**Lab 11**

**1)Title: Introduction to Packet Analysis using Wireshark**

**2) Objective**

The primary objectives of this lab are:

* To understand the fundamental concepts of network packet analysis.
* To gain familiarity with the Wireshark interface and its core functionalities for capturing and analyzing network traffic.
* To apply display filters to isolate specific types of traffic for detailed inspection.
* To identify basic network protocols and analyze their behavior for both diagnostic and security purposes.

**3) Theory**

**Packet Analysis** is the process of capturing, inspecting, and interpreting data packets as they travel across a network. It is a critical skill for network administrators and security analysts, used for troubleshooting network problems, identifying performance bottlenecks, and detecting malicious activity.

**Wireshark** is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Wireshark captures network packets in real-time and presents them in a human-readable format, allowing users to drill down into the details of each packet's headers and payload.

The typical Wireshark interface consists of three main panes:

1. **Packet List Pane:** Displays a summary of each captured packet.
2. **Packet Details Pane:** Shows the protocols and fields of the selected packet in a hierarchical tree structure.
3. **Packet Bytes Pane:** Displays the raw data of the selected packet in hexadecimal and ASCII.

**4) Procedures/Commands/Screenshots of Output**

For this report, I have chosen to perform the following four analysis exercises:

* **Exercise 1: Basic Packet Capture and Protocol Hierarchy Analysis**
* **Exercise 2: Analyzing DNS Query and Response**
* **Exercise 3: Analyzing ICMP Ping Request and Reply**
* **Exercise 4: Filtering HTTP and HTTPS Traffic**

**Exercise 1: Basic Packet Capture and Protocol Hierarchy Analysis**

**Objective:** To capture baseline network traffic and identify the most common protocols in use.

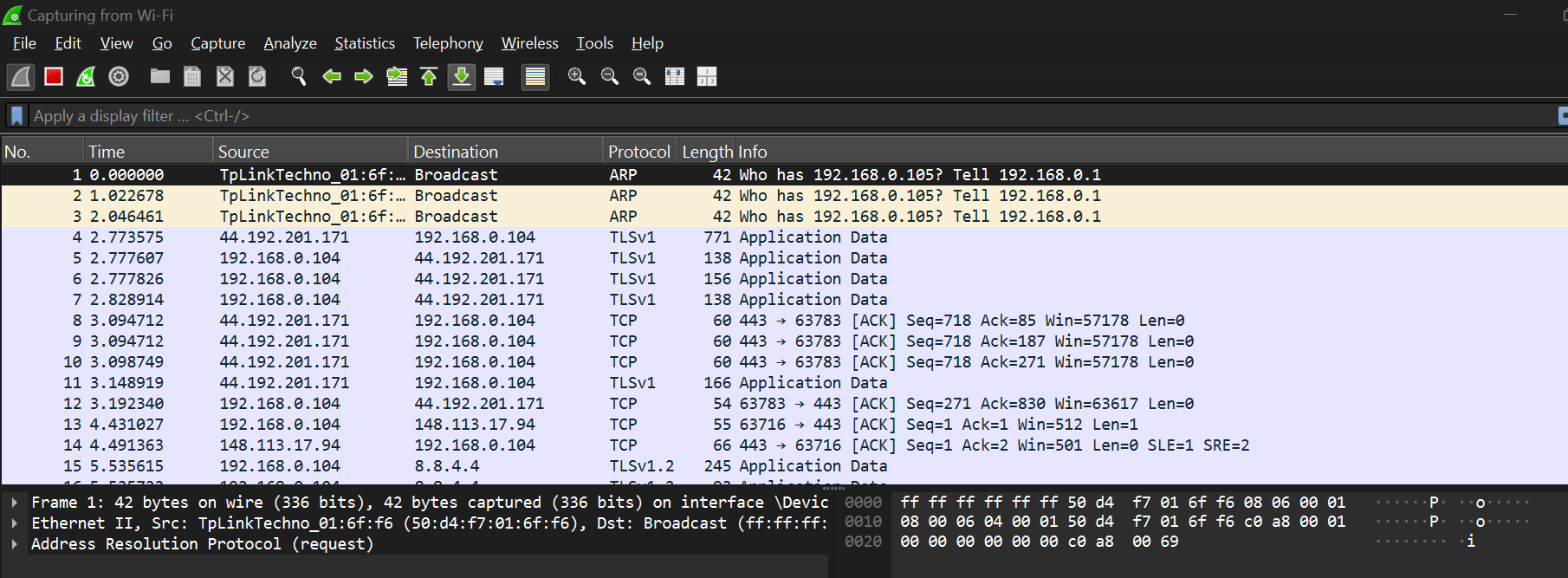
**Procedure:**

1. Opened Wireshark with Administrator privileges.
2. Selected the active network interface (Wi-Fi).

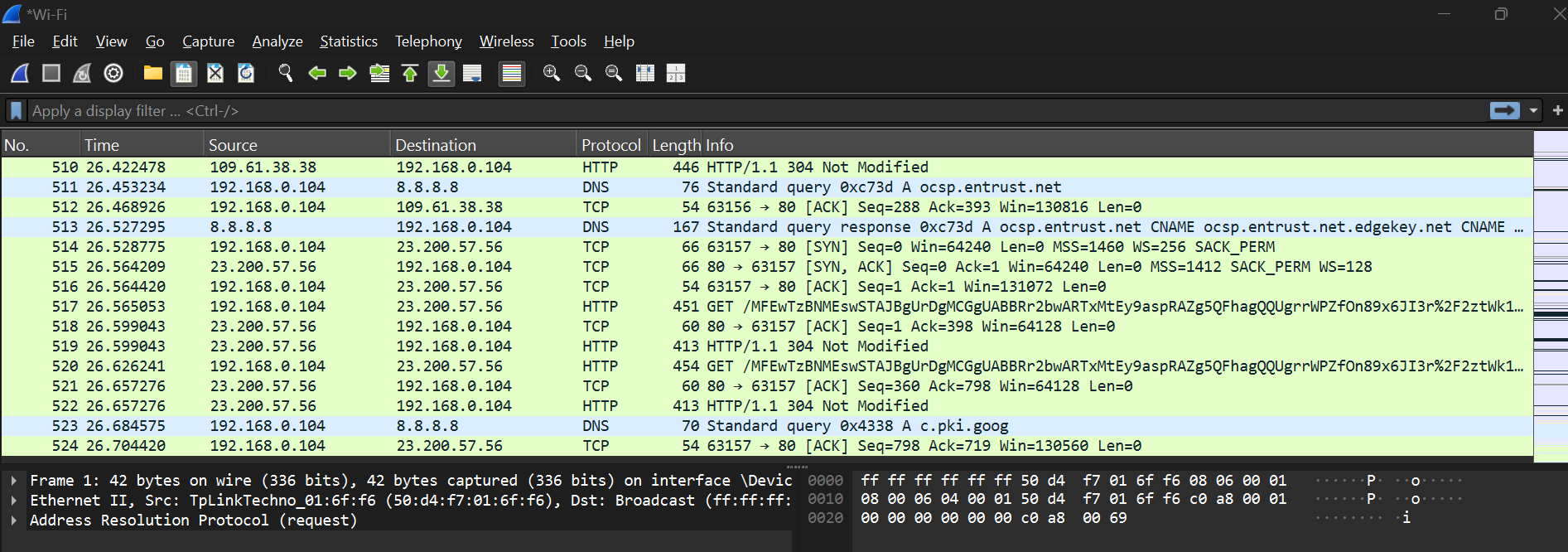
A screenshot of a computer

AI-generated content may be incorrect.

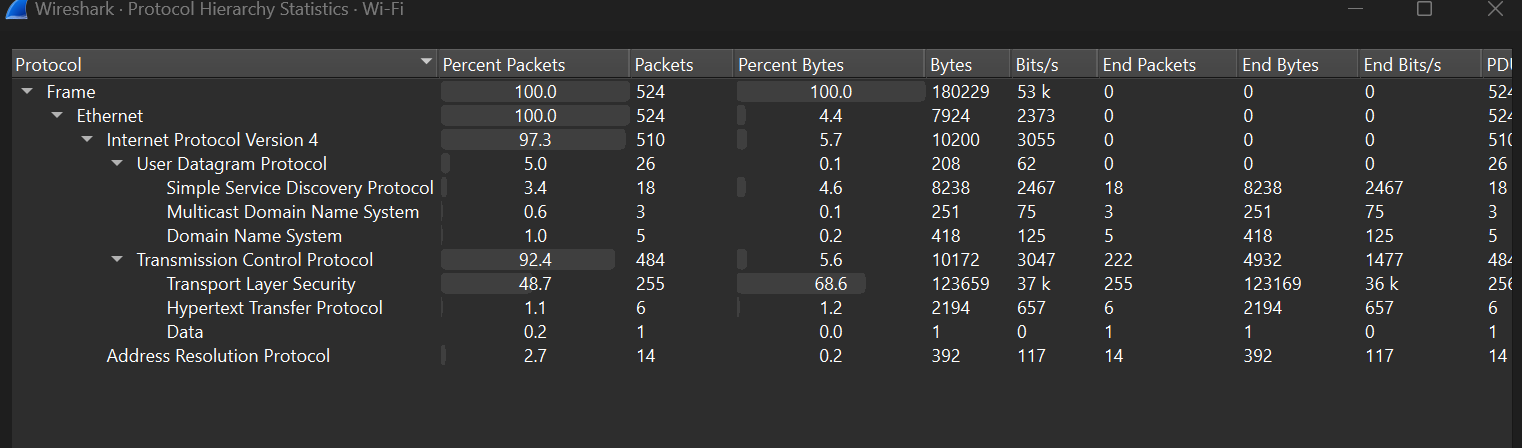
1. Clicked the blue shark fin button to start the capture. ( Generated traffic by browsing a website. )



1. Clicked the red stop button after about 15 seconds to end the capture.



1. Navigated to **Statistics > Protocol Hierarchy** to view the distribution of protocols.

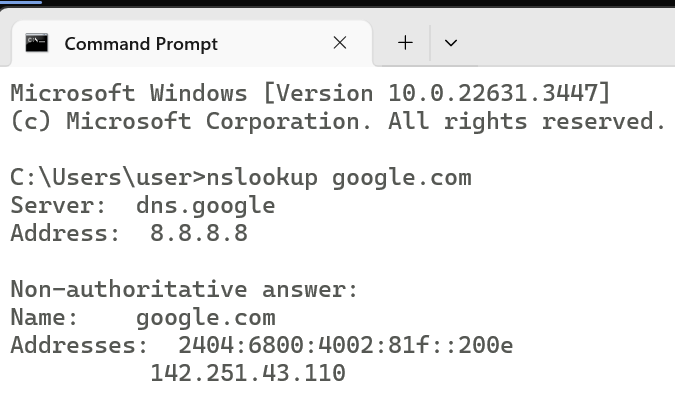


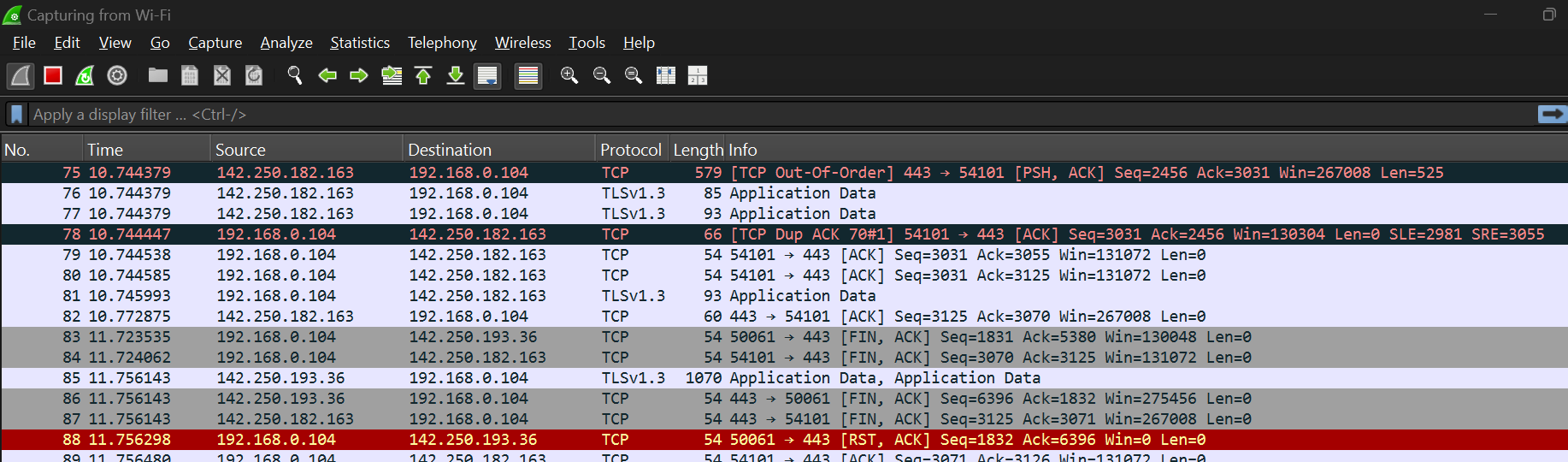
**Observation:**  
The Protocol Hierarchy window provided a clear breakdown of all captured traffic. As expected, the majority of traffic was TCP, which encapsulated protocols like TLS (for HTTPS) and HTTP. A significant portion was also DNS (UDP) for domain name resolution. This overview is invaluable for quickly understanding the composition of network traffic.

**Exercise 2: Analyzing DNS Query and Response**

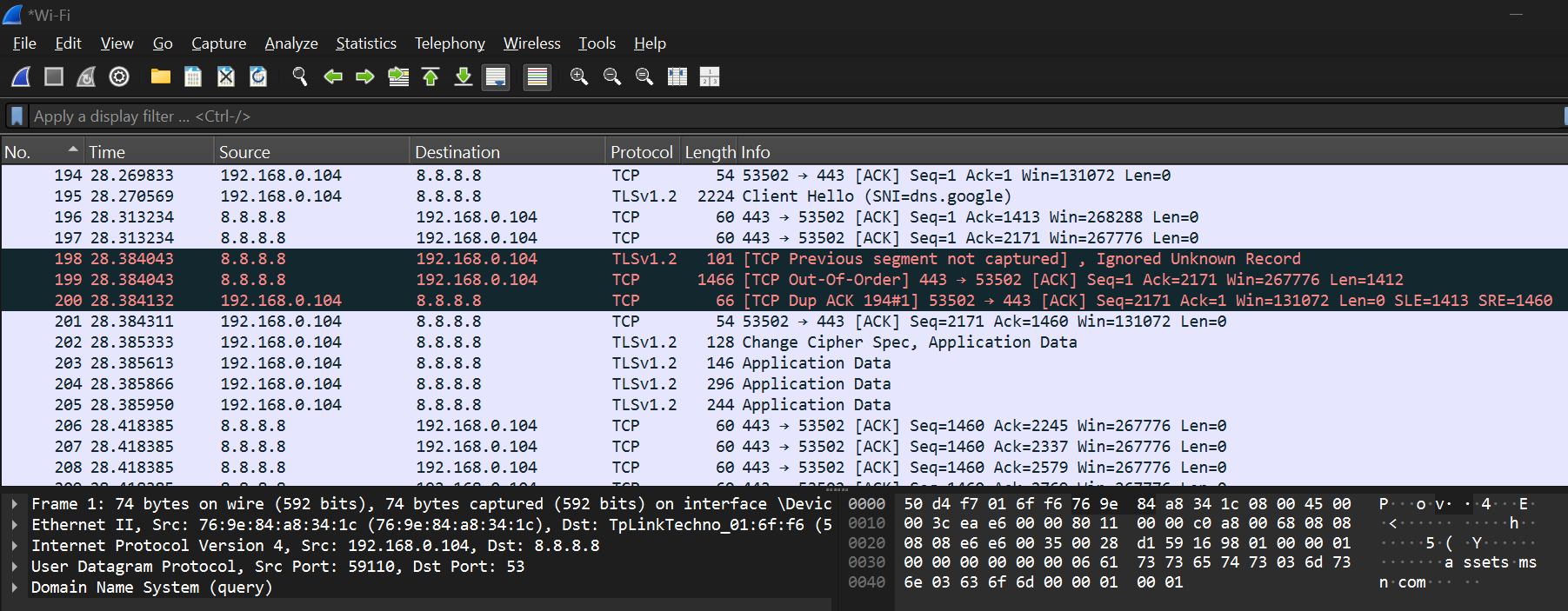
**Objective:** To observe how DNS resolves a domain name to an IP address.

**Procedure:**

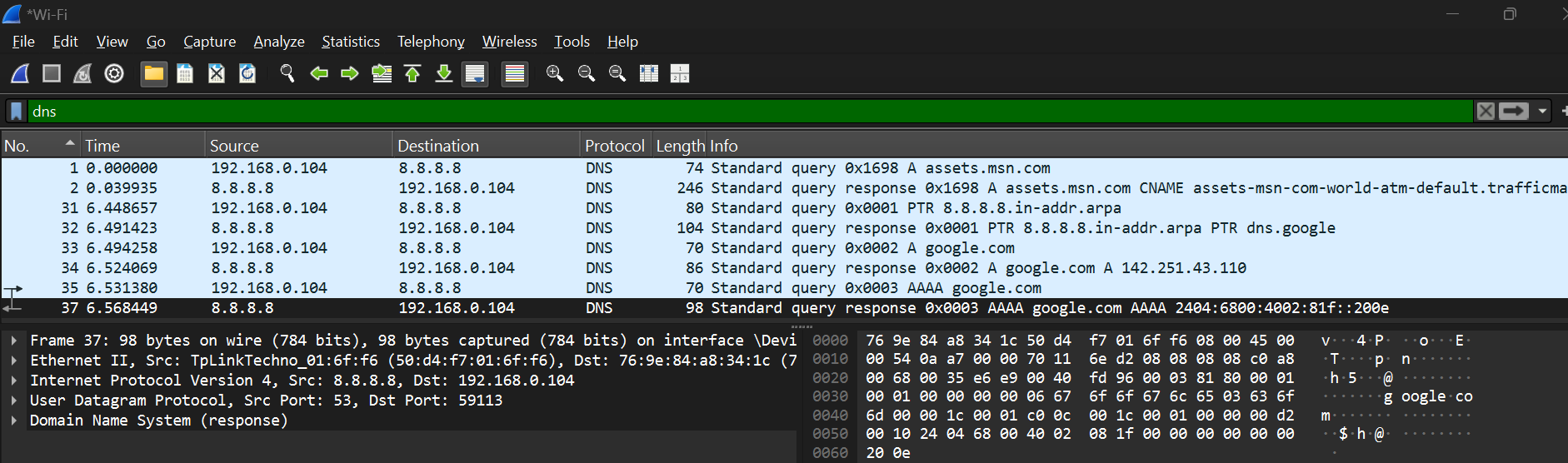
1. Started a new packet capture in Wireshark.
2. Opened a command prompt and executed the command nslookup google.com to generate DNS traffic. 



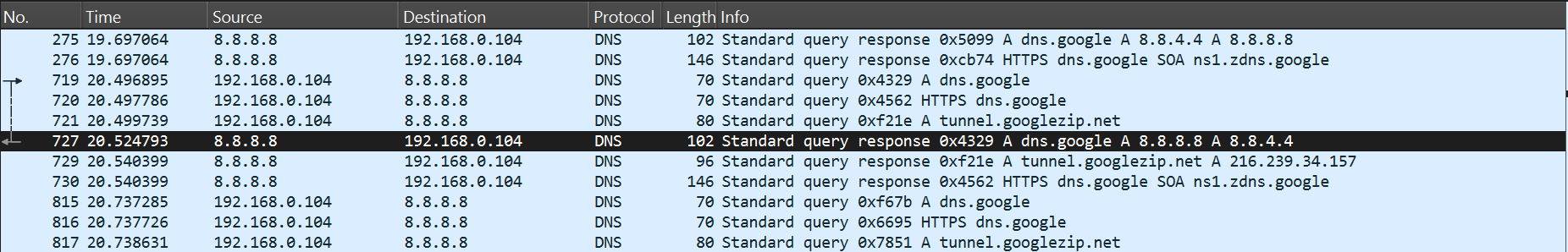
1. Stopped the capture.

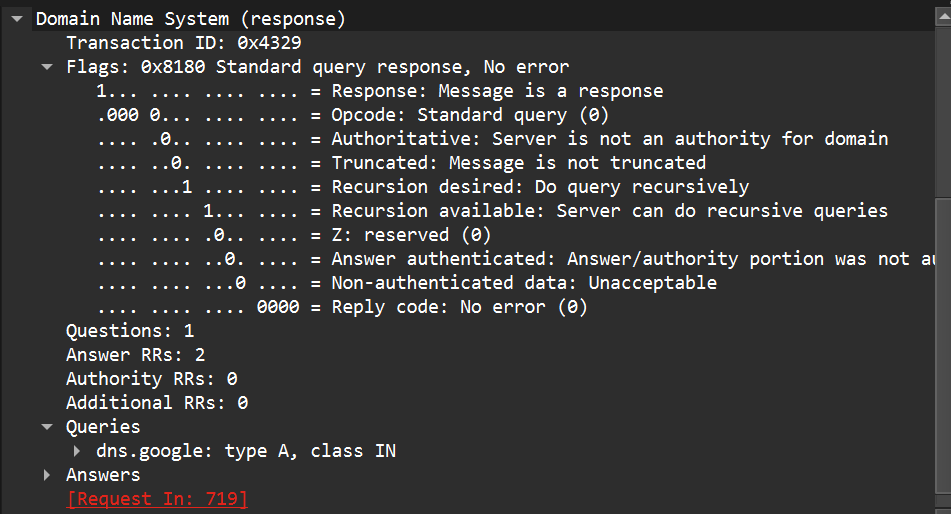


1. Applied the display filter: dns

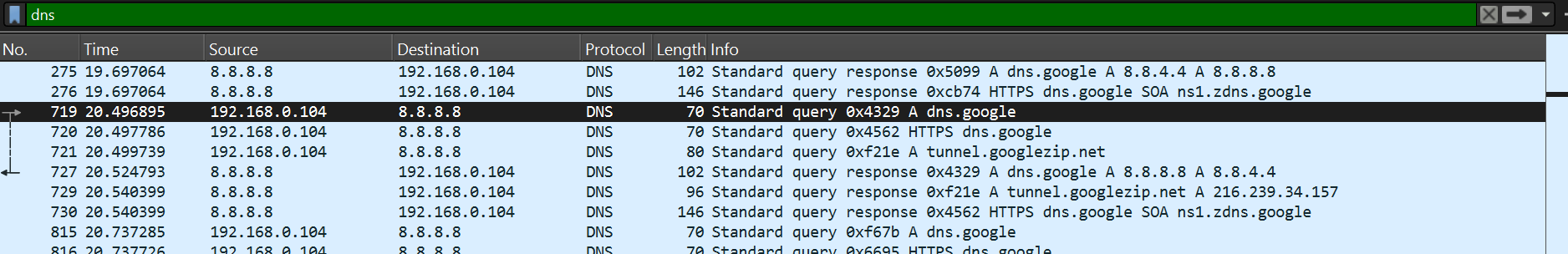


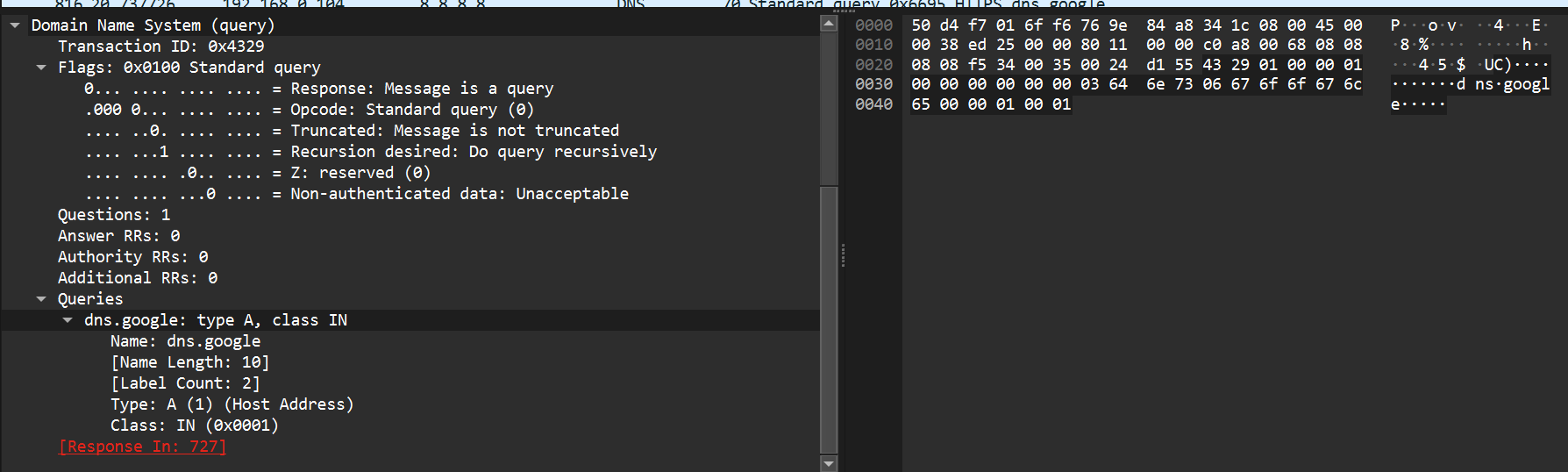
1. Selected a DNS query packet and expanded the **Domain Name System (query)** section in the details pane to view the questioned domain name.





1. Selected the corresponding DNS response packet and expanded the **Domain Name System (response)** section to view the resolved IP address(es).



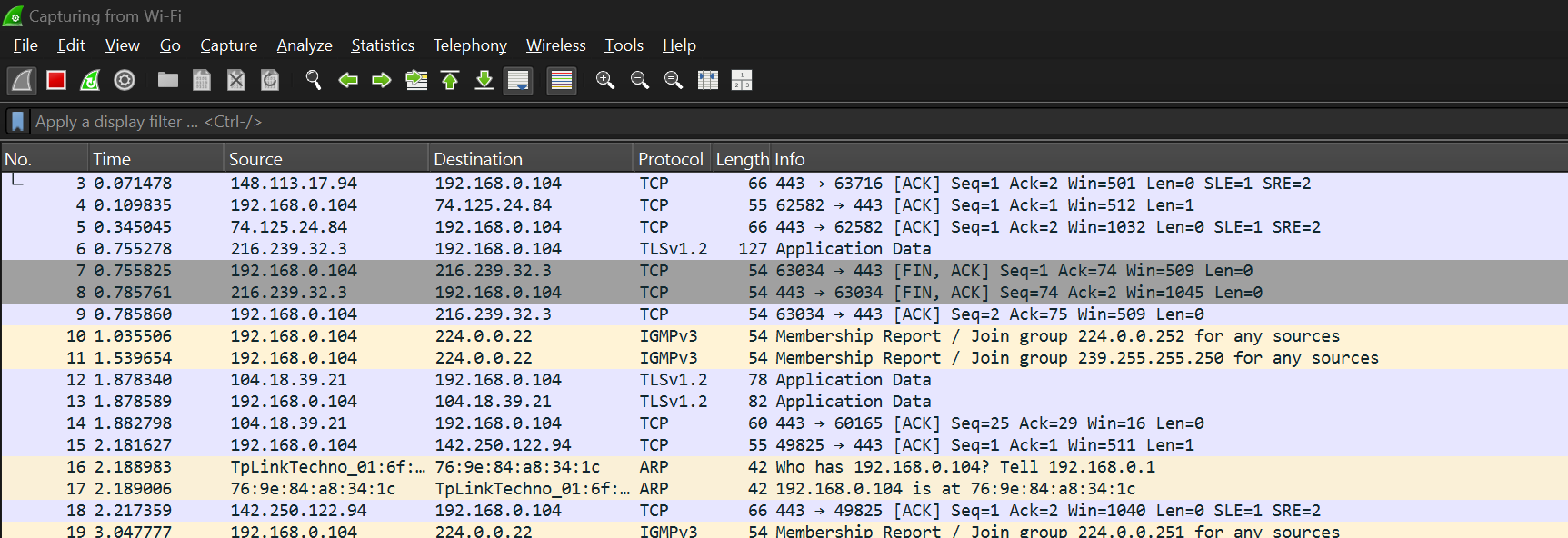


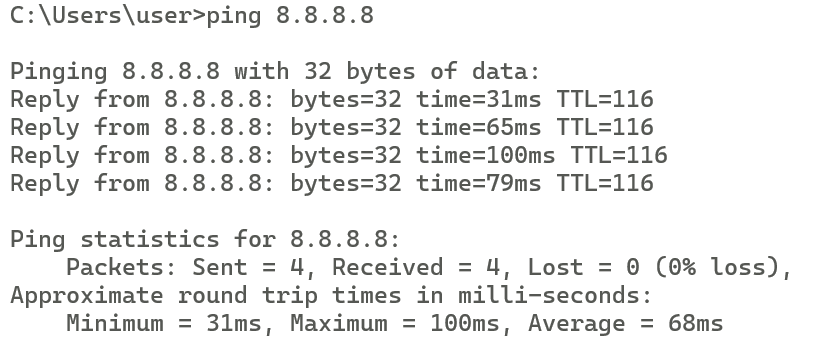
**Observation:**  
The analysis clearly showed a standard DNS transaction. The query packet was sent to a predefined DNS server, asking for the A record of "google.com". The response packet was received from the same server, containing the IPv4 address answer. This confirms the unencrypted nature of standard DNS traffic.

**Exercise 3: Analyzing ICMP Ping Request and Reply**

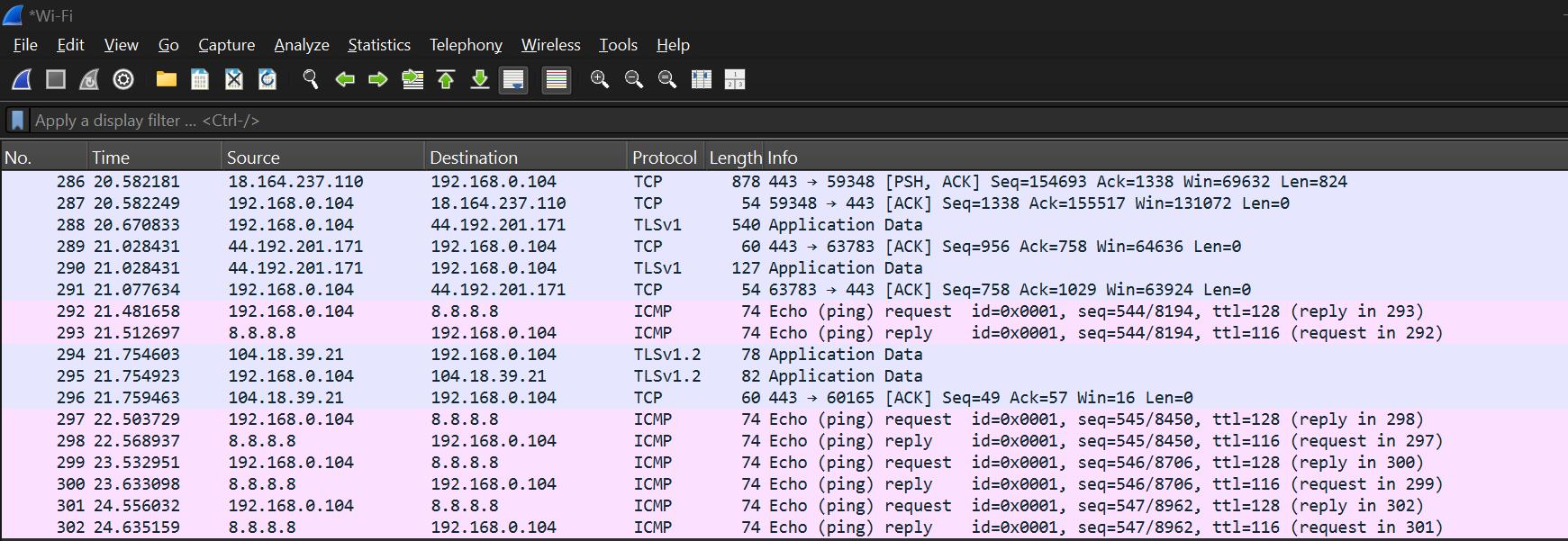
**Objective:** To demonstrate the use of ICMP for network diagnostics.

**Procedure:**

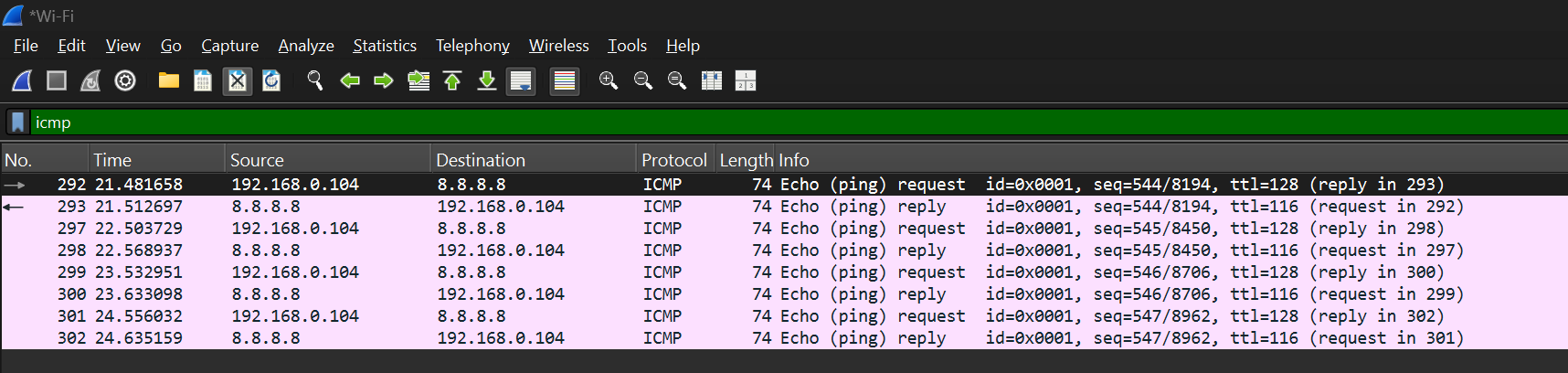
1. Started a new packet capture. 
2. Opened a command prompt and executed the command ping 8.8.8.8 (Google's DNS server).



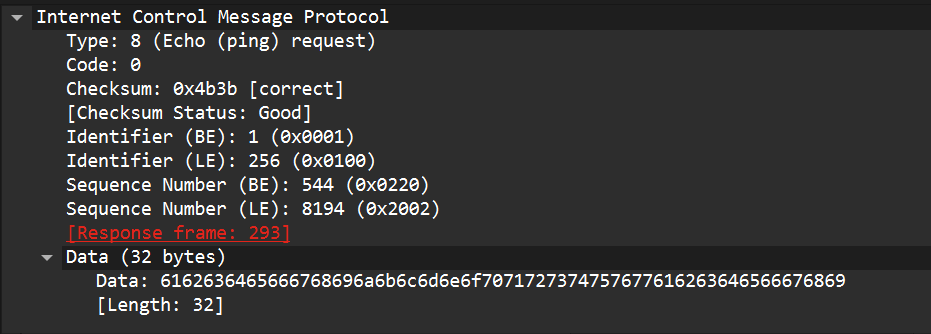
1. Stopped the capture after the ping command completed.

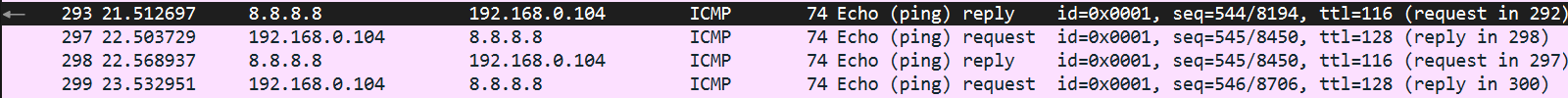


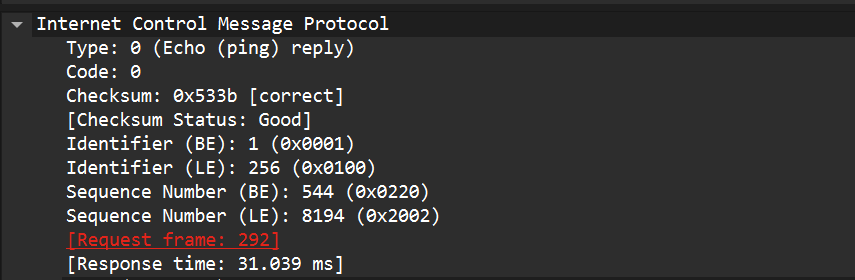
1. Applied the display filter: icmp.



1. Examined an Echo (ping) Request packet (icmp.type == 8) and its corresponding Echo Reply packet (icmp.type == 0).







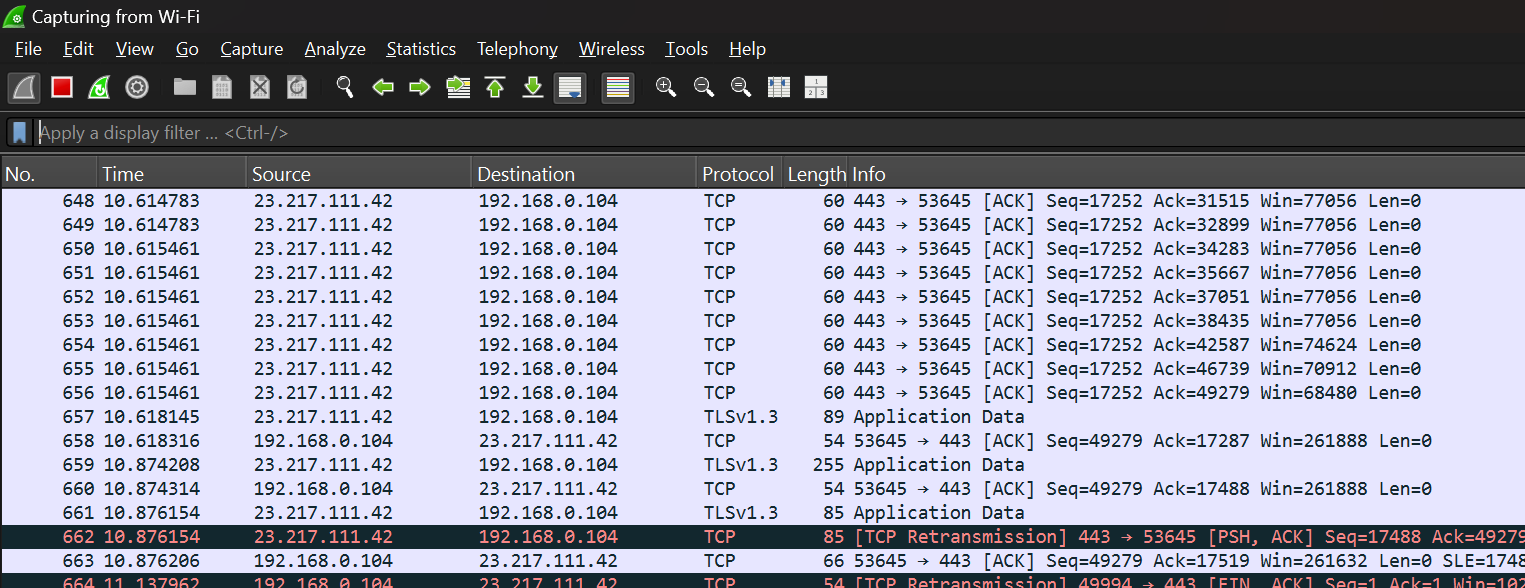
**Observation:**  
The capture successfully displayed the ICMP Echo Request packets generated by the ping command and the Echo Reply packets returned by the destination host (8.8.8.8). The details pane showed important data such as the sequence and identifier numbers, which are used to match requests with replies. It was also discovered that ICMP wasn’t originally found with Wi-Fi and after configuring it with Ethernet, the ICMP was available. Only ICMPV6 was found with Wi-Fi configuration.

**Exercise 4: SSL/TLS Handshake Analysis**

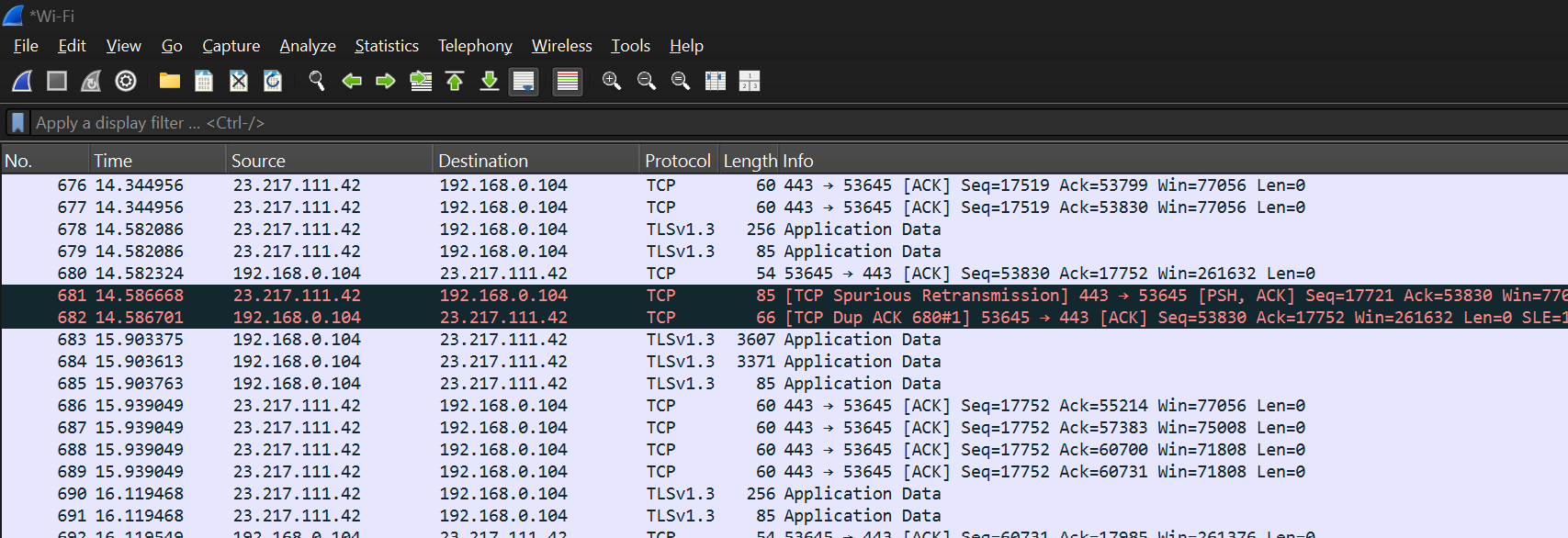
**Objective:**  
To explain how a secure session is established using SSL/TLS and observe the handshake process in Wireshark.

**Procedure:**

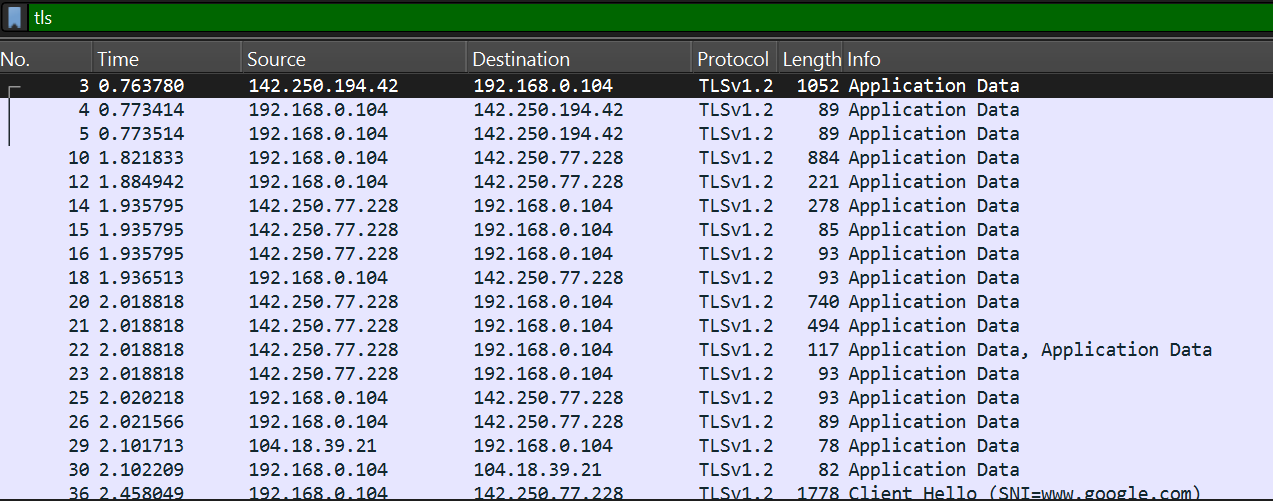
1. Started a new packet capture in Wireshark.



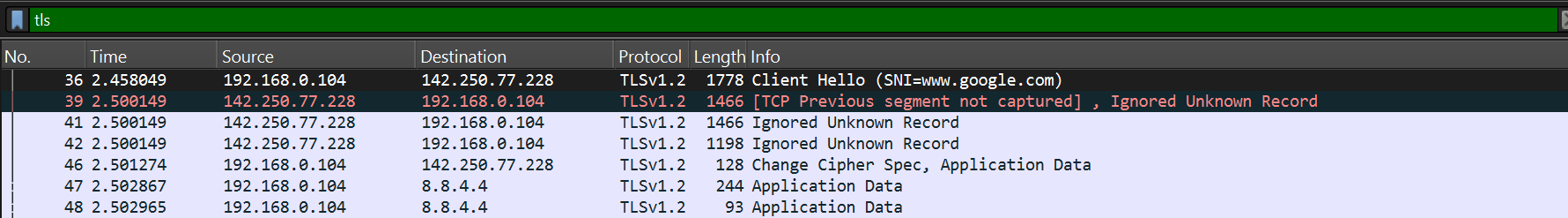
1. Opened a web browser and visited a secure HTTPS website (e.g., [https://www.google.com](https://www.google.com?utm_source=chatgpt.com)).
2. Stopped the capture after the page fully loaded.

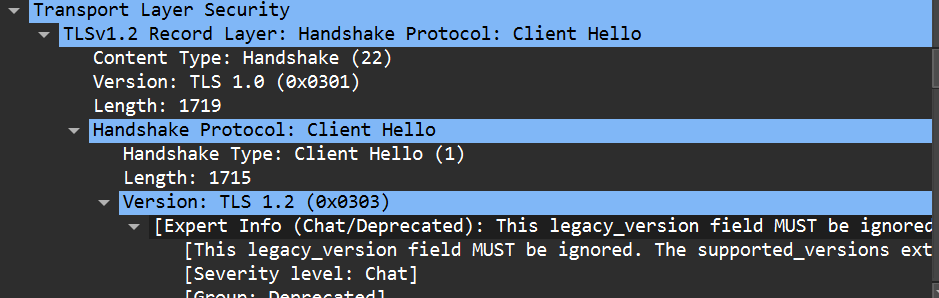


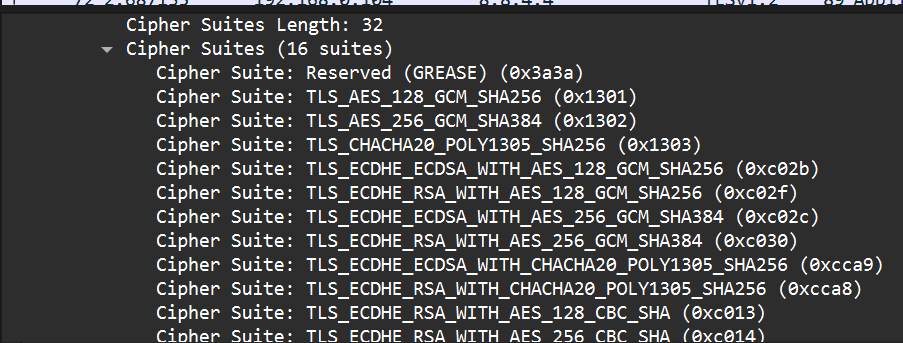
1. Applied the display filter: tls (or ssl in older Wireshark versions).



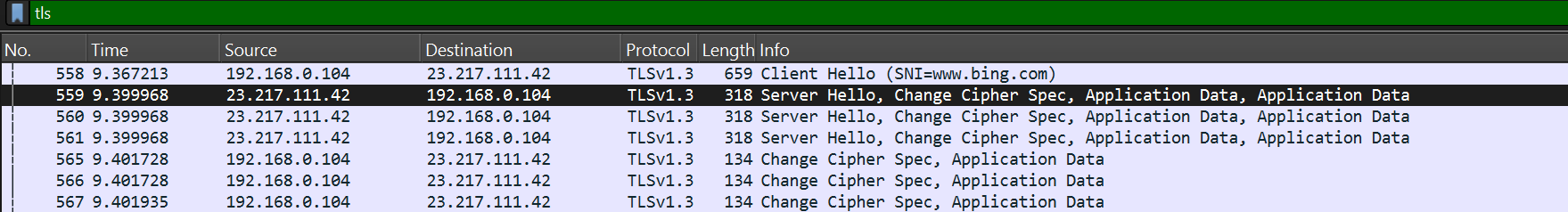
1. Selected a ClientHello packet and expanded the **Transport Layer Security** section to view details such as supported cipher suites and protocol version.

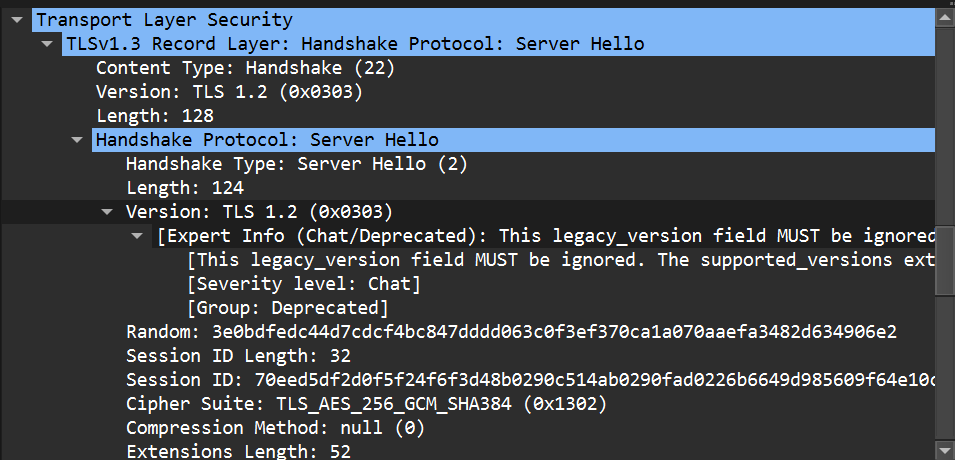






1. Selected the corresponding ServerHello packet and expanded the **Transport Layer Security** section to view the chosen cipher suite, session ID, and server certificate information.





**Observation:**  
In the ServerHello packet, we expanded the TLS details in Wireshark. The chosen cipher suite was TLS\_AES\_256\_GCM\_SHA384, confirming the server selected an encryption algorithm supported by the client. The Session ID uniquely identifies this session. Extensions such as supported\_versions and key\_share indicate TLS version negotiation and key exchange parameters. This confirms the server’s part in establishing a secure session

.

**5) Conclusion**

This lab provided a hands-on introduction to the powerful tool Wireshark. Through the exercises performed, basic capture, DNS, ICMP, and HTTP/S analysis. I gained practical experience in capturing live network traffic and using filters to isolate specific protocols with Ethernet and Wi-Fi configurations. I observed the clear-text nature of protocols like HTTP and DNS, underscoring the security risks they pose, and contrasted this with the encryption provided by TLS used in HTTPS.