

This lab session covers the usage of the Wireshark application to monitor and capture the outgoing and incoming packets from a network connection (WIFI, ethernet, etc.). Specifically, students should be able to analyze HTTP, HTTPS, TCP/IP, and UDP protocols using Wireshark, a network protocol analyzer, and draw conclusions.

Pre-lab Preparation:

1. Review the basics and the structure of HTTP, TCP/IP, and UDP protocols,
2. Install Wireshark and ensure it is running on your computer,
3. Create an online, *publically accessible* Git repository to host and upload your work in the labs. We recommend you use GitHub or GitLab.

Lab Activities:

Part 1: Capturing HTTP Traffic.

Task 1: Start Wireshark and capture packets.

Step 1: Open Wireshark.

Step 2: Select the network interface connected to the internet (e.g., Ethernet or Wi-Fi).

Step 3: Click the "Start Capturing Packets" button (the shark fin icon).

Step 4: Open your favorite web browser and navigate to (<http://neverssl.com/>) website.

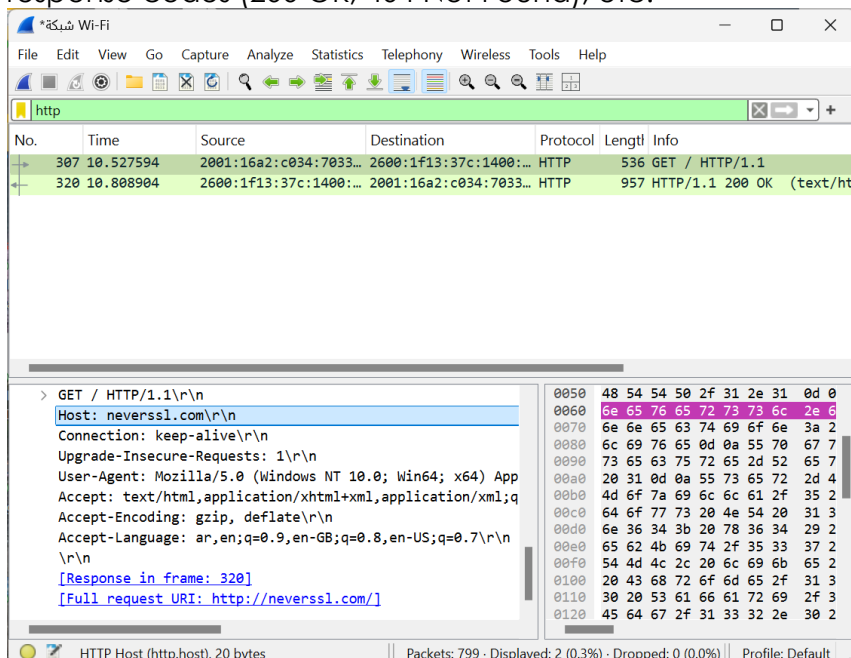
Step 5: After the website has fully loaded, stop capturing packets by clicking the red stop button in Wireshark.

Task 2: Filter HTTP packets and analyze them.

Step 1: In the filter bar, type http and press Enter. This filters out only the HTTP packets from the capture.

Step 2: Select any HTTP packet to view its details.

Step 3: Observe the HTTP request and response messages. Note the method (GET, POST), URL, response codes (200 OK, 404 Not Found), etc.



Part 2: Analyzing TCP/IP Traffic.**Task 1: Filter TCP packets**

Step 1: Clear the previous filter and type TCP to focus on TCP packets.

Step 2: Select a TCP packet related to your HTTP request/response.

Step 3: Right-click on the packet and select "Follow" -> "TCP Stream".

Step 4: This shows the entire conversation between the client and server.

Task 2: Analyze TCP handshake and investigate Data Transfer and Termination

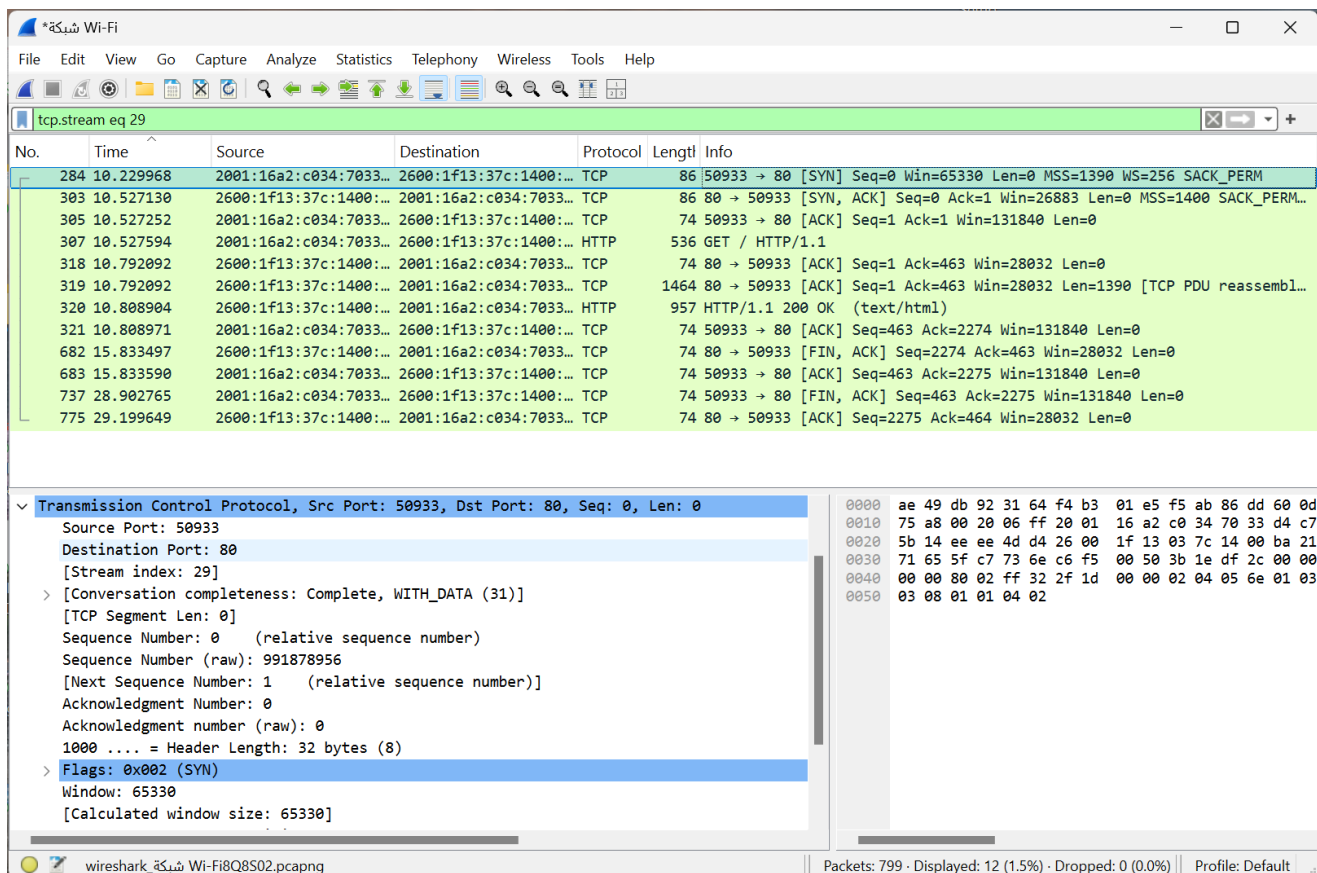
Step 1: Find and select packets related to the TCP three-way handshake:

- SYN: Initiates a connection.
- SYN-ACK: Acknowledges and responds to the SYN.
- ACK: Acknowledges the SYN-ACK and establishes the connection.

Step 2: Note the sequence and acknowledgment numbers. Screenshot and upload your image to your online git repository.

Step 3: Observe the data packets exchanged between the client and server. Take a screenshot and upload it to your online git repo.

Step 4: Look at the TCP termination process (FIN, ACK packets).



Wireshark interface showing a TCP stream (eq 29) with packets 284 to 775. The selected packet (No. 284) is a SYN packet from 2001:16a2:c034:7033 to 2600:1f13:37c:1400. The packet details show the TCP header with Seq=0, Win=65330, Len=0, and Flags=0x002 (SYN). The packet bytes are displayed in hexadecimal and ASCII.

Transmission Control Protocol, Src Port: 50933, Dst Port: 80, Seq: 0, Len: 0

Source Port: 50933
Destination Port: 80
[Stream index: 29]
> [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 991878956
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 ... = Header Length: 32 bytes (8)
> Flags: 0x002 (SYN)
Window: 65330
[Calculated window size: 65330]

0000 ae 49 db 92 31 64 f4 b3 01 e5 f5 ab 86 dd 60 0d
0010 75 a8 00 20 06 ff 20 01 16 a2 c0 34 70 33 d4 c7
0020 5b 14 ee ee 4d d4 26 00 1f 13 03 7c 14 00 ba 21
0030 71 65 5f c7 73 6e c6 f5 00 50 3b 1e df 2c 00 00
0040 00 00 80 02 ff 32 2f 1d 00 00 02 04 05 6e 01 03
0050 03 08 01 01 04 02

Part 3: Capturing and Analyzing UDP Traffic

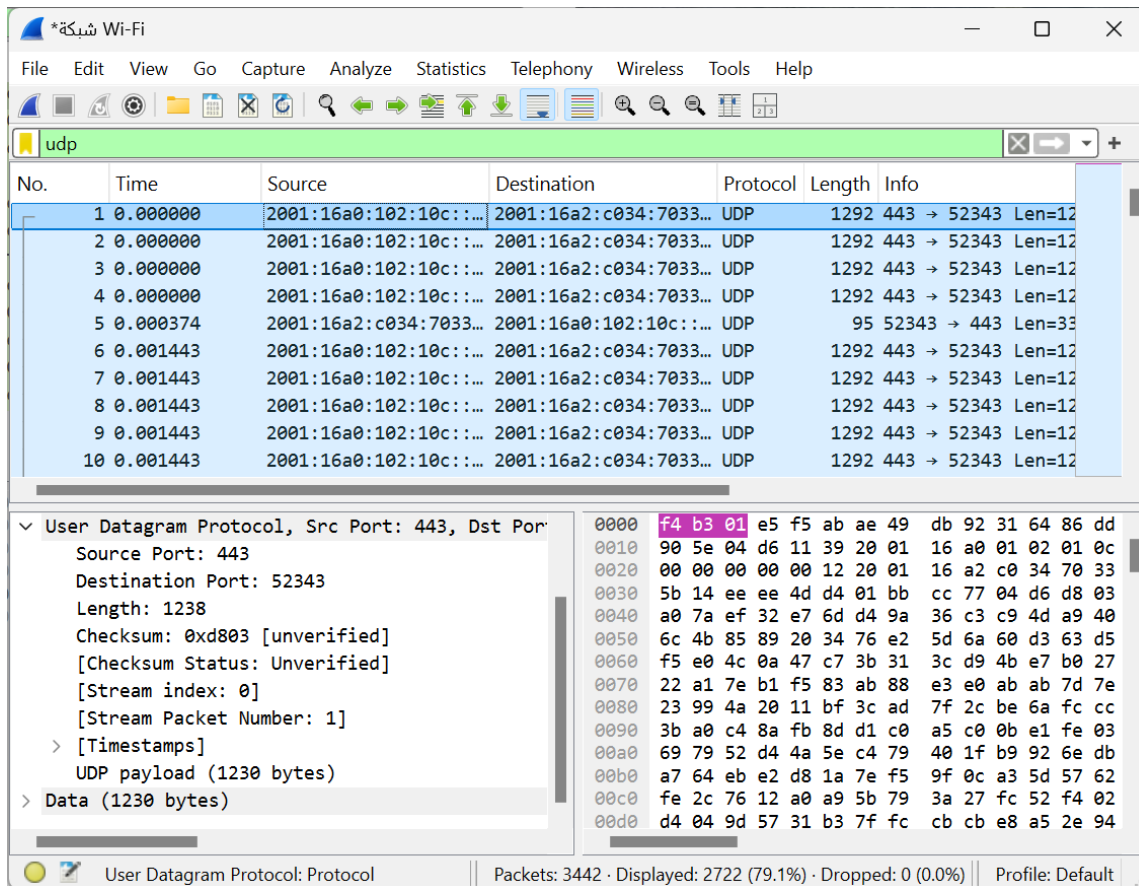
Task 1: Generate UDP traffic and capture packet

Step 1: Open a network application that uses UDP (e.g., streaming video, VoIP software, or custom script).

Step 2: Start the application to generate UDP traffic.

Step 3: Start capturing packets in Wireshark while the UDP application is running.

Step 4: After sufficient traffic is generated, stop capturing packets.



The image shows a Wireshark capture of UDP traffic on a Wi-Fi network. The packet list shows 10 packets, all of which are UDP. The selected packet (No. 1) is a User Datagram Protocol packet with Source Port 443 and Destination Port 52343. The packet length is 1238 bytes. The packet details pane shows the User Datagram Protocol header and the data payload (1230 bytes). The packet bytes pane shows the raw data in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
2	0.000000	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
3	0.000000	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
4	0.000000	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
5	0.000374	2001:16a2:c034:7033...	2001:16a0:102:10c::...	UDP	95	52343 → 443 Len=33
6	0.001443	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
7	0.001443	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
8	0.001443	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
9	0.001443	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12
10	0.001443	2001:16a0:102:10c::...	2001:16a2:c034:7033...	UDP	1292	443 → 52343 Len=12

Selected packet details:

- User Datagram Protocol, Src Port: 443, Dst Port: 52343
- Source Port: 443
- Destination Port: 52343
- Length: 1238
- Checksum: 0xd803 [unverified]
- [Checksum Status: Unverified]
- [Stream index: 0]
- [Stream Packet Number: 1]
- > [Timestamps]
- UDP payload (1230 bytes)
- > Data (1230 bytes)

Packet bytes:

```
0000 f4 b3 01 e5 f5 ab ae 49 db 92 31 64 86 dd
0010 90 5e 04 d6 11 39 20 01 16 a0 01 02 01 0c
0020 00 00 00 00 00 12 20 01 16 a2 c0 34 70 33
0030 5b 14 ee ee 4d d4 01 bb cc 77 04 d6 d8 03
0040 a0 7a ef 32 e7 6d d4 9a 36 c3 c9 4d a9 40
0050 6c 4b 85 89 20 34 76 e2 5d 6a 60 d3 63 d5
0060 f5 e0 4c 0a 47 c7 3b 31 3c d9 4b e7 b0 27
0070 22 a1 7e b1 f5 83 ab 88 e3 e0 ab ab 7d 7e
0080 23 99 4a 20 11 bf 3c ad 7f 2c be 6a fc cc
0090 3b a0 c4 8a fb 8d d1 c0 a5 c0 0b e1 fe 03
00a0 69 79 52 d4 4a 5e c4 79 40 1f b9 92 6e db
00b0 a7 64 eb e2 d8 1a 7e f5 9f 0c a3 5d 57 62
00c0 fe 2c 76 12 a0 a9 5b 79 3a 27 fc 52 f4 02
00d0 d4 04 9d 57 31 b3 7f fc cb cb e8 a5 2e 94
```

Task 2: Filter and analysis UDP Packets

Step 1: In the filter bar, type UDP and press Enter.

Step 2: This filters out only the UDP packets from the capture.

Step 3: Select any UDP packet to view its details.

Step 4: Observe the source and destination ports, length, and data.

Step 5: Compare the simplicity of UDP headers with TCP headers.

CS471 – Web Technologies (Laboratory)		Lab 1
		The Internet Protocols

Part 4: Comparing TCP and UDP by filling in the following tables. Save your work (e.g., in an MS Word document), and upload it to your online git repo.

Task 1: Fill in the following table and provide reasons.

	TCP or UDP	Reasons
Reliability and Connection Establishment	TCP	TCP is more reliable and uses 3-way handshaking and initiates a connection before starting to communicate while UDP doesn't.
Data Integrity and Ordering	TCP	TCP have sequence and ack numbers to make sure the data are received and in the correct order.

Task 2: Identify the use Cases and Performance of TCP and UDP.

	TCP	UDP
Use cases	Web pages and messages	Streaming services, video call, online games
Performance	Slower than UDP but make sure the data received and handle errors and data integrity	Fast but can't handle the error and have no data integrity