

Universidade do Minho

Comunicação por Computador

TP1 – Protocolos da Camada de Transporte

MIEI-3º Ano- 2º semestre

Grupo 3

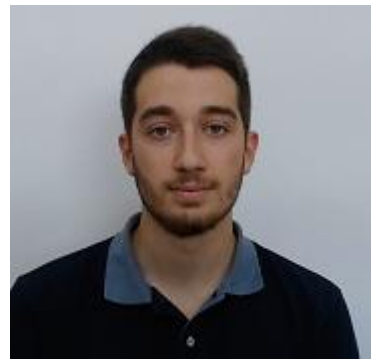
PL1



Catarina Gil
A85266



Margarida Campos
A85166



Filipe Oliveira
A80330

Questão 1

Inclua no relatório uma tabela em que identifique, para cada comando executado, qual o protocolo de aplicação, o protocolo de transporte, porta de atendimento e overhead de transporte, como ilustrado no exemplo seguinte:

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)
Ping	-----	-----	-----	-----
tracert	-----	UDP	33434-33450	8 (33.3%)
telnet	telnet	TCP	23	20 (50%)
ftp	ftp	TCP	21	20 (40.8%)
Tftp	Tftp	UDP	69	8 (27.8%)
Browser/http	http	TCP	80	20 (50%)
nslookup	DNS	UDP	53	8 (33.9%)
ssh	ssh	TCP	22	20 (50%)

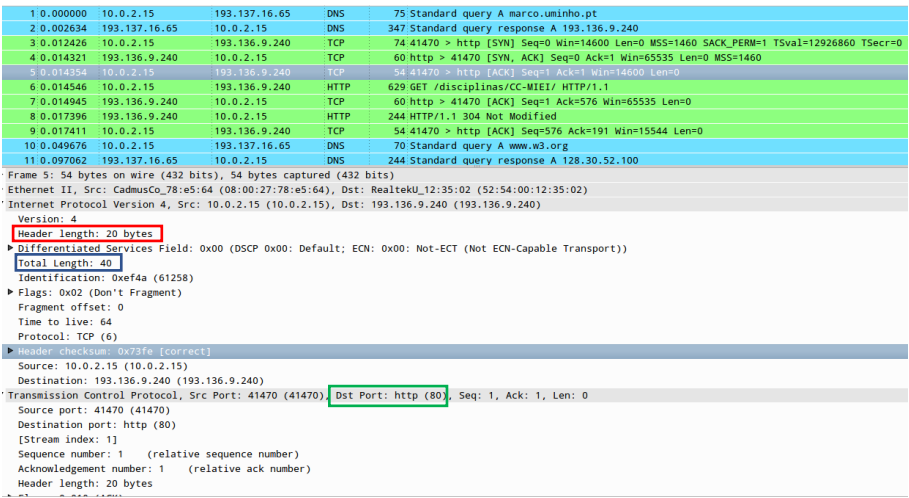


Figura 1- Captura do Wireshark usando o comando HTTP

A figura 1 mostra o protocolo de aplicação http. Podemos verificar que o protocolo de transporte usado é o TCP. A porta de atendimento é a 80 (analisado na figura pelo quadrado verde). Dado que o *Header Length* (quadrado vermelho) é 20 bytes e o *Total Length* é 40 (quadrado azul) , a percentagem de overhead é $20/40 = 0.5$.

Nas restantes aplicações foi aplicado o mesmo raciocínio sendo posteriormente apresentadas as respetivas capturas do wireshark.

1	0.000000	10.0.2.15	193.137.16.65	DNS	75 Standard query A cc2020.ddns.net
2	0.954441	193.137.16.65	10.0.2.15	DNS	172 Standard query response A 193.136.9.183
3	0.955402	10.0.2.15	193.137.16.65	DNS	86 Standard query PTR 183.9.136.193.in-addr.arpa
4	0.958849	193.137.16.65	10.0.2.15	DNS	410 Standard query response PTR dhcp-43.uminho.pt
5	0.959269	10.0.2.15	193.136.9.183	TCP	74 33077 > ftp [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=1
6	0.963772	193.136.9.183	10.0.2.15	TCP	60 ftp > 33077 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
7	0.963871	10.0.2.15	193.136.9.183	TCP	54 33077 > ftp [ACK] Seq=1 Ack=1 Win=14600 Len=0
8	0.989901	193.136.9.183	10.0.2.15	FTP	74 Response: 220 (vsFTPD 2.3.5)
9	0.990111	10.0.2.15	193.136.9.183	TCP	54 33077 > ftp [ACK] Seq=1 Ack=21 Win=14600 Len=0
10	4.953295	10.0.2.15	193.136.9.183	FTP	63 Request: USER cc
11	4.953657	193.136.9.183	10.0.2.15	TCP	60 ftp > 33077 [ACK] Seq=21 Ack=10 Win=65535 Len=0
12	4.960562	193.136.9.183	10.0.2.15	FTP	88 Response: 331 Please specify the password.
13	4.960654	10.0.2.15	193.136.9.183	TCP	54 33077 > ftp [ACK] Seq=10 Ack=55 Win=14600 Len=0
▶ Differentiated Services Field: 0x10 (DSCP 0x04: Unknown DSCP; ECN: 0x00: Not-ECT (Not ECT-Capable Transport)) Total Length: 49 Identification: 0x28ca (10442) ▶ Flags: 0x02 (Don't Fragment) Fragment offset: 0 Time to live: 64 Protocol: TCP (6) ▶ Header checksum: 0x3a9f [correct] Source: 10.0.2.15 (10.0.2.15) Destination: 193.136.9.183 (193.136.9.183)					
▼ Transmission Control Protocol, Src Port: 33077, Dst Port: ftp (21) Seq: 1, Ack: 21, Len: 9 Source port: 33077 (33077) Destination port: ftp (21) [Stream index: 2] Sequence number: 1 (relative sequence number) [Next sequence number: 10 (relative sequence number)] Acknowledgement number: 21 (relative ack number) Header length: 20 bytes ▶ Flags: 0x018 (PSH, ACK) Window size value: 14600 [Calculated window size: 14600] [Window size scaling factor: -2 (no window scaling used)]					

Figura 2- Captura no Wireshark utilizando o comando FTP

1	0.000000	10.0.2.15	193.137.16.65	DNS	75 Standard query AAAA cc2020.ddns.net
2	0.051571	193.137.16.65	10.0.2.15	DNS	135 Standard query response
3	0.052620	10.0.2.15	193.137.16.65	DNS	93 Standard query AAAA cc2020.ddns.net.eduroam.uminho.pt
4	0.055928	193.137.16.65	10.0.2.15	DNS	147 Standard query response, No such name
5	0.056317	10.0.2.15	193.137.16.65	DNS	75 Standard query A cc2020.ddns.net
6	0.113088	193.137.16.65	10.0.2.15	DNS	172 Standard query response A 193.136.9.183
7	0.164263	10.0.2.15	193.136.9.183	TFTP	86 Read Request, File: file1, Transfer type: octet, tsize=000=0\000, blk
8	5.007436	CadmusCo_78:e5:64	RealtekU_12:35:02	ARP	42 Who has 10.0.2.2? Tell 10.0.2.15
9	5.007859	RealtekU_12:35:02	CadmusCo_78:e5:64	ARP	60 10.0.2.2 is at 52:54:00:12:35:02
10	6.170978	10.0.2.15	193.136.9.183	TFTP	86 Read Request, File: file1, Transfer type: octet, tsize=000=0\000, blk
▶ Frame 10: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) ▶ Ethernet II, Src: CadmusCo_78:e5:64 (08:00:27:78:e5:64), Dst: RealtekU_12:35:02 (52:54:00:12:35:02) ▼ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 193.136.9.183 (193.136.9.183) Version: 4 Header length: 20 bytes ▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECT-Capable Transport)) Total Length: 72 Identification: 0xe5b4 (58804) ▶ Flags: 0x02 (Don't Fragment) Fragment offset: 0 Time to live: 64 Protocol: UDP (17) ▶ Header checksum: 0x7da2 [correct] Source: 10.0.2.15 (10.0.2.15) Destination: 193.136.9.183 (193.136.9.183)					
▼ User Datagram Protocol, Src Port: 53786 (53786), Dst Port: tftp (69) Source port: 53786 (53786) Destination port: tftp (69) Length: 52 ▶ Checksum: 0xd793 [validation disabled]					

Figura 3- Captura Wireshark utilizando o comando TFTP

5 0.016035	10.0.2.15	193.137.16.65	DNS	75 Standard query A cc2020.ddns.net
6 0.180760	193.137.16.65	10.0.2.15	DNS	172 Standard query response A 193.136.9.183
7 0.181357	10.0.2.15	193.136.9.183	TCP	74 55946 > telnet [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=13268001 TSecr=0
8 0.187371	193.136.9.183	10.0.2.15	TCP	60 telnet > 55946 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
9 0.187491	10.0.2.15	193.136.9.183	TCP	54 55946 > telnet [ACK] Seq=1 Ack=1 Win=14600 Len=0
10 0.188909	10.0.2.15	193.136.9.183	TELNET	81 Telnet Data ...
11 0.192544	193.136.9.183	10.0.2.15	TCP	60 telnet > 55946 [ACK] Seq=1 Ack=28 Win=65535 Len=0
12 7.548248	10.0.2.15	193.136.9.183	TELNET	58 Telnet Data ...
13 7.548860	193.136.9.183	10.0.2.15	TCP	60 telnet > 55946 [ACK] Seq=1 Ack=32 Win=65535 Len=0

▶ Frame 9: 54 bytes on wire (432 bits), 54 bytes captured (432 bits)
 ▶ Ethernet II, Src: CadmusCo_78:e5:64 (08:00:27:78:e5:64), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
 ▶ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 193.136.9.183 (193.136.9.183)
 Version: 4
 Header length: 20 bytes
 ▶ Differentiated Services Field: 0x10 (DSCP 0x04: Unknown DSCP; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
 Total Length: 40
 Identification: 0x01ed (493)
 Flags: 0x02 (Don't Fragment)
 Fragment offset: 0
 Time to live: 64
 Protocol: TCP (6)
 ▶ Header checksum: 0x6185 [correct]
 Source: 10.0.2.15 (10.0.2.15)
 Destination: 193.136.9.183 (193.136.9.183)
 ▶ Transmission Control Protocol, Src Port: 55946 (55946), Dst Port: telnet (23) Seq: 1, Ack: 1, Len: 0
 Source port: 55946 (55946)
 Destination port: telnet (23)
 [Stream index: 3]
 Sequence number: 1 (relative sequence number)
 Acknowledgement number: 1 (relative ack number)
 Header length: 20 bytes
 ▶ Flags: 0x010 (ACK)
 Window size value: 14600

Figura 4- Captura Wireshark utilizando o comando TELNET

20 0.212343	193.136.9.183	10.0.2.15	SSHv2	366 Server: New Keys
21 0.212407	10.0.2.15	193.136.9.183	TCP	54 37662 > ssh [ACK] Seq=1395 Ack=1338 Win=17712 Len=0
22 0.293960	10.0.2.15	193.136.9.183	SSHv2	70 Client: New Keys
23 0.294870	193.136.9.183	10.0.2.15	TCP	60 ssh > 37662 [ACK] Seq=1338 Ack=1411 Win=65535 Len=0
24 0.294900	10.0.2.15	193.136.9.183	TCP	102 [TCP segment of a reassembled PDU]
25 0.295672	193.136.9.183	10.0.2.15	TCP	60 ssh > 37662 [ACK] Seq=1338 Ack=1459 Win=65535 Len=0
26 0.300045	193.136.9.183	10.0.2.15	TCP	102 [TCP segment of a reassembled PDU]
27 0.300080	10.0.2.15	193.136.9.183	TCP	54 37662 > ssh [ACK] Seq=1459 Ack=1386 Win=17712 Len=0
28 0.334663	10.0.2.15	193.136.9.183	TCP	118 [TCP segment of a reassembled PDU]
29 0.335243	193.136.9.183	10.0.2.15	TCP	60 ssh > 37662 [ACK] Seq=1386 Ack=1523 Win=65535 Len=0

▶ Frame 21: 54 bytes on wire (432 bits), 54 bytes captured (432 bits)
 ▶ Ethernet II, Src: CadmusCo_78:e5:64 (08:00:27:78:e5:64), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
 ▶ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 193.136.9.183 (193.136.9.183)
 Version: 4
 Header length: 20 bytes
 ▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
 Total Length: 40
 Identification: 0xeb20 (60192)
 Flags: 0x02 (Don't Fragment)
 Fragment offset: 0
 Time to live: 64
 Protocol: TCP (6)
 ▶ Header checksum: 0x7861 [correct]
 Source: 10.0.2.15 (10.0.2.15)
 Destination: 193.136.9.183 (193.136.9.183)
 ▶ Transmission Control Protocol, Src Port: 37662 (37662), Dst Port: ssh (22) Seq: 1395, Ack: 1338, Len: 0
 Source port: 37662 (37662)
 Destination port: ssh (22)
 [Stream index: 3]
 Sequence number: 1395 (relative sequence number)
 Acknowledgement number: 1338 (relative ack number)
 Header length: 20 bytes
 ▶ Flags: 0x010 (ACK)
 Window size value: 17712
 Calculated window size: 17712

Figura 5- Captura Wireshark utilizando o comando SSH

1	0.000000	10.0.2.15	193.137.16.65	DNS	73 Standard query A www.uminho.pt
2	0.003671	193.137.16.65	10.0.2.15	DNS	345 Standard query response A 193.137.9.114
3	5.000981	CadmusCo_78:e5:64	RealtekU_12:35:02	ARP	42 Who has 10.0.2.2? Tell 10.0.2.15
4	5.002345	RealtekU_12:35:02	CadmusCo_78:e5:64	ARP	60 10.0.2.2 is at 52:54:00:12:35:02

▶ Frame 1: 73 bytes on wire (584 bits), 73 bytes captured (584 bits)

▶ Ethernet II, Src: CadmusCo_78:e5:64 (08:00:27:78:e5:64), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)

▼ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 193.137.16.65 (193.137.16.65)

Version: 4

Header length: 20 bytes

▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))

Total Length: 59

Identification: 0xbc1e (48158)

▶ Flags: 0x00

Fragment offset: 0

Time to live: 64

Protocol: UDP (17)

▶ Header checksum: 0xe0ba [correct]

Source: 10.0.2.15 (10.0.2.15)

Destination: 193.137.16.65 (193.137.16.65)

▼ User Datagram Protocol, Src Port: 41214 (41214), Dst Port: domain (53)

Source port: 41214 (41214)

Destination port: domain (53)

Length: 39

▶ Checksum: 0xde11 [validation disabled]

Figura 6- Captura Wireshark utilizando o comando nslookup

18	0.029129	10.0.2.15	193.136.19.254	UDP	74 Source port: 51982 Destination port: 33442
19	0.029385	10.0.2.15	193.136.19.254	UDP	74 Source port: 42329 Destination port: 33443
20	0.029631	10.0.2.15	193.136.19.254	UDP	74 Source port: 34156 Destination port: 33444
21	0.029854	10.0.2.15	193.136.19.254	UDP	74 Source port: 46604 Destination port: 33445
22	0.030090	10.0.2.15	193.136.19.254	UDP	74 Source port: 55882 Destination port: 33446
23	0.030385	10.0.2.15	193.136.19.254	UDP	74 Source port: 54194 Destination port: 33447
24	0.030611	10.0.2.15	193.136.19.254	UDP	74 Source port: 53813 Destination port: 33448
25	0.030848	10.0.2.15	193.136.19.254	UDP	74 Source port: 56452 Destination port: 33449
26	0.031414	10.0.2.15	193.137.16.65	DNS	81 Standard query PTR 2.2.0.10.in-addr.arpa
27	0.033256	193.137.16.65	10.0.2.15	DNS	144 Standard query response, No such name
28	0.189007	10.0.2.15	193.136.19.254	UDP	101 Standard query PTR 2.2.0.10.in-addr.arpa "OM" question

▶ Frame 24: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)

▶ Ethernet II, Src: CadmusCo_78:e5:64 (08:00:27:78:e5:64), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)

▼ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 193.136.19.254 (193.136.19.254)

Version: 4

Header length: 20 bytes

▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))

Total Length: 60

Identification: 0xa177 (41335)

▶ Flags: 0x00

Fragment offset: 0

Time to live: 5

Protocol: UDP (17)

▶ Header checksum: 0x32a5 [correct]

Source: 10.0.2.15 (10.0.2.15)

Destination: 193.136.19.254 (193.136.19.254)

▼ User Datagram Protocol, Src Port: 53813 (53813), Dst Port: 33448 (33448)

Source port: 53813 (53813)

Destination port: 33448 (33448)

Length: 40

▶ Checksum: 0xe1ce [validation disabled]

Figura 7- Captura wireshark utilizando o comando traceroute

Questão 2

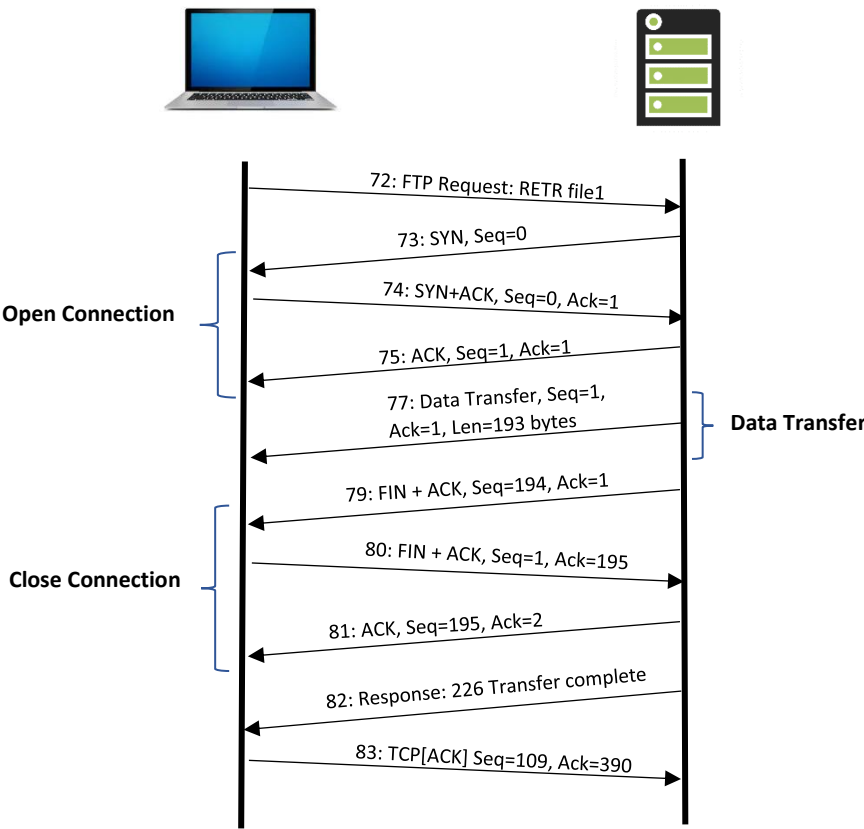
Uma representação num diagrama temporal das transferências da file1 por FTP e TFTP respetivamente. Se for caso disso, identifique as fases de estabelecimento de conexão, transferência de dados e fim de conexão. Identifica também claramente os tipos de segmentos trocados e os números de sequência usados quer nos dados como nas confirmações.

Resposta:

FTP

No.	Time	Source	Destination	Protocol	Length	Info
69	128.084783	10.3.3.1	10.1.1.1	FTP	97	Response: 200 Switching to Binary mode.
70	128.084823	10.1.1.1	10.3.3.1	FTP	89	Request: PORT 10,1,1,1,218,225
71	128.085146	10.3.3.1	10.1.1.1	FTP	117	Response: 200 PORT command successful. Consider using PASV.
72	128.085174	10.1.1.1	10.3.3.1	FTP	78	Request: RETR file1
73	128.085731	10.3.3.1	10.1.1.1	TCP	74	ftp-data > 56033 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=11596533 TSecr=0 WS=
74	128.085731	10.1.1.1	10.3.3.1	TCP	74	56033 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=11596533
75	128.086020	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56033 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=11596533 TSecr=11596533
76	128.086073	10.3.3.1	10.1.1.1	FTP	130	Response: 150 Opening BINARY mode data connection for file1 (193 bytes).
77	128.086081	10.3.3.1	10.1.1.1	FTP-DAT	259	FTP Data: 193 bytes
78	128.086091	10.1.1.1	10.3.3.1	TCP	66	56033 > ftp-data [ACK] Seq=1 Ack=194 Win=15552 Len=0 TSval=11596534 TSecr=11596533
79	128.086113	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56033 [FIN, ACK] Seq=194 Ack=1 Win=14608 Len=0 TSval=11596533 TSecr=11596533
80	128.086300	10.1.1.1	10.3.3.1	TCP	66	56033 > ftp-data [FIN, ACK] Seq=1 Ack=195 Win=15552 Len=0 TSval=11596534 TSecr=11596533
81	128.086871	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56033 [ACK] Seq=195 Ack=2 Win=14608 Len=0 TSval=11596534 TSecr=11596534
82	128.086878	10.3.3.1	10.1.1.1	FTP	90	Response: 226 Transfer complete.
83	128.086891	10.1.1.1	10.3.3.1	TCP	66	59256 > ftp [ACK] Seq=108 Ack=390 Win=14608 Len=0 TSval=11596534 TSecr=11596533
84	130.007670	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
85	130.378628	fe80::200:ff:feaa:1ff02::5		OSPF	90	Hello Packet
86	132.264207	10.1.1.1	10.3.3.1	FTP	72	Request: QUIT
87	132.266283	10.3.3.1	10.1.1.1	FTP	80	Response: 221 Goodbye.

Figura 8- Captura de Tráfego de transferência de file1 por FTP no portátil1

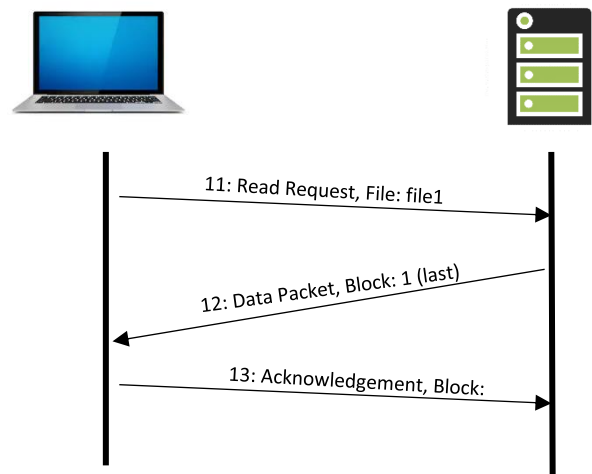


A partir da análise de tráfego (figura 8) e do diagrama temporal desenvolvido a partir dessa captura podemos verificar que: após feito o pedido (FTP request), o servidor estabelece a conexão enviando um pacote SYN com o número de sequência 0. O utilizador responde com um SYN+ACK. Posteriormente, o servidor envia um ACK de volta ao utilizador. Depois de todos os dados pretendidos serem enviados, é iniciada a fase de encerramento de conexão. A flag FIN, presente nesta fase, indica que se pretende terminar a conexão e fica à espera de uma resposta ACK.

TFTP

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
2	0.684537	fe80::200:ff:feaa:1ff02::5	224.0.0.5	OSPF	90	Hello Packet
3	10.000148	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
4	10.716638	fe80::200:ff:feaa:1ff02::5	224.0.0.5	OSPF	90	Hello Packet
5	20.001066	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
6	20.726775	fe80::200:ff:feaa:1ff02::5	224.0.0.5	OSPF	90	Hello Packet
7	30.001166	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
8	30.732838	fe80::200:ff:feaa:1ff02::5	224.0.0.5	OSPF	90	Hello Packet
9	32.365763	00:00:00:aa:00:1b	Broadcast	ARP	42	Who has 10.1.1.254? Tell 10.1.1.1
10	32.365917	00:00:00:aa:00:11	00:00:00:aa:00:1b	ARP	42	10.1.1.254 is at 00:00:00:aa:00:11
11	32.365917	10.1.1.1	10.3.3.1	TFTP	56	Read Request, File: file1, Transfer type: octet
12	32.366453	10.3.3.1	10.1.1.1	TFTP	239	Data Packet, Block: 1 (last)
13	32.366491	10.1.1.1	10.3.3.1	TFTP	46	Acknowledgement, Block: 1
14	37.373021	00:00:00:aa:00:11	00:00:00:aa:00:1b	ARP	42	Who has 10.1.1.1? Tell 10.1.1.254
15	37.373051	00:00:00:aa:00:1b	00:00:00:aa:00:11	ARP	42	10.1.1.1 is at 00:00:00:aa:00:1b
16	40.001312	10.1.1.254	224.0.0.5	OSPF	78	Hello Packet
17	40.737332	fe80::200:ff:feaa:1ff02::5	224.0.0.5	OSPF	90	Hello Packet

Figura 9- Captura de Tráfego de transferência de file1 por TFTP no portátil1



Com base na captura de tráfego (figura 9) e no correspondente diagrama temporal podemos verificar que o utilizador envia um Read Request para o servidor contendo variadas informações entre as quais o nome do ficheiro. O servidor responde com um pacote de dados. Quando o utilizador recebe os dados, envia um pacote ACK para o servidor.

Questão 3

Com base nas experiências realizadas, distinga e compare sucintamente as quatro aplicações de transferência de ficheiros que usou nos seguintes pontos (i) uso da camada de transporte; (ii) eficiência na transferência; (iii) complexidade; (iv) segurança;

Resposta:

	SFTP	FTP	TFTP	HTTP
Uso da camada de transporte	TCP	TCP	UDP	TCP
Eficiência na transferência	O facto de ser fiável tem um impacto negativo na eficiência.	É fiável pelo que terá um maior overhead.	Possui um overhead mais baixo pois não se encarrega pela entrega dos dados.	Possui grande eficiência.
Complexidade	É bastante complexo uma vez que possui múltiplas funcionalidades.	Contribui para a segurança no processo de transferência de dados. É, portanto, complexo.	Como o protocolo é UDP, existem menos funcionalidades disponíveis. Como tal, a complexidade é reduzida.	Complexidade reduzida.
Segurança	É seguro uma vez que aplica encriptação e autenticação dos dados.	Recorre à autenticação, mas a segurança é reduzida.	Não possui técnicas de autenticação nem de encriptação. Tem por isso um nível de segurança bastante reduzido.	Recorre à autenticação, mas a segurança é reduzida.

Questão 4

As características das ligações de rede têm uma enorme influência nos níveis de Transporte e de Aplicação. Discuta, relacionando a resposta com as experiências realizadas, as influências das situações de perda ou duplicação de pacotes IP no desempenho global de Aplicações fiáveis (se possível, relacionando com alguns dos mecanismos de transporte envolvidos).

Resposta:

55	44.667488	10.2.2.1	10.3.3.1	FTP	78 Request: RETR file1
56	44.672959	10.3.3.1	10.2.2.1	TCP	74 ftp-data > 58042 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=13413210 TSecr=0 WS=16
57	44.672979	10.2.2.1	10.3.3.1	TCP	74 58042 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=13413210 TSecr=13413210 WS=16
58	44.679334	10.3.3.1	10.2.2.1	TCP	66 ftp-data > 58042 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=13413211 TSecr=13413210
59	44.679510	10.3.3.1	10.2.2.1	FTP	130 Response: 150 Opening BINARY mode data connection for file1 (193 bytes).
60	44.680034	10.3.3.1	10.2.2.1	FTP-DAT	259 FTP Data: 193 bytes
61	44.680034	10.3.3.1	10.2.2.1	TCP	66 ftp-data > 58042 [FIN, ACK] Seq=194 Ack=1 Win=14608 Len=0 TSval=13413211 TSecr=13413210
62	44.680034	10.2.2.1	10.3.3.1	TCP	66 58042 > ftp-data [ACK] Seq=1 Ack=194 Win=15552 Len=0 TSval=13413211 TSecr=13413211
63	44.680035	10.2.2.1	10.3.3.1	TCP	66 58042 > ftp-data [FIN, ACK] Seq=1 Ack=195 Win=15552 Len=0 TSval=13413211 TSecr=13413211
64	44.685401	10.3.3.1	10.2.2.1	TCP	66 ftp-data > 58042 [ACK] Seq=195 Ack=2 Win=14608 Len=0 TSval=13413213 TSecr=13413211
65	44.685470	10.3.3.1	10.2.2.1	FTP	90 Response: 226 Transfer complete.
66	44.685492	10.2.2.1	10.3.3.1	TCP	66 47993 > ftp [ACK] Seq=107 Ack=390 Win=14608 Len=0 TSval=13413213 TSecr=13413211
67	48.242352	10.2.2.1	10.3.3.1	FTP	72 Request: QUIT
68	48.249147	10.3.3.1	10.2.2.1	FTP	80 Response: 221 Goodbye.

Figura 10- Captura de Tráfego de transferência da file1 por FTP em alpha

13	30.020540	fe80::200:ff:feaa:11:11:11:11:11	ff02::5	OSPF	90 Hello Packet
14	33.181914	10.2.2.1	10.3.3.1	TFTP	56 Read Request, File: file1, Transfer type: octet
15	33.197381	10.3.3.1	10.2.2.1	TFTP	239 Data Packet, Block: 1 (last)
16	33.197563	10.2.2.1	10.3.3.1	TFTP	46 Acknowledgement, Block: 1
17	38.207410	00:00:00 aa:00:12 00:00:00 aa:00:19		ARP	42 Who has 10.2.2.1? Tell 10.2.2.254

Figura 11- Captura de Tráfego de transferência da file1 por TFTP em alpha

Como já visto anteriormente, a aplicação FTP utiliza o protocolo de transporte TCP e a aplicação TFTP utiliza o protocolo de transporte UDP. Nas figuras apresentadas verificamos que estes dois protocolos tratam a ação de transferência de ficheiros de formas distintas.

O UDP não possui mecanismos de deteção e recuperação de perdas de pacotes, pelo que se forem perdidos, os protocolos que correm em cima do UDP poderão identificar e corrigir estas folhas. Para tal poderá ser atribuído um número de identificação ao pacote de modo a verificar a sua receção.

Por contrapartida, o TPC possui tais mecanismos de deteção e recuperação, assim caso haja uma perda de pacotes estes serão retransmitidos. O TCP garante que todos os pacotes são entregues.

Conclusão

Com a elaboração deste trabalho foi possível observar e estudar os diferentes protocolos de aplicação e os respectivos protocolos de transporte (TCP e UDP), assim como o overhead e as portas de atendimento associadas.

Posteriormente observamos a transferência de um ficheiro recorrendo a serviços diferentes (SFTP, FTP, TFTP, HTTP). Deste modo foi possível identificar as diferentes fases para a sua conceção entre as quais: estabelecimento de conexão, transferência de dados e fim de conexão. Numa fase final distinguimos como o TCP e o UDP lidam com a perda e duplicação de pacotes.