Protocolos da Camada de Transporte

Carolina Cunha, Hugo Faria, João Diogo Mota

University of Minho, Department of Informatics, 4710-057 Braga, Portugal
e-mail:{a80142,a81283, a80791}@alunos.uminho.pt

CONTEÚDO

1. QUESTÕES E RESPOSTAS

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:

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	-	-	-	-
traceroute	-	UDP	33445	8 (40 – 32 data)
telnet	-	TCP	23	40 (20 + 20 options)
ftp	FTP	TCP	21	32 (20 + 12 options)
Tftp	TFTP	UDP	69	8 (22 – 14 checksum)
Browser/http	HTTP	TCP	80	32 (20 + 12 options)
nslookup	DNS	UDP	53	8 (39 – 31 checksum)
ssh	SSH	TCP	22	32 (20 + 12 options)
sftp	SSH	TCP	22	32 (20 + 12 options)

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).

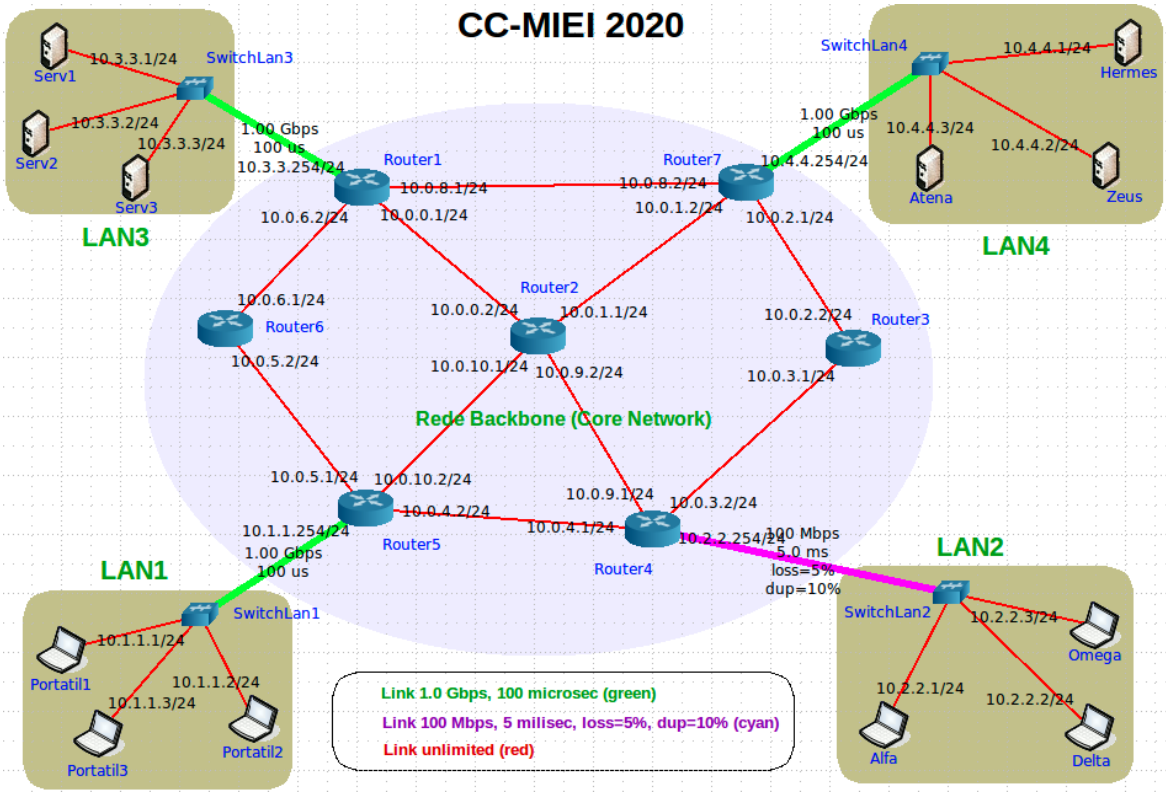
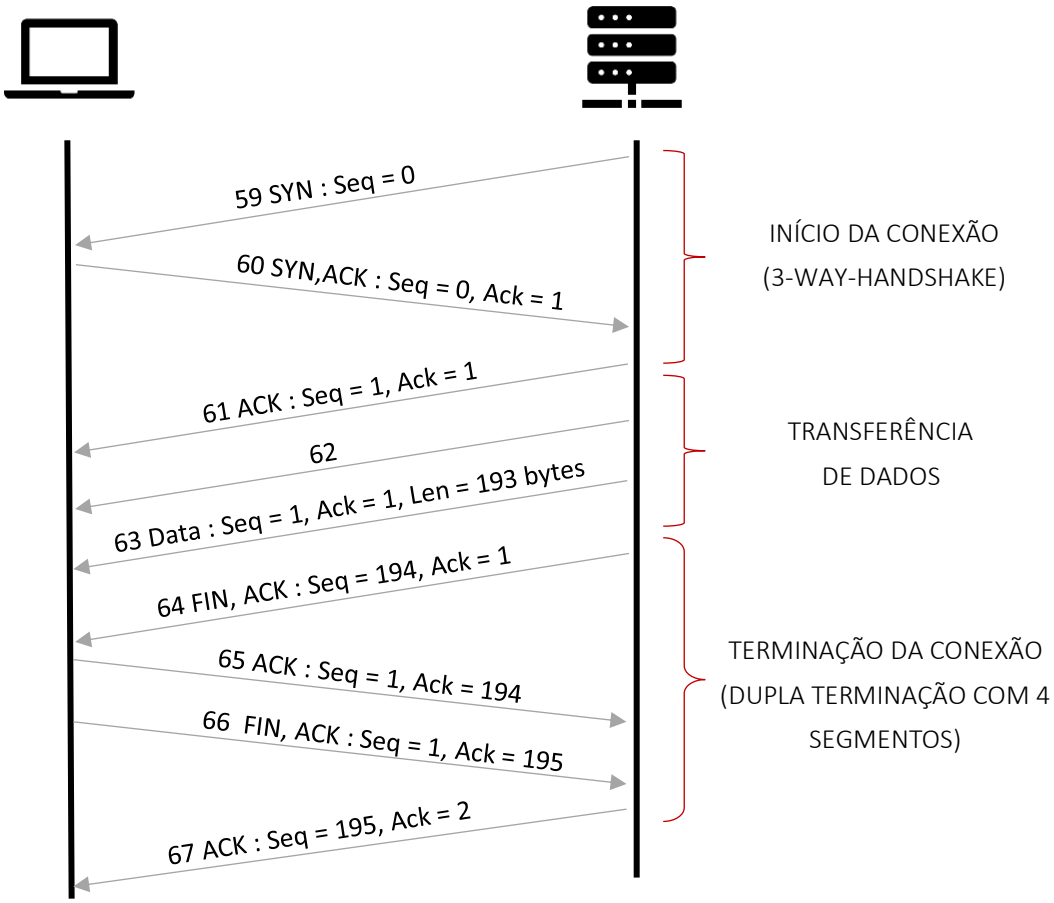


Figura 1: Topologia utilizada

58	78.323766	10.1.1.1	10.3.3.1	FTP	78	Request: RETR file1
59	78.324035	10.3.3.1	10.1.1.1	TCP	74	ftp-data > 56537 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=2102760 TSecr=0 WS=16
60	78.324211	10.1.1.1	10.3.3.1	TCP	74	56537 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=2102760 TSecr=
61	78.324362	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56537 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=2102760 TSecr=2102760
62	78.324433	10.3.3.1	10.1.1.1	FTP	130	Response: 150 Opening BINARY mode data connection for file1 (193 bytes).
63	78.324437	10.3.3.1	10.1.1.1	FTP-DAT	259	FTP Data: 193 bytes
64	78.324491	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56537 [FIN, ACK] Seq=194 Ack=1 Win=14608 Len=0 TSval=2102760 TSecr=2102760
65	78.324815	10.1.1.1	10.3.3.1	TCP	66	56537 > ftp-data [ACK] Seq=1 Ack=194 Win=15552 Len=0 TSval=2102760 TSecr=2102760
66	78.324816	10.1.1.1	10.3.3.1	TCP	66	56537 > ftp-data [FIN, ACK] Seq=1 Ack=195 Win=15552 Len=0 TSval=2102760 TSecr=2102760
67	78.325399	10.3.3.1	10.1.1.1	TCP	66	ftp-data > 56537 [ACK] Seq=195 Ack=2 Win=14608 Len=0 TSval=2102760 TSecr=2102760
68	78.325404	10.3.3.1	10.1.1.1	FTP	90	Response: 226 Transfer complete.

Figura 2: Captura dos pacotes da transferência do file1 por FTP

Diagrama temporal da transferência do file1 por FTP.



A conexão entre o Portatil1 e o Serv1 é estabelecida através de um 3-way-handshake, uma vez que o FTP recorre ao TCP como protocolo de transporte. Dado que o TCP é orientado à conexão, é garantida uma troca fiável e ordenada dos dados.

A observação dos pacotes capturados permite verificar o envio de um *Request* por parte do Portatil1 a requisitar a transferência do *file1*. Este pedido será respondido pelo Serv1 com um pacote [SYN], especificando o número de sequência inicial [Seq = 0] e iniciando a conexão. O Portatil1 recebe o pacote e responde com um segmento [SYN,ACK], em que é alocado espaço de armazenamento e especificado o número de sequência inicial [Seq = 0]. Por fim, o Serv1 recebe o segmento [SYN,ACK], respondendo com um segmento [ACK].

Após concluída esta troca de pacotes, é dada como iniciada a conexão. Os pacotes 62 e 63 representam a transferência do *file1* do Serv1 para o Portatil1. Depois de enviado o ficheiro, o Serv1 indica que quer terminar a conexão. O Portatil1 envia uma resposta a indicar a receção do *file1*, através do [ACK] com Ack = 194, e envia um outro pacote a aceitar e a indicar que quer terminar a conexão. Por fim, o Serv1 responde, aceitando a terminação da conexão.

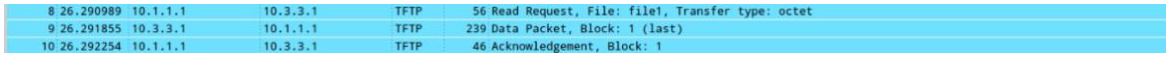
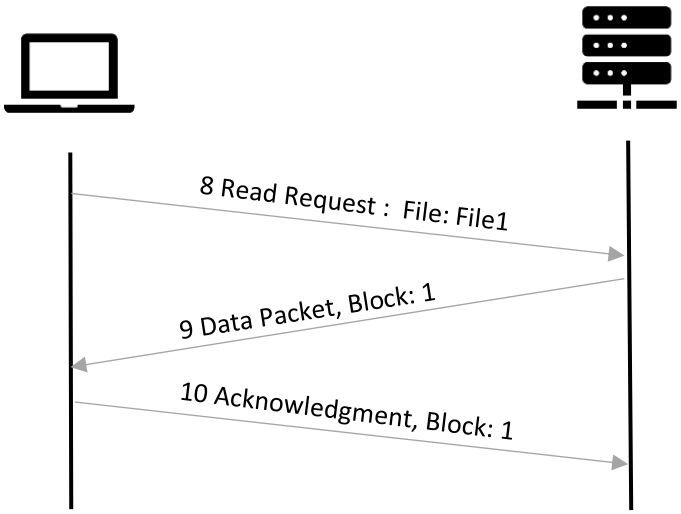


Figura 3: Captura dos pacotes da transferência do file1 por TFTP

Diagrama temporal da transferência do file1 por TFTP.



Quando em comparação com a transferência realizada pelo FTP, a concretizada pelo TFTP é bastante mais simples, uma vez que não existe estabelecimento nem terminação de conexão.

A análise dos pacotes capturados, permite confirmar a simplicidade do TFTP, que recorre ao UDP como protocolo de transporte. Deste modo, o Portatil1 envia um *Read Request*, onde é realizado o pedido de transferência do *file1*. De seguida, o Serv1 envia um pacote *Data Packet*, contendo o ficheiro. Por fim, o Portatil1 responde com um *Acknowledgement,* informando o Serv1 da receção do ficheiro.

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;

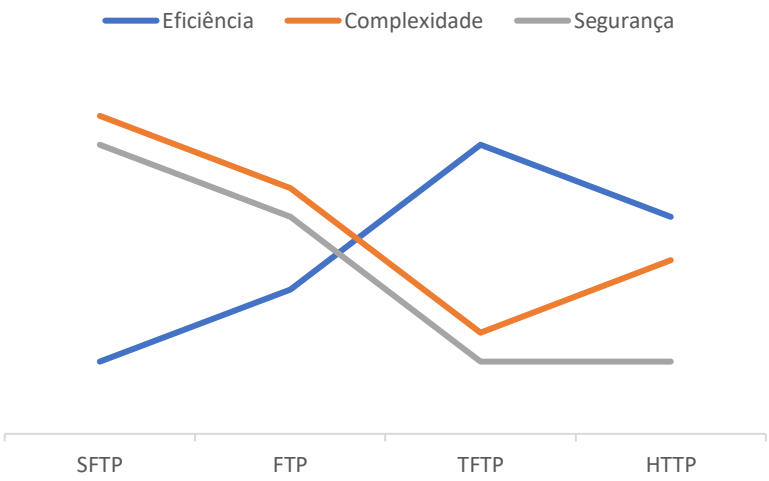
As quatro aplicações de transferência de ficheiros utilizadas foram o **FTP**, **SFTP**, **TFTP** e **HTTP**.

O SFTP tem como protocolo de transporte o TCP e é a aplicação com maior segurança, dado que utiliza meios de autenticação e de encriptação para a transferência de ficheiros (Figura 17). Por este motivo, é também a aplicação mais complexa, o que implica ser a aplicação com menor eficiência.

O FTP é uma aplicação com alguma segurança, uma vez que requer uma autenticação de modo a que seja possível realizar a transferência de ficheiros, mas não realiza qualquer encriptação (Figura 18). Dado que utiliza o TCP como protocolo de transporte, detém alguma complexidade e menor eficiência.

O TFTP é uma aplicação com baixa complexidade, uma vez que não recorre a medidas de segurança e de autenticação e, por esse motivo, apenas necessita da troca de três pacotes para realizar a transferência de ficheiros (Figura 3). Devido à inexistência de métodos de segurança, que torna o TFTP uma aplicação pouco segura, bem como o uso do UDP como protocolo de transporte (não garante a entrega dos dados; não efetua controlo de fluxo ou controlo de conexão), o TFTP é a aplicação mais eficiente na transferência de ficheiros.

Por fim, o HTTP, analogamente ao TFTP, também não usufrui de medidas de segurança e autenticação, pelo que se trata de uma aplicação pouco segura (Figura 19). No entanto, tem como protocolo de transporte o TCP, tendo a capacidade de deteção e correção de erros nos pacotes, sendo mais complexo do que o TFTP. Consequentemente, é menos eficiente do que o TFTP.



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).

   (Nota: Para responder a esta pergunta deve em primeiro lugar efetuar as transferências pedidas no enunciado, quer a partir do sistema Portatil1 na LAN1, quer do sistema Alfa na LAN2, pois só assim poderá ligar esta resposta à prática. Na topologia, o sistema Alfa tem conetividade ao Backbone através de um link que funciona com perdas, atrasos e duplicações, que é o link entre o switch SwitchLan2 e o router Router4. Nos testes podem mesmo ajustar esses parâmetros.)

Assim como indicado na topologia CORE fornecida e apresentada na Figura 1, podemos observar que as seguintes ligações não apresentam qualquer tipo de probabilidade de obtenção de erros de transporte, tal como a comunicação dos *routers* entre si: o SwitchLan1 (presente na LAN1) e o Router5; o SwitchLan3 (que se encontra na LAN3) e Router1; e o SwitchLan4 (visível na LAN4) e Router7. Por outro lado, o *link* entre o Router4 e o SwitchLan2, presente na LAN2, apresenta uma percentagem de perdas de 5% e de duplicações de 10%, sendo por isso, ao contrário das restantes ligações expostas, esperada a presença de erros aquando do envio de pacotes por esta ligação.

Para comprovar e analisar estas possíveis perdas e duplicações, foi enviado um ficheiro (*file2*) a partir do Serv1 (servidor encontrado na LAN3) até Alfa (*host* presente na LAN2), através do comando "wget http://10.3.3.1/file2 " (Figura 15), de modo a que a comunicação passasse pela ligação anteriormente referida, onde poderiam existir problemas. De modo a facilitar a análise, foi iniciada uma captura no *Wireshark* antes de ser introduzido o comando para iniciar a transferência do *file2*. Um excerto da captura pode ser visto na (Figura 13).

Uma vez que a transferência do ficheiro está a ser realizada sobre o protocolo de transporte TCP, é previsto que qualquer erro presente num pacote que seja detetado, será corrigido, uma vez que este protocolo apresenta controlo de erro, fluxo e congestão.

Tal como era expectável, é possível visualizar perdas (veja-se de exemplo o pacote 202 da Figura 13, "TCP Previous Segment Lost"), e ACKs duplicados (sirva de exemplo o pacote 205 da Figura 13, "TCP Dup ACK"), que ocorrem quando o pacote esperado não é recebido (por exemplo, Seq = 114, Ack = 24843 (pacote 201) e Seq = 114, Ack = 24843 (pacote 205)). É ainda possível verificar pacotes retransmitidos por *Fast Retransmission.* Este tipo de retransmissão ocorre quando o emissor recebe três ACKs duplicados e, por este motivo, supõe que o segmento respetivo foi perdido. A existência de pacotes recebidos com ordem trocada ("*Out-Of-Order*", Figura 14) pode igualmente constituir uma complicação na transferência de ficheiros, uma vez que o *buffer* do recetor

irá ficar com “buracos”. A possibilidade de ocorrência destas trocas deve-se ao balanço de tráfego quando se verifica a existência de vários percursos com o mesmo número de saltos desde a origem até ao destino. Por fim, é também visível a retransmissão de pacotes, possivelmente devido à excedência do *timeout* do pacote.

Com base nas capturas observadas, conclui-se que a baixa fiabilidade de um percurso a percorrer pelos pacotes implica que estes estão mais sujeitos a erros (perdas/duplicações). Desta forma, o TCP procura aperceber-se de situações de congestão através da receção de ACKs duplicados e da ocorrência de *timeouts*, recorrendo a mecanismos de prevenção/minimização destas situações.

Uma vez que a ocorrência de erros provoca uma constante retransmissão dos mesmos pacotes, o número de RTT (*Round Trip Time)* vai ser significativamente maior, aumentando o tempo mínimo para o ficheiro transferido ser totalmente recebido. Desta forma, irá ser visível um decréscimo no desempenho global de aplicações fiáveis.

Por outro lado, a transferência do ficheiro *file1* para o Portatil1 (Figura 12: Captura dos pacotes da transferência do file1 por Portatil1 não apresentou quaisquer perdas ou duplicações de pacotes, devido às características das ligações apresentadas na topologia. A análise das Figura 15 e Figura 16 permitiu confirmar que o desempenho global da transferência do ficheiro de maior dimensão (*file2*) é maior quando a transferência é realizada pelo Portatil1 (0.7segundos) do que pelo Alfa (10segundos).

2. CONCLUSÕES

A elaboração deste trabalho prático permitiu aprofundar os conteúdos relativos à camada de transporte e aplicação abordados ao longo das aulas teóricas.

Assim, estudou-se de que modo o TCP (*Transmission Control Protocol*) e o UDP (*User Datagram Protocol*) diferem entre si, bem como as situações em que recorrer a cada um. A complexidade do TCP, que permite o controlo de erros, controlo de fluxo e controlo de congestão, garante a segurança e fiabilidade da troca de dados, permitindo obter a noção de fluxo. Por outro lado, a eficiência do UDP devido à sua baixa complexidade (troca de dados não fiável e desordenada) revela-se de extrema importância para aplicações onde a rapidez é crucial (por exemplo, *streaming*, aplicações em tempo real, entre outros), permitindo às aplicações ter um maior controlo sobre o envio de dados (quando enviar e quantos bytes a enviar).

A consolidação destes conhecimentos permitiu compreender que tipos de protocolos utilizar dependendo da finalidade de cada aplicação, bem como as regras segundo as quais estes protocolos se regem de forma a realizar a transferência de dados.

3. ANEXOS

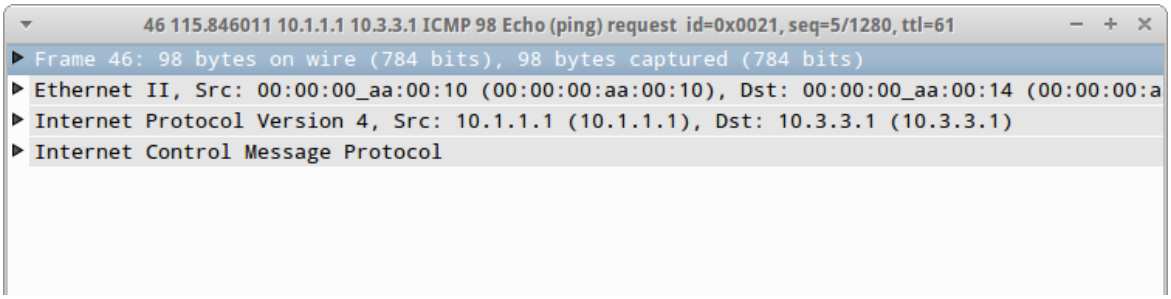


Figura 4: Pacote capturado após comando ping

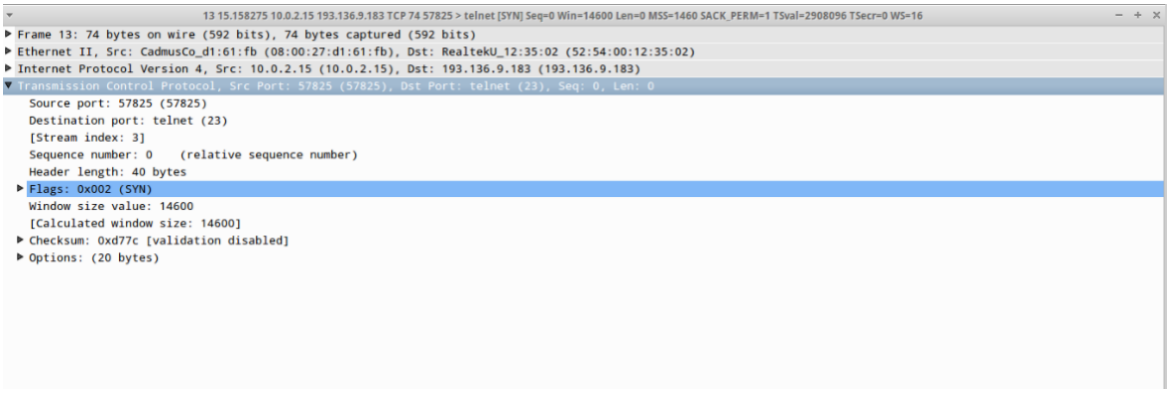


Figura 5: Pacote capturado após comando telnet

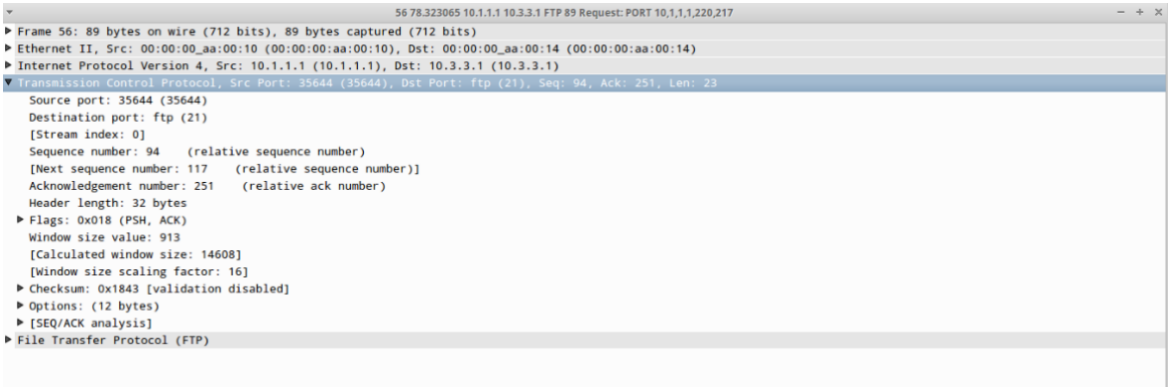


Figura 6: Pacote capturado após comando FTP

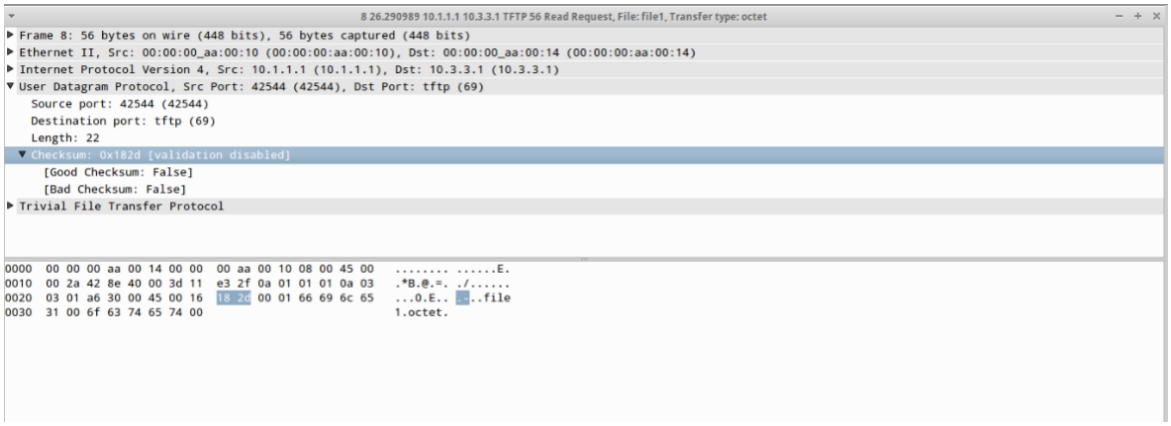


Figura 7: Pacote capturado após comando TFTP

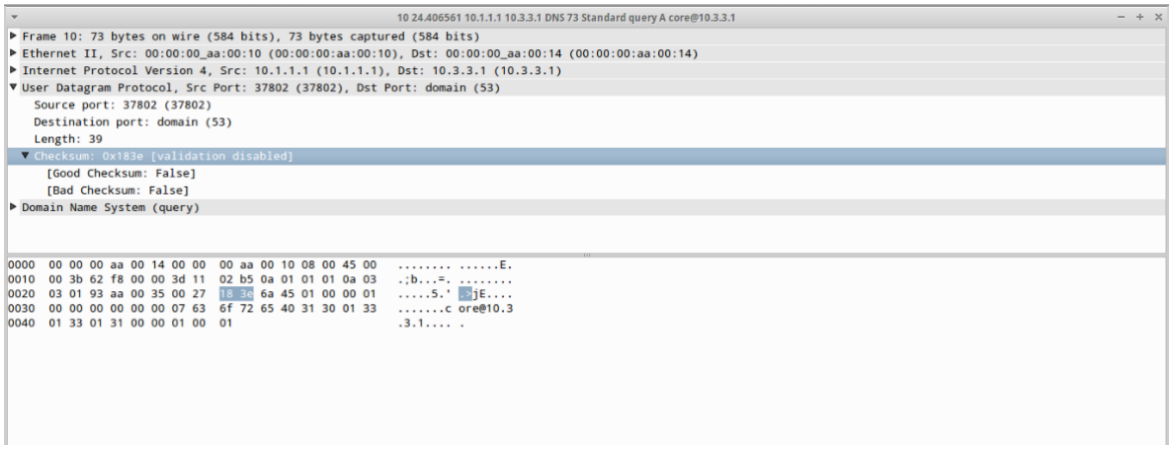


Figura 8: Pacote capturado após comando nslookup

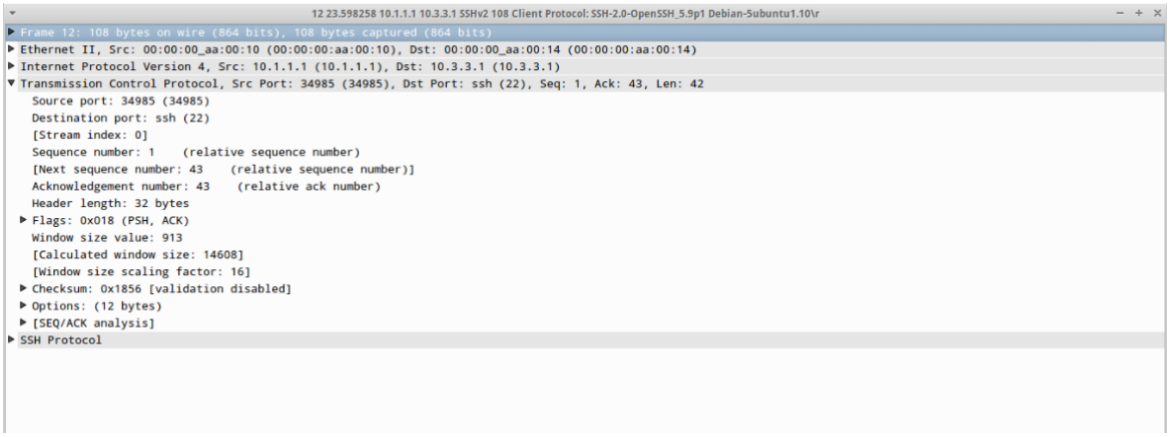


Figura 9: Pacote capturado após comando SSH

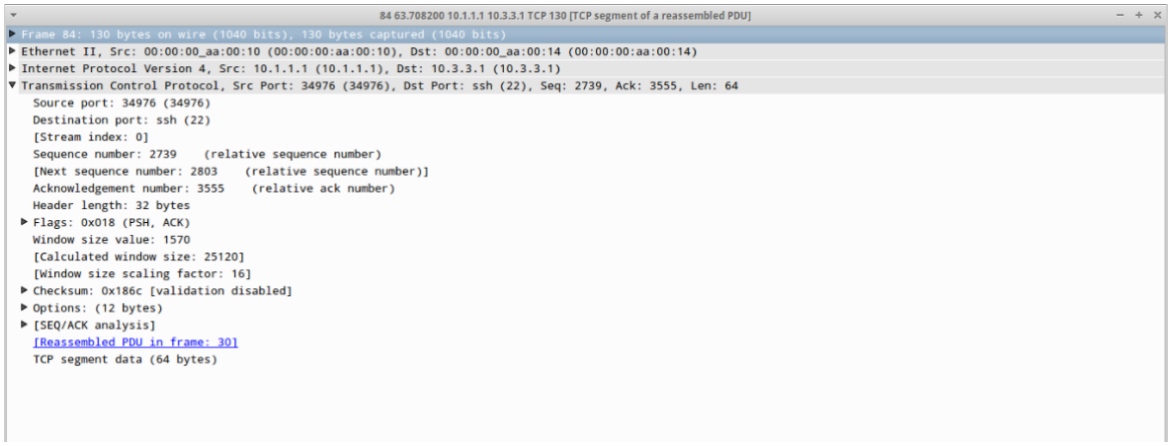


Figura 10: Pacote capturado após comando SFTP

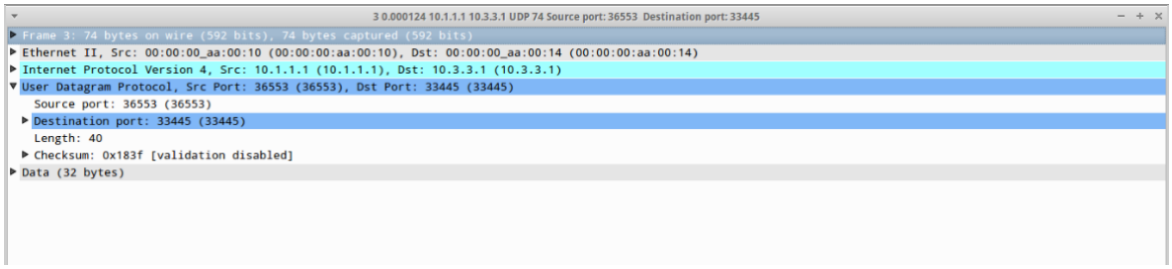


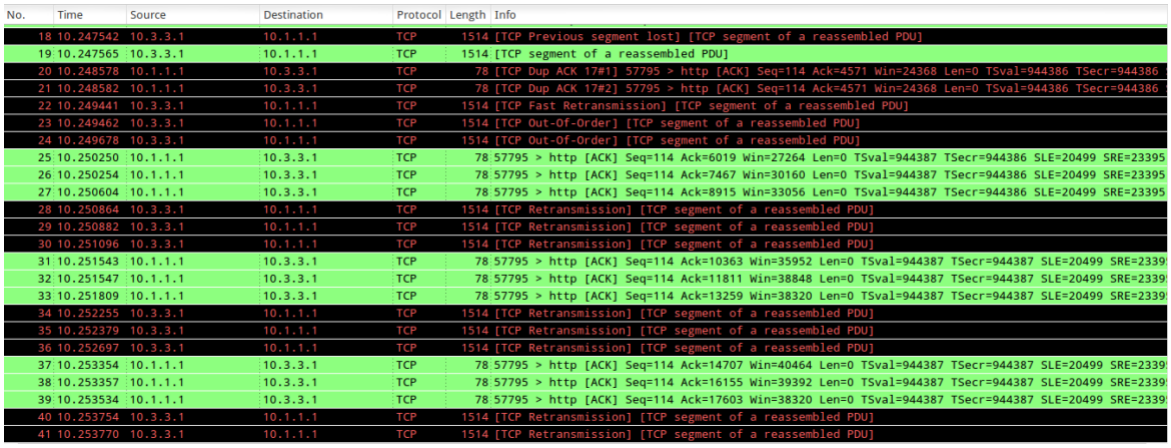
Figura 11: Pacote capturado após comando traceroute

3	7.222716	10.1.1.1	10.3.3.1	TCP	74	57793 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=896117 TSecr=0 WS=16
4	7.223679	10.3.3.1	10.1.1.1	TCP	74	http > 57793 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=896117 TSecr=89611
5	7.223873	10.1.1.1	10.3.3.1	TCP	66	57793 > http [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=896117 TSecr=896117
6	7.224083	10.1.1.1	10.3.3.1	HTTP	179	GET /file1 HTTP/1.1
7	7.224462	10.3.3.1	10.1.1.1	TCP	66	http > 57793 [ACK] Seq=1 Ack=114 Win=14480 Len=0 TSval=896117 TSecr=896117
8	7.224466	10.3.3.1	10.1.1.1	TCP	289	[TCP segment of a reassembled PDU]
9	7.224507	10.3.3.1	10.1.1.1	HTTP	259	HTTP/1.1 200 Ok (text/plain)
10	7.224635	10.1.1.1	10.3.3.1	TCP	66	57793 > http [ACK] Seq=114 Ack=224 Win=15680 Len=0 TSval=896117 TSecr=896117
11	7.225102	10.1.1.1	10.3.3.1	TCP	66	57793 > http [FIN, ACK] Seq=114 Ack=418 Win=16752 Len=0 TSval=896118 TSecr=896117
12	7.225595	10.3.3.1	10.1.1.1	TCP	66	http > 57793 [ACK] Seq=418 Ack=115 Win=14480 Len=0 TSval=896118 TSecr=896118

Figura 12: Captura dos pacotes da transferência do file1 por Portatil1

200	17.998463	10.2.2.1	10.3.3.1	TCP	66	58322 > http [ACK] Seq=114 Ack=23395 Win=39392 Len=0 TSval=946318 TSecr=946318
201	17.999766	10.2.2.1	10.3.3.1	TCP	66	58322 > http [ACK] Seq=114 Ack=24843 Win=38320 Len=0 TSval=946318 TSecr=946318
202	17.999905	10.3.3.1	10.2.2.1	TCP	1514	[TCP Previous segment lost] [TCP segment of a reassembled PDU]
203	18.000131	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]
204	18.011139	10.2.2.1	10.3.3.1	TCP	78	[TCP Window Update] 58322 > http [ACK] Seq=114 Ack=24843 Win=41744 Len=0 TSval=946320 TSecr=94631
205	18.011144	10.2.2.1	10.3.3.1	TCP	78	[TCP Dup ACK 204#1] 58322 > http [ACK] Seq=114 Ack=24843 Win=41744 Len=0 TSval=946320 TSecr=94631
206	18.011604	10.2.2.1	10.3.3.1	TCP	78	[TCP Dup ACK 204#2] 58322 > http [ACK] Seq=114 Ack=24843 Win=41744 Len=0 TSval=946320 TSecr=94631
207	18.012763	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]
208	18.012793	10.3.3.1	10.2.2.1	TCP	1514	[TCP Fast Retransmission] [TCP segment of a reassembled PDU]
209	18.020616	10.2.2.1	10.3.3.1	TCP	78	58322 > http [ACK] Seq=114 Ack=26291 Win=40304 Len=0 TSval=946327 TSecr=946327 SLE=27739 SRE=3208
210	18.021238	10.3.3.1	10.2.2.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
211	18.021251	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]
212	18.029095	10.2.2.1	10.3.3.1	TCP	66	58322 > http [ACK] Seq=114 Ack=32083 Win=39392 Len=0 TSval=946330 TSecr=946329
213	18.029097	10.2.2.1	10.3.3.1	TCP	66	58322 > http [ACK] Seq=114 Ack=33531 Win=38320 Len=0 TSval=946330 TSecr=946329
214	18.036336	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]
215	18.047937	10.2.2.1	10.3.3.1	TCP	66	58322 > http [ACK] Seq=114 Ack=34979 Win=42256 Len=0 TSval=946333 TSecr=946331
216	18.048416	10.3.3.1	10.2.2.1	TCP	1514	[TCP Previous segment lost] [TCP segment of a reassembled PDU]
217	18.058919	10.2.2.1	10.3.3.1	TCP	78	[TCP Dup ACK 215#1] 58322 > http [ACK] Seq=114 Ack=34979 Win=42256 Len=0 TSval=946336 TSecr=94633
218	18.058921	10.2.2.1	10.3.3.1	TCP	78	[TCP Dup ACK 215#2] 58322 > http [ACK] Seq=114 Ack=34979 Win=42256 Len=0 TSval=946336 TSecr=94633
219	18.068912	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]
220	18.079149	10.2.2.1	10.3.3.1	TCP	78	[TCP Dup ACK 215#3] 58322 > http [ACK] Seq=114 Ack=34979 Win=42256 Len=0 TSval=946341 TSecr=94633
221	18.104529	10.3.3.1	10.2.2.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
222	18.126855	10.2.2.1	10.3.3.1	TCP	78	58322 > http [ACK] Seq=114 Ack=36427 Win=40816 Len=0 TSval=946350 TSecr=946344 SLE=37875 SRE=4077
223	18.179421	10.3.3.1	10.2.2.1	TCP	1514	[TCP segment of a reassembled PDU]

Figura 13: Excerto da captura dos pacotes da transferência do file1 por Alfa



No.	Time	Source	Destination	Protocol	Length	Info
18	10.247542	10.3.3.1	10.1.1.1	TCP	1514	[TCP Previous segment lost] [TCP segment of a reassembled PDU]
19	10.247565	10.3.3.1	10.1.1.1	TCP	1514	[TCP segment of a reassembled PDU]
20	10.248578	10.1.1.1	10.3.3.1	TCP	78	[TCP Dup ACK 17#1] 57795 > http [ACK] Seq=114 Ack=4571 Win=24368 Len=0 TSval=944386 TSecr=944386
21	10.248582	10.1.1.1	10.3.3.1	TCP	78	[TCP Dup ACK 17#2] 57795 > http [ACK] Seq=114 Ack=4571 Win=24368 Len=0 TSval=944386 TSecr=944386
22	10.249441	10.3.3.1	10.1.1.1	TCP	1514	[TCP Fast Retransmission] [TCP segment of a reassembled PDU]
23	10.249462	10.3.3.1	10.1.1.1	TCP	1514	[TCP Out-Of-Order] [TCP segment of a reassembled PDU]
24	10.249678	10.3.3.1	10.1.1.1	TCP	1514	[TCP Out-Of-Order] [TCP segment of a reassembled PDU]
25	10.250250	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=6019 Win=27264 Len=0 TSval=944387 TSecr=944386 SLE=20499 SRE=23395
26	10.250254	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=7467 Win=30160 Len=0 TSval=944387 TSecr=944386 SLE=20499 SRE=23395
27	10.250604	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=8915 Win=33056 Len=0 TSval=944387 TSecr=944386 SLE=20499 SRE=23395
28	10.250864	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
29	10.250882	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
30	10.251096	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
31	10.251543	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=10363 Win=35952 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
32	10.251547	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=11811 Win=38848 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
33	10.251809	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=13259 Win=38320 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
34	10.252255	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
35	10.252379	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
36	10.252697	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
37	10.253354	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=14707 Win=40464 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
38	10.253357	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=16155 Win=39392 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
39	10.253534	10.1.1.1	10.3.3.1	TCP	78	57795 > http [ACK] Seq=114 Ack=17603 Win=38320 Len=0 TSval=944387 TSecr=944387 SLE=20499 SRE=2339
40	10.253754	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]
41	10.253770	10.3.3.1	10.1.1.1	TCP	1514	[TCP Retransmission] [TCP segment of a reassembled PDU]

Figura 14: Excerto 2 da captura dos pacotes da transferência do file1 por Alfa

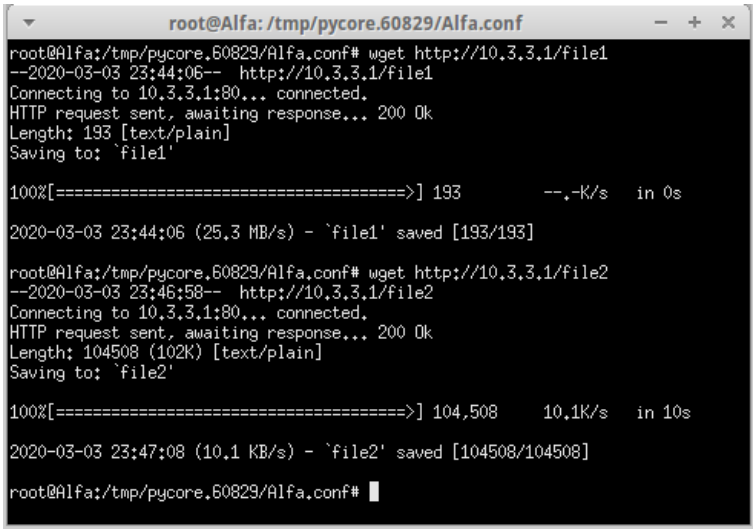


Figura 15: Transferência dos file1 e file2 por Alfa

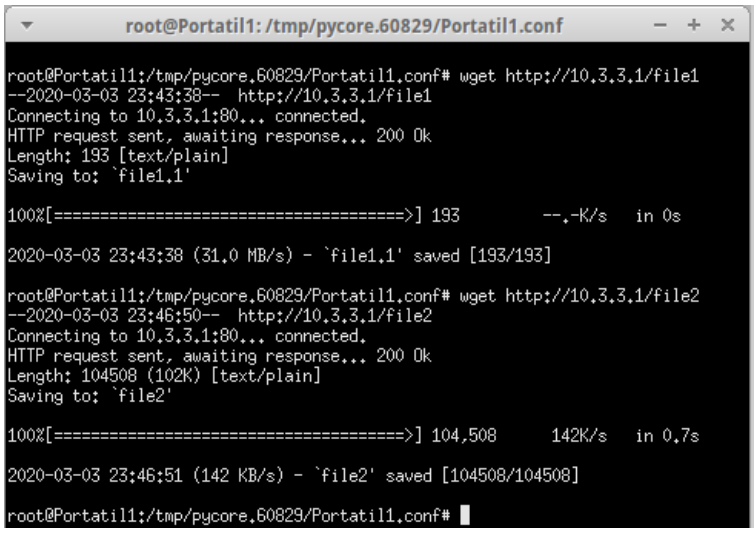
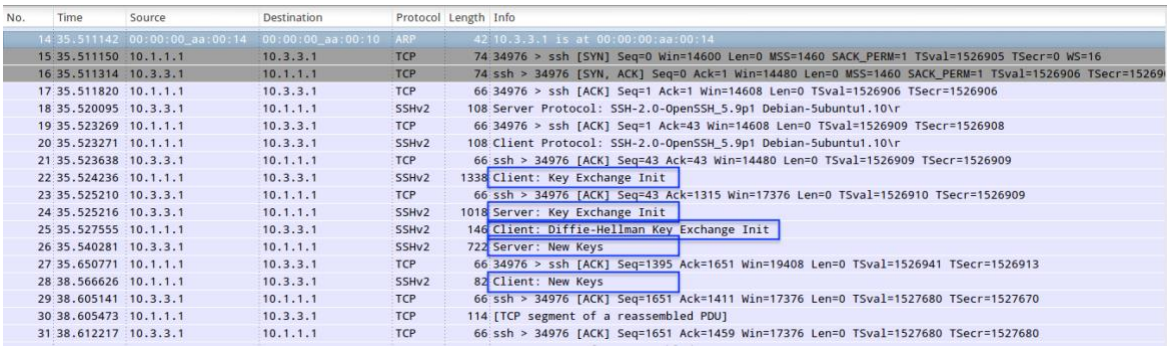
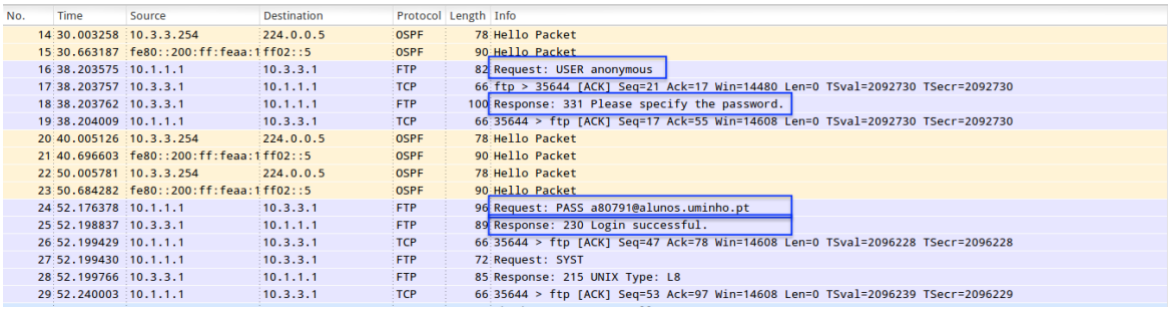


Figura 16: Transferência dos file1 e file2 por Portatil1



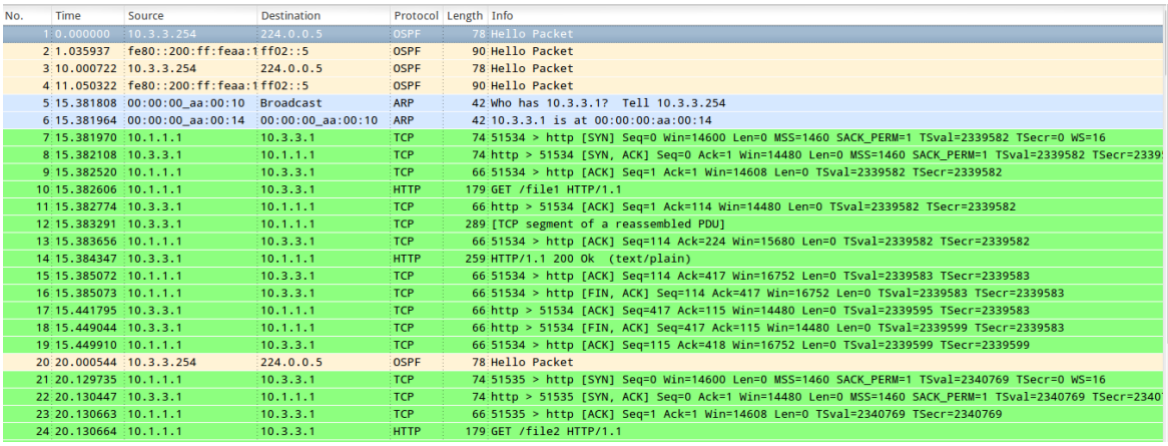
No.	Time	Source	Destination	Protocol	Length	Info
14	35.511142	00:00:00_aa:00:14	00:00:00_aa:00:10	ARP	42	10.3.3.1 is at 00:00:00:aa:00:14
15	35.511150	10.1.1.1	10.3.3.1	TCP	74	34976 > ssh [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=1526905 TSecr=0 WS=16
16	35.511314	10.3.3.1	10.1.1.1	TCP	74	ssh > 34976 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=1526906 TSecr=15269
17	35.511820	10.1.1.1	10.3.3.1	TCP	66	34976 > ssh [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=1526906 TSecr=1526906
18	35.520095	10.3.3.1	10.1.1.1	SSHv2	108	Server Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.10\r
19	35.523269	10.1.1.1	10.3.3.1	TCP	66	34976 > ssh [ACK] Seq=1 Ack=43 Win=14608 Len=0 TSval=1526909 TSecr=1526908
20	35.523271	10.1.1.1	10.3.3.1	SSHv2	108	Client Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.10\r
21	35.523638	10.3.3.1	10.1.1.1	TCP	66	ssh > 34976 [ACK] Seq=43 Ack=43 Win=14480 Len=0 TSval=1526909 TSecr=1526909
22	35.524236	10.1.1.1	10.3.3.1	SSHv2	1338	Client: Key Exchange Init
23	35.525210	10.3.3.1	10.1.1.1	TCP	66	ssh > 34976 [ACK] Seq=43 Ack=1315 Win=17376 Len=0 TSval=1526910 TSecr=1526909
24	35.525216	10.3.3.1	10.1.1.1	SSHv2	1018	Server: Key Exchange Init
25	35.527555	10.1.1.1	10.3.3.1	SSHv2	146	Client: Diffie-Hellman Key Exchange Init
26	35.540281	10.3.3.1	10.1.1.1	SSHv2	722	Server: New Keys
27	35.650771	10.1.1.1	10.3.3.1	TCP	66	34976 > ssh [ACK] Seq=1395 Ack=1651 Win=19408 Len=0 TSval=1526941 TSecr=1526913
28	38.566626	10.1.1.1	10.3.3.1	SSHv2	82	Client: New Keys
29	38.605141	10.3.3.1	10.1.1.1	TCP	66	ssh > 34976 [ACK] Seq=1651 Ack=1411 Win=17376 Len=0 TSval=1527680 TSecr=1527670
30	38.605473	10.1.1.1	10.3.3.1	TCP	114	[TCP segment of a reassembled PDU]
31	38.612217	10.3.3.1	10.1.1.1	TCP	66	ssh > 34976 [ACK] Seq=1651 Ack=1459 Win=17376 Len=0 TSval=1527680 TSecr=1527680

Figura 17: Segurança STFP



No.	Time	Source	Destination	Protocol	Length	Info
14	30.003258	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
15	30.663187	fe80::200:ff:feaa:1	ff02::5	OSPF	90	Hello Packet
16	38.203575	10.1.1.1	10.3.3.1	FTP	82	Request: USER anonymous
17	38.203757	10.3.3.1	10.1.1.1	TCP	66	ftp > 35644 [ACK] Seq=21 Ack=17 Win=14480 Len=0 TSval=2092730 TSecr=2092730
18	38.203762	10.3.3.1	10.1.1.1	FTP	100	Response: 331 Please specify the password.
19	38.204009	10.1.1.1	10.3.3.1	TCP	66	35644 > ftp [ACK] Seq=17 Ack=55 Win=14608 Len=0 TSval=2092730 TSecr=2092730
20	40.005126	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
21	40.696603	fe80::200:ff:feaa:1	ff02::5	OSPF	90	Hello Packet
22	50.005781	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
23	50.684282	fe80::200:ff:feaa:1	ff02::5	OSPF	90	Hello Packet
24	52.176378	10.1.1.1	10.3.3.1	FTP	96	Request: PASS a80791@alunos.uminho.pt
25	52.198837	10.3.3.1	10.1.1.1	FTP	89	Response: 230 Login successful.
26	52.199429	10.1.1.1	10.3.3.1	TCP	66	35644 > ftp [ACK] Seq=47 Ack=78 Win=14608 Len=0 TSval=2096228 TSecr=2096228
27	52.199430	10.1.1.1	10.3.3.1	FTP	72	Request: SYST
28	52.199766	10.3.3.1	10.1.1.1	FTP	85	Response: 215 UNIX Type: L8
29	52.240003	10.1.1.1	10.3.3.1	TCP	66	35644 > ftp [ACK] Seq=53 Ack=97 Win=14608 Len=0 TSval=2096239 TSecr=2096229

Figura 18: Segurança FTP



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
2	1.035937	fe80::200:ff:feaa:1	ff02::5	OSPF	90	Hello Packet
3	10.000722	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
4	11.050322	fe80::200:ff:feaa:1	ff02::5	OSPF	90	Hello Packet
5	15.381808	00:00:00_aa:00:10	Broadcast	ARP	42	Who has 10.3.3.1? Tell 10.3.3.254
6	15.381964	00:00:00_aa:00:14	00:00:00_aa:00:10	ARP	42	10.3.3.1 is at 00:00:00:aa:00:14
7	15.381970	10.1.1.1	10.3.3.1	TCP	74	51534 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=2339582 TSecr=0 WS=16
8	15.382108	10.3.3.1	10.1.1.1	TCP	74	http > 51534 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=2339582 TSecr=2339
9	15.382520	10.1.1.1	10.3.3.1	TCP	66	51534 > http [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=2339582 TSecr=2339582
10	15.382606	10.1.1.1	10.3.3.1	HTTP	179	GET /file1 HTTP/1.1
11	15.382774	10.3.3.1	10.1.1.1	TCP	66	http > 51534 [ACK] Seq=1 Ack=114 Win=14480 Len=0 TSval=2339582 TSecr=2339582
12	15.383291	10.3.3.1	10.1.1.1	TCP	289	[TCP segment of a reassembled PDU]
13	15.383656	10.1.1.1	10.3.3.1	TCP	66	51534 > http [ACK] Seq=114 Ack=224 Win=15680 Len=0 TSval=2339582 TSecr=2339582
14	15.384347	10.3.3.1	10.1.1.1	HTTP	259	HTTP/1.1 200 Ok (text/plain)
15	15.385072	10.1.1.1	10.3.3.1	TCP	66	51534 > http [ACK] Seq=114 Ack=417 Win=16752 Len=0 TSval=2339583 TSecr=2339583
16	15.385073	10.1.1.1	10.3.3.1	TCP	66	51534 > http [FIN, ACK] Seq=114 Ack=417 Win=16752 Len=0 TSval=2339583 TSecr=2339583
17	15.441795	10.3.3.1	10.1.1.1	TCP	66	http > 51534 [ACK] Seq=417 Ack=115 Win=14480 Len=0 TSval=2339595 TSecr=2339583
18	15.449044	10.3.3.1	10.1.1.1	TCP	66	http > 51534 [FIN, ACK] Seq=417 Ack=115 Win=14480 Len=0 TSval=2339599 TSecr=2339583
19	15.449910	10.1.1.1	10.3.3.1	TCP	66	51534 > http [ACK] Seq=115 Ack=418 Win=16752 Len=0 TSval=2339599 TSecr=2339599
20	20.000544	10.3.3.254	224.0.0.5	OSPF	78	Hello Packet
21	20.129735	10.1.1.1	10.3.3.1	TCP	74	51535 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=2340769 TSecr=0 WS=16
22	20.130447	10.3.3.1	10.1.1.1	TCP	74	http > 51535 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=2340769 TSecr=2340
23	20.130663	10.1.1.1	10.3.3.1	TCP	66	51535 > http [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=2340769 TSecr=2340769
24	20.130664	10.1.1.1	10.3.3.1	HTTP	179	GET /file2 HTTP/1.1

Figura 19: Transferência file1 HTTP