CPE 435: OPERATING SYSTEMS LABORATORY.

Lab10

Introduction to Wireshark and Packet Analysis.

Submitted by: Dan Otieno.

Date of Experiment: 03/24/23.

Report Deadline: 03/31/23.

Demonstration Deadline: 03/31/23.

Introduction:

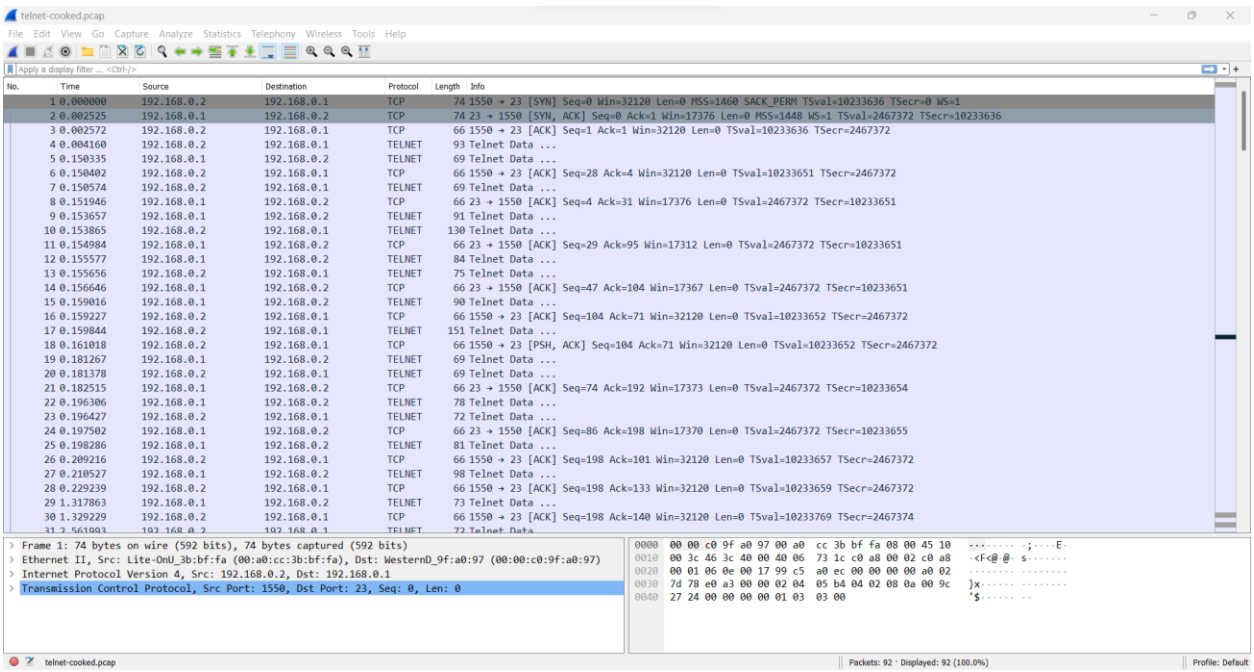
The purpose of this lab was to understand the utilities we use to debug and analyze software.

Results & Observation:

Subtask 1:

Description:

The goal for this assignment was to download the telnet pcap file from the Wireshark website samples and open the file in the Wireshark window, the following screenshot captures the window when I opened the Wireshark file:



1. How many packets are captured in the .pcap file that you loaded? **92 packets.**

2. List all the communicating parties in the .pcap file. Can you also identify the ports being used by each of them? **There are two communicating parties in the file 192.168.0.1 (port 23) and 192.168.0.2 (port 1550)**.

Ethernet · 2 | IPv4 · 2 | IPv6 | TCP · 2 | UDP

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
192.168.0.1	23	92	7.566 KiB	44	4.183 KiB	48	3.384 KiB
192.168.0.2	1550	92	7.566 KiB	48	3.384 KiB	44	4.183 KiB

3. What protocols are used for communication by the communicating parties ? **TCP.**

4. What is the total duration of the communication? (You may want to see the first and last frame) – **39.5713s**.

```
92 39.571274      192.168.0.1          192.168.0.2          TCP
```

5. What is the frame length and number of the longest frame transferred? Who is the source and destination of that packet? **Frame Number: 1, Frame Length (74 bytes or 592 bits), sent from Port 1550 to Port 23**.

```
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: Lite-OnU_3b:bf:fa (00:a0:cc:3b:bf:fa), Dst: WesternD_9f:a0:97 (00:00:c0:9f:a0:97)
> Internet Protocol Version 4, Src: 192.168.0.2, Dst: 192.168.0.1
v Transmission Control Protocol, Src Port: 1550, Dst Port: 23, Seq: 0, Len: 0
    Source Port: 1550
    Destination Port: 23
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 2579865836
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x002 (SYN)
    Window: 32120
    [Calculated window size: 32120]
```

Subtask 2:

6. Select frame number 8. Who is the sender and receiver of this frame? **Sender is Port 23, and Receiver is Port 1550**.

```
> Frame 8: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: WesternD_9f:a0:97 (00:00:c0:9f:a0:97), Dst: Lite-OnU_3b:bf:fa (00:a0:cc:3b:bf:fa)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.2
v Transmission Control Protocol, Src Port: 23, Dst Port: 1550, Seq: 4, Ack: 31, Len: 0
    Source Port: 23
    Destination Port: 1550
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 4    (relative sequence number)
    Sequence Number (raw): 401695553
    [Next Sequence Number: 4    (relative sequence number)]
    Acknowledgment Number: 31    (relative ack number)
    Acknowledgment number (raw): 2579865867
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
    Window: 17376
    [Calculated window size: 17376]
```

7. On the window that appears below the listing of all the frames (as shown below), expand Internet Protocol Version 4. What is the Time To Live for frame 8? **64**. What does this mean? **This is the period or duration that the data in frame 8 can exist or "live" in the network before it is discarded**.

```
Time to Live: 64
Protocol: TCP (6)
```

8. Select frame 8 again. Right click on it, and select Follow > TCP Stream. What information can you see? What is the username and password that is transferred? **Login history for yahoo, login credentials, duration, and data transmission details. Username is "fake" and Password is "user"**.

Wireshark · Follow TCP Stream (tcp.stream eq 0) · telnet-cooked.pcap

```
........... ..!.."..'.....#..%..%........... ..!.."..".........P. ...."......b........b....         B.
........................."......'.....#..&..&..$..&..&..$.. .....#.....'........... .9600,9600....#.bam.zing.org:
0.0....'..DISPLAY.bam.zing.org:0.0......xterm-color.............!................"............
OpenBSD/i386 (oof) (ttyp2)

login: fake
......Password:user

......Last login: Sat Nov 27 20:11:43 on ttyp2 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (OOF) #4: Tue Oct 12 20:42:32 CDT 1999

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

$ /sbin/ping www.yahoo.com
PING www.yahoo.com (204.71.200.67): 56 data bytes
64 bytes from 204.71.200.67: icmp_seq=0 ttl=241 time=69.885 ms
64 bytes from 204.71.200.67: icmp_seq=1 ttl=241 time=73.591 ms
64 bytes from 204.71.200.67: icmp_seq=2 ttl=241 time=72.302 ms
64 bytes from 204.71.200.67: icmp_seq=3 ttl=241 time=73.493 ms
64 bytes from 204.71.200.67: icmp_seq=4 ttl=241 time=75.068 ms
64 bytes from 204.71.200.67: icmp_seq=5 ttl=241 time=70.239 ms
..........
.--- www.yahoo.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 69.885/72.429/75.068 ms
$ ls
$ ls -a
.          ..         .cshrc     .login     .mailrc    .profile   .rhosts
$ exit
```

9. Repeat the same procedure in telnet-raw.pcap. Find the login information used to verify credentials. (Select frame 8 again). **Username: .."……..."ffaakkee, Password: user**.

Wireshark · Follow TCP Stream (tcp.stream eq 0) · telnet-raw.pcap

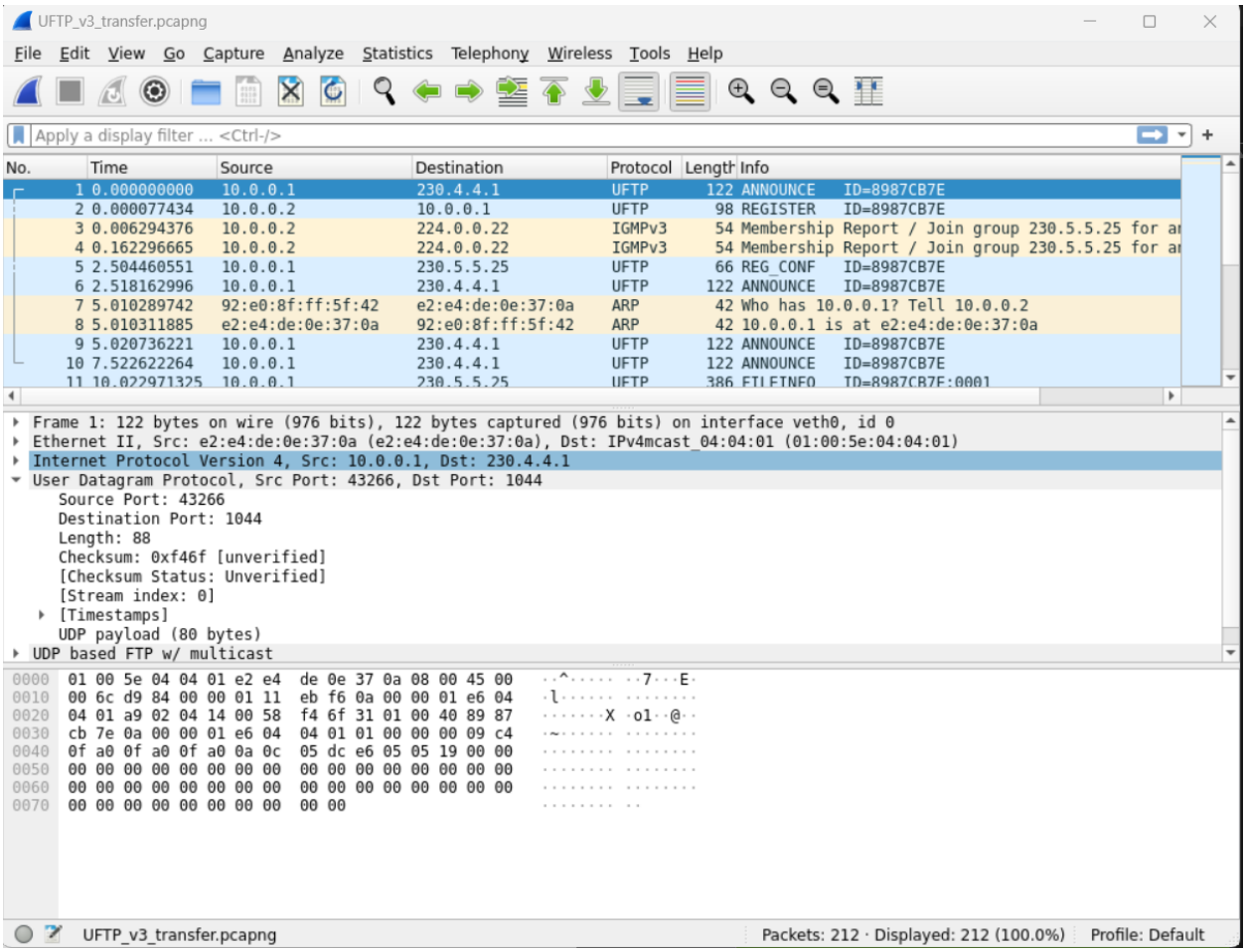
```
........... ..!.."..'.....#..%..%........... ..!.."..".........P. ...."......b........b....         B.
........................"......'.....#..&..&..$..&..&..$.. .....#.....'........... .9600,9600....#.bam.zing.org:
0.0....'..DISPLAY.bam.zing.org:0.0......xterm-color.............!................"............
OpenBSD/i386 (oof) (ttyp1)

login: .."........"ffaakkee
.
Password:user
.
Last login: Thu Dec  2 21:32:59 on ttyp1 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (OOF) #4: Tue Oct 12 20:42:32 CDT 1999

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

$ llss
.
$ llss  --aa
.
.         ..         .cshrc    .login    .mailrc   .profile  .rhosts
$ //ssbbiinn//ppiinngg  wwwwww..yyaahhoooo..ccoomm
.
PING www.yahoo.com (204.71.200.74): 56 data bytes
64 bytes from 204.71.200.74: icmp_seq=0 ttl=239 time=73.569 ms
64 bytes from 204.71.200.74: icmp_seq=1 ttl=239 time=71.099 ms
64 bytes from 204.71.200.74: icmp_seq=2 ttl=239 time=68.728 ms
64 bytes from 204.71.200.74: icmp_seq=3 ttl=239 time=73.122 ms
64 bytes from 204.71.200.74: icmp_seq=4 ttl=239 time=71.276 ms
64 bytes from 204.71.200.74: icmp_seq=5 ttl=239 time=75.831 ms
64 bytes from 204.71.200.74: icmp_seq=6 ttl=239 time=70.101 ms
64 bytes from 204.71.200.74: icmp_seq=7 ttl=239 time=74.528 ms
64 bytes from 204.71.200.74: icmp_seq=9 ttl=239 time=74.514 ms
64 bytes from 204.71.200.74: icmp_seq=10 ttl=239 time=75.188 ms
64 bytes from 204.71.200.74: icmp_seq=11 ttl=239 time=72.925 ms
...^C
.--- www.yahoo.com ping statistics ---
```

58 client pkts, 78 server pkts, 106 turns.

Entire conversation (2001 bytes) | Show data as ASCII

10. What do you think is wrong with these two files that you analyzed? How can you not allow anyone to know your password that you send for authentication? **The username and password data was too easy to find in the data stream, this would be resolved by encrypting the data to avoid vulnerability.**

11. Load the file uftp_v3_transfer.pcapng. The protocol used is UFTP. What is UFTP? Can you identify two parties that are involved in file transfer? (Use your intelligent guessing)



UFTP is an encrypted multicast file transfer program, designed to transfer files securely, reliably, and efficiently to multiple receivers simultaneously. This is useful for distributing large files to many receivers and is especially useful for data distribution over a satellite link (with two way communication), where the inherent delay makes any TCP based communication highly inefficient. SOURCE. The two parties involved in this transfer are Port 43266 (10.0.0.1) and Port 1044 (230.4.4.1).

Ethernet · 5 | IPv4 · 5 | IPv6 | TCP | UDP · 4

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
10.0.0.1	43266	206	286.586 KiB	202	286.285 KiB	4	308 bytes
10.0.0.2	1044	4	308 bytes	4	308 bytes	0	0 bytes
230.4.4.1	1044	4	488 bytes	0	0 bytes	4	488 bytes
230.5.5.25	1044	198	285.809 KiB	0	0 bytes	198	285.809 KiB

12. Write differences between TCP and UDP. **SOURCE.**

- **Connection-oriented vs Connectionless: TCP is a connection-oriented protocol, meaning that a connection is established between the two devices before any data is sent. UDP is a connectionless protocol, meaning that data can be sent without establishing a connection first.**

- **Reliability: TCP is a reliable protocol as it ensures that all packets are received and in the correct order. If a packet is lost or damaged, TCP will automatically resend it until it is received correctly. UDP, on the other hand, is an unreliable protocol, as it does not guarantee that all packets will be received or in the correct order.**
- **Flow control: TCP uses flow control mechanisms to prevent data overload on the receiving end. It controls the rate at which data is sent so that the receiver can process it without being overwhelmed. UDP does not have built-in flow control mechanisms.**
- **Speed: UDP is generally faster than TCP because it doesn't have the overhead of ensuring reliability and flow control.**
- **Usage: TCP is commonly used for applications that require reliable, ordered delivery of data, such as web browsing, email, and file transfers. UDP is commonly used for applications that require fast, real-time communication, such as online gaming, video conferencing, and voice over IP (VoIP).**

Subtask 3:

Description:

For this assignment, we work on encrypted protocol.

13. What is the difference between https:// and http://? What is the encryption standard used by them, if any? **HTTP stands for Hypertext Transfer Protocol, and it is a protocol – or a prescribed order and syntax for presenting information – used for transferring data over a network. Most information that is sent over the Internet, including website content and API calls, uses the HTTP protocol. HTTPS is HTTP with encryption and verification. The only difference between the two protocols is that HTTPS uses TLS (SSL) to encrypt normal HTTP requests and responses, and to digitally sign those requests and responses. As a result, HTTPS is far more secure than HTTP. A website that uses HTTP has http:// in its URL, while a website that uses HTTPS has https://. SOURCE.**

14. Download the file mysql_complete_pcap. Is it encrypted? Please justify. **As in the screenshot below, the data does not look encrypted, the TCP stream does not reveal any clear encryption protocol, and we can also see explicit table entries in the TCP stream.**

mysql_complete.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

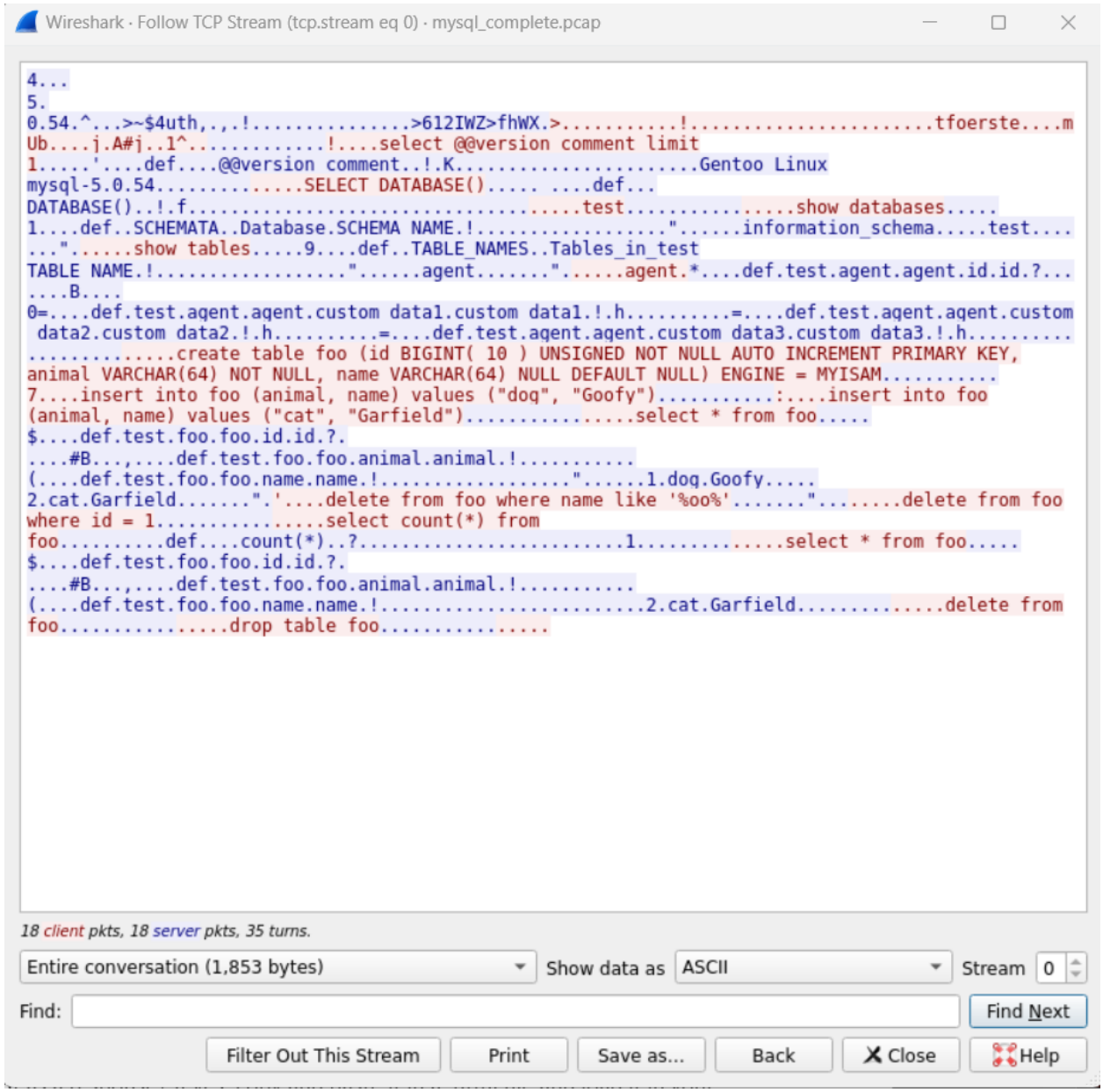
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.254	192.168.0.254	TCP	74	56162 → 3306 [SYN] Seq=0 Win=32792 Len=0 MSS=16396 SACK_PERM TSval=15785614 TSecr=0 WS=64
2	0.000046	192.168.0.254	192.168.0.254	TCP	74	3306 → 56162 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=16396 SACK_PERM TSval=15785614 TSecr=15785614 WS=64
3	0.000077	192.168.0.254	192.168.0.254	TCP	66	56162 → 3306 [ACK] Seq=1 Ack=1 Win=32832 Len=0 TSval=15785614 TSecr=15785614
4	0.000265	192.168.0.254	192.168.0.254	MySQL	122	Server Greeting proto=10 version=5.0.54
5	0.000286	192.168.0.254	192.168.0.254	TCP	66	56162 → 3306 [ACK] Seq=1 Ack=57 Win=32832 Len=0 TSval=15785614 TSecr=15785614
6	0.000559	192.168.0.254	192.168.0.254	MySQL	132	Login Request user=tfoerste
7	0.000583	192.168.0.254	192.168.0.254	TCP	66	3306 → 56162 [ACK] Seq=57 Ack=67 Win=32768 Len=0 TSval=15785614 TSecr=15785614
8	0.000695	192.168.0.254	192.168.0.254	MySQL	77	Response OK
9	0.000893	192.168.0.254	192.168.0.254	MySQL	103	Request Query
10	0.001051	192.168.0.254	192.168.0.254	MySQL	162	Response TABULAR Response
11	0.040792	192.168.0.254	192.168.0.254	TCP	66	56162 → 3306 [ACK] Seq=104 Ack=164 Win=32832 Len=0 TSval=15785655 TSecr=15785615
12	5.698832	192.168.0.254	192.168.0.254	MySQL	88	Request Query
13	5.699011	192.168.0.254	192.168.0.254	MySQL	130	Response TABULAR Response
14	5.699035	192.168.0.254	192.168.0.254	TCP	66	56162 → 3306 [ACK] Seq=126 Ack=228 Win=32832 Len=0 TSval=15791313 TSecr=15791313
15	5.699226	192.168.0.254	192.168.0.254	MySQL	75	Request Use Database
16	5.699324	192.168.0.254	192.168.0.254	MySQL	77	Response OK
17	5.699573	192.168.0.254	192.168.0.254	MySQL	85	Request Query
18	5.699998	192.168.0.254	192.168.0.254	MySQL	174	Response TABULAR Response
19	5.700180	192.168.0.254	192.168.0.254	MySQL	82	Request Query
20	5.700418	192.168.0.254	192.168.0.254	MySQL	160	Response TABULAR Response

```
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 192.168.0.254, Dst: 192.168.0.254
v Transmission Control Protocol, Src Port: 56162, Dst Port: 3306, Seq: 0, Len: 0
    Source Port: 56162
    Destination Port: 3306
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 3436755789
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x002 (SYN)
    Window: 32792
    [Calculated window size: 32792]
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 3c 65 c3 40 00 40 06  51 ac c0 a8 00 fe c0 a8   .<e.@.@. Q.......
0020  00 fe db 62 0c ea cc d8  bb 4d 00 00 00 00 a0 02   ...b.... .M......
0030  80 18 ba 51 00 00 02 04  40 0c 04 02 08 0a 00 f0   ...Q.... @.......
0040  de 8e 00 00 00 00 01 03  03 06                     ........ ..
```

Frame (frame), 74 bytes

Packets: 57 · Displayed: 57 (100.0%)



15. Download the file mysql-ssl-larger.pcapng. Is it encrypted? Please justify. **The file is encrypted with TLSv1 (Transport Layer Security) protocol. We also cannot see explicit data in the stream output, but rather ASCII characters.**

mysql-ssl-larger.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

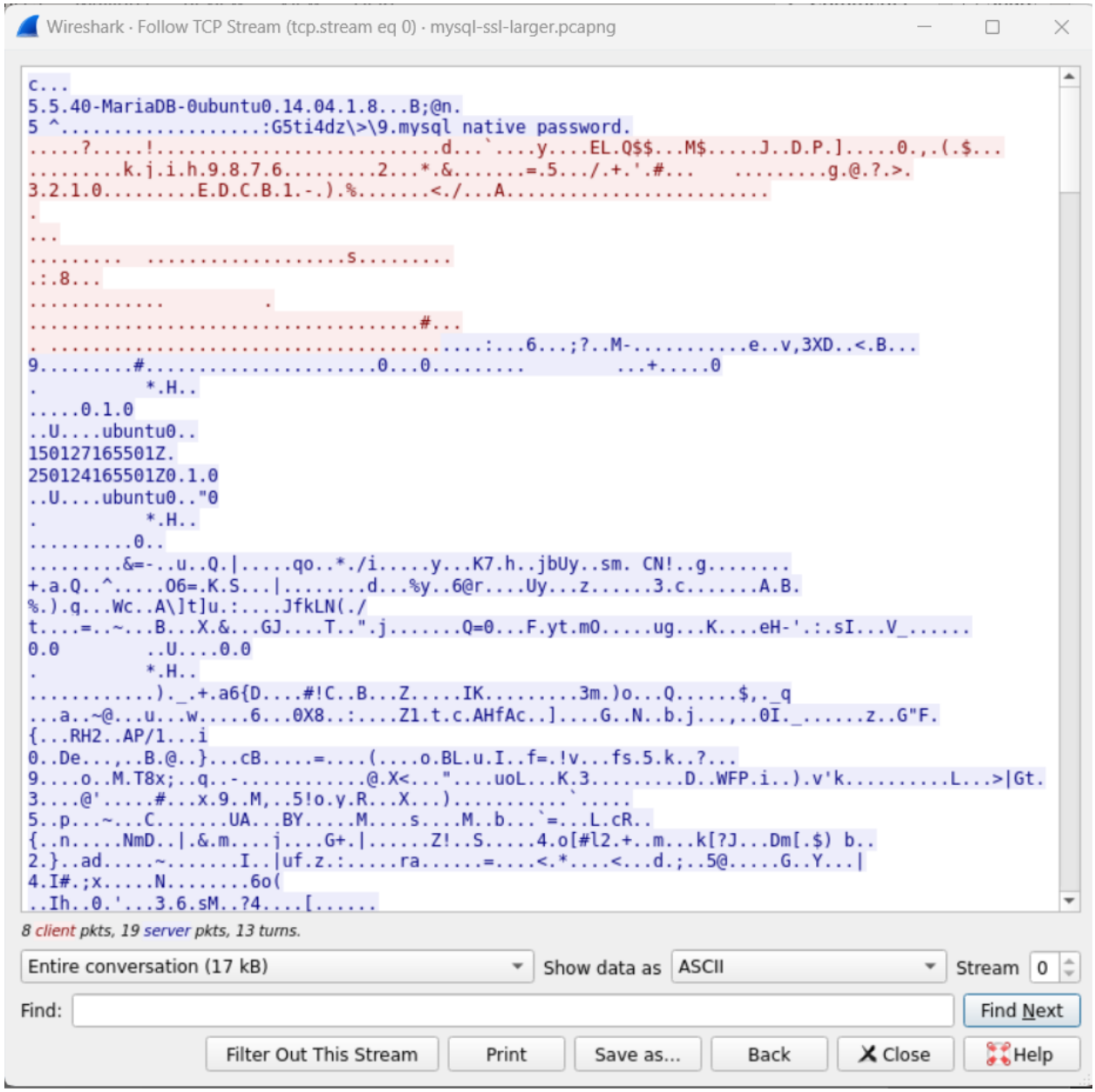
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.2.102	192.168.2.101	TCP	74	34543 → 3306 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=709524415 TSecr=0 WS=128
2	0.000250827	192.168.2.101	192.168.2.102	TCP	74	3306 → 34543 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM TSval=212591205 TSecr=709524415 WS=128
3	0.000309653	192.168.2.102	192.168.2.101	TCP	66	34543 → 3306 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=709524415 TSecr=212591205
4	0.001346031	192.168.2.101	192.168.2.102	MySQL	169	Server Greeting proto=10 version=5.5.40-MariaDB-0ubuntu0.14.04.1
5	0.001449759	192.168.2.102	192.168.2.101	TCP	66	34543 → 3306 [ACK] Seq=1 Ack=104 Win=29312 Len=0 TSval=709524416 TSecr=212591206
6	0.004658740	192.168.2.102	192.168.2.101	MySQL	102	Login Request user=
7	0.004849622	192.168.2.101	192.168.2.102	TCP	66	3306 → 34543 [ACK] Seq=104 Ack=37 Win=29056 Len=0 TSval=212591207 TSecr=709524417
8	0.007728558	192.168.2.102	192.168.2.101	TLSv1	427	Client Hello
9	0.007938820	192.168.2.101	192.168.2.102	TCP	66	3306 → 34543 [ACK] Seq=104 Ack=398 Win=30080 Len=0 TSval=212591207 TSecr=709524418
10	0.011278899	192.168.2.101	192.168.2.102	TLSv1	1261	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
11	0.012606576	192.168.2.102	192.168.2.101	TLSv1	212	Certificate, Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
12	0.013465035	192.168.2.101	192.168.2.102	TLSv1	300	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
13	0.013793749	192.168.2.102	192.168.2.101	TLSv1	220	Application Data, Application Data
14	0.014366523	192.168.2.101	192.168.2.102	TLSv1	140	Application Data, Application Data
15	0.014489885	192.168.2.102	192.168.2.101	TLSv1	172	Application Data, Application Data
16	0.015058844	192.168.2.101	192.168.2.102	TLSv1	220	Application Data, Application Data
17	0.015214278	192.168.2.102	192.168.2.101	TLSv1	156	Application Data, Application Data
18	0.015753882	192.168.2.101	192.168.2.102	TLSv1	476	Application Data, Application Data
19	0.016096026	192.168.2.102	192.168.2.101	TLSv1	156	Application Data, Application Data
20	0.018570955	192.168.2.101	192.168.2.102	TLSv1	1514	Application Data

```
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth0, id 0
> Ethernet II, Src: Clevo_aa:83:da (00:90:f5:aa:83:da), Dst: HewlettP_18:01:14 (00:11:0a:18:01:14)
> Internet Protocol Version 4, Src: 192.168.2.102, Dst: 192.168.2.101
v Transmission Control Protocol, Src Port: 34543, Dst Port: 3306, Seq: 0, Len: 0
    Source Port: 34543
    Destination Port: 3306
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 280452901
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x002 (SYN)
    Window: 29200
```

```
0000  00 11 0a 18 01 14 00 90  f5 aa 83 da 08 00 45 00   ........ ......E.
0010  00 3c 24 9a 40 00 40 06  90 06 c0 a8 02 66 c0 a8   .<$.@.@. .....f..
0020  02 65 86 ef 0c ea 10 b7  5f 25 00 00 00 00 a0 02   .e...... _%......
0030  72 10 86 4a 00 00 02 04  05 b4 04 02 08 0a 2a 4a   r..J.... ......*J
0040  7b bf 00 00 00 00 01 03  03 07                     {....... ..
```

Frame (frame), 74 bytes

Packets: 42 · Displayed: 42 (100.0%)

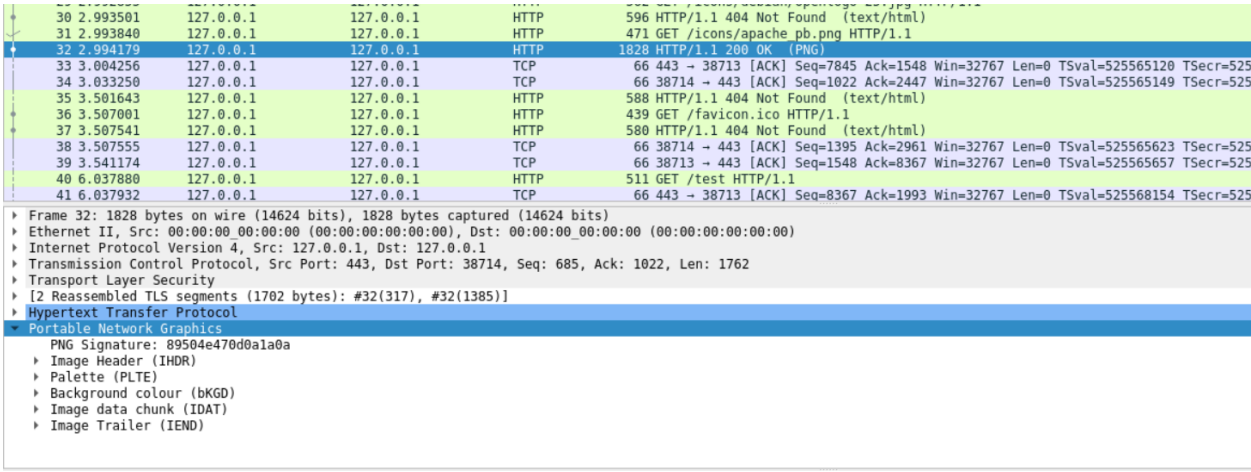


16. Download the zipped file snakeoil2_070531.tgz. Extract the content in your local folder.

Load the .pcap file in wireshark. Is it encrypted? **Yes, the file is encrypted, as evidenced by the https encryption modes.**

17. Perform the decryption of the .pcap file as demonstrated in class by the instructor.

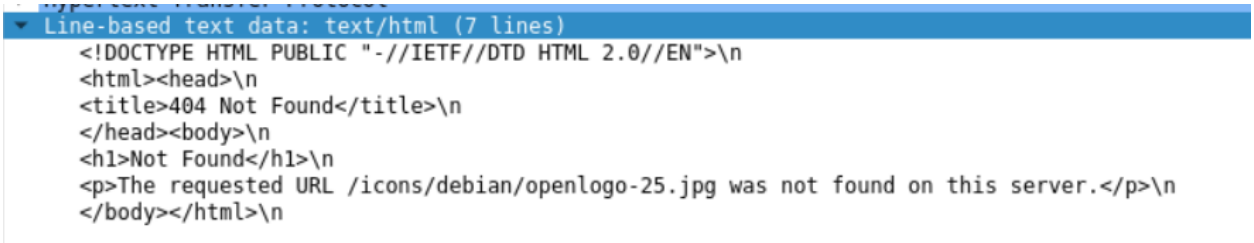
    1. What frame number requests the image apache_pb.png? **Frame 31.**

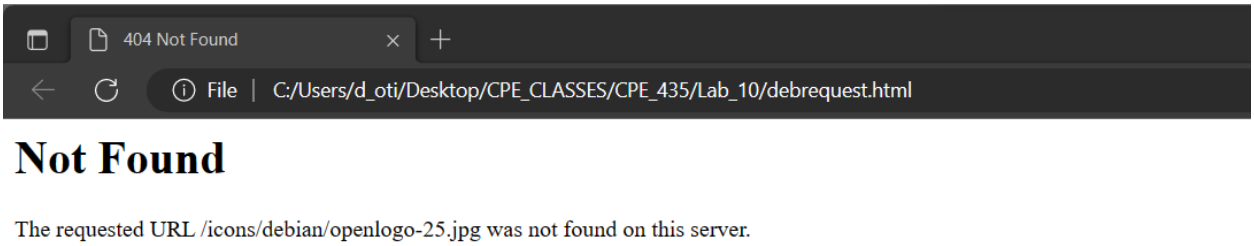


2. Does the server provide the image? What is the status code that implies the response has a payload? **Yes, the server provided the image in png format. The status code is 200 (also in screenshot above, on Frame 32).**

3. Attach the image apache_pb.png to your report.

18. What is the response that the server provided when requested for openlogo-25.jpg? Can you see the html code sent as a response? If yes, copy and paste it in a .html file and load it in your favorite browser. Attach the screenshot of how the response looks like in the web browser.





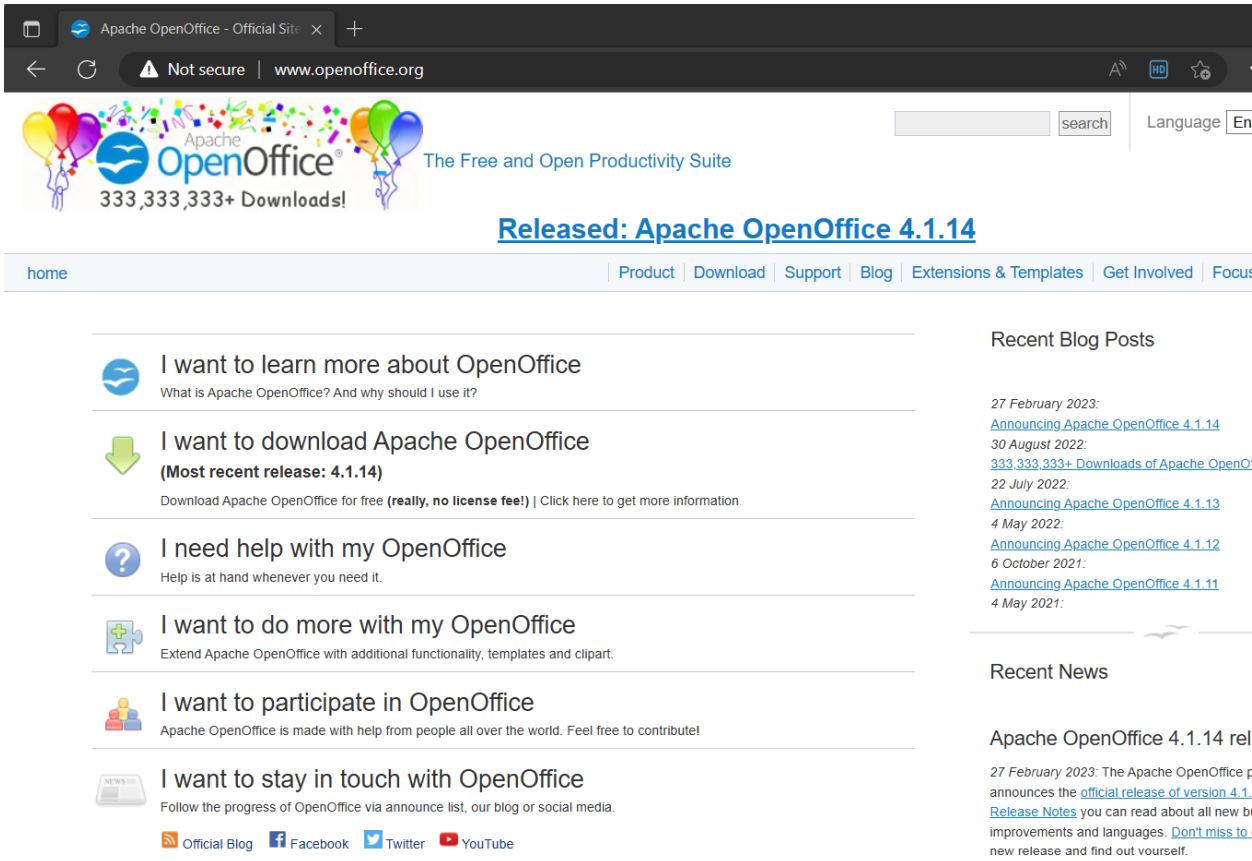
Subtask 4:

19. The first thing that you will do is capture packets. You can use wireshark or tcpdump to capture packets. While you can capture packets from wireshark, I suggest you use tcpdump so that you can be familiar with a new tool. Following are the procedures that you will follow:

1. Find the interface that is connected to the internet. Do ifconfig in the terminal and select the one which is connected to the internet. Wireshark should show you the interface in its GUI.

2. Start packet capture in tcpdump using tcpdump -i <interface> -s 65535 -w <filename>. Or select the bluefin below File menu in wireshark after you select the interface if you wish to use wireshark. **Wireshark used.**

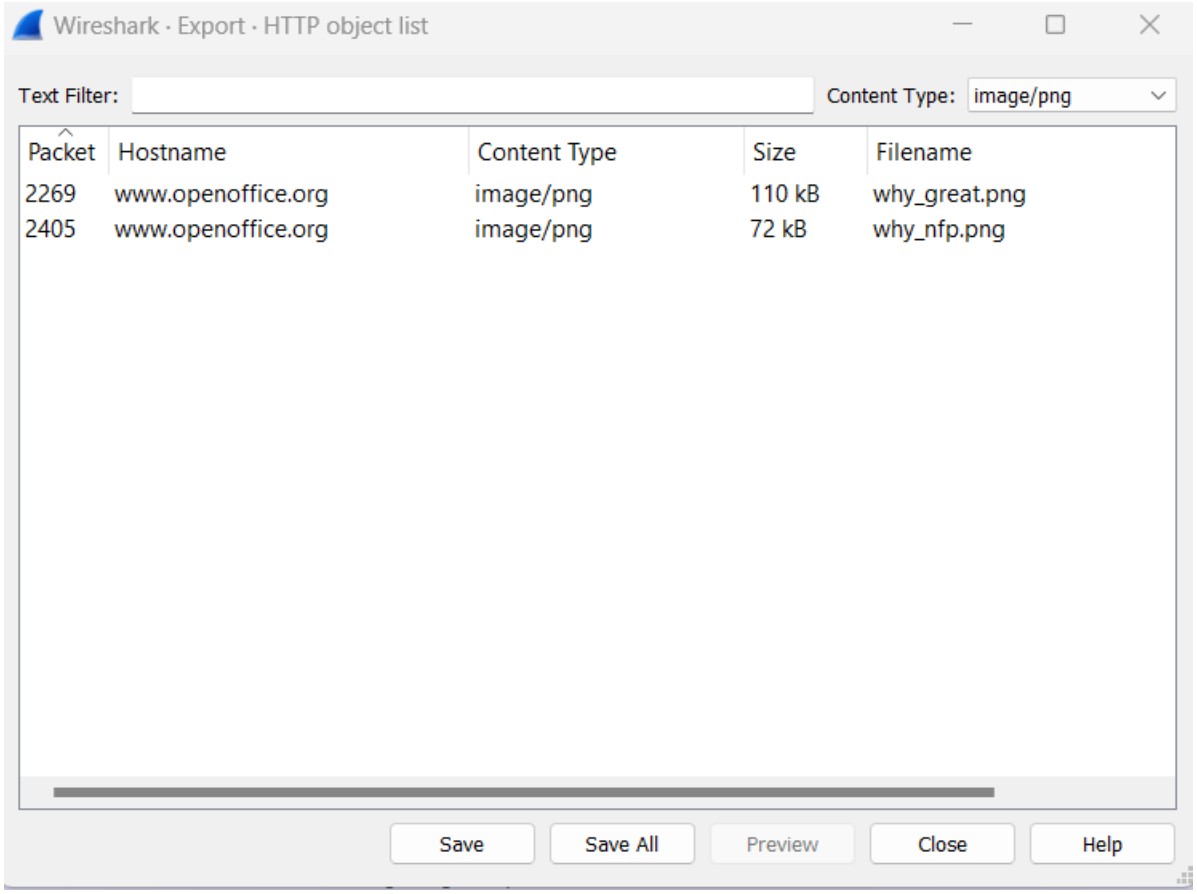
3. Please visit the website http://www.openoffice.org/. What is wrong with this website? **The website is NOT secure. (We can also tell by the URL starting with http instead of https)**.



4. After it is completely loaded, stop the capture. You may want to navigate around the website before stopping the capture. You can select the button in Wireshark GUI or kill the tcpdump process if you are using tcpdump.

5. Load the file in wireshark. If you are using wireshark, it is already loaded.

6. Try to find at least two images that are sent by the server to your machine and attach them to your report.


Wireshark · Export · HTTP object list
Text Filter:
Content Type: image/png
Packet
Hostname
Content Type
Size
Filename
2269 www.openoffice.org image/png 110 kB why_great.png
2405 www.openoffice.org image/png 72 kB why_nfp.png
Save
Save All
Preview
Close
Help

7. What are the vulnerabilities of the website that you can see right away? **The website is not secure, that is instantly indicated on the URL bar. The website also contains links to various social networking platforms, so a user's personal information may be compromised if they were to enter their credentials on those sites**.

8. Repeat similar operation for https://www.uah.edu/. Attach two images sent from the server to your machine if you can. If you cannot view any images, comment on why this might be**. There were no images from this test because it is a secure website and images are encrypted files.**

Wireshark · Export · HTTP object list

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
98	192.168.0.1:34691	text/xml	2943 bytes	WANCfg.xml
120	192.168.0.240:2869	text/xml	142 bytes	bzvzaqckpg
132	192.168.0.1:34691	text/xml	309 bytes	CmnIfCfg
134	192.168.0.1:34691	text/xml	352 bytes	CmnIfCfg
146	192.168.0.1:34691	text/xml	313 bytes	CmnIfCfg
148	192.168.0.1:34691	text/xml	368 bytes	CmnIfCfg
359	www.msftncsi.com	text/plain	14 bytes	ncsi.txt
381	192.168.0.1:34691	text/xml	2943 bytes	WANCfg.xml
404	192.168.0.240:2869	text/xml	142 bytes	jkskayayqv
413	192.168.0.1:34691	text/xml	309 bytes	CmnIfCfg
415	192.168.0.1:34691	text/xml	352 bytes	CmnIfCfg
422	192.168.0.1:34691	text/xml	313 bytes	CmnIfCfg
424	192.168.0.1:34691	text/xml	368 bytes	CmnIfCfg
431	192.168.0.1:34691	text/xml	309 bytes	CmnIfCfg
433	192.168.0.1:34691	text/xml	352 bytes	CmnIfCfg
441	192.168.0.1:34691	text/xml	313 bytes	CmnIfCfg
443	192.168.0.1:34691	text/xml	368 bytes	CmnIfCfg
454	192.168.0.1:34691	text/xml	309 bytes	CmnIfCfg

Save Save All Preview Close Help