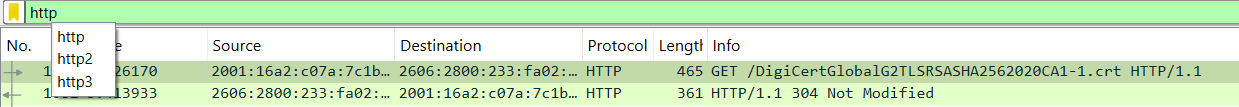
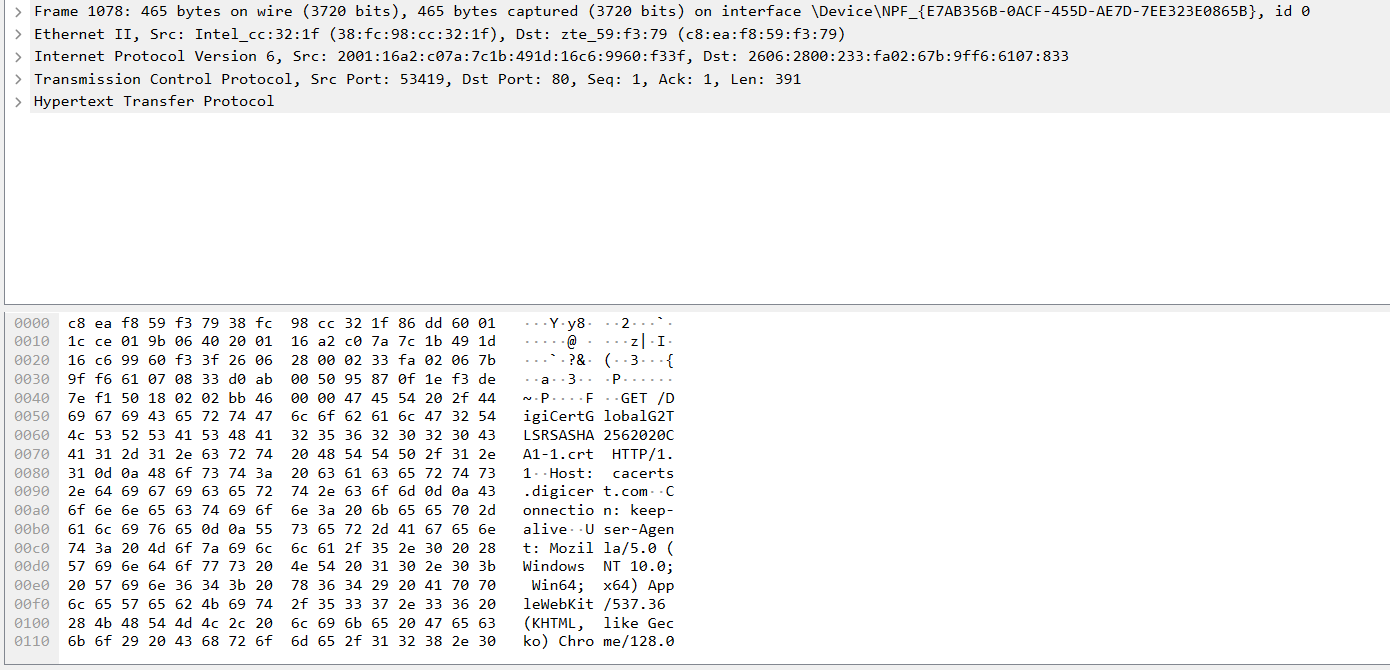
**Lab week 2**

**Task 2:**

**Step 1:**



**Step 2:**



**Step 3**:

1. The methods:

GET

1. URL

<https://myqu.qu.edu.sa>

/DigiCertGlobalG2TLSRSASHA2562020CA1-1.crt

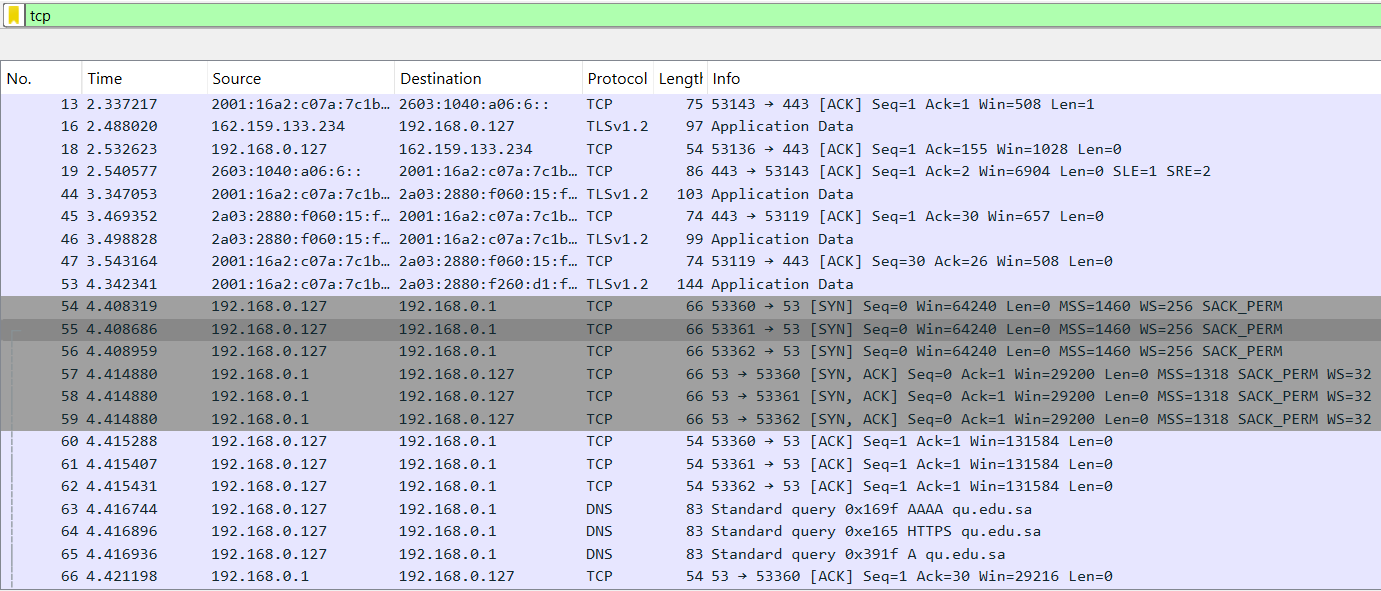
1. Response codes:

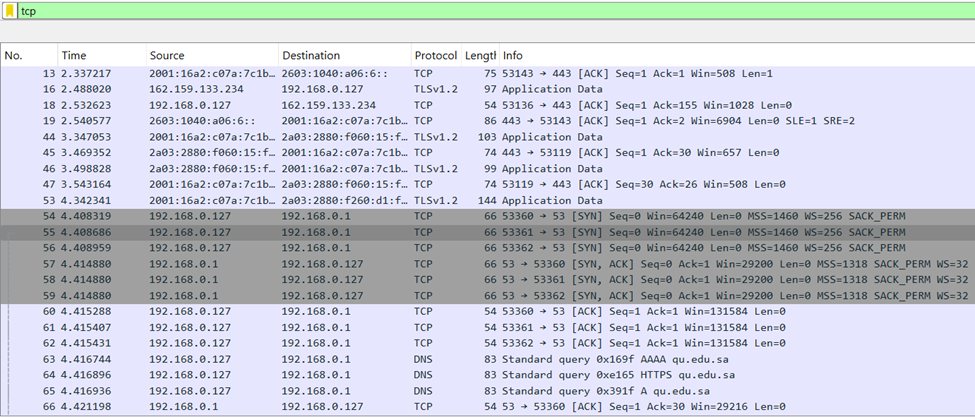
HTTP/1.1 200 OK

HTTP/1.1 304 Not Modified

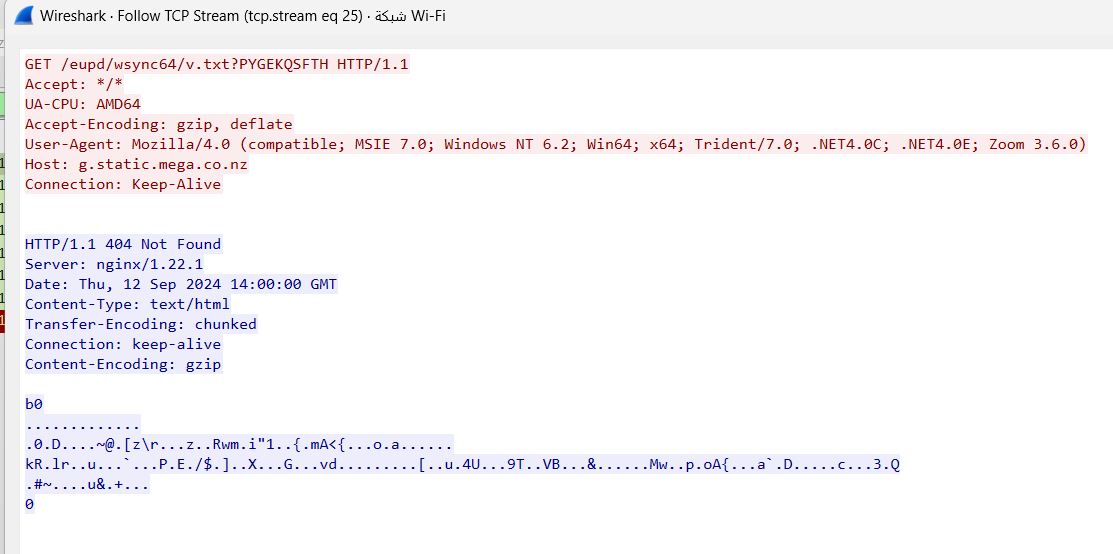
**Part 2:**

**Task 1:**

**Step 1:** 

**Step 2****:** 

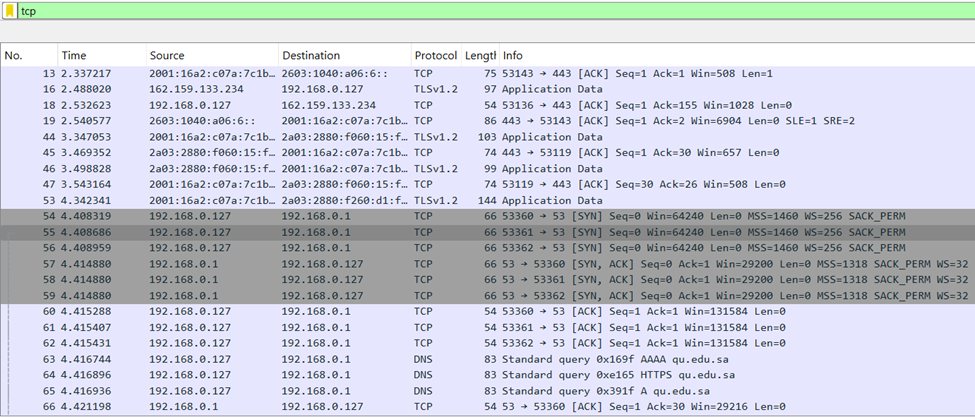


**Step 3:** 

**Step 4:**

**Task 2:**

**Step 1:**





**Step 2:**

**First Packet (SYN from client)**:

Seq=0, Ack=0

**Second Packet (SYN-ACK from server)**:

Seq=0, Ack=1

**Third Packet (ACK from client)**:

Seq=1, Ack=1

**Step 4:**

1. **Client FIN Packet:**
   1. **Description:** Client initiates termination
   2. **Packet:** [FIN, ACK] Seq=1 Ack=1 Win=513 Len=0
2. **Server ACK Packet:**
   1. **Description:** Server acknowledges client’s FIN
   2. **Packet:** [ACK] Seq=1 Ack=2 Win=11953 Len=0
3. **Server FIN Packet:**
   1. **Description:** Server initiates termination
   2. **Packet:** [FIN, ACK] Seq=2 Ack=2 Win=11953 Len=0
4. **Client ACK Packet:**
   1. **Description:** Client acknowledges server’s FIN
   2. **Packet:**  [ACK] Seq=2 Ack=3 Win=64582 Len=0

**Part 3:**

**Task 1: Generate UDP traffic and capture packets**

**Step 1:** Open a network application that uses UDP (**YOUTUBE**).

**Step 2:** Start the application to generate UDP traffic.

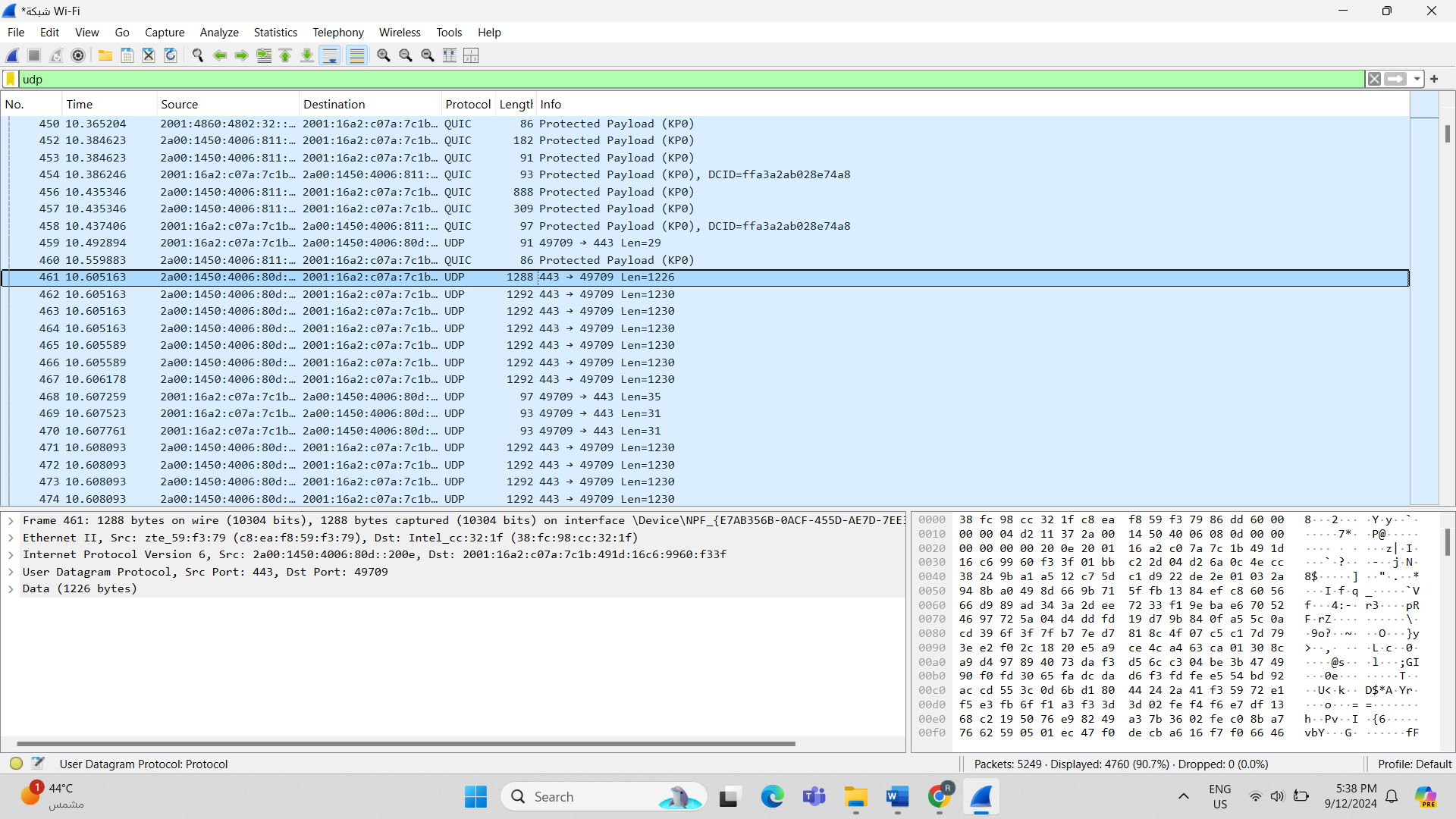
**Step 3:** Start capturing packets in Wireshark while the UDP application is running.

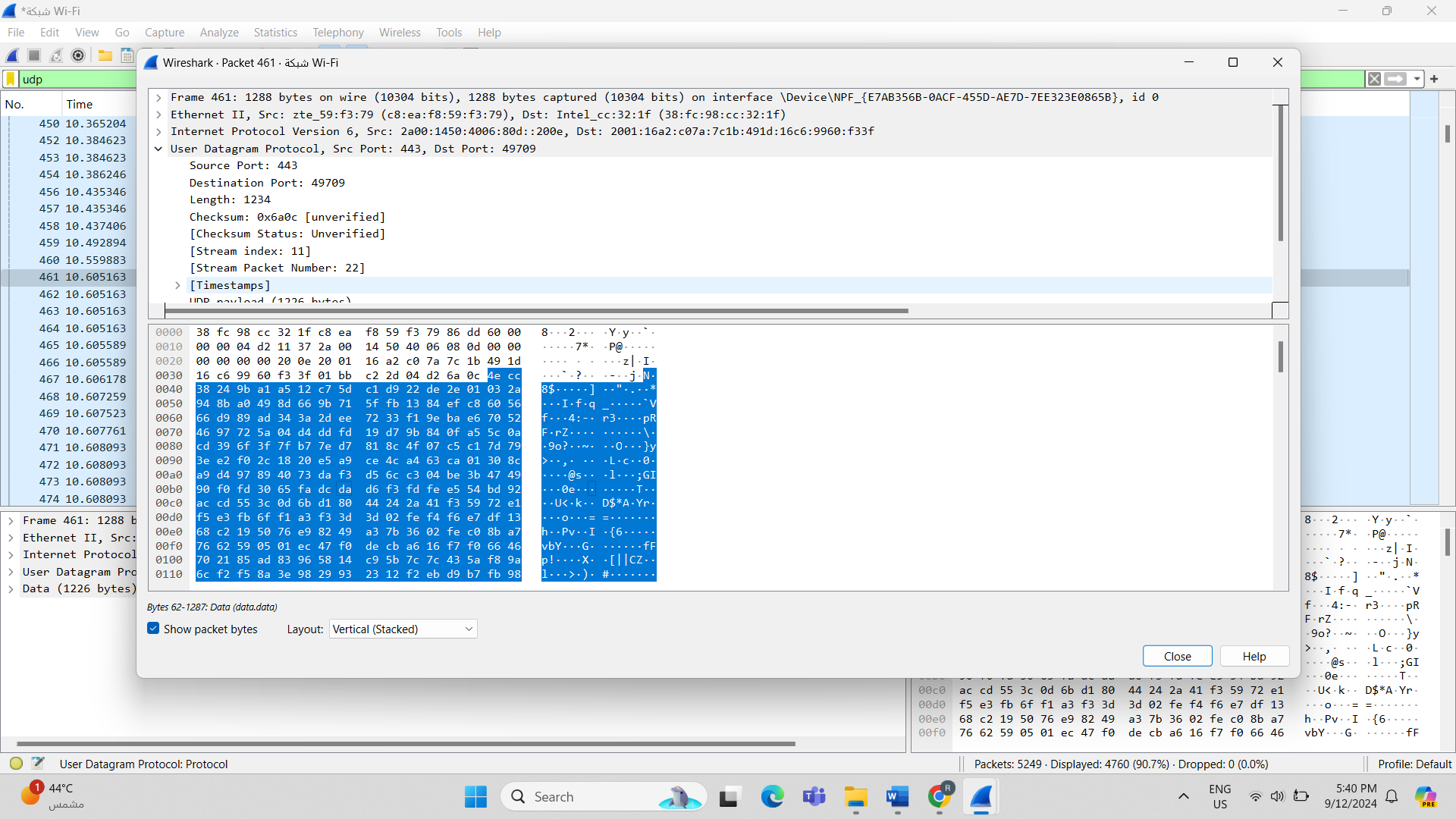
**Step 4:** After sufficient traffic is generated, stop capturing packets.

**Task 2: Filter and analysis UDP Packets**

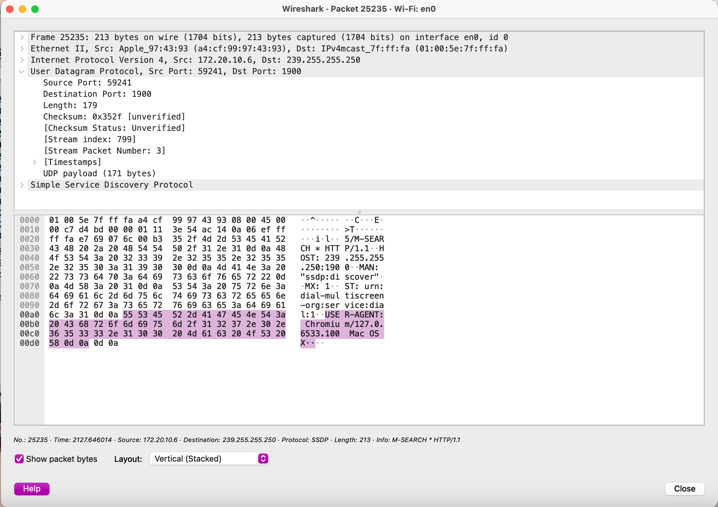
**Step 1:** In the filter bar, type UDP and press Enter.

**Step 2:**



**Step 3:** 

**Step 4:**



**Step 5:**

 **UDP is simpler** due to its smaller header (8 bytes), fewer fields, lack of connection management, and absence of error recovery and flow control mechanisms.

 **TCP is more complex** because it provides reliable, connection-oriented communication, ensuring data integrity, order, and congestion management.

**Part 4:**

**Task 1:**

|  |  |  |
| --- | --- | --- |
|  | TCP or UDP | Reasons |
| Reliability and Connection  Establishment | **TCP** | Uses a connection-oriented approach through a three-way handshake (SYN, SYN-ACK, ACK). This ensures data is reliably delivered, acknowledged, and retransmitted if necessary. |
| Data Integrity and Ordering | **TCP** | Provides in-sequence data delivery using sequence numbers. It also guarantees integrity through error-checking mechanisms like checksums. |

**Task 2:**

|  |  |  |
| --- | --- | --- |
|  | TCP | UDP |
| Use Cases | Designed for applications that require secure and reliable communication, such as transferring files (FTP), sending emails (SMTP), or browsing the web (HTTP/HTTPS). TCP ensures all data is delivered accurately and in sequence. | Best suited for scenarios where low latency is crucial and minor data loss is acceptable, such as live video streaming, online multiplayer gaming, or Voice over IP (VoIP). UDP prioritizes speed over reliability. |
| Performance | Offers reliable data transfer with error-checking, data retransmission, and in-order delivery, but tends to introduce higher latency due to the connection establishment and flow control processes. | Provides faster data transmission with lower latency since it doesn't require connection setup or acknowledgment of received packets. However, this comes at the cost of potential data loss or out-of-order delivery. |

**Student: Rana Aljasser**

**ID: 411201818**