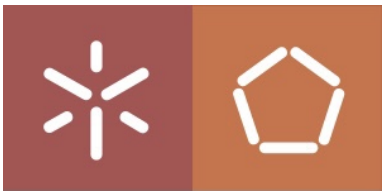


Comunicações por Computador

22/23

TP1: Protocolos da Camada de Transporte

Grupo 6.02

13 de outubro de 2022

Patrícia Pereira (A89578) Meriem Khammassi (A85829)

Índice

Lista de Figuras

1 Parte A

A primeira parte do Trabalho consistiu na configuração e utilização de serviços de transferência de ficheiros.

Para a realização do mesmo utilizamos, usando a *VirtualBox*, a máquina virtual fornecida pelos docentes *XubunCORE_7_5*, que contém o emulador core que nos permite realizar este trabalho prático.

À medida que executamos os comandos enumerados no enunciado, usamos o *wireshark*, para capturar o tráfego em certos nodos da topologia *CC-Topo-2022-2023.imn*. Sendo assim possível responder às questões que se encontram na Parte B deste relatório.

2 Parte B

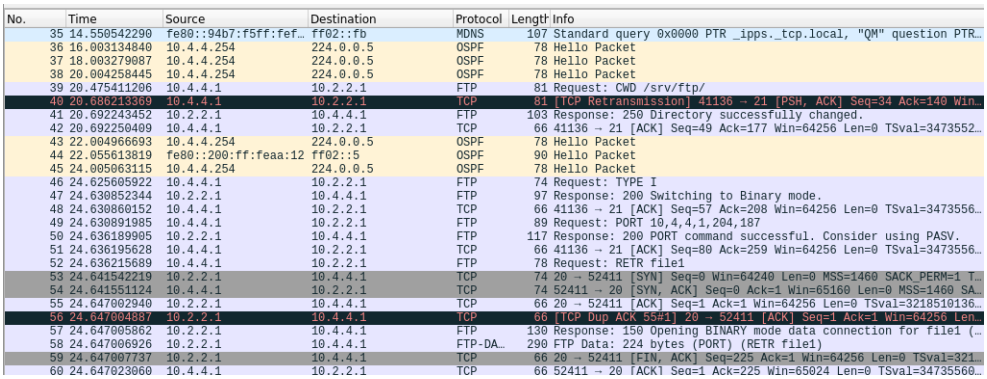
2.1 Questão 1

De que forma as perdas e duplicações de pacotes afetaram o desempenho das aplicações? Que camada lidou com esses problemas: transporte ou aplicação? Responda com base nas experiências feitas e nos resultados observados.

Cada vez que acontece uma perda de um pacote, o protocolo TCP obriga a que um novo pacote seja enviado, causando um atraso na chegada e no processamento de dados. Isto motiva a que haja overhead associado e alterações no desempenho.

Posto isto, podemos afirmar que a camada que lida com as perdas é a camada de transporte.

Na figura seguinte podemos observar que aconteceu, durante a a captura de tráfego no portátil Grilo, uma duplicação e uma retransmissão, pelo protocolo TCP.

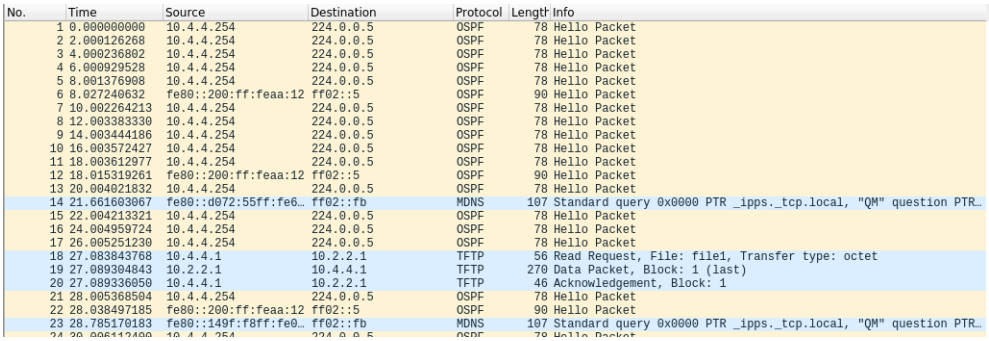


No.	Time	Source	Destination	Protocol	Length	Info
35	14.550542290	fe80::94b7:f5ff:fef…	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR…
36	16.003134840	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
37	18.003279087	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
38	20.004258445	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
39	20.475411206	10.4.4.1	10.2.2.1	FTP	81	Request: CWD /srv/ftp/
40	20.680213369	10.4.4.1	10.2.2.1	TCP	81	[TCP Retransmission] 41136 → 21 [PSH, ACK] Seq=34 Ack=140 Win…
41	20.692243452	10.2.2.1	10.4.4.1	FTP	103	Response: 250 Directory successfully changed.
42	20.692250409	10.4.4.1	10.2.2.1	TCP	66	41136 → 21 [ACK] Seq=49 Ack=177 Win=64256 Len=0 TSval=3473552…
43	22.004966693	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
44	22.055613819	fe80::200:ff:feaa:12	ff02::5	OSPF	90	Hello Packet
45	24.005063115	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
46	24.625605922	10.4.4.1	10.2.2.1	FTP	74	Request: TYPE I
47	24.630852344	10.2.2.1	10.4.4.1	FTP	97	Response: 200 Switching to Binary mode.
48	24.630860152	10.4.4.1	10.2.2.1	TCP	66	41136 → 21 [ACK] Seq=57 Ack=208 Win=64256 Len=0 TSval=3473556…
49	24.630891985	10.4.4.1	10.2.2.1	FTP	89	Request: PORT 10,4,4,1,204,187
50	24.636189905	10.2.2.1	10.4.4.1	FTP	117	Response: 200 PORT command successful. Consider using PASV.
51	24.636195628	10.4.4.1	10.2.2.1	TCP	66	41136 → 21 [ACK] Seq=80 Ack=259 Win=64256 Len=0 TSval=3473556…
52	24.636215689	10.4.4.1	10.2.2.1	FTP	78	Request: RETR file1
53	24.641542219	10.2.2.1	10.4.4.1	TCP	74	20 → 52411 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
54	24.641551124	10.4.4.1	10.2.2.1	TCP	74	52411 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
55	24.647002940	10.2.2.1	10.4.4.1	TCP	66	20 → 52411 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3218510136…
56	24.647004887	10.2.2.1	10.4.4.1	TCP	66	[TCP Dup ACK 55#1] 20 → 52411 [ACK] Seq=1 Ack=1 Win=64256 Len…
57	24.647005862	10.2.2.1	10.4.4.1	FTP	130	Response: 150 Opening BINARY mode data connection for file1 (…
58	24.647006926	10.2.2.1	10.4.4.1	FTP-DA…	290	FTP Data: 224 bytes (PORT) (RETR file1)
59	24.647007737	10.2.2.1	10.4.4.1	TCP	66	20 → 52411 [FIN, ACK] Seq=225 Ack=1 Win=64256 Len=0 TSval=321…
60	24.647023060	10.4.4.1	10.2.2.1	TCP	66	52411 → 20 [ACK] Seq=1 Ack=225 Win=65024 Len=0 TSval=34735560…

Figura 1: Captura de tráfego no portátil Grilo usando FTP

No entanto ao utilizar TFTP, protocolo UDP, não há confirmação de receção do pacote, sendo que há possibilidade de não chegar ao destino a totalidade da informação.

Não chegando toda a informação pretendida, as perdas e duplicações de pacotes terão de ser tratadas pela camada de aplicação, afetando de forma negativa o desempenho.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
2	2.000126268	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
3	4.000236802	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
4	6.000929528	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
5	8.001376908	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
6	8.027240632	fe80::200:ff:feaa:12	ff02::5	OSPF	90	Hello Packet
7	10.002264213	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
8	12.003383330	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
9	14.003444186	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
10	16.003572427	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
11	18.003612977	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
12	18.015319261	fe80::200:ff:feaa:12	ff02::5	OSPF	90	Hello Packet
13	20.004021832	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
14	21.661603067	fe80::d072:55ff:fe6…	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR…
15	22.004213321	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
16	24.004959724	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
17	26.005251230	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
18	27.083843768	10.4.4.1	10.2.2.1	TFTP	56	Read Request, File: file1, Transfer type: octet
19	27.089304843	10.2.2.1	10.4.4.1	TFTP	270	Data Packet, Block: 1 (last)
20	27.089336050	10.4.4.1	10.2.2.1	TFTP	46	Acknowledgement, Block: 1
21	28.005368504	10.4.4.254	224.0.0.5	OSPF	78	Hello Packet
22	28.038497185	fe80::200:ff:feaa:12	ff02::5	OSPF	90	Hello Packet
23	28.785170183	fe80::149f:f8ff:fe0…	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR…

Figura 2: Captura de tráfego no portátil Grilo usando TFTP

2.2 Questão 2

Obtenha a partir do Wireshark, ou desenhe manualmente, um diagrama temporal para a transferência do ficheiro file1 por FTP realizada em A.3. Foque-se apenas na transferência de dados [ftp-data] e não na conexão de controlo (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 tanto nos dados como nas confirmações.

O serviço FTP utiliza a porta 20 para lidar com dados, e a porta 21 para estabelecer conexão. Assim de forma a focar-nos na transferência de dados, no tráfego capturado filtramos a porta 20.

tcp.port==20

No.	Time	Source	Destination	Protocol	Length	Info
53	24.641542219	10.2.2.1	10.4.4.1	TCP	74	20 → 52411 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
54	24.641551124	10.4.4.1	10.2.2.1	TCP	74	52411 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
55	24.647002940	10.2.2.1	10.4.4.1	TCP	66	20 → 52411 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3218510136…
56	24.647004887	10.2.2.1	10.4.4.1	TCP	66	[TCP Dup ACK 55#1] 20 → 52411 [ACK] Seq=1 Ack=1 Win=64256 Len…
58	24.647006926	10.2.2.1	10.4.4.1	FTP-DA…	290	FTP Data: 224 bytes (PORT) (RETR file1)
59	24.647007737	10.2.2.1	10.4.4.1	TCP	66	20 → 52411 [FIN, ACK] Seq=225 Ack=1 Win=64256 Len=0 TSval=321…
60	24.647023060	10.4.4.1	10.2.2.1	TCP	66	52411 → 20 [ACK] Seq=1 Ack=225 Win=65024 Len=0 TSval=34735560…
61	24.647192219	10.4.4.1	10.2.2.1	TCP	66	52411 → 20 [FIN, ACK] Seq=1 Ack=226 Win=65024 Len=0 TSval=347…
62	24.652708338	10.2.2.1	10.4.4.1	TCP	66	20 → 52411 [ACK] Seq=226 Ack=2 Win=64256 Len=0 TSval=32185101…

Figura 3: Captura de tráfego na porta 20 usando FTP

Com estes resultados podemos obter o seguinte gráfico temporal.

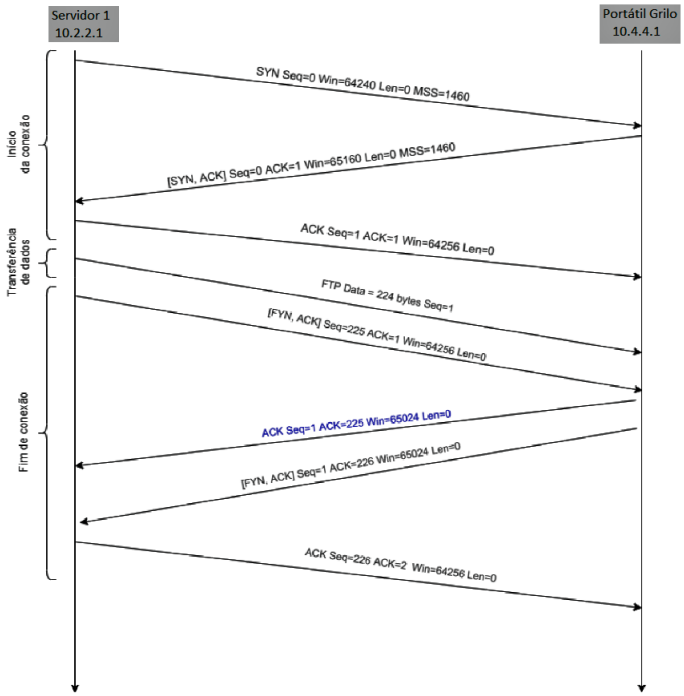


Figura 4: Diagrama Temporal

2.3 Questão 3

Obtenha a partir do Wireshark, ou desenhe manualmente, um diagrama temporal para a transferência do ficheiro file1 por TFTP realizada em A.4. 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 tanto nos dados como nas confirmações.

Para obter os resultados desta questão utilizamos a mesma técnica da Questão anterior, isto é, aplicar o filtro *tcp.port==20*. Com o resultado podemos obter o seguinte gráfico temporal.

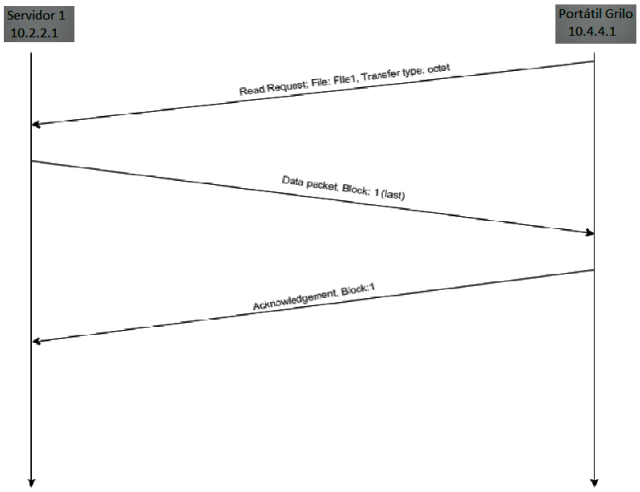


Figura 5: Diagrama Temporal

2.4 Questão 4

Compare sucintamente as quatro aplicações de transferência de ficheiros que usou, tendo em consideração os seguintes aspetos: (i) identificação da camada de transporte; (ii) eficiência; (iii) complexidade; (iv) segurança.;

(i) **identificação da camada de transporte**

- **SFTP**: Utiliza o protocolo TCP
- **FTP**: Utiliza o protocolo TCP
- **TFTP**: Utiliza o protocolo UDP
- **HTTP**: Utiliza o protocolo TCP

(ii) **Eficiência**

- **FTP**: Uma vez que é utilizado o protocolo TCP, é garantido que, através do uso de pacotes *acknowledge*, o segmento será transmitido. Contudo, há uma perda de eficiência, visto que é necessário esperar pelo *acknowledge* para continuar.
- **SFTP**: Idêntico ao FTP, mas os dados são encriptados.
- **TFTP**: Em consequência do uso do protocolo UDP por parte desta aplicação, esta torna-se menos viável. Não sendo, por essa razão, possível averiguar se o pacote foi entregue com sucesso ou não, sendo necessário, por vezes, a retransmissão dos mesmos. No entanto, em caso de sucesso, este é mais o rápido que o FTP.
- **HTTP**: Este protocolo permite que vários HTTP requests sejam enviados numa única ligação TCP, sem que seja necessário esperar pelas respostas correspondentes.

(iii) **Complexidade**

- **SFTP**: Uma vez que o protocolo SFTP é muito fiável, e possibilita encriptação, transferência e gestão de dados, exige overhead, este protocolo revela-se muito complexo.
- **FTP**: Em virtude de que o protocolo FTP é capaz de suportar múltiplos pedidos de transferência de dados concorrentemente em que realiza uma nova conexão para cada uma das transferências, é necessário que existam diferentes velocidades de transferência. A elevada frequência de novas conexões torna este protocolo bastante complexo.
- **TFTP**: Através do nome deste protocolo TFTP, podemos concluir que este é uma alternativa simplificada do protocolo anterior. O protocolo TFTP, para além de ser mais simples, suporta muito menos funcionalidades do que a versão com maior complexidade. Para além disto, baseando-se também no facto deste protocolo ser não orientado à conexão, este protocolo é simples.
- **HTTP** Mesmo com mais complexidade introduzida no HTTP/2.0 por encapsular mensagens HTTP em quadros (frames), o HTTP foi projetado para ser simples e legível às pessoas, trabalhando com *Request's* e *Response's* e não tendo encriptação.

(iv) **Segurança**

- **SFTP:** O SFTP oferece uma proteção extra aos arquivos e alterações feitas na hospedagem. Nada obstante, o SFTP utiliza-se da tecnologia SSH para autenticar o contacto e estabelecer conexões seguras entre as máquinas. O SSH usa uma arquitetura em camadas, em termos de segurança, a camada de transporte fornece a encriptação de dados e a autenticação do servidor. A camada de autenticação está encarregue de manusear a autenticar o utilizador, assim, afirma-se que este protocolo consegue garantir uma elevada segurança.
- **FTP:** Este protocolo utiliza autenticação, não proporcionando encriptação de dados, tornando-o vulnerável a ter bastantes falhas na segurança. Como resultado, e devido às transmissões não serem encriptadas, este protocolo é extremamente inseguro.
- **TFTP:** Protocolo que não fornece autenticação. Posto isto, como não protege os dados a serem transferidos, é considerado relativamente inseguro.
- **HTTP:** Protocolo da camada de aplicação que não é encriptado, tendo a informação representada em texto. Por esta razão, embora utilize autenticação, é vulnerável a adúlteros dos dados, não garantindo segurança a esse nível.

2.5 Questão 5

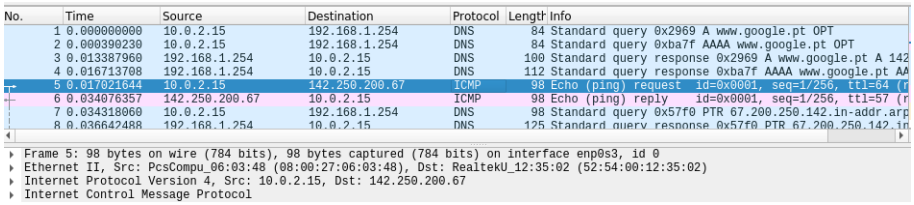
Com base na captura de pacotes feita, preencha a seguinte tabela, identificando para cada aplicação executada, qual o protocolo de aplicação, o protocolo de transporte, porta de atendimento e overhead de transporte.

Comando	Protocolo de Aplicação	Protocolo de Transporte	Porta de Atendimento	Overhead de Transporte (em bytes)
ping	-	-	-	-
traceroute	-	UDP	33446	8
telnet	TELNET	TCP	23	20
ftp	FTP	TCP	21	20
tftp	TFTP	UDP	69	8
wget/lynx	HTTP	TCP	80	20
nslookup	DNS	UDP	53	8
ssh	SSHv2	TCP	22	20

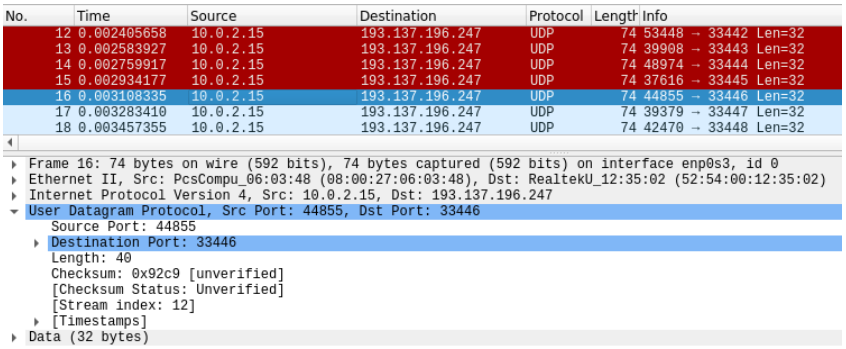
Tabela 1: Tabela que identifica, para cada aplicação executada, os protocolos e o overhead

Para preencher a tabela solicitada recorremos à análise das seguintes capturas:

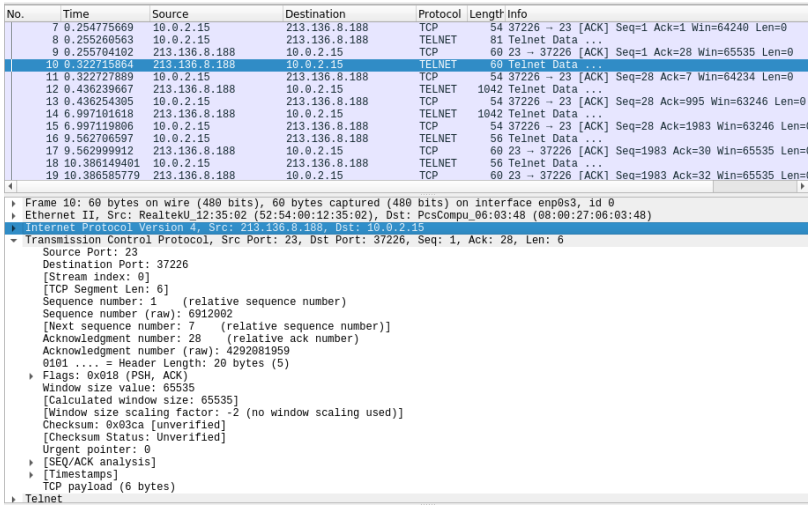
- **ping**:



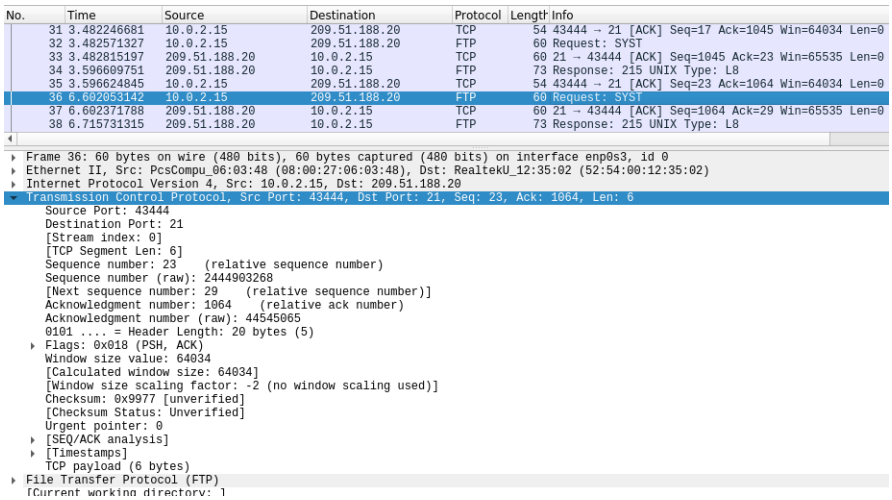
- **traceroute**:



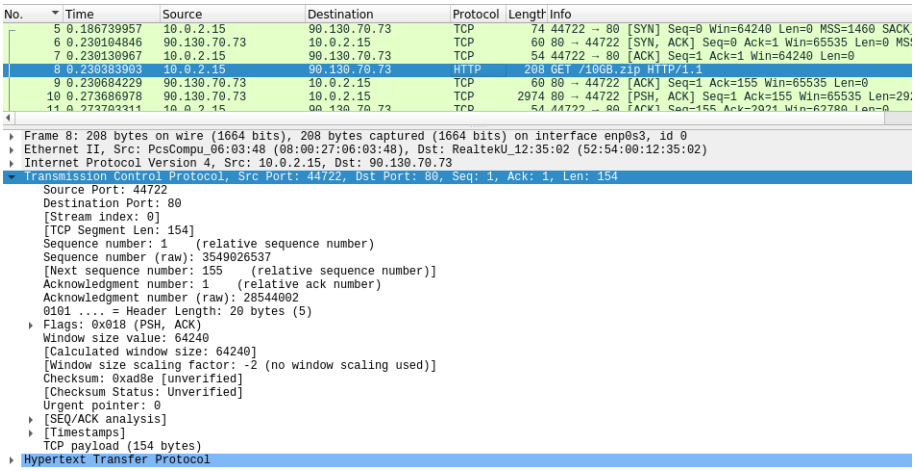
- **telnet**:



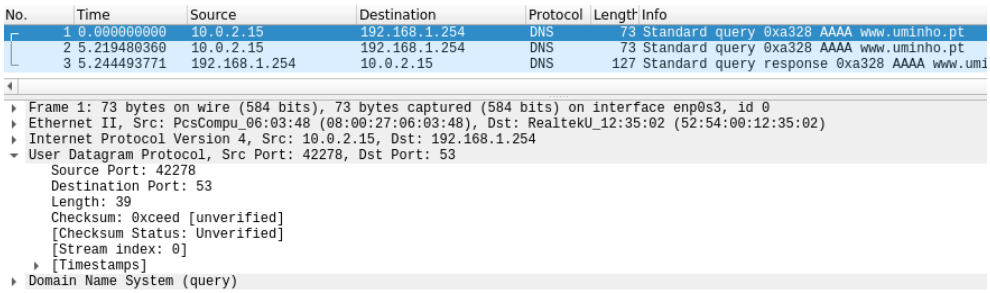
- **ftp**:



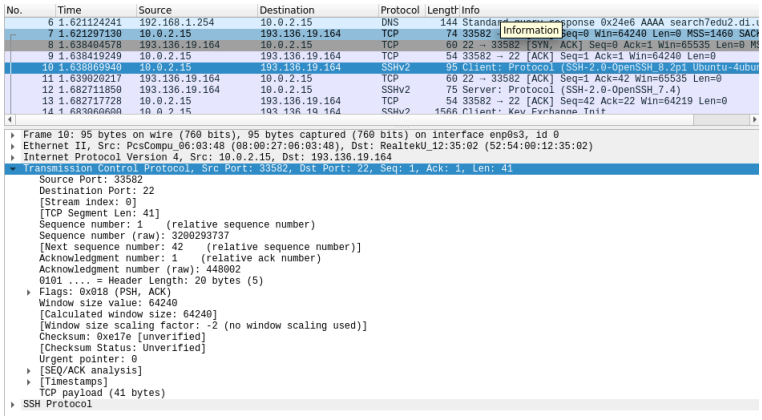
- **http**:



- **nslookup**:



- **ssh**:


No. Time Source Destination Protocol Length Info
6 1.621124241 192.168.1.254 10.0.2.15 DNS 144 Standard query response 0x24e6 AAAA search7edu2.di.
7 1.621297130 10.0.2.15 193.136.19.164 TCP 74 33582 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SAC
8 1.638404578 193.136.19.164 10.0.2.15 TCP 60 22 → 33582 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MS
9 1.638419249 10.0.2.15 193.136.19.164 TCP 54 33582 → 22 [ACK] Seq=1 Ack=1 Win=64240 Len=0
10 1.638869940 10.0.2.15 193.136.19.164 SSHv2 95 Client: Protocol (SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubu
11 1.639020217 193.136.19.164 10.0.2.15 TCP 60 22 → 33582 [ACK] Seq=1 Ack=42 Win=65535 Len=0
12 1.682711850 193.136.19.164 10.0.2.15 SSHv2 75 Server: Protocol (SSH-2.0-OpenSSH_7.4)
13 1.682717728 10.0.2.15 193.136.19.164 TCP 54 33582 → 22 [ACK] Seq=42 Ack=22 Win=64219 Len=0
14 1.683060600 10.0.2.15 193.136.19.164 SSHv2 1566 Client: Key Exchange Init
Information
Frame 10: 95 bytes on wire (760 bits), 95 bytes captured (760 bits) on interface enp0s3, id 0
Ethernet II, Src: PcsCompu_06:03:48 (08:00:27:06:03:48), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.19.164
Transmission Control Protocol, Src Port: 33582, Dst Port: 22, Seq: 1, Ack: 1, Len: 41
Source Port: 33582
Destination Port: 22
[Stream index: 0]
[TCP Segment Len: 41]
Sequence number: 1 (relative sequence number)
Sequence number (raw): 3200293737
[Next sequence number: 42 (relative sequence number)]
Acknowledgment number: 1 (relative ack number)
Acknowledgment number (raw): 448002
0101 .... = Header Length: 20 bytes (5)
Flags: 0x018 (PSH, ACK)
Window size value: 64240
[Calculated window size: 64240]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0xe17e [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
[SEQ/ACK analysis]
[Timestamps]
TCP payload (41 bytes)
SSH Protocol