

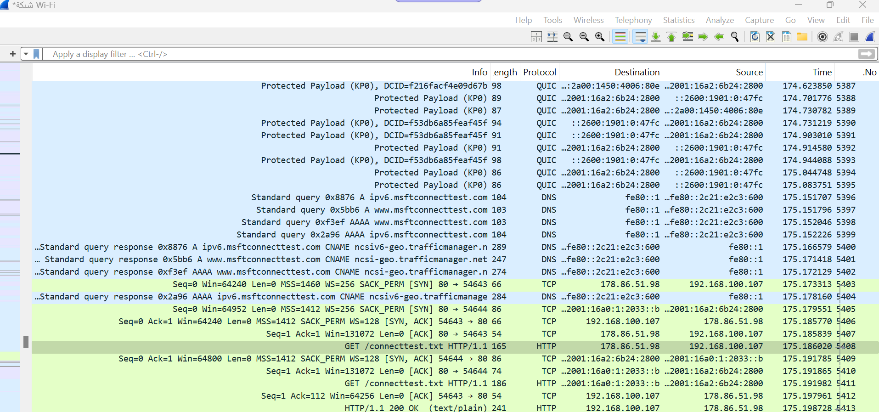
**Lab Week 2**

**The Internet Protocols**

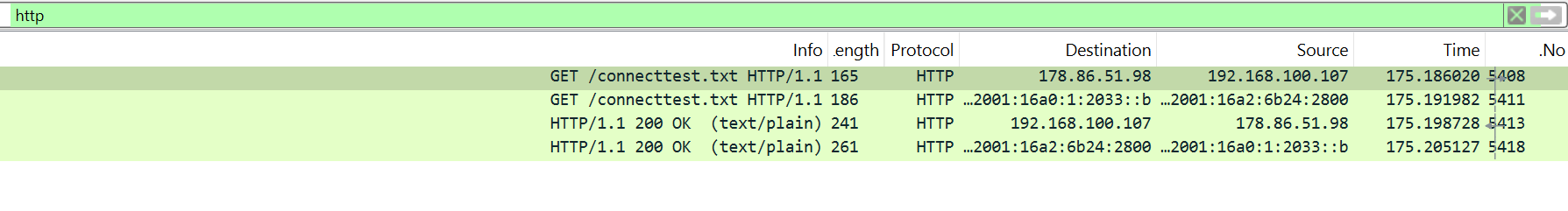
|  |
| --- |
| Shahad Nasser Alwatban |

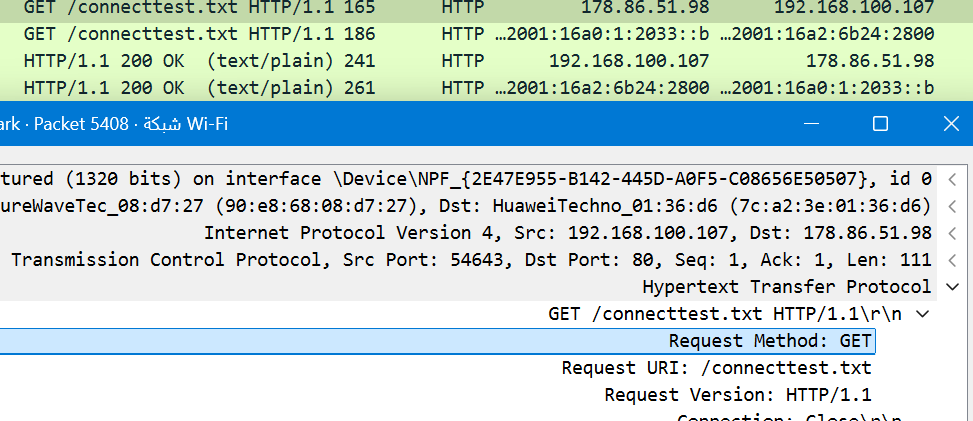
**Part 1: Capturing HTTP Traffic.**

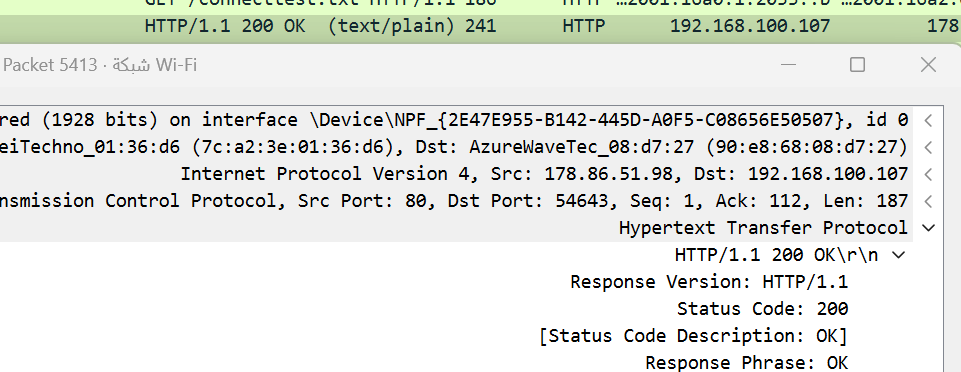
**Task 1:** Start Wireshark and capture packets.

****"I opened Wireshark and Selected the network interface connected to the internet (Wi Fi) and them I Capturing Packets and then Open web browser and navigate to (https://qu.edu.sa) website after the website has fully loaded, stop capturing packets "

**Task 2:** Filter HTTP packets and analyze them.

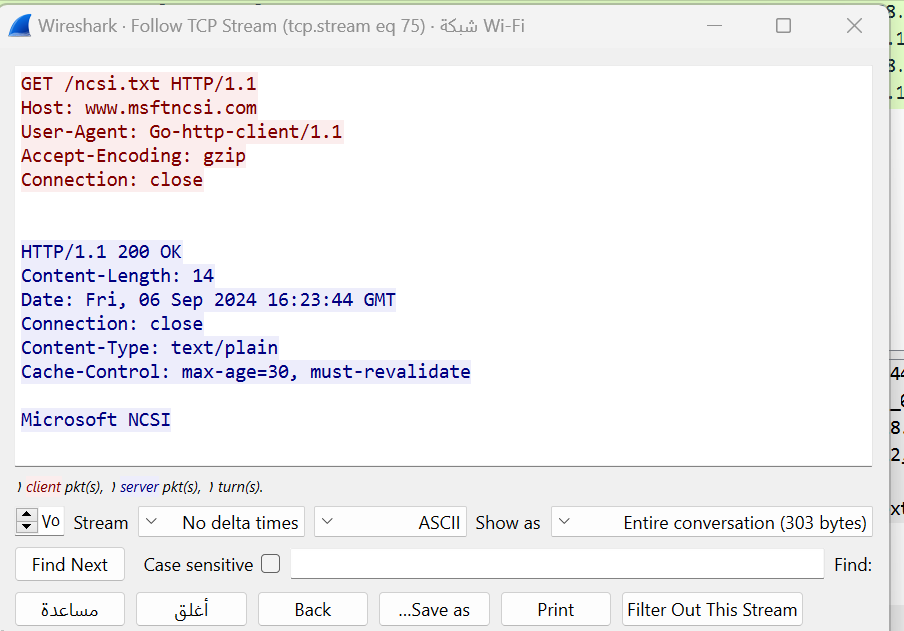
In filter bar, I filter only HTTP packets from the capture, and I found HTTP request messages (GET) also another one with response codes (200 OK):

Here is HTTP request with GET method.

Here is HTTP response with response 200 ok.

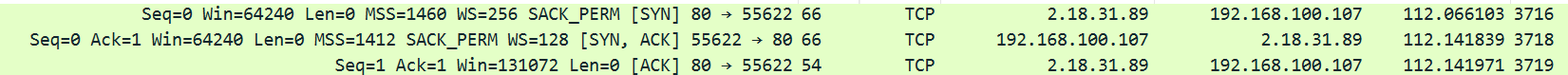
**Part 2: Analyzing TCP/IP Traffic.**

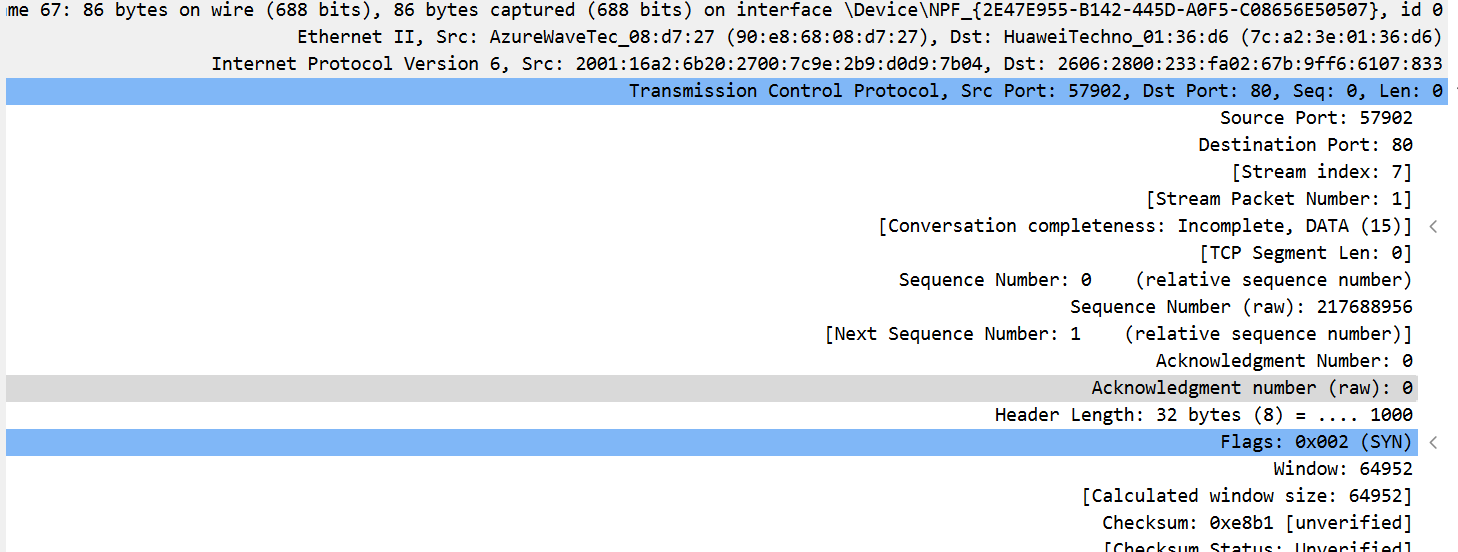
**Task 1:** Filter TCP packets

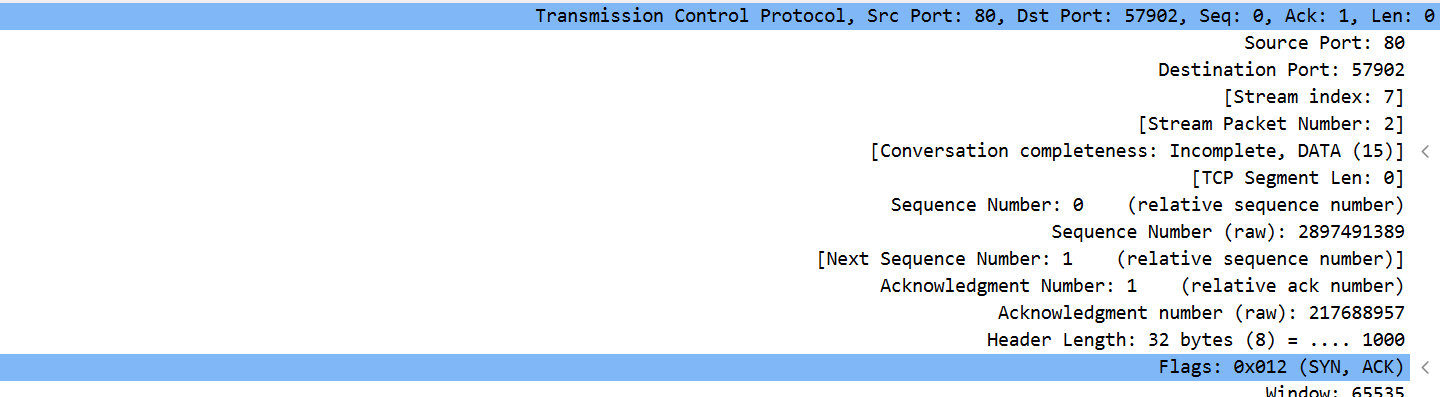
Here I Select a TCP packet related to HTTP and select "Follow" -> "TCP Stream", This shows the entire conversation between the client and server.

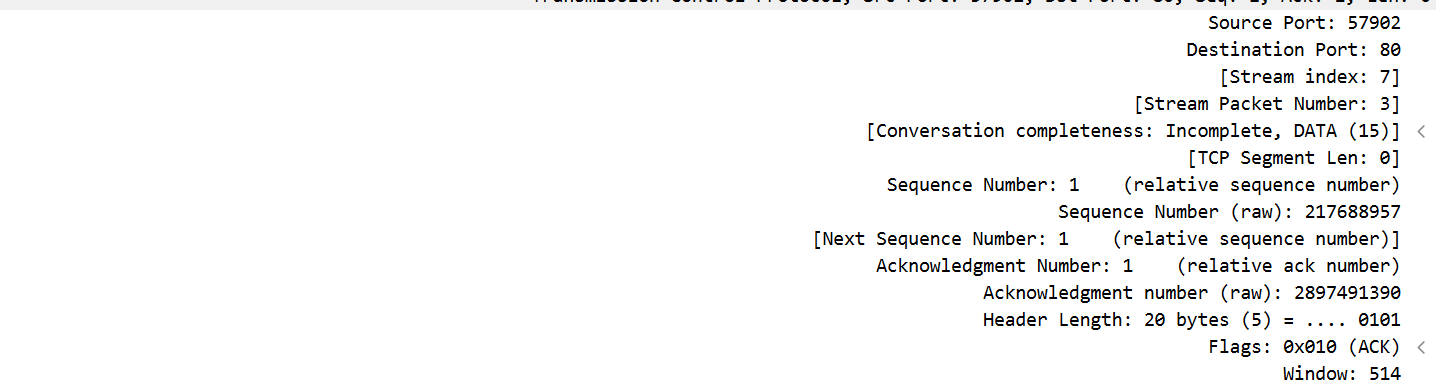
**Task 2:** Analyze TCP handshake and investigate Data Transfer and Termination

I Find 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.)

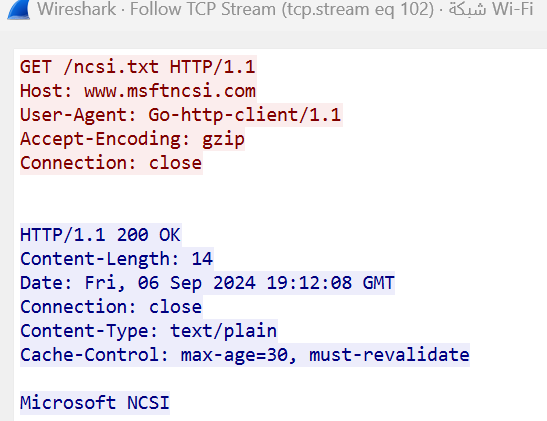
Here sequence and acknowledgment numbers.

SYN : the sequence and acknowledgment numbers

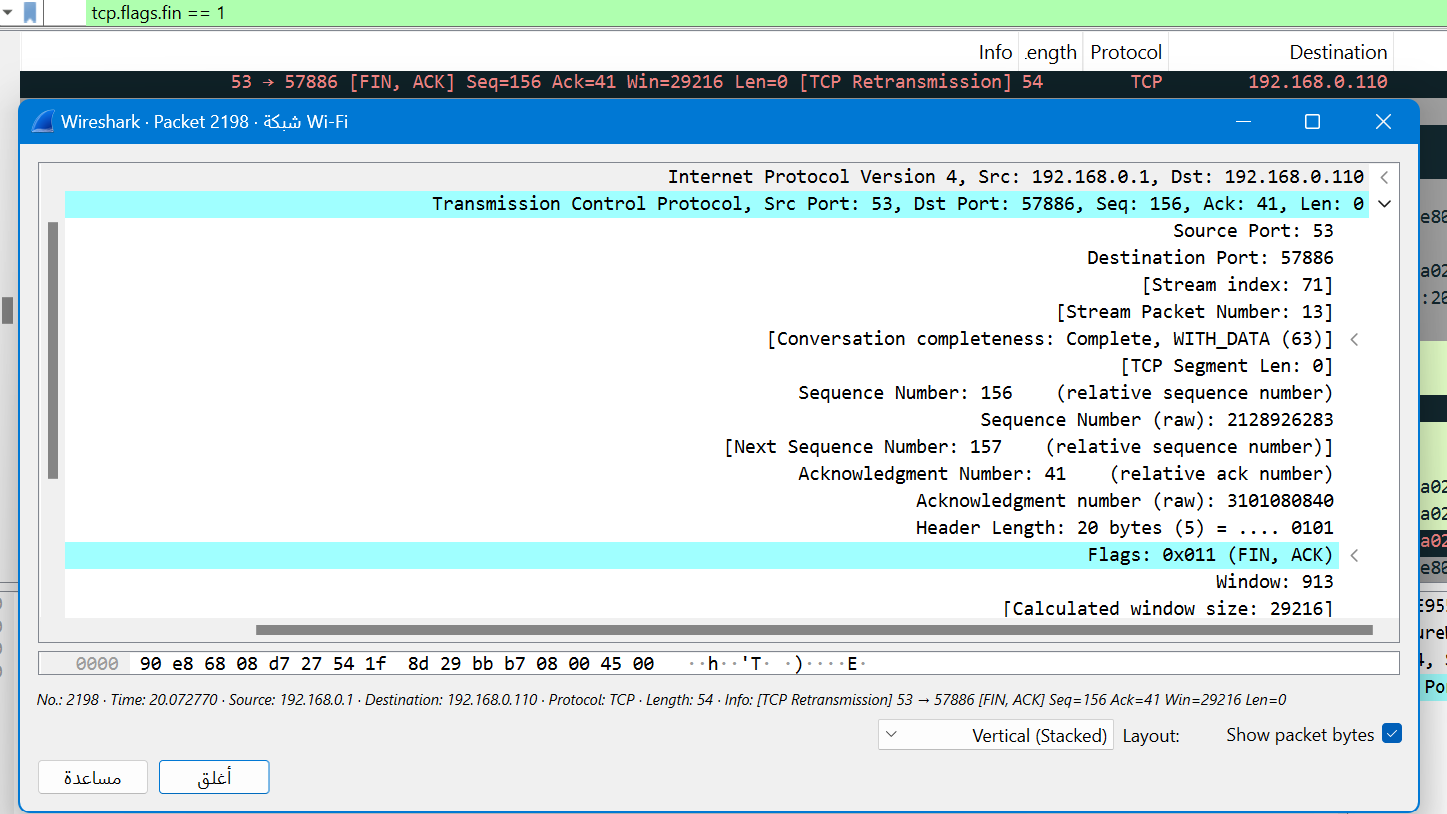
SYN-ACK: the sequence and acknowledgment numbers

ACK: the sequence and acknowledgment numbers

I use the "Follow TCP Stream" option to view the entire conversation between the client and server. This feature will show the data exchanged in a more readable format.

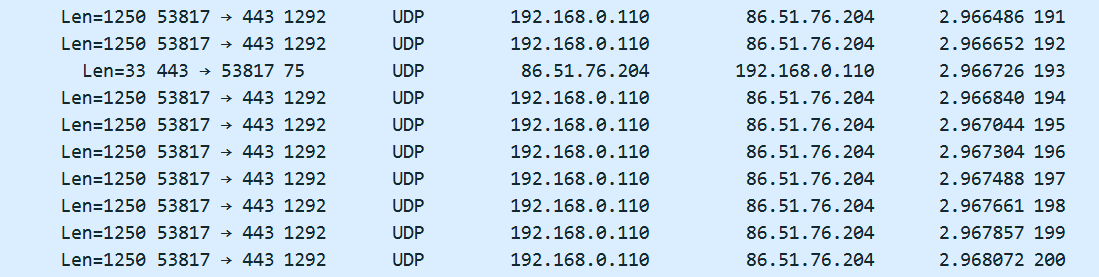


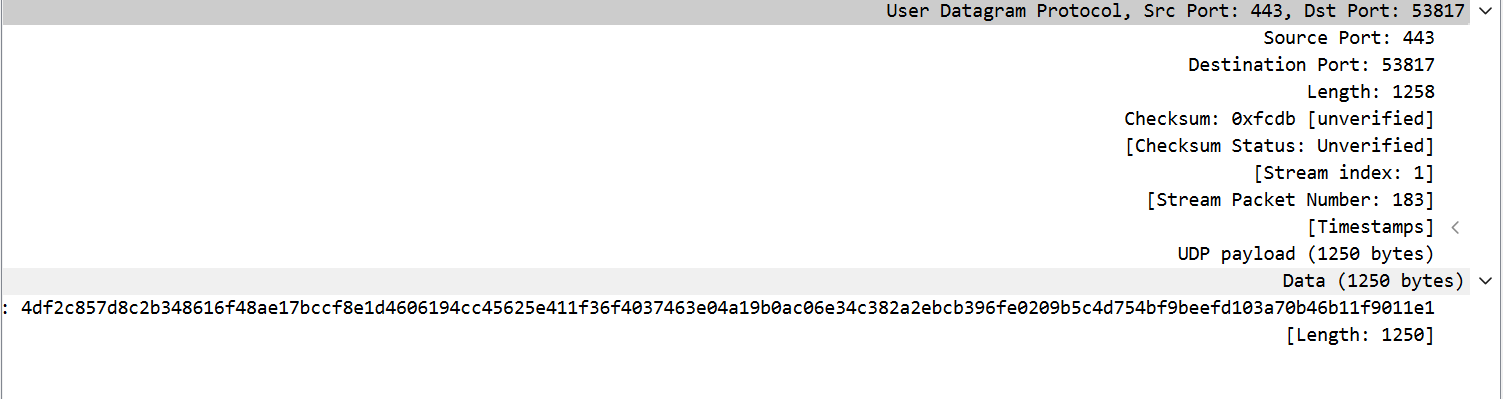
I looked at the TCP termination process (FIN, ACK packets) by using " tcp.flags.fin == 1" in the filter.



**Part 3: Capturing and Analyzing UDP Traffic**

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

In the filter bar, I type UDP

Observe the source and destination ports, length, and data.

**Compare the simplicity of UDP headers with TCP headers:**

The UDP header is much simpler, consisting of only 4 fields while TCP header is significantly more complex because TCP offers features like connection establishment, reliability, flow control, and congestion control. The TCP header consists of 10 mandatory fields

**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 uses acknowledgments (ACKs), Error Detection and Correction and Sequencing, unlike UDP |
| Data Integrity and Ordering | TCP | Each TCP segment includes a checksum for error detection, Guaranteed Ordering, Acknowledgments and Retransmissions, Error Detection and Correction unlike UDP |

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

|  |  |  |
| --- | --- | --- |
|  | TCP | UDP |
| Use case | * Web browsing * Email * File Transfer | * Live video * Online gamming * Audio streaming |
| Performance | Reliability, Throughput, Latency, Overhead, Connection-Oriented Nature, Data Ordering and Performance Optimization Techniques | Optional checksum enables error detection, while minimal overhead, lack of connection setup, and small headers ensure high throughput, low latency, and reduced overhead. |