Universidade do Minho

Licenciatura em Engenharia Informática

Comunicações Por Computador

Grupo 45

TP1:Protocolos da Camada de Transporte

Maria Eugénia Bessa Cunha (A93264)

Vicente Gonçalves Moreira (A93296)

Tânia Filipa Soares Teixeira (A89613)

Outubro 2021

1 Questões e Respostas

1.1 Questões Parte I.

1.1.1 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.

O desempenho das aplicações é afetado pelas perdas e duplicações devido à ineficiente comunicação. Estas anomalias no processo de transporte resultaram em atrasos na comunicação devido à necessidade de reenviar e/ou sincronizar os pacotes, o que também pode levar a ocorrer duplicação destes. A camada que garantiu o normal funcionamento da comunicação e que efetuou eventuais correções foi a camada de transporte, como se pode verifcar no seguinte exemplo.

Figura 1: Perda/Duplicação de pacotes entre 'Grilo' e 'Servidor1'

5	6.707209300	10.4.4.1	10.2.2.1	FTP	74	Request: TYPE I
6	6.707410002	10.2.2.1	10.4.4.1	FTP	97	Response: 200 Switching to Binary mode.
7	6.712553858	10.4.4.1	10.2.2.1	TCP	66	38400 → 21 [ACK] Seq=9 Ack=32 Win=502 Len=0 TSval=1316876299 …
8	6.712620841	10.4.4.1	10.2.2.1	FTP	89	Request: PORT 10,4,4,1,163,241
9	6.712840338	10.2.2.1	10.4.4.1	FTP	117	Response: 200 PORT command successful. Consider using PASV.
10	6.717927479	10.4.4.1	10.2.2.1	TCP	66	38400 → 21 [ACK] Seq=32 Ack=83 Win=502 Len=0 TSval=1316876305…
11	6.717988139	10.4.4.1	10.2.2.1	FTP	78	Request: RETR file1
12	6.717988880	10.4.4.1	10.2.2.1	TCP	78	[TCP Retransmission] 38400 → 21 [PSH, ACK] Seq=32 Ack=83 Win=…
13	6.718249497	10.2.2.1	10.4.4.1	TCP	78	21 → 38400 [ACK] Seq=83 Ack=44 Win=510 Len=0 TSval=3138262800…
14	6.718402777	10.2.2.1	10.4.4.1	TCP	74	20 → 41969 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
15	6.723482115	10.4.4.1	10.2.2.1	TCP	74	41969 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
16	6.723623961	10.2.2.1	10.4.4.1	TCP	66	20 → 41969 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3138262805…
17	6.723730265	10.2.2.1	10.4.4.1	FTP	130	Response: 150 Opening BINARY mode data connection for file1 (…
18	6.724361511	10.2.2.1	10.4.4.1	FTP-DA…	290	FTP Data: 224 bytes (PORT) (RETR file1)
19	6.724365972	10.2.2.1	10.4.4.1	TCP	66	20 → 41969 [FIN, ACK] Seq=225 Ack=1 Win=64256 Len=0 TSval=313…
20	6.729506073	10.4.4.1	10.2.2.1	TCP	66	41969 → 20 [ACK] Seq=1 Ack=225 Win=65024 Len=0 TSval=13168763…
21	6.729625500	10.4.4.1	10.2.2.1	TCP	66	41969 → 20 [FIN, ACK] Seq=1 Ack=226 Win=65024 Len=0 TSval=131…
22	6.729626293	10.4.4.1	10.2.2.1	TCP	66	[TCP Out-Of-Order] 41969 → 20 [FIN, ACK] Seq=1 Ack=226 Win=65…
23	6.729856277	10.2.2.1	10.4.4.1	TCP	66	20 → 41969 [ACK] Seq=226 Ack=2 Win=64256 Len=0 TSval=31382628…
24	6.729858938	10.2.2.1	10.4.4.1	TCP	66	[TCP Dup ACK 23#1] 20 → 41969 [ACK] Seq=226 Ack=2 Win=64256 L…
25	6.729859819	10.2.2.1	10.4.4.1	FTP	90	Response: 226 Transfer complete.
26	6.734978416	10.4.4.1	10.2.2.1	TCP	54	41969 → 20 [RST] Seq=2 Win=0 Len=0
27	6.734979426	10.4.4.1	10.2.2.1	TCP	66	38400 → 21 [ACK] Seq=44 Ack=171 Win=502 Len=0 TSval=131687632…

1.1.2 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.

Figura 2: Transferência do 'file1' utilizando FTP

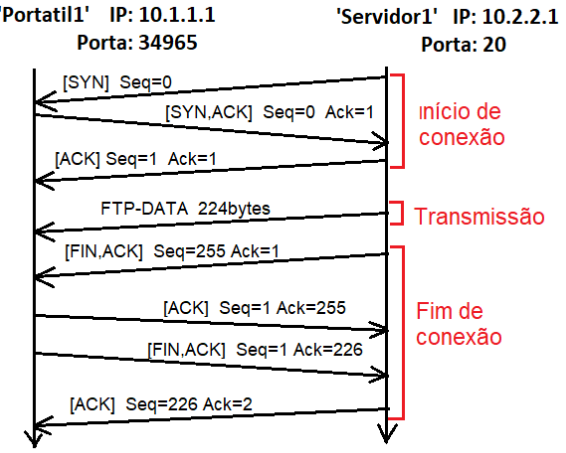


Figura 3: Packets wire-shark (Referência para o diagrama anterior)

35	11.679247986	10.2.2.1	10.1.1.1	TCP	74 20 → 34965 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
36	11.679406514	10.1.1.1	10.2.2.1	TCP	74 34965 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
37	11.679523529	10.2.2.1	10.1.1.1	TCP	66 20 → 34965 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3620143200…
38	11.679610816	10.2.2.1	10.1.1.1	FTP	130 Response: 150 Opening BINARY mode data connection for file1 (…
39	11.679612783	10.2.2.1	10.1.1.1	FTP-DA…	290 FTP Data: 224 bytes (PORT) (RETR file1)
40	11.679788936	10.2.2.1	10.1.1.1	TCP	66 20 → 34965 [FIN, ACK] Seq=225 Ack=1 Win=64256 Len=0 TSval=362…
41	11.679808511	10.1.1.1	10.2.2.1	TCP	66 51168 → 21 [ACK] Seq=78 Ack=295 Win=502 Len=0 TSval=189468682…
42	11.679809304	10.1.1.1	10.2.2.1	TCP	66 34965 → 20 [ACK] Seq=1 Ack=225 Win=65024 Len=0 TSval=18946868…
43	11.680102253	10.1.1.1	10.2.2.1	TCP	66 34965 → 20 [FIN, ACK] Seq=1 Ack=226 Win=65024 Len=0 TSval=189…
44	11.680302375	10.2.2.1	10.1.1.1	TCP	66 20 → 34965 [ACK] Seq=226 Ack=2 Win=64256 Len=0 TSval=36201432…

1.1.3 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

Figura 4: Transferência do 'file1' utilizando TFTP

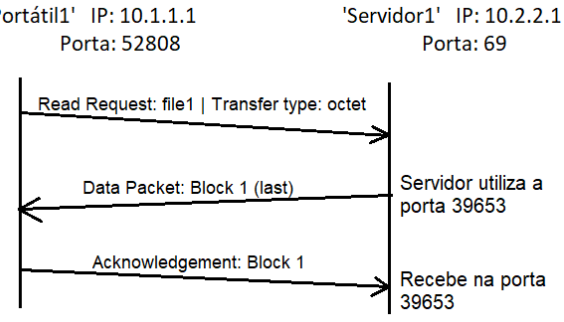
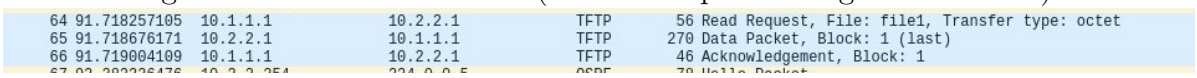


Figura 5: Packets wire-shark (Referência para o diagrama anterior)



64	91.718257105	10.1.1.1	10.2.2.1	TFTP	56	Read Request, File: file1, Transfer type: octet
65	91.718676171	10.2.2.1	10.1.1.1	TFTP	270	Data Packet, Block: 1 (last)
66	91.719004109	10.1.1.1	10.2.2.1	TFTP	46	Acknowledgement, Block: 1

1.1.4 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;

FTP - ***(i)***TCP ; ***(ii)*** (2^{o}) 1.054ms para a transferência; ***(iii)***Complexo devido as várias respostas e sistemas de transmissão extras (utilização de dois canais de comunicação); ***(iv)***Pouca seguranca pois não utiliza encriptação, sendo possível "ver"os conteúdos dos pacotes.

TFTP - ***(i)***UDP; ***(ii)*** (1^{o}) 0.747ms para a transferência; ***(iii)***Pouco complexo, dado que não é necessário tanta transmissão de informação; ***(iv)***Pouca segurança pois não utiliza encriptação.

SFTP - ***(i)***TCP; ***(ii)*** (4^{o}) 47.8ms para a transferência; ***(iii)***Complexo devido ao uso de encriptação e às várias respostas e pacotes utilizados; ***(iv)***Muito seguro pois encripta os pacotes;

HTTP - (i)TCP; ***(ii)*** (3^{o}) 10.468ms para a transferência; ***(iii)*** Baixa complexidade em relação a FTP e SFTP, dado que não utiliza tantos pacotes ou portas extras para transmissão;***(iv)*** Pouca segurança dado que os pacotes não são encriptados

1.2 Questões parte II.

1.2.1 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	Protocolo de transporte	Porta de atendimento	Overhead de transporte em bytes**
ping	———	——*	———	——
traceroute	———	——*	———	——
telnet	TELNET	TCP	23	1600 (80 packets)
ftp	FTP	TCP	21	480 (24 packets)
tftp	TFTP	UDP	69	24 (3 packets)
http(browser)	HTTP	TCP	80	1700 (85 packets)
nslookup	DNS	UDP	53	32 (4 packets)
ssh	SSHv2	TCP	22	1740 (87 packets)

*O comando *ping* e *traceroute* utilizaram o protocolo *ICMP*, no entanto, este é um protocolo de rede, logo não o incluímos.

**Para calcular o *overhead*, foi efetuada a comunicação e após a captura desta utilizamos o comando *"tcp/udp.port == XX"* e avaliamos o número de packets totais utilizados. (Packets * Tamanho do cabeçalho UPD/TCP) (Uma vez que TCP pode ter um tamanho de cabeçalho variável entre 20-60 bytes, calculamos uma estimativa minima com o tamanho de cabeçalho 20 bytes)

Figura 6: Exemplo do comando Telnet

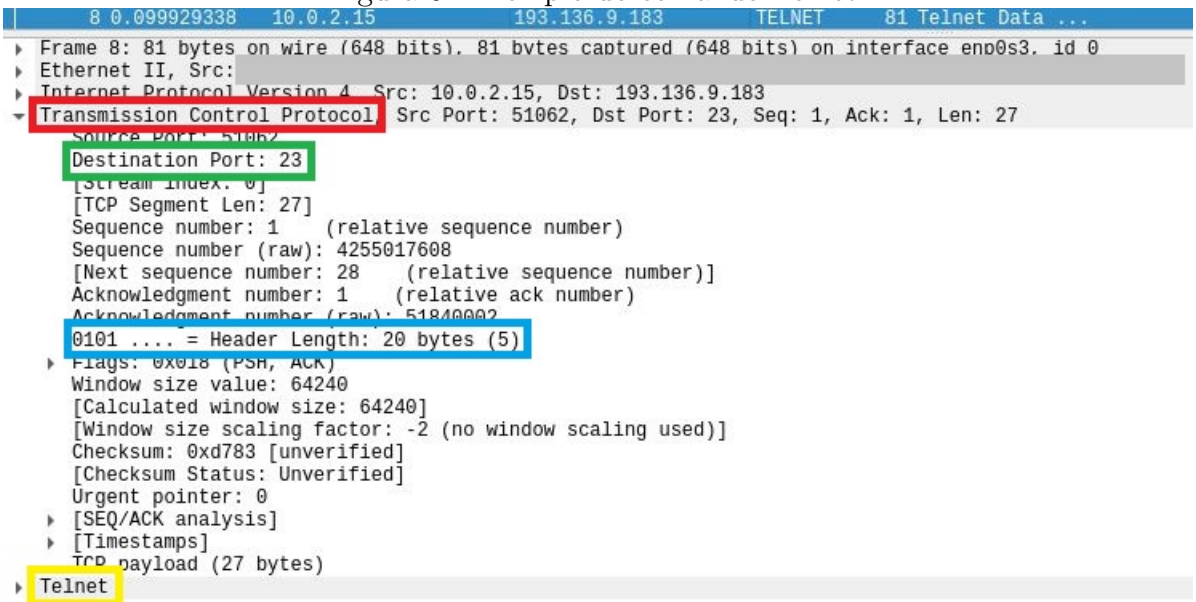


Figura 7: Exemplo do comando NSLookup

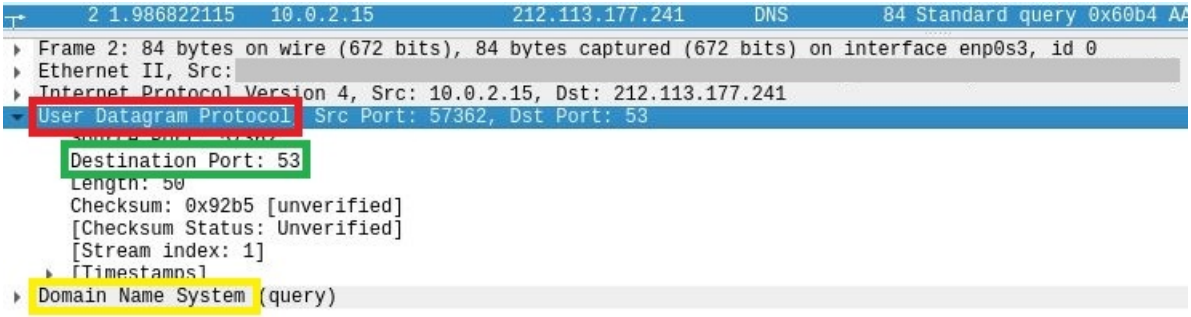


Figura 8: Exemplo do comando FTP

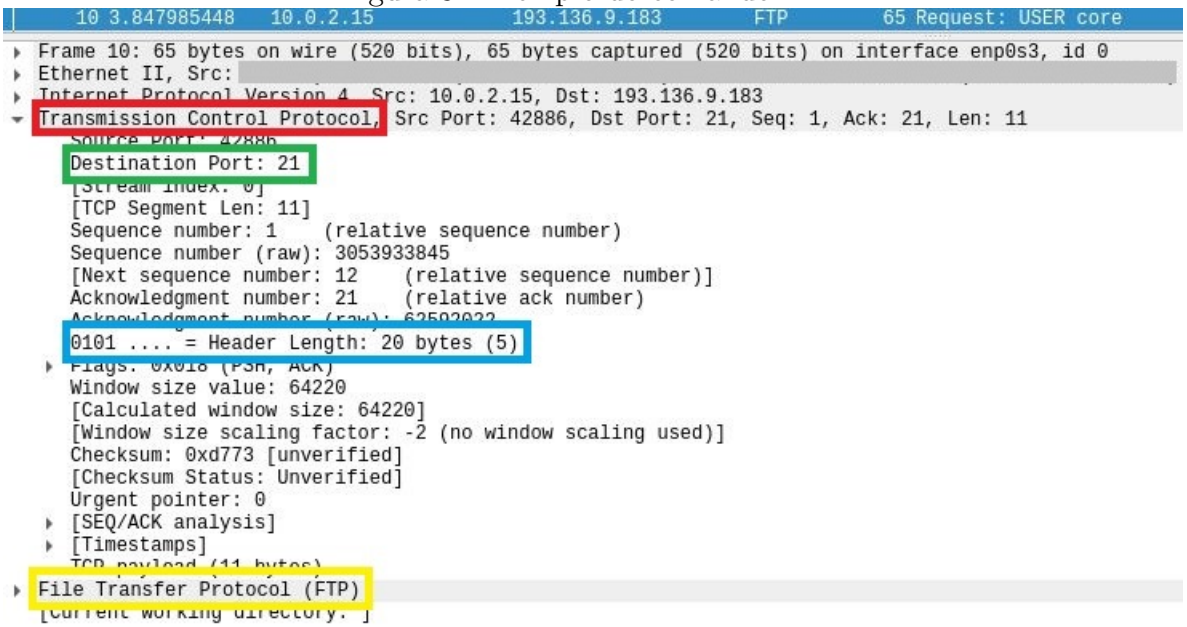
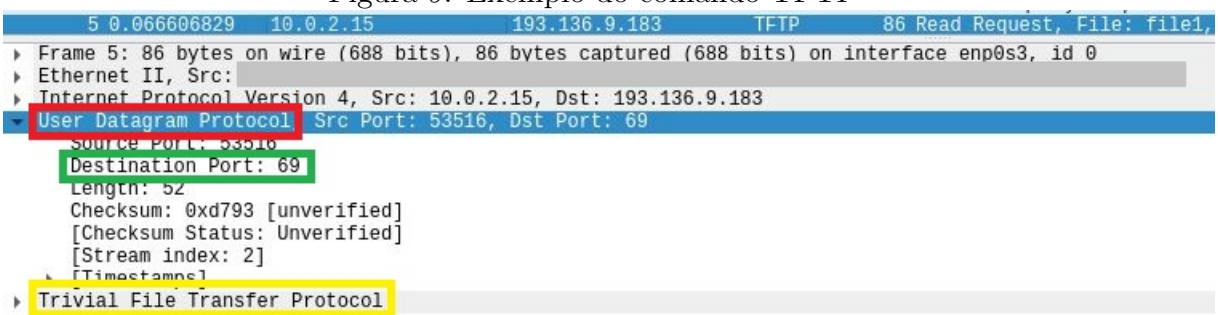


Figura 9: Exemplo do comando TFTP



2 Conclusão

Durante a realização deste trabalho podemos perceber, consolidar e trabalhar os conteúdos e conceitos lecionados nas aulas teóricas relativos a protocolos de transporte *TCP* e *UDP*, expandindo, desta forma, os nossos conhecimentos relativos ao funcionamento destes.

Além do mais, com recurso ao *Core* e ao *Wireshark*, tivemos a possibilidade de avaliar e analisar estes mesmos protocolos, capturando o tráfego de packets e assim elaborar diagramas temporais, o que nos permitiu aprofundar os conhecimentos referentes às *flags* utilizadas para estabelecer conexão e a confirmação da receção dos dados transferidos.

Finalmente, na elaboração das respostas, pudemos avaliar e comparar o funcionamento de alguns protocolos de aplicação como *FTP* e *HTTP* o que nos proporcionou uma necessidade de estudo mais detalhado destas, reforçando assim o nosso conhecimento nesta área.