Wireshark Network Traffic Analysis Report

Task Name: Task 5 – Capture and Analyze Network Traffic Using Wireshark

Target Site: “http://demo.testfire.net” **Tool**

Used: Wireshark (on Kali Linux) **Capture**

File: “traffic_<files>.pcap”

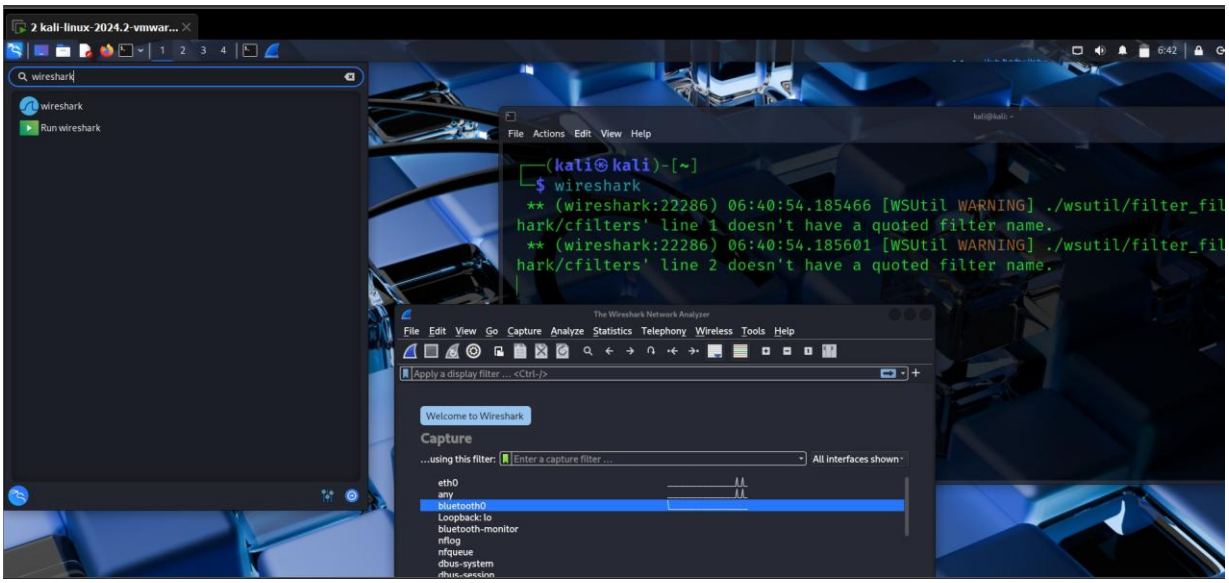
Objective:

The objective of this task is to perform live packet capturing using Wireshark, analyze the network packets and identifying basic protocols and traffic types.

1. Setting up Wireshark

Wireshark is a network packet analyzer tool that is publicly available and comes pre-installed in Kali Linux by default.

We can start Wireshark by typing wireshark in the Kali terminal or by opening it from the applications menu. Once it starts, it asks us to choose a network interface to capture traffic from. For this task, we selected the eth0 interface.



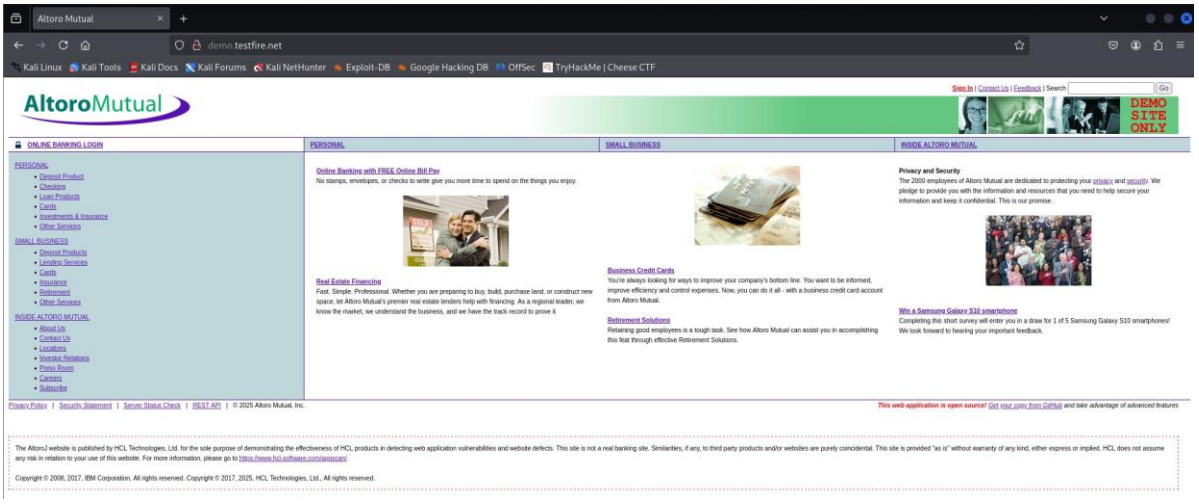
2. Capturing the Packets for a particular domain.

- Packets are small pieces of data that move across a network. They carry part of the message along with source and destination info. At the end, they're reassembled to complete the data.
- The configurations are set, now to hit the search engine with our demo site "demo.testfire.net" and view its network's traffic in wireshark and capture the packets for our analyzation.
- After selecting the interface (eth0), a window appears where live network packets start scrolling as they are captured. We can watch the traffic being captured in real time.

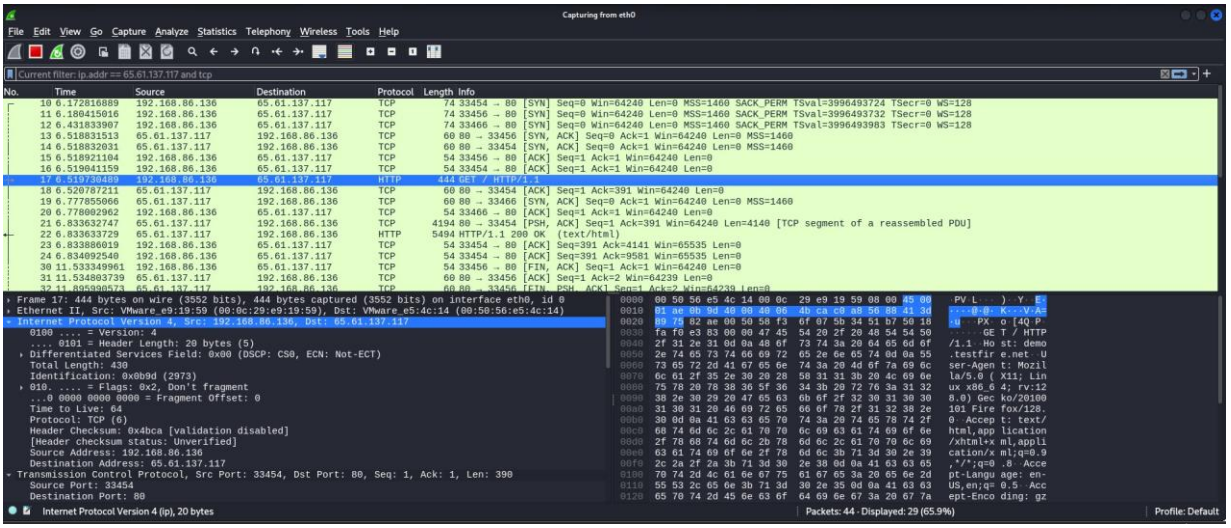
  The Wireshark window is divided into three panels :-

  - The top panel shows the list of captured packets.
  - The middle panel displays details and related protocols of the selected packet.
  - The bottom panel shows the raw data (bytes) inside the packet.

Demo site for our network analyzation:-



Wireshark capturing the network packets :-

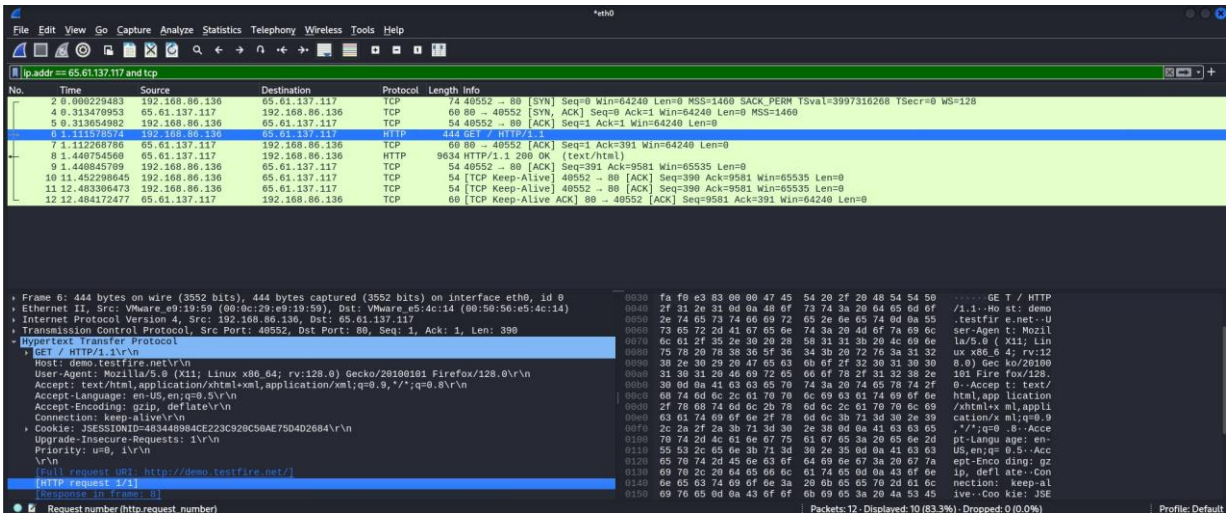


2. Traffic Filtering

Wireshark captures a large number of network requests, so to analyze a specific one, we need to use special keywords to filter the traffic like :-

- “ip.addr □□ 65.61.67.117” – shows all packets to or from that IP.
- “tcp.port □□ 80” – shows traffic on port 80 (HTTP).
- “ip.addr □□ 65.61.67.117 and tcp.port □□ 80” – traffic from/to that IP on port 80.

Network being filtered with specified key words :-



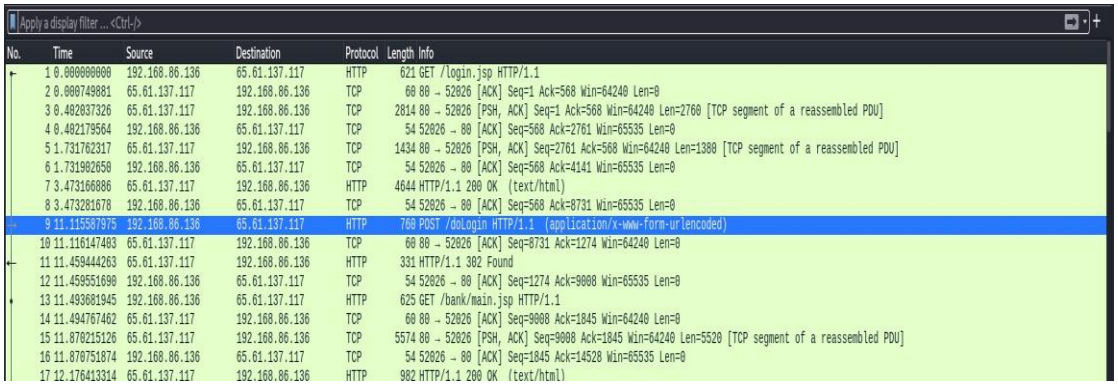
3. Packet List Overview

The Packet List Panel is the top section in Wireshark where all captured packets are shown in a list. Each row represents one packet, and each column shows specific details about it.

The fields listed in packet list panel are described as follows :-

- **No.** – This is the packet number, assigned in the order packets are captured.
- **Time** – Shows when the packet was captured. The time starts from when Wireshark begins capturing and helps measure the gap between packets.
- **Source** – Shows the IP address of the device that sent the packet.
- **Destination** –Shows the IP address of the device that is receiving the packet.
- **Protocol** –Tells which network protocol is used, such as TCP, HTTP, DNS, or ICMP. This shows what kind of communication the packet is part of.
- **Length** – Shows the size of the packet in bytes.
- **Info** – Gives a brief summary of what the packet is doing, such as request type, port numbers, or flags.

Packet Panel List :-

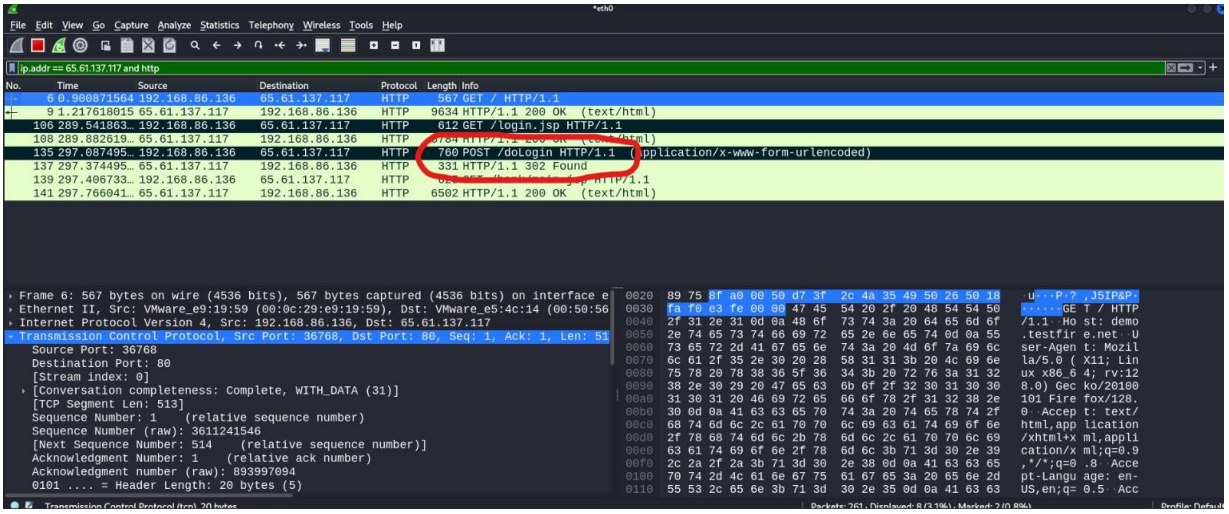


4. Capturing different Protocol's packets

HTTP (Hyper Text Transfer Protocol) :-

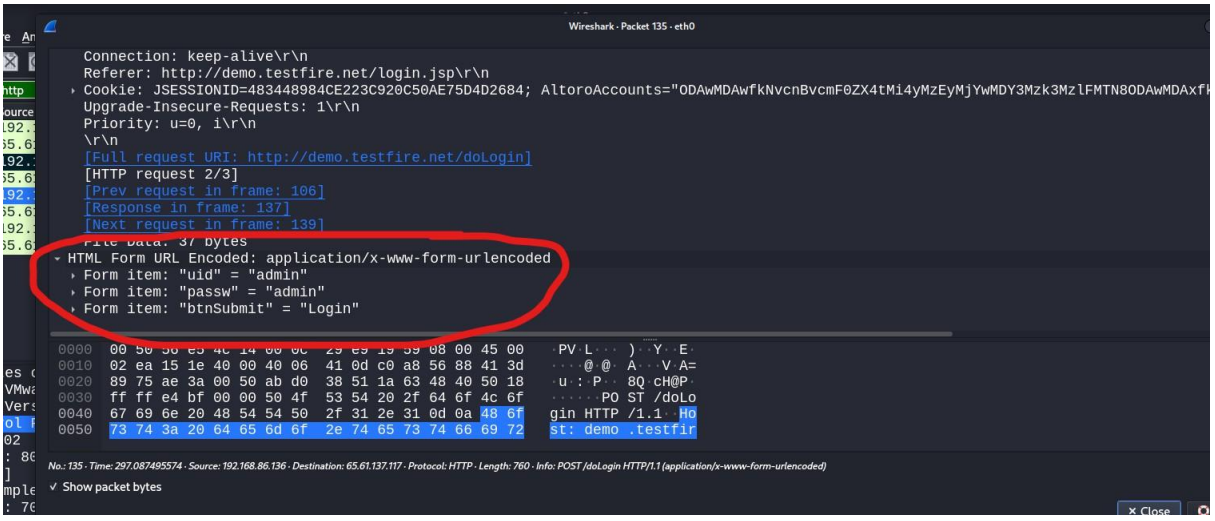
The basic protocol used for communication between our **web browser** and **web servers**. As we all know that this protocol is not safe as it does not provide with Encryption layer.

HTTP Login request packet captured of demo site :-

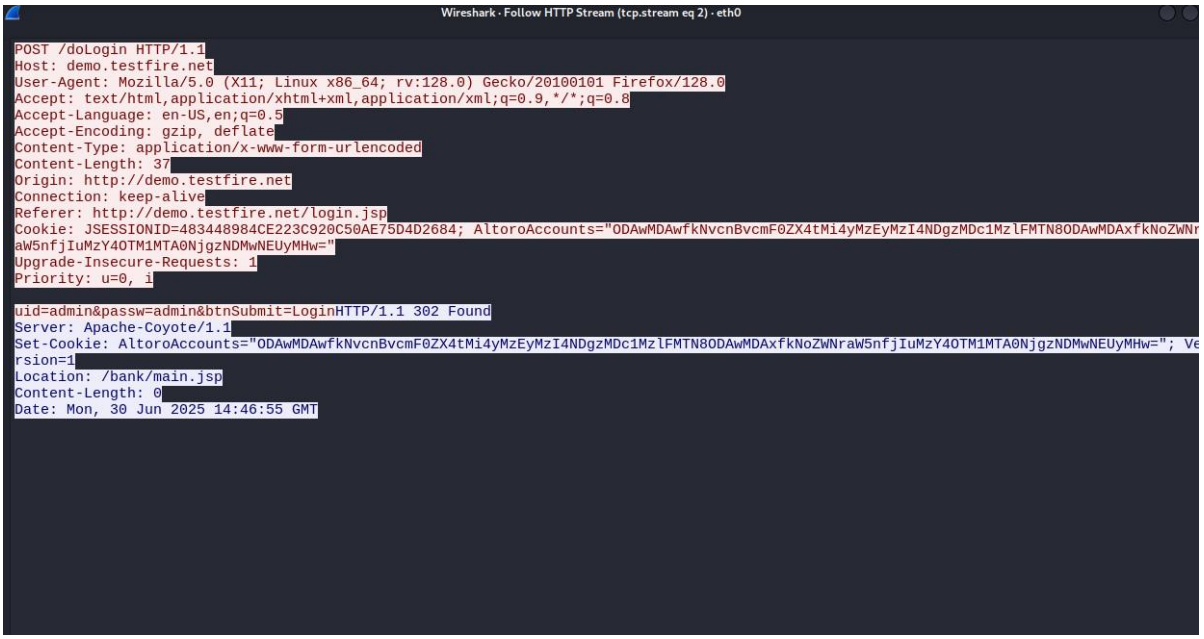


As we can see, I have marked the HTTP login packets captured in Wireshark. This shows that a login attempt was made on our demo site. Since HTTP does not provide any encryption, as mentioned earlier, it can easily expose our confidential data to anyone monitoring the network.

Lets analyze the "/doLogin packet" :-



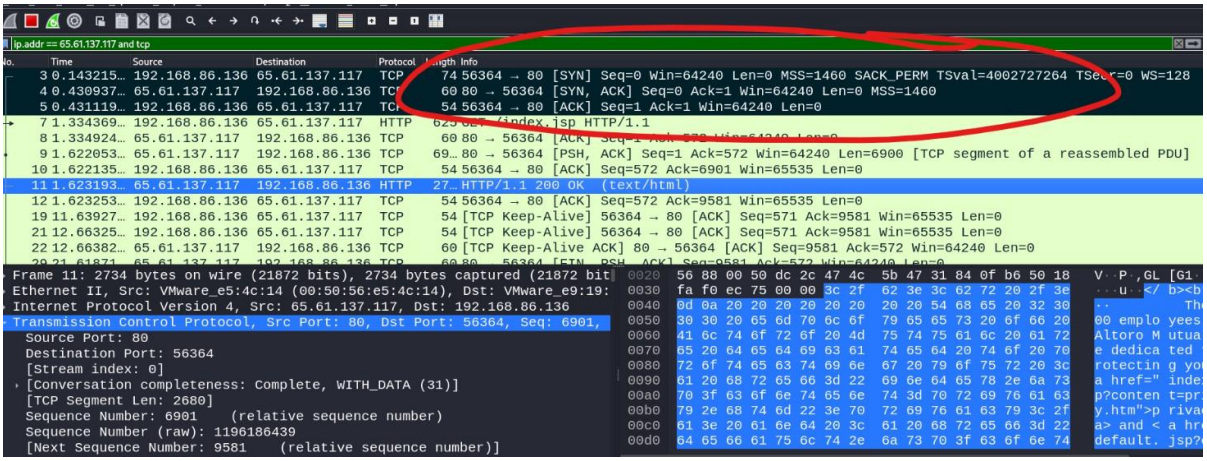
As mentioned earlier the request package exposes our confidential info.



TCP (Transmission Control Protocol) :-

It is used to send data across a network in a reliable way. TCP breaks the data into packets, makes sure all packets arrive, and puts them back together in the correct order. If any packets are lost or damaged, TCP will resend them.

TCP 3-way handshake request package captured :-



The TCP 3-way handshake is the process used to start a reliable connection between two devices on a network.

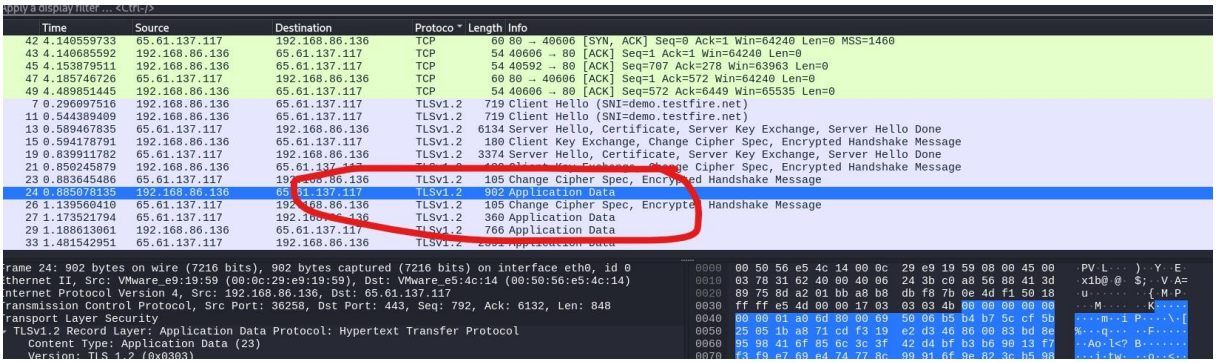
1. First, the sender sends a message called SYN to request a connection
2. The receiver replies with SYN-ACK to accept the request
3. The sender then sends an ACK to confirm the connection is ready

This three-step process makes sure both sides are ready to send and receive data safely.

HTTPS(Hyper Text Transfer Protocol Secure) :-

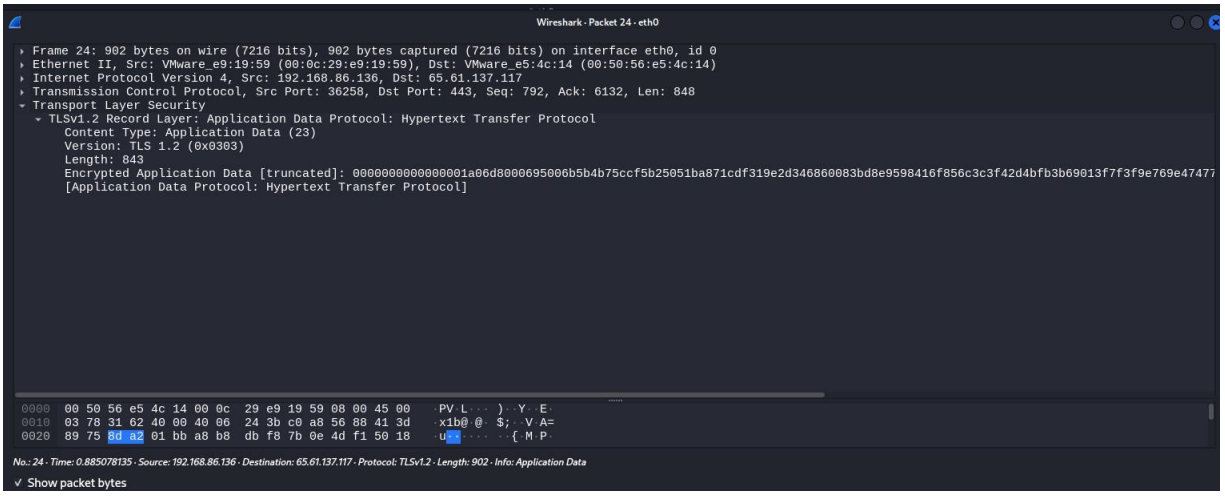
HTTPS stands for Hypertext Transfer Protocol Secure. It is the secure version of HTTP and uses encryption to keep data safe while it travels between your browser and a website. This helps protect things like passwords and personal information from being seen or stolen by attackers.

HTTPS Login request packet captured of demo site :-

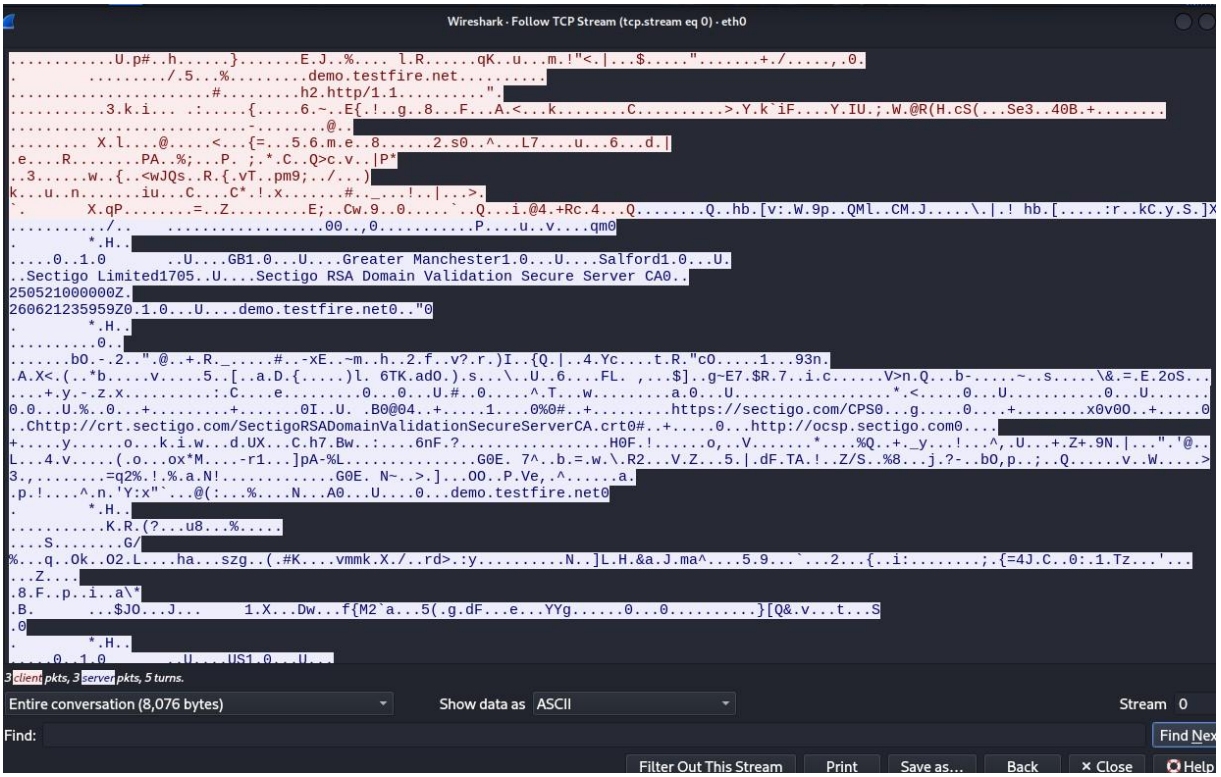


We can see that the HTTPS request made is being encrypted by adding a TLS (Transport layer security), which encodes out data from unauthorize monitoring as seen below.

Lets analyze this packet :-



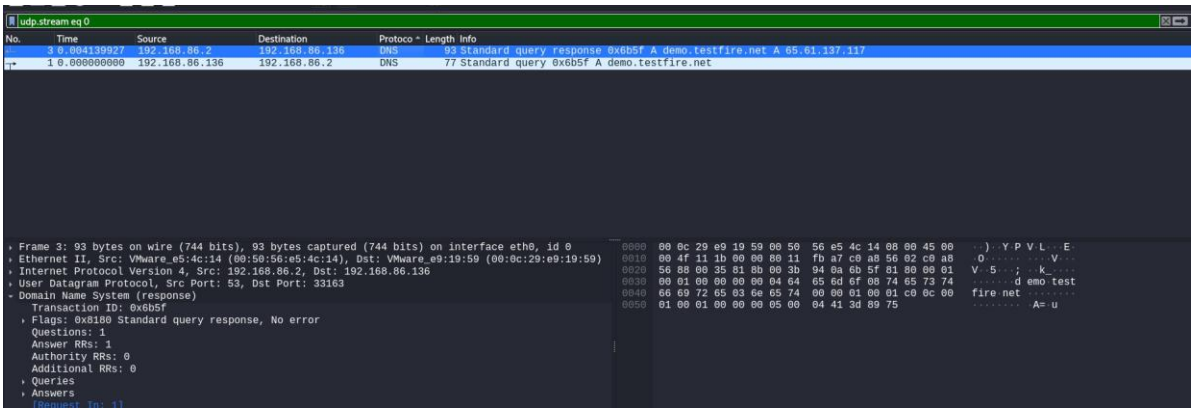
As mentioned earlier, the request packet converts our raw data into encrypted format.



DNS(Domain name system)) :-

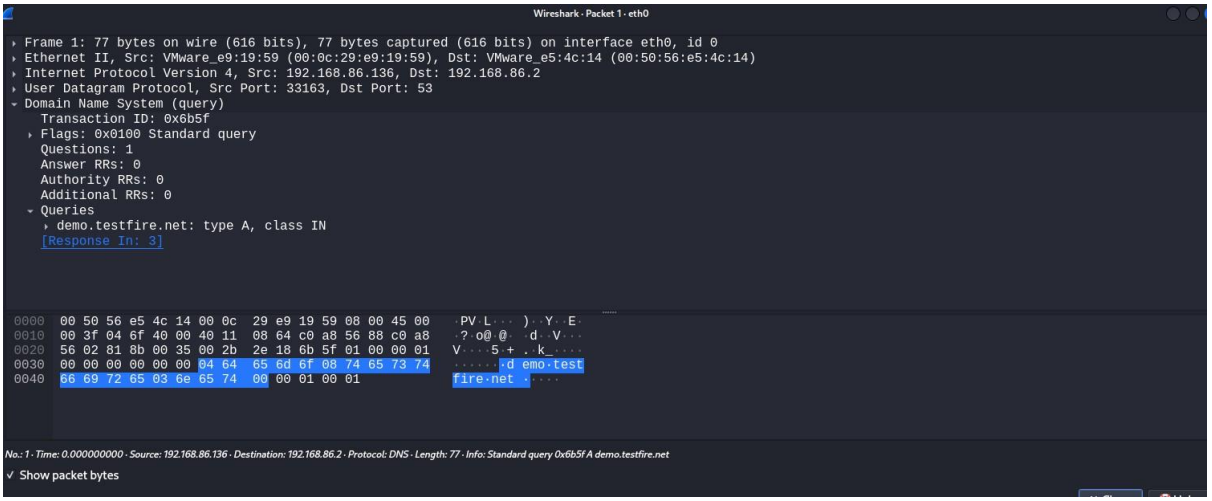
DNS stands for Domain Name System. It works like the phonebook of the internet. When we type a website name like google.com, DNS finds the correct IP address for that site so our browser can connect to it. Without DNS, we would have to remember the number based IP addresses of every website.

DNS request packet captured :-



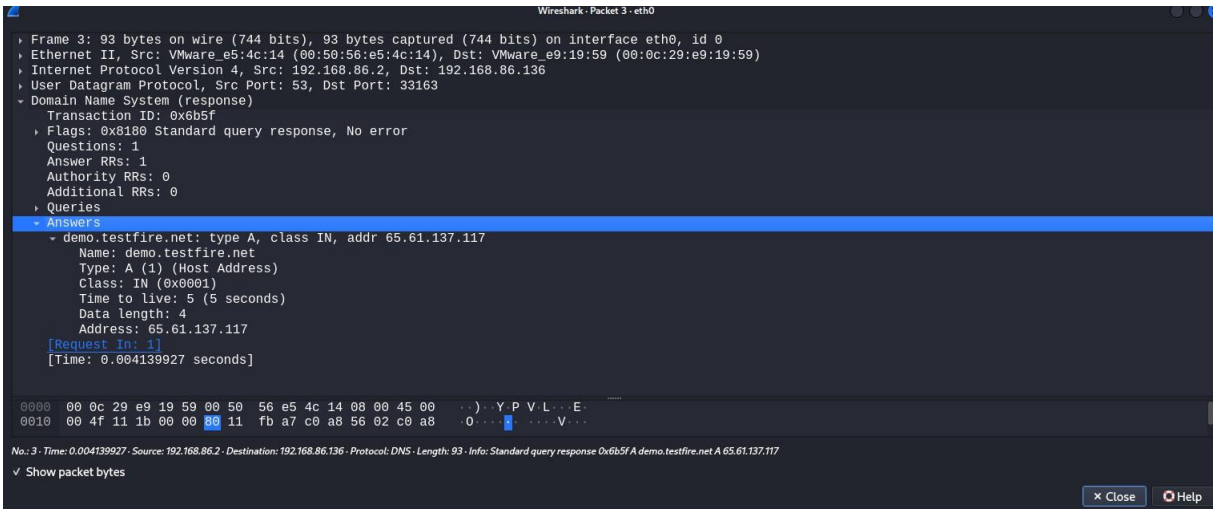
As mentioned earlier, the DNS protocol helps us find the IP address for the website or service we are trying to reach. In the figure above, we can see two DNS packets. One is making a request to find the IP address, and the other is responding with the answer to that request.

Lets analyze 1st packet :-



The analysis of the first packet shows that the request is asking for the domain name "demo.testfire.net", which is similar to sending a query to find its IP address.

lets analyze 2nd packet :-



In response to the first packet, this packet replies with the IP address "65.61.137.117", similar to answering a question that was asked by the 1st packet.

5. Packet Analysis

❖ **HTTP Packet Analysis**

Packet No: 106, 108, 135

Source IP: 192.168.82.136

Destination IP: 65.61.137.117

Info: /login.jsp, /doLogin.jsp

Purpose: These packets show requests made to the login page and form submission, indicating a user is trying to log into the website using the HTTP protocol.

❖ **TCP Handshake Analysis**

Packets: 3 (SYN), 4 (SYN-ACK), 5 (ACK)

Source Port: 40952

Destination Port: 80

Purpose: These three packets show the standard TCP handshake used to establish a stable and reliable connection between the client and the web server.

❖ **HTTPS Packet Analysis**

Packet No: 106, 108, 135

Source IP: 192.168.82.136

Destination IP: 65.61.137.117

Info: /login.jsp, /doLogin.jsp

Purpose: These packets represent secure web requests. If HTTPS was in use, the data would be encrypted, protecting sensitive login information during transmission.

❖ **DNS Packet Analysis**

Packet No: 1, 3

Query: demo.testfire.net

Response: 65.61.137.117

Purpose: These packets show the DNS resolution process where the domain name is translated into an IP address before the connection is made.

6. Observation

- The site uses HTTP instead of HTTPS, which means data like usernames and passwords can be seen in plain text over the network.
- Multiple HTTP requests were made to pages such as /login.jsp and /index.jsp, showing user activity on the site.
- DNS queries were made before the HTTP connections, which is normal as the system first resolves the domain name to an IP address.
- A complete TCP 3-way handshake was observed before any data was exchanged, confirming that a proper connection was established.

7. Conclusion

By using Wireshark, we were able to watch how our system talked to the website demo.testfire.net step by step. We saw the process start with a DNS request to find the website's IP address, then a proper connection setup using the TCP handshake, followed by the actual page visits using HTTP.

The DNS packets showed that the domain demo.testfire.net was linked to the IP address 65.61.137.117. Then, a full connection was made using the usual 3-step handshake method. After that, multiple HTTP requests were sent to pages like /login.jsp and /doLogin.jsp, showing that a login attempt took place. Since the site used HTTP instead of HTTPS, all the information sent was visible in plain text, including sensitive details like login data.

Overall, this task helped us understand how data moves through a network and how tools like Wireshark can help us see what's really happening behind the scenes. It also made it clear why using secure websites (with HTTPS) is important to keep our data safe.