**TP1: Protocolos da Camada de Transporte**

Henrique José Carvalho Faria

José André Martins Pereira

University of Minho, Department of Informatics,

4710-057 Braga, Portugal e-mail: [{a82200,a82880}@alunos.uminho.pt](mailto:%7ba82145,a82880,a81744%7d@alunos.uminho.pt)

**Exercício 1:**

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

**Resposta:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Comando usado (Aplicação)** | **Protocolo de Aplicação (se ap** | **Protocolo de transporte (se aplicável)** | **Porta de atendimento (se aplicável)** | **Overhead de transporte em bytes (se aplicável)** |
| **Ping** | Não tem aplicação | Não tem protocolo de transporte porque corre diretamente em ICMP | Não tem porta | 20 bytes |
| **traceroute** | Não tem aplicação | UDP | 33434 | 8 bytes |
| **telnet** | TELNET | TCP | 23 | 20 bytes |
| **ftp** | FTP | TCP | 21 (20 para data) | 20 bytes |
| **tftp** | TFTP | UDP | 69 | 8 bytes |
| **Browser/http** | HTTP | TCP | 80 | 32 bytes |
| **nslookup** | DNS | UDP | 53 | 8 bytes |
| **ssh** | SSHv2 | TCP | 22 | 20 bytes |
| **Outras:** |  |  |  |  |
| **sftp** | TLSv1.2 | TCP | 5223 | 32 bytes |
|  |  |  |  |  |

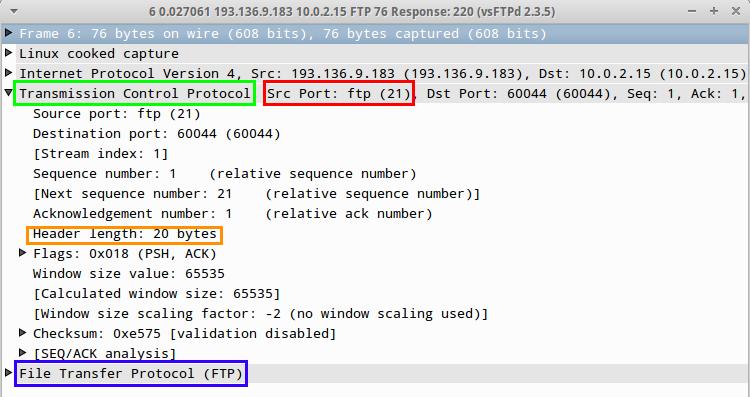
****

Figura 1 – Informação do FTP.

A Figura 1, representa a captura feita com o Wireshark, durante a execução do comando ***ftp gr2018.ddns.net,*** onde se consegue verificar o protocolo da aplicação **FTP** (retângulo azul), o protocolo de transporte **TCP** (*Transmission Control Protocol*) (retângulo verde), a porta de atendimento, **21** (retângulo vermelho) e por fim, não menos importante, o overhead de transporte, **20 bytes** (retângulo laranja).

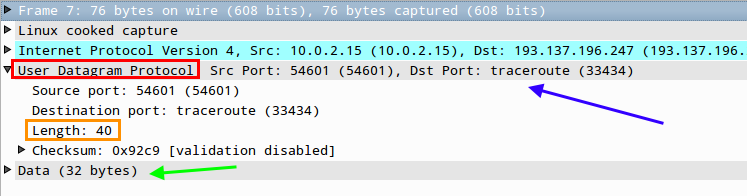


Figura 2 – Informação do traceroute.

A Figura 2, representa a captura durante a execução do comando *traceroute* [*www.fccn.pt*](http://www.fccn.pt), o qual não tem protocolo de aplicação, o protocolo de transporte é UDP, pelo que se pode obsvervar no retângulo vermelho, a porta de atendimento é a 33434, como se pode verificar na seta azul, por fim, o campo **Length**= 40 bytes (retângulo laranja), ou seja, o **overhead** de transporte é 40 – **payload**, ou seja, 40 – 32 = 8 bytes (que normalmente é sempre o mesmo quando o protocolo de transporte é UDP), pelo que tem-se 8/40 = 0.20 = 20% de overhead.

Os restantes comandos, são semelhantes, pelo que não se achou necessário a sua apresentação e justificação.

**Exercício 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. (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).

**Resposta:**

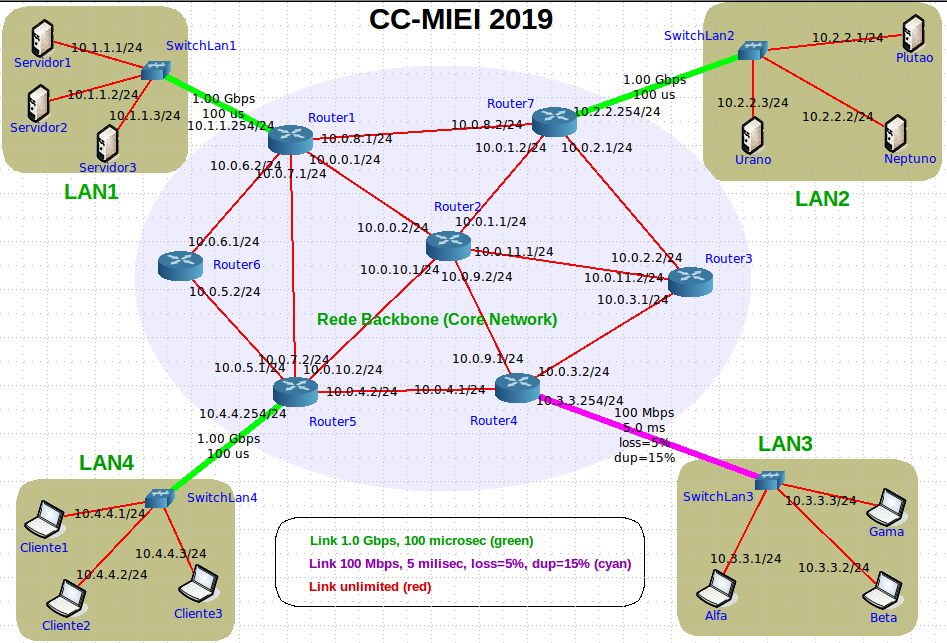
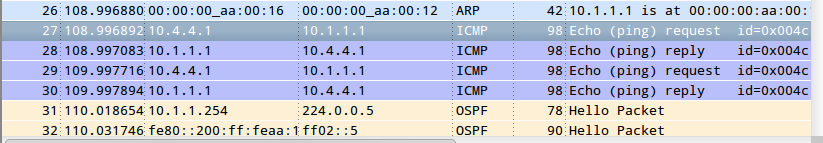


Figura 3 – Topologia utilizada.

Inicialmente verificou-se a conetividade entre o Cliente1 e Servidor1, capturando com o Wireshark no Router1, durante a execução do comando ping entre o Cliente1(10.4.4.1) e Servidor1(10.1.1.1), e obteve-se a resposta que se pode observar na Figura 4. O mesmo se verifica entre o host Alfa(10.3.3.1) eo Servidor1, como se pode ver na Figura 5.

Figura 4 – Captura no Router1 durante a execução do comando ping 10.1.1.1.

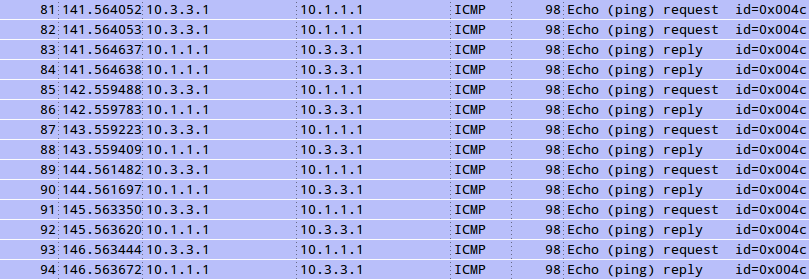
Figura 5 – Captura no Router1 durante a execução do comando ping 10.1.1.1.

Figura 6 – Captura no wireshark do comando TFTP.

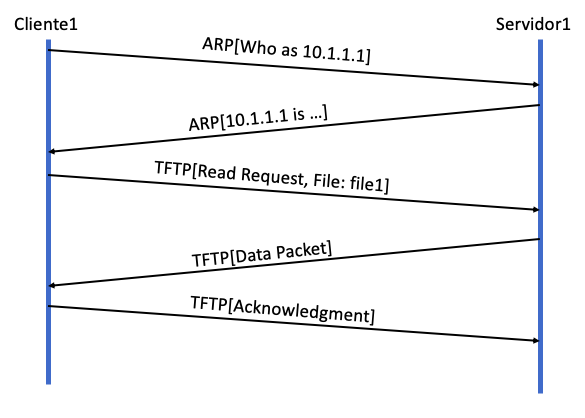


Figura 7 – Diagrama temporal de transferência do TFTP.

Observando o diagrama temporal de transferências do comando TFTP, verifica-se que não é complexo, começando pelo pedido de transferência, através do pacote TFTP Request do file1, ao qual o Servidor1 “responde” com um pacote TFTP Data Packet, com o file1, e o host Cliente1, envia um ACK a informar o servidor que recebeu o ficheiro file1.

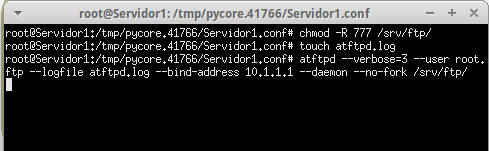


Figura 8 - Inicialização do servidor 1.

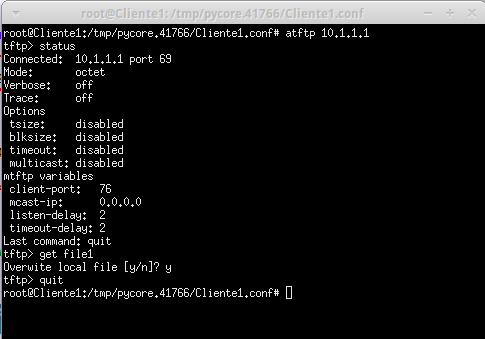


Figura 9 – Cliente 1 faz transferência do file1.

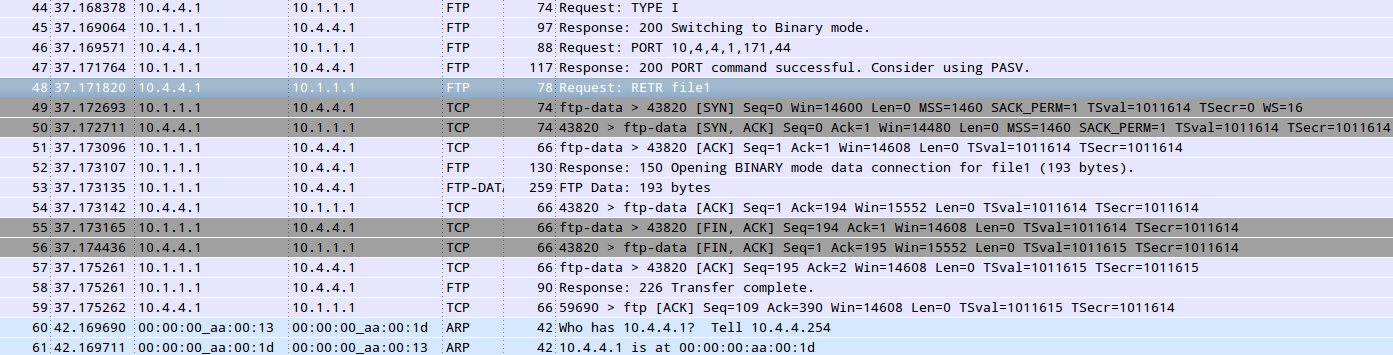


Figura 10 – Captura wireshark do FTP.

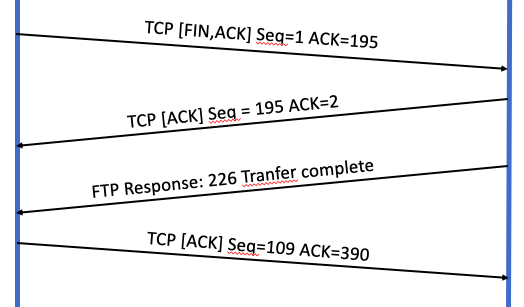
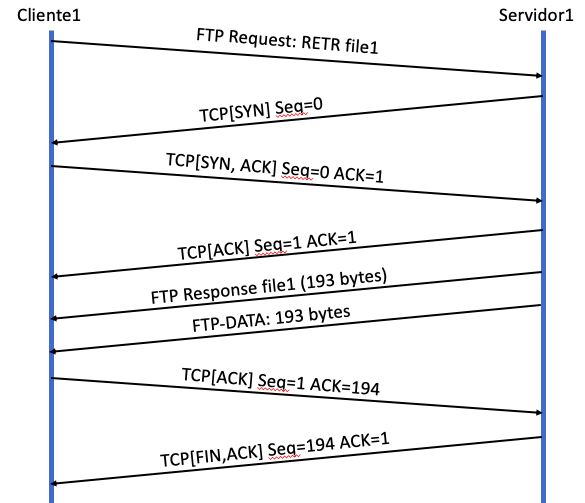


Figura 11 – Diagrama temporal de transferências do FTP.

Observando o diagrama temporal de transferências do comando FTP, verifica-se que inicialmente é enviado um pacote FTP Request, ao qual o servidor denominado Servidor1 “responde” com um pacote TCP [SYN] para se estabelecer a sincronização, ao qual o host Cliente1 responde com TCP [SYN, ACK], e de seguida, o servidor, envia TCP [ACK] com Seq=1, Ack=1, com a finalidade de informar o host que vai iniciar o envio do ficheiro.

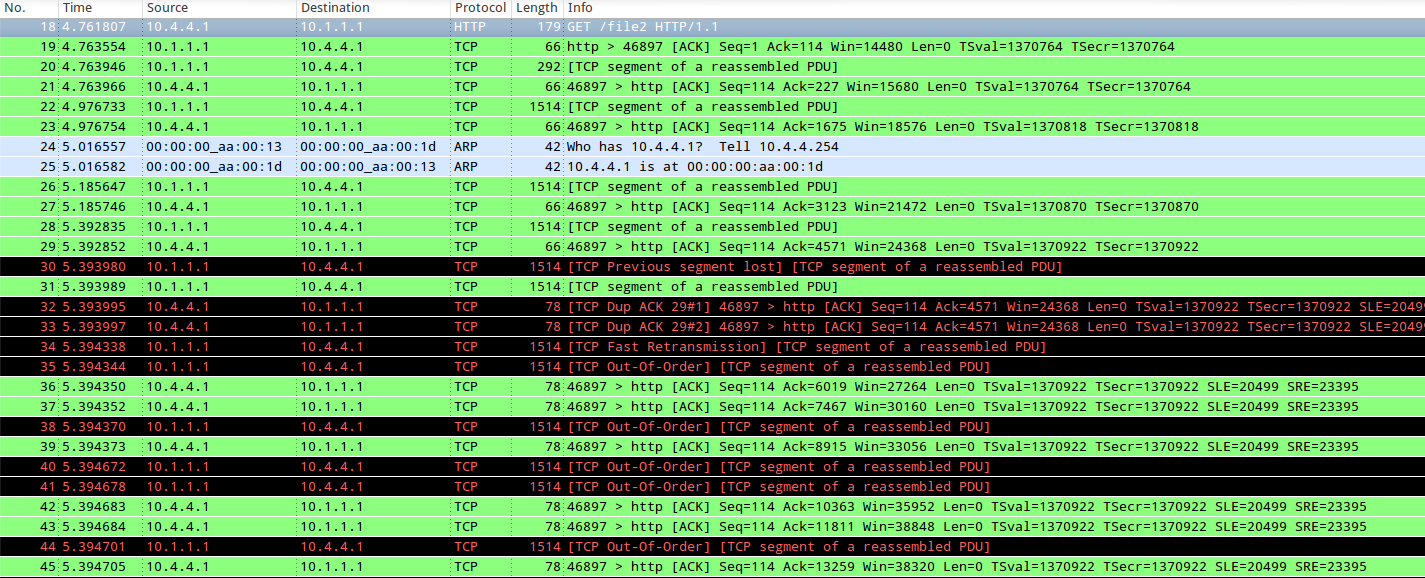
O servidor, começa a enviar o ficheiro com o FTP Response file1, e envia o pacote com o ficheiro, FTP-DATA.

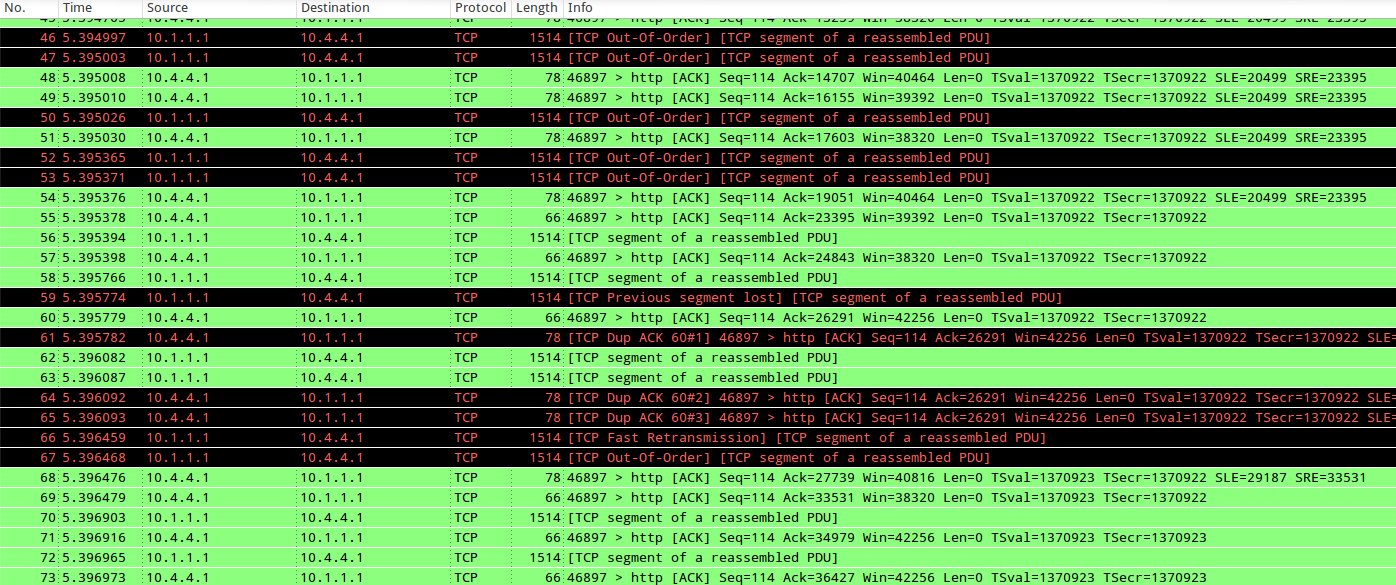
O Cliente1, informa que recebeu o file1, através do TCP [ACK] Seq=1, Ack=194, pelo que o servidor responde com TCP [FIN,ACK] Seq=194, Ack=1, a informar a finalização, ao qual o cliente, “responde” com Ack=195. De seguida o servidor, envia o pacote TCP [ACK] Seq=195, Ack=2, e para terminar o servidor informa que a transferência está concluída com FTP Response, e o cliente envia um TCP [ACK] Seq=109, ACK=390, a informa que recebeu.

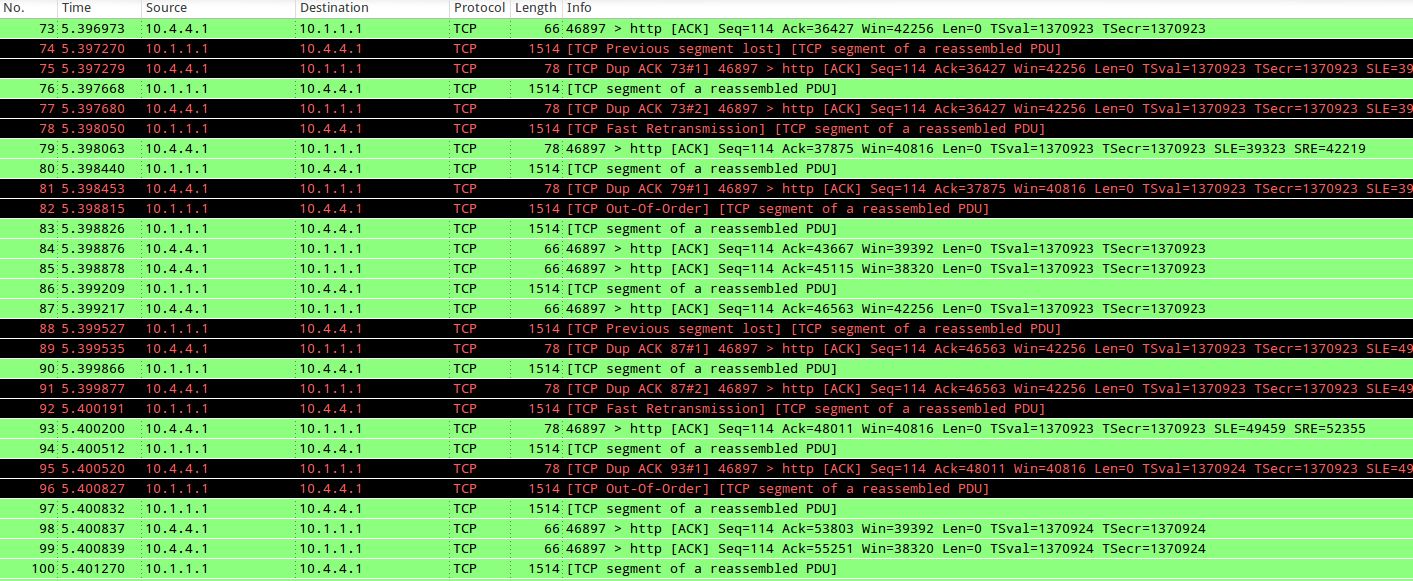
**Exercício 3:**

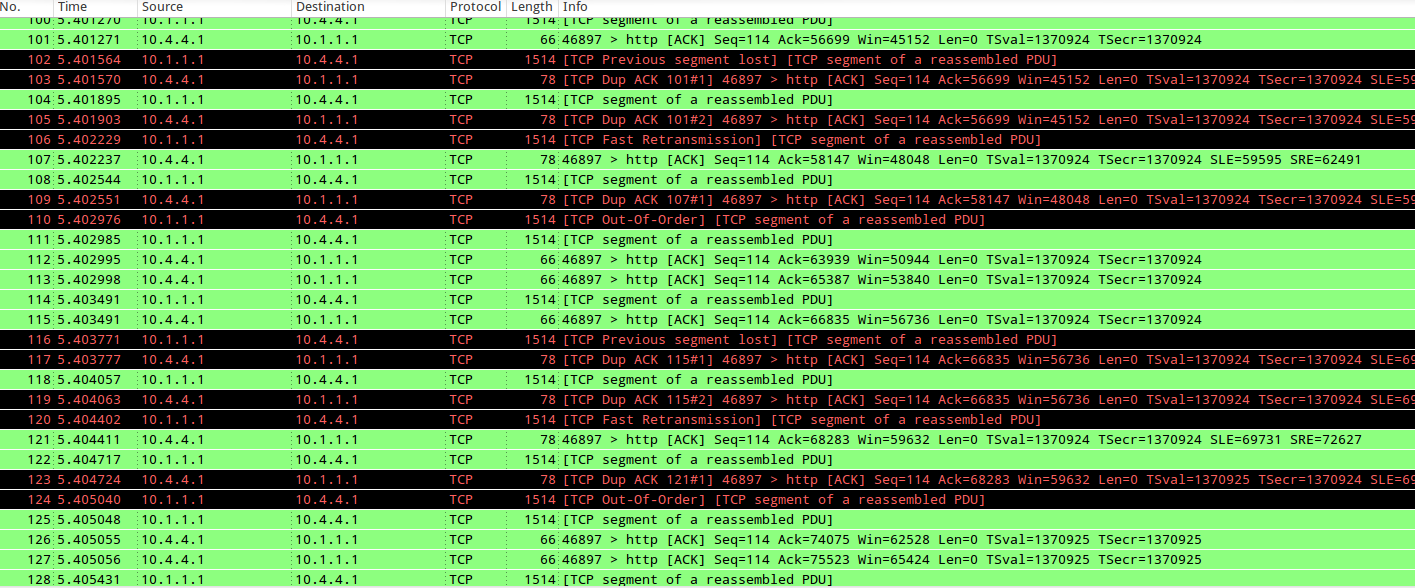
Com base nas experiências realizadas, distinga e compare sucintamente as quatro aplicações de transferência de ficheiros que usou nos seguintes pontos (i) uso da camada de transporte; (ii) eficiência na transferência; (iii) complexidade; (iv) segurança;

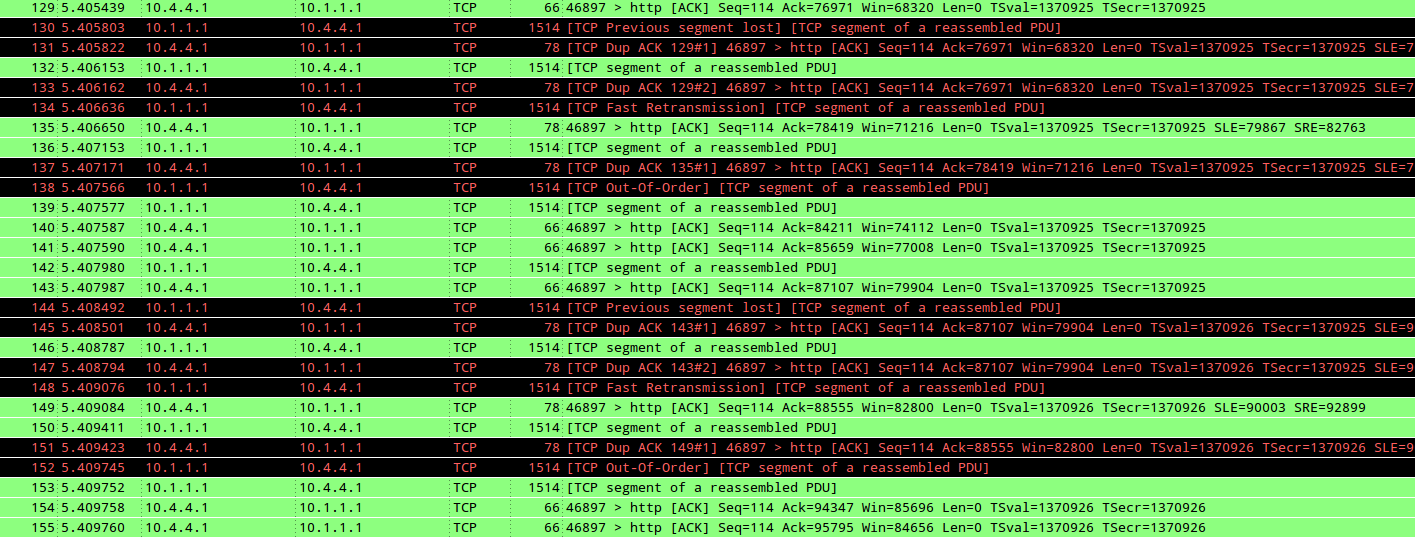
**Resposta:**











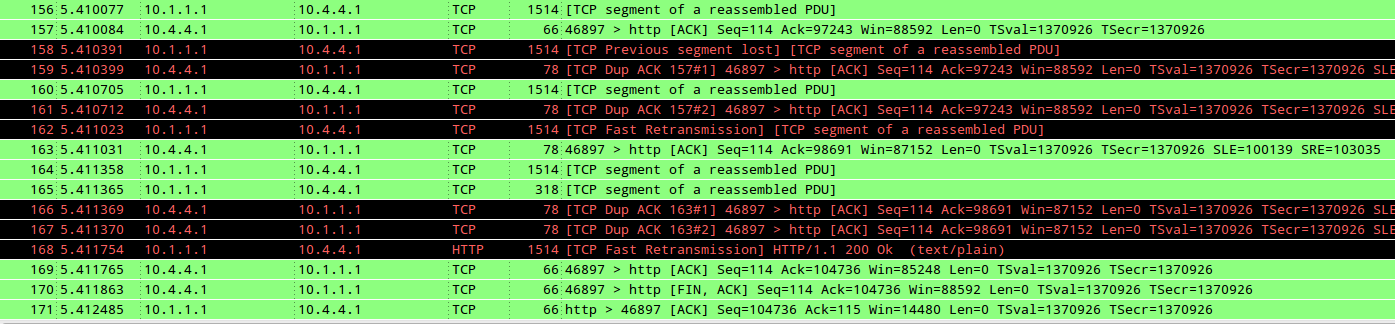


Figura 12 – Troca de mensagens HTTP.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocolos** | **Uso da camada de transporte** | **Eficiência na transferência** | **complexidade** | **segurança** |
| **FTP** | **TCP** | **3º mais eficiente** | **3º mais complexo** | **Login** |
| **TFTP** | **UDP** | **1º mais eficiente** | **4º mais complexo** |  |
| **SFTP** | **TCP** | **4º mais eficiente** | **1º mais complexo** | **Login + encriptação** |
| **HTTP** | **TCP** | **2º mais eficiente** | **2º mais complexo** |  |

**Eficiência na transferência:**

Devido ás características de cada protocolo, o TFTP é o mais eficiente por ser mais rápido a transferir os dados, quer por não implementar segurança quer por usar UDP não fazendo verificação na receção de erros.

Em segundo vem o HTTP que não implementa segurança mas utiliza TCP, verificando os pacotes mal os recebe.

Em 3º lugar temos o FTP que utiliza login e TCP e em último temos o SFTP que para além de TCP e login utiliza também encriptação.

**Complexidade:**



Figura 13 – Complexidade TFTP.

Em matéria de complexidade o menos complexo trata-se do TFTP que faz uso apenas de 3 frames para além dos ARPs trocados visto que não faz uso de medidas de segurança ou autenticação, análogamente o HTTP não usa medidas de segurança ou encriptação, mas usa o protocolo TCP na camada de transporte acabando por fazer tratamento de erros na receção dos frames ao contrário do TFTP

Por fazer uso do login e de encriptação o SFTP é o mais complexo e segue-se o FTP em complexidade visto que só usa login e não encriptação.

**Segurança:**



Figura 14 - SFTP – pedido de password

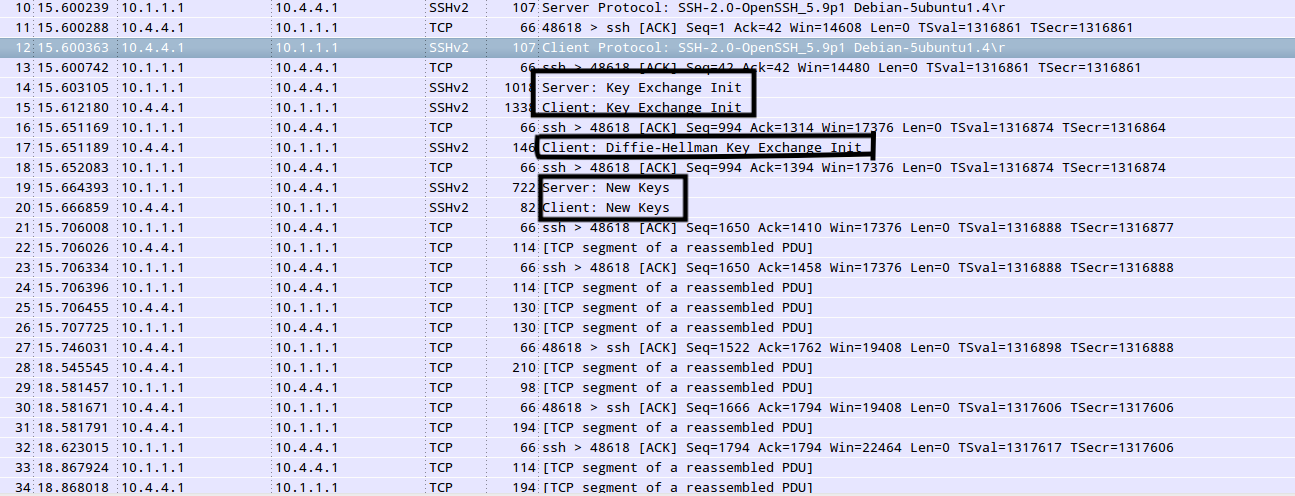


Figura 15 - SFTP – uso de encriptação

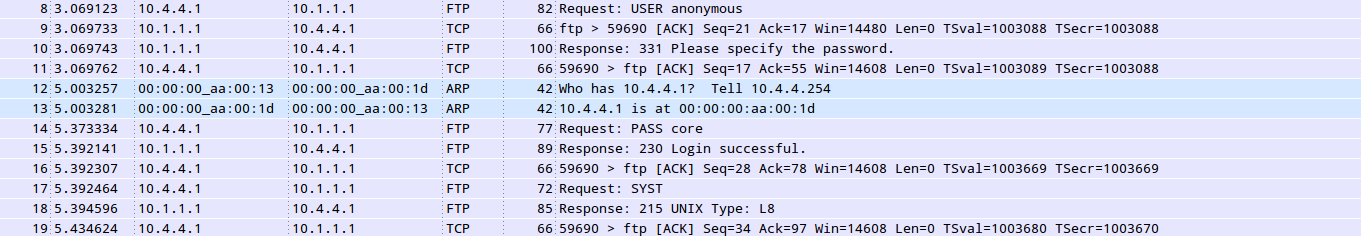


Figura 16 - FTP – uso de autenticação

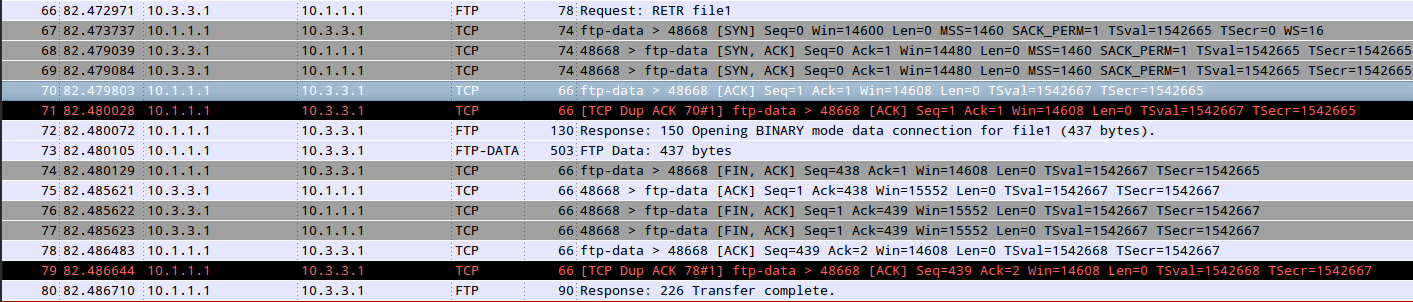
No caso da segurança o mais seguro é o SFTP pois para além de utilizar autenticação, usa também encriptação na troca de dados. Em segundo lugar fica o FTP que utiliza apenas login para garantir a autenticidade do utilizador.

Tanto o TFTP como o HTTP são menos seguros visto que não fazem uso nem de autenticação nem de encriptação.

**Exercício 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:**

Figura 17 – Captura do FTP entre o host Alfa e o Servidor1.

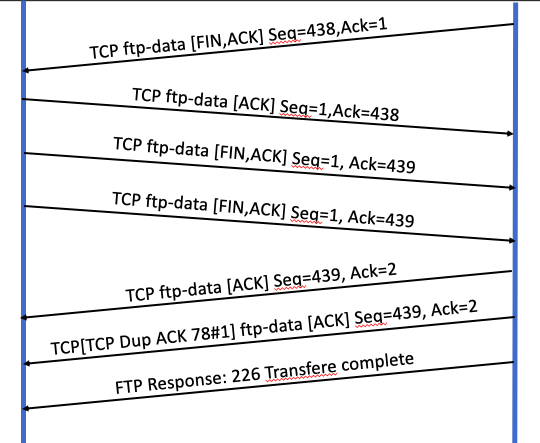
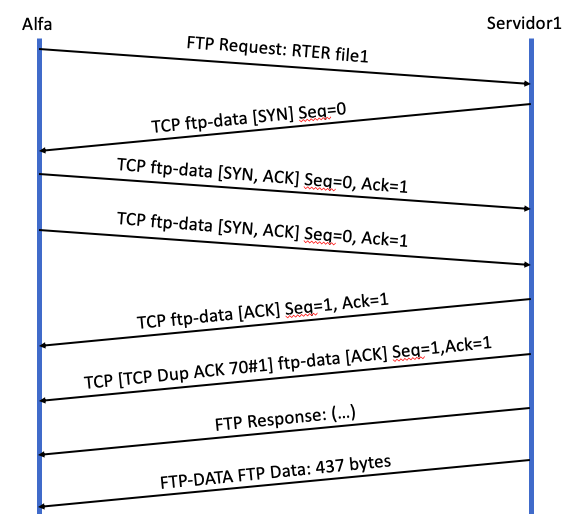


Figura 18 – Diagrama de transferências do FTP entre o host Alfa e o Sevidor1.

Na figura 17, está representado o diagrama de transferências entre o host Alfa1 e o Servidor1, através da aplicação FTP. Inicialmente é feito um REQUEST [RTER file1], ao qual o Servidor1 envia um TCP[SYN] com Seq = 0.

Ao receber este pacote o host Alfa, envia dois pacotes TCP [SYN, ACK], com a mesma informação, pelo que o Servidor1 ao receber os mesmos, envia um pacote TCP [Dup ACK 70#1] ftp-data [ACK] com **Ack = 1**, com o objetivo de “avisar” o Alfa de algo não está bem, pois houve **duplicação** de dois pacotes. Depois é enviado o file1 através de **FTP-DATA (437 bytes).**

O servidor informa o host Alfa da finalização da transferência, pelo que o cliente, envia inicialmente um TCP ftp-data [ACK] e depois dois TCP ftp-data [FIN,ACK] Seq=1, Ack=439. O servidor, ao receber, vai identificar com uma duplicação, e informar o cliente do mesmo com TCP [TCP Dup ACK 78#1] ftp-data [ACK] Seq=439, Ack=2.

Por fim, o servidor, envia um FTP Response, a informar o cliente de que a transferência foi completada.

**Conclusões:**

Em suma, os protocolos de transportes TCP e UDP tem bastantes diferenças. O protocolo TCP - *Transmission Control Protocol,* é caracterizado por ser um protocolo da camada de transpote muito completo, pois tem controlo de fluxo, erros e congestão e não está dependente de um circuito pré-determinado.

Por outro lado, o protocolo UDP - *User Datagram Protocol* é caracterizado por ser um protocolo simples da camada de transporte, pois não tem os controlos tal como o TCP, pelo que o circuito é pré-determinado.

No entanto, o protocolo UDP tem imensas vantagens para aplicações, onde a velocidade de transmissão é importante, como por exemplo streaming, pois não ”perde” tempo em controlo de erros, etc..., com UDP tenho controlo do que mando e quando mando.

A utilização de cada um dos protocolos de transporte, depende do tipo de aplicação e qual a sua finalidade, e assim, com estas informações, consegue-se determinar o protocolo mais adequado.

Por fim, a nível de segurança, descobriu-se que a aplicação mais segura é o SFTP, de seguida o FTP, que usa o protocolo de transporte TCP, depois o HTTP, e por fim o TFTP que não tem segurança nenhum, pelo que usa o protocolo UDP.