TP1 Protocolos da Camada de Transporte

Alexandre Martins, João Bernardo Freitas, and João Luís Amorim

University of Minho, Department of Informatics, 4710-057 Braga, Portugal
e-mail: {a77523,a74814,a74806}@alunos.uminho.pt

1 Introdução

O principal objectivo deste trabalho é o aprofundamento de conhecimentos em protocolos de aplicação **TCP** e **UDP** bem como os protocolos de transferência **TFTP**, **FTP**, **SFTP** e **HTTP**,para tal utilizamos ferramentas como **Wireshark** e **CORE**.

2 Questões e Respostas

2.1 Questão 1

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	x	ICMP	x	0+20=20
Traceroute	x	UDP	33521	0+20=20
Telnet	telnet	TCP	23	20+20=40
FTP	ftp	TCP	21	20+20=40
TFTP	ssl/hhtp-over-tls	TCP	443	20+20=40
browser/http	http	TCP	80	20+20=40
nslookup	DNS	UDP	53	0+20=20
ssh	ssh	TCP	22	20+20=40

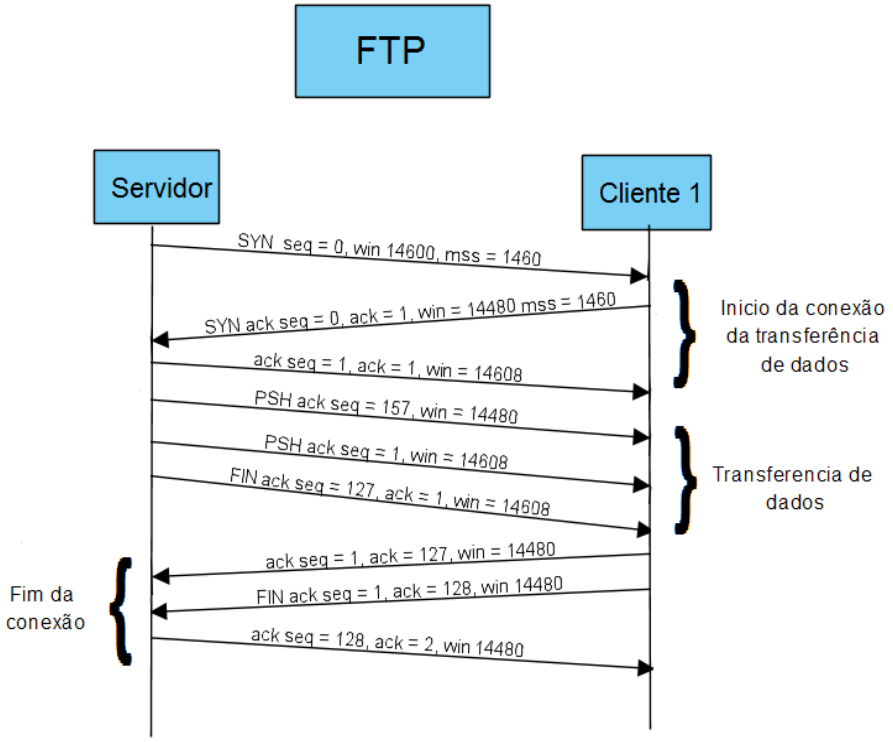
2.2 Questão 2

"Uma representação num diagrama temporal das transferências da file1 por FTP e TFTP respectivamente. 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. (Nota: a transferência por FTP envolve mais que uma conexão FTP, nomeadamente uma de controlo [ftp] e outra de dados [ftp-data]. Faça o diagrama apenas para a conexão de transferência de dados do ficheiro mais pequeno)"

A elaboração dos diagramas temporais da transferência do file1 é feita através da análise do tráfego capturado utilizando o Wireshark, tráfego este que foi obtido após a execução das instruções mencionadas no enunciado.

Representação Temporal da Transferência do File1

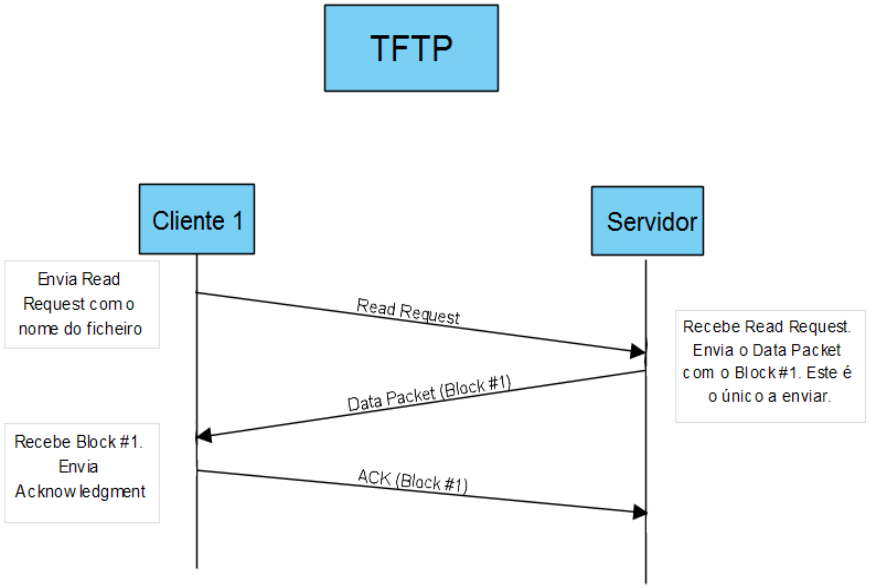
FTP:



Tal como já mencionado, o diagrama é obtido após a análise do tráfego capturado para a transferência por FTP, tal como é mostrado de seguida (a figura corresponde apenas à conexão de transferência de dados para o file1 e não à totalidade do tráfego):

33	37.728980	10.1.1.1	10.4.4.1	TCP	74	ftp-data > 45749 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=621285 TSecr=0 WS=16
34	37.729508	10.4.4.1	10.1.1.1	TCP	74	45749 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=621285 TSecr=621285 WS=16
35	37.729929	10.1.1.1	10.4.4.1	TCP	66	ftp-data > 45749 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=621285 TSecr=621285
36	37.729975	10.1.1.1	10.4.4.1	FTP	105	Response: 150 Here comes the directory listing.
37	37.729976	10.1.1.1	10.4.4.1	FTP-DAT	192	FTP Data: 126 bytes
38	37.729977	10.1.1.1	10.4.4.1	TCP	66	ftp-data > 45749 [FIN, ACK] Seq=127 Ack=1 Win=14608 Len=0 TSval=621285 TSecr=621285
39	37.730470	10.4.4.1	10.1.1.1	TCP	66	45749 > ftp-data [ACK] Seq=1 Ack=127 Win=14480 Len=0 TSval=621285 TSecr=621285
40	37.730471	10.4.4.1	10.1.1.1	TCP	66	45749 > ftp-data [FIN, ACK] Seq=1 Ack=128 Win=14480 Len=0 TSval=621285 TSecr=621285
41	37.731008	10.1.1.1	10.4.4.1	TCP	66	ftp-data > 45749 [ACK] Seq=128 Ack=2 Win=14608 Len=0 TSval=621285 TSecr=621285

TFTP:



Tal como já mencionado, o diagrama é obtido após a análise do tráfego capturado para a transferência por TFTP, tal como é mostrado de seguida:

33	134.709354	10.4.4.1	10.1.1.1	TFTP	56	Read Request, File: file1, Transfer type: octet
34	134.712749	10.1.1.1	10.4.4.1	TFTP	483	Data Packet, Block: 1 (last)
35	134.713446	10.4.4.1	10.1.1.1	TFTP	46	Acknowledgement, Block: 1

2.3 Questão 3

"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;"

TFTP *Trivial File Transfer Protocol* é o protocolo mais simples dos quatro estudados neste trabalho, é geralmente utilizado para transferir ficheiros de pequena dimensão e utiliza **UDP** *User Datagram Protocol* como protocolo de aplicação. Devido á sua simplicidade é o protocolo mais rápido e mais fácil de programar, infelizmente esta simplicidade vem com um custo na segurança, nomeadamente, não existe autenticação de utilizador nem encriptação de mensagens e visto que é utilizado **UDP**, não se verifica se o receptor recebeu o pacote ou não, permitindo que haja perda de pacotes, baixando a sua fiabilidade.

FTP *File Transfer Protocol* é um protocolo da camada de transporte semelhante ao **TFTP** sendo que uma das principais diferenças entre os dois está no facto de utilizar **TCP** *Transmission Control Protocol* em vez de **UDP**. O uso de **TCP** tem como consequência o aumento do tamanho dos pacotes, devido ao *overhead*, e também do tempo de transferência de ficheiros, visto que o protocolo **TCP** necessita que haja comunicação entre o servidor e o receptor para confirmar que todos os pacotes são entregues aumentando a sua fiabilidade. Porém tal como **TFTP** não encripta as mensagens mas ao contrário deste é necessário haver autenticação do utilizador.

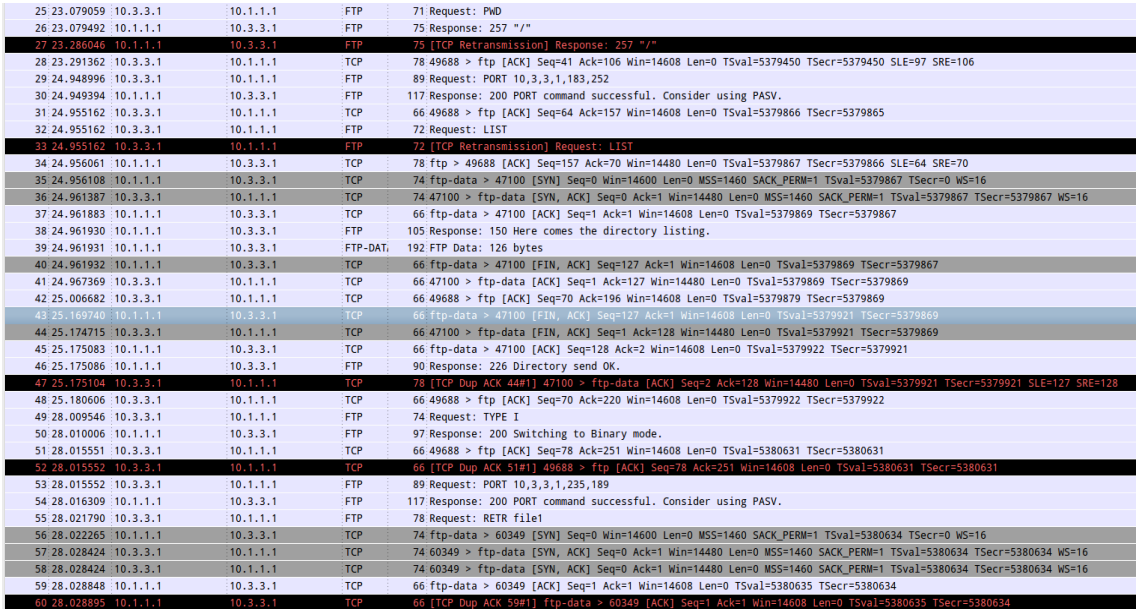
SFTP *SSH File Transfer Protocol* é um protocolo da camada de transporte baseado no protocolo **FTP** sendo que a principal diferença entre os dois é o facto de que **SFTP** encripta todas as mensagens através do protocolo **SSH** (Secure Shell). Visto que é baseado no protocolo **FTP** utiliza **TCP** como protocolo de aplicação. O facto de encriptar os pacotes obriga o aumento do tempo de transferências ainda mais, comparado com **FTP**, aumentando também a segurança.

HTTP *Hypertext Transfer Protocol* foi o ultimo protocolo que nós experimentamos que, tal como **FTP** e **SFTP** utiliza **TCP** como protocolo de aplicação, logo é tão fiável quanto esses dois. No que toca á velocidade **HTTP** é equivalente a **FTP**, sendo que este último tem a vantagem no que toca á transferência de um só ficheiro de grande tamanho, no entanto **HTTP** é mais rápido quando é necessário transferir vários ficheiros de tamanho mais reduzido. No que toca a segurança é semelhante a **TFTP**, visto que, não efectua autenticação de utilizador nem encripta os pacotes.

2.4 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)."

Protocolos de transferência que utilizem **TCP** têm disponíveis sistemas de verificação de erros, onde, no caso de haver perda de pacotes ou algum erro de transmissão, existem métodos de os descobrir e pedir ao servidor para efectuar uma retransmissão. Isto é feito através dos números de sequência, onde o receptor irá repetir um ACK com o número de sequência do último pacote recebido correctamente, o servidor ao receber mensagens constantes com com um ACK repetido irá reenviar o pacote imediatamente a seguir ao ACK. Isto leva a uma perda de velocidade nas comunicações pois são necessários campos de verificação que aumentam o *overhead*, e consequentemente o tamanho do pacote, mas em contrapartida aumenta a fiabilidade da transmissão, dando mais garantias que a comunicação é feita correctamente.
Como podemos observar na seguinte imagem, era suposto uma transferência, que começou na transmissão 35, terminar na transmissão 44, visto que o cliente enviou **FIN, ACK**, mas, devido a erros na transmissão, o servidor não recebeu essa mensagem e, tal como podemos observar, no pacote 45 começa a transferir o segundo ficheiro, o cliente ao receber um pacote com *Seq=128* apesar de ter na transmissão anterior enviado um pacote com *Ack=128* percebeu que tinha ocorrido um erro. Para avisar o servidor que da ocorrência de um erro na transmissão envia um **Dup Ack**, de forma indicar o servidor onde é que este ocorreu. Visto que o pacote 56 é extremamente semelhante ao 35 podemos afirmar que esse pacote corresponde ao reenvio do pacote 35.



25	23.079059	10.3.3.1	10.1.1.1	FTP	71 Request: PWD
26	23.079492	10.1.1.1	10.3.3.1	FTP	75 Response: 257 "/"
27	23.286046	10.1.1.1	10.3.3.1	FTP	75 [TCP Retransmission] Response: 257 "/"
28	23.291362	10.3.3.1	10.1.1.1	TCP	78 49688 > ftp [ACK] Seq=41 Ack=106 Win=14608 Len=0 TSval=5379450 TSecr=5379450 SLE=97 SRE=106
29	24.948996	10.3.3.1	10.1.1.1	FTP	89 Request: PORT 10,3,3,1,183,252
30	24.949394	10.1.1.1	10.3.3.1	FTP	117 Response: 200 PORT command successful. Consider using PASV.
31	24.955162	10.3.3.1	10.1.1.1	TCP	66 49688 > ftp [ACK] Seq=64 Ack=157 Win=14608 Len=0 TSval=5379866 TSecr=5379865
32	24.955162	10.3.3.1	10.1.1.1	FTP	72 Request: LIST
33	24.955162	10.3.3.1	10.1.1.1	FTP	72 [TCP Retransmission] Request: LIST
34	24.956061	10.1.1.1	10.3.3.1	TCP	78 ftp > 49688 [ACK] Seq=157 Ack=70 Win=14480 Len=0 TSval=5379867 TSecr=5379866 SLE=64 SRE=70
35	24.956108	10.1.1.1	10.3.3.1	TCP	74 ftp-data > 47100 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=5379867 TSecr=0 WS=16
36	24.961387	10.3.3.1	10.1.1.1	TCP	74 47100 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=5379867 TSecr=5379867 WS=16
37	24.961883	10.1.1.1	10.3.3.1	TCP	66 ftp-data > 47100 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=5379869 TSecr=5379867
38	24.961930	10.1.1.1	10.3.3.1	FTP	105 Response: 150 Here comes the directory listing.
39	24.961931	10.1.1.1	10.3.3.1	FTP-DAT	192 FTP Data: 126 bytes
40	24.961932	10.1.1.1	10.3.3.1	TCP	66 ftp-data > 47100 [FIN, ACK] Seq=127 Ack=1 Win=14608 Len=0 TSval=5379869 TSecr=5379867
41	24.967369	10.3.3.1	10.1.1.1	TCP	66 47100 > ftp-data [ACK] Seq=1 Ack=127 Win=14480 Len=0 TSval=5379869 TSecr=5379869
42	25.006682	10.3.3.1	10.1.1.1	TCP	66 49688 > ftp [ACK] Seq=70 Ack=196 Win=14608 Len=0 TSval=5379879 TSecr=5379869
43	25.169740	10.1.1.1	10.3.3.1	TCP	66 ftp-data > 47100 [FIN, ACK] Seq=127 Ack=1 Win=14608 Len=0 TSval=5379921 TSecr=5379869
44	25.174715	10.3.3.1	10.1.1.1	TCP	66 47100 > ftp-data [FIN, ACK] Seq=1 Ack=128 Win=14480 Len=0 TSval=5379921 TSecr=5379869
45	25.175083	10.1.1.1	10.3.3.1	TCP	66 ftp-data > 47100 [ACK] Seq=128 Ack=2 Win=14608 Len=0 TSval=5379922 TSecr=5379921
46	25.175086	10.1.1.1	10.3.3.1	FTP	90 Response: 226 Directory send OK.
47	25.175104	10.3.3.1	10.1.1.1	TCP	78 [TCP Dup ACK 44#1] 47100 > ftp-data [ACK] Seq=2 Ack=128 Win=14480 Len=0 TSval=5379921 TSecr=5379921 SLE=127 SRE=128
48	25.180606	10.3.3.1	10.1.1.1	TCP	66 49688 > ftp [ACK] Seq=70 Ack=220 Win=14608 Len=0 TSval=5379922 TSecr=5379922
49	28.009546	10.3.3.1	10.1.1.1	FTP	74 Request: TYPE I
50	28.010006	10.1.1.1	10.3.3.1	FTP	97 Response: 200 Switching to Binary mode.
51	28.015551	10.3.3.1	10.1.1.1	TCP	66 49688 > ftp [ACK] Seq=78 Ack=251 Win=14608 Len=0 TSval=5380631 TSecr=5380631
52	28.015552	10.3.3.1	10.1.1.1	TCP	66 [TCP Dup ACK 51#1] 49688 > ftp [ACK] Seq=78 Ack=251 Win=14608 Len=0 TSval=5380631 TSecr=5380631
53	28.015552	10.3.3.1	10.1.1.1	FTP	89 Request: PORT 10,3,3,1,235,189
54	28.016309	10.1.1.1	10.3.3.1	FTP	117 Response: 200 PORT command successful. Consider using PASV.
55	28.021790	10.3.3.1	10.1.1.1	FTP	78 Request: RETR file1
56	28.022265	10.1.1.1	10.3.3.1	TCP	74 ftp-data > 60349 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=5380634 TSecr=0 WS=16
57	28.028424	10.3.3.1	10.1.1.1	TCP	74 60349 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=5380634 TSecr=5380634 WS=16
58	28.028424	10.3.3.1	10.1.1.1	TCP	74 60349 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=5380634 TSecr=5380634 WS=16
59	28.028848	10.1.1.1	10.3.3.1	TCP	66 ftp-data > 60349 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=5380635 TSecr=5380634
60	28.028895	10.1.1.1	10.3.3.1	TCP	66 [TCP Dup ACK 59#1] ftp-data > 60349 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=5380635 TSecr=5380634

Por outro lado nos protocolos que utilizem **UDP** não há verificação nem correcção de erros que o que faz com que aumente bastante a velocidade de transmissão, no entanto leva a uma menor fiabilidade. A comunicação entre servidor e cliente é feita em três pacotes, primeiro o cliente pede para transferir um ficheiro, de seguida o servidor envia o ficheiro e por fim o cliente envia uma confirmação que recebeu.

27	119.781500	10.3.3.1	10.1.1.1	TFTP	56	Read Request, File: file1, Transfer type: octet
28	119.785843	10.1.1.1	10.3.3.1	TFTP	483	Data Packet, Block: 1 (last)
29	119.791276	10.3.3.1	10.1.1.1	TFTP	46	Acknowledgement, Block: 1

Por isso **UDP** é normalmente utilizado em aplicações onde velocidade é o mais importante, Streaming, jogos online, DNS, VoIP e **TCP** é utilizado nas aplicações onde é necessário que a informação enviada esteja exacta EMail,Web.

3 Conclusão

Neste trabalho aprofundou-se a compreensão do funcionamento dos diferentes protocolos de transferência e de aplicação. Em particular, estudou-se os protocolos de transferência **TFTP**, **FTP**, **SFTP** e **HTTP** as suas vantagens bem como as desvantagens, também estudamos os protocolos de aplicação **TCP** e **UDP** dando mais importância às diferenças relativas à fiabilidade e velocidade de transmissão.