seq=67 Ack=507 Win=63784 Le
70	36.194933446	10.0.2.15	193.136.9.33	TELNET	55	Telnet Data ...
71	36.195358704	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=507 Ack=68 Win=65535 Le
72	36.221368738	193.136.9.33	10.0.2.15	TELNET	60	Telnet Data ...
73	36.221387689	10.0.2.15	193.136.9.33	TCP	54	34478 → 23 [ACK] Seq=68 Ack=508 Win=63784 Le
74	36.262052334	10.0.2.15	193.136.9.33	TELNET	55	Telnet Data ...
75	36.262556461	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=508 Ack=69 Win=65535 Le
76	36.311622122	193.136.9.33	10.0.2.15	TELNET	60	Telnet Data ...
77	36.311655226	10.0.2.15	193.136.9.33	TCP	54	34478 → 23 [ACK] Seq=69 Ack=509 Win=63784 Le
78	36.897898819	10.0.2.15	193.136.9.33	TELNET	56	Telnet Data ...
79	36.898191770	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=509 Ack=71 Win=65535 Le
80	36.920848897	193.136.9.33	10.0.2.15	TELNET	66	Telnet Data ...
81	36.920866907	10.0.2.15	193.136.9.33	TCP	54	34478 → 23 [ACK] Seq=71 Ack=521 Win=63784 Le
82	37.702957275	10.0.2.15	193.136.9.33	TELNET	55	Telnet Data ...
83	37.703404280	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=521 Ack=72 Win=65535 Le
84	38.197395238	10.0.2.15	193.136.9.33	TELNET	55	Telnet Data ...
85	38.197905326	193.136.9.33	10.0.2.15	TCP	60	23 → 34478 [ACK] Seq=521 Ack=73 Win=65535 Le

```
Acknowledgment number (raw): 21248509
0101 .... = Header Length: 20 bytes (5)
▸ Flags: 0x018 (PSH, ACK)
Window size value: 63784
[Calculated window size: 63784]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0xd6d3 [unverified]
[Checksum Status: Unverified]
```

Figure 10: Captura Telnet



No.	Time	Source	Destination	Protocol	Length	Info
67	8.843765669	10.0.2.15	88.157.128.22	NTP	90	NTP Version 4, client
68	8.852268528	88.157.128.22	10.0.2.15	NTP	90	NTP Version 4, server
69	10.843314391	10.0.2.15	88.157.128.22	NTP	90	NTP Version 4, client
70	10.853584267	88.157.128.22	10.0.2.15	NTP	90	NTP Version 4, server
71	12.843520623	10.0.2.15	88.157.128.22	NTP	90	NTP Version 4, client
72	12.854859251	88.157.128.22	10.0.2.15	NTP	90	NTP Version 4, server
73	14.843522490	10.0.2.15	88.157.128.22	NTP	90	NTP Version 4, client
74	14.853932759	88.157.128.22	10.0.2.15	NTP	90	NTP Version 4, server
75	17.843544810	10.0.2.15	109.48.74.248	NTP	90	NTP Version 4, client
76	17.869145538	109.48.74.248	10.0.2.15	NTP	90	NTP Version 4, server
77	19.960203602	10.0.2.15	192.168.1.254	DNS	84	Standard query 0x00ab A www.uminho.pt OPT
78	19.971448756	192.168.1.254	10.0.2.15	DNS	100	Standard query response 0x00ab A www.uminho.
79	19.972377040	10.0.2.15	192.168.1.254	DNS	84	Standard query 0xcd94 AAAA www.uminho.pt OPT
80	19.983773047	192.168.1.254	10.0.2.15	DNS	147	Standard query response 0xcd94 AAAA www.umin
81	20.071630720	PcsCompu_06:03:48	RealtekU_12:35:02	ARP	42	Who has 10.0.2.2? Tell 10.0.2.15
82	20.072574569	RealtekU_12:35:02	PcsCompu_06:03:48	ARP	60	10.0.2.2 is at 52:54:00:12:35:02
83	57.844381041	10.0.2.15	192.168.1.254	DNS	92	Standard query 0xab63 A 1.ubuntu.pool.ntp.or
84	57.844757299	10.0.2.15	192.168.1.254	DNS	92	Standard query 0x15a7 AAAA 1.ubuntu.pool.ntp
85	57.912742227	192.168.1.254	10.0.2.15	DNS	140	Standard query response 0xab63 A 1.ubuntu.po
86	57.924336076	192.168.1.254	10.0.2.15	DNS	162	Standard query response 0x15a7 AAAA 1.ubuntu
87	57.924598140	10.0.2.15	194.117.47.44	NTP	90	NTP Version 4, client
88	57.940056318	194.117.47.44	10.0.2.15	NTP	90	NTP Version 4, server
89	61.843629059	10.0.2.15	194.117.47.44	NTP	90	NTP Version 4, client

```
▾ Domain Name System (query)
Transaction ID: 0x00ab
▸ Flags: 0x0100 Standard query
Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 1
▸ Queries
```

Figure 11: Captura NSLookUp

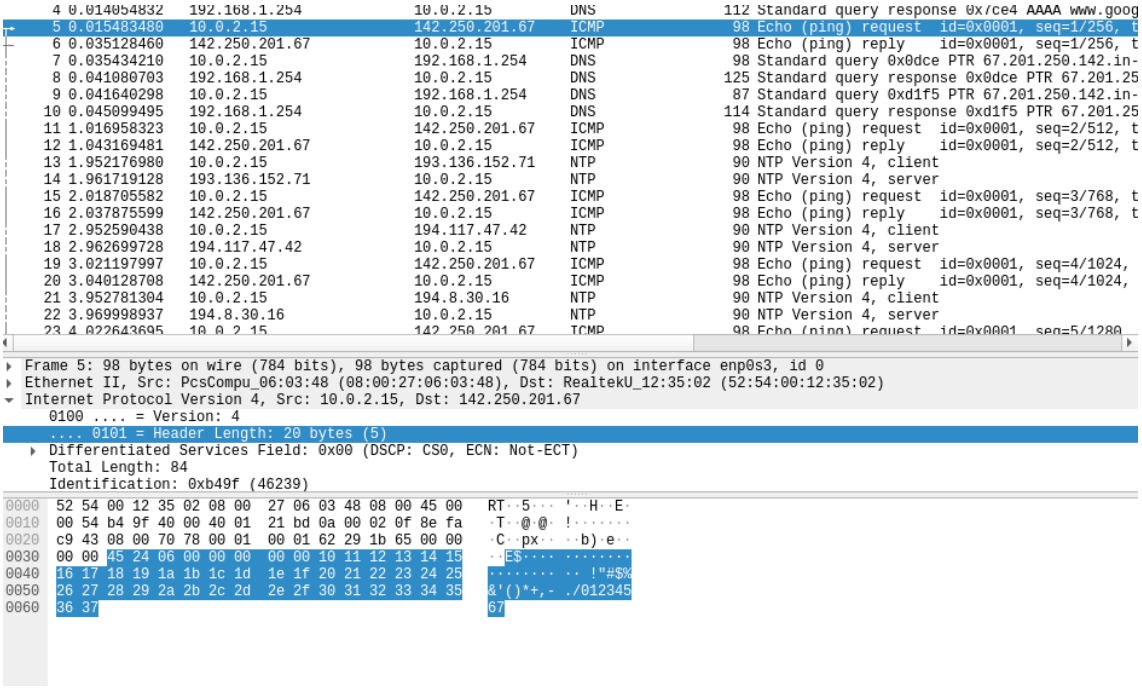


Figure 12: Captura Ping

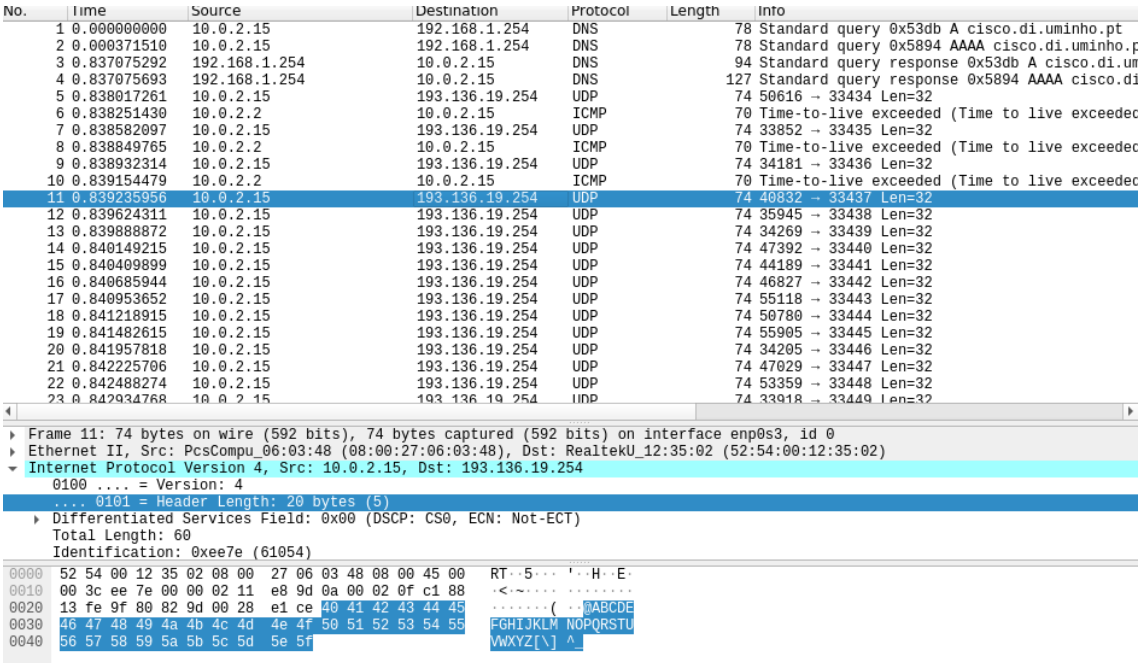


Figure 13: captura Traceroute

3 Conclusões

Após a realização e conclusão deste trabalho prático, é seguro dizer que foi um ótimo projeto para aprofundar conceitos relativos aos diferentes protocolos estudados, sendo estes referentes tanto à camada de transporte como à camada de aplicação, para além de obtermos uma melhor perceção das vantagens e desvantagens que cada um destes protocolos traz, o que, por consequência, nos ajuda a perceber quando devem ser utilizados. O facto de conseguirmos analisar estes conceitos em tempo real, através do uso das ferramentas aconselhadas, torna toda a experiência mais simples e percetível.

Desta forma, consideramos que o fizemos um bom trabalho e esperamos poder vir a utilizar e aprofundar todos os conceitos estudados no futuro.