==SSL==

"When we visit a secure website like Amazon (which uses HTTPS), our browser and the website create a secure connection using the SSL/TLS protocol.

In Wireshark, we can capture this process — it's called the **SSL/TLS handshake**.
You'll see packets like **'Client Hello'** and **'Server Hello'**.

- The **Client Hello** message is sent by your computer to say, 'Hello, I want to connect securely, and here are the encryption methods I support.'

- The **Server Hello** comes from the website (server) saying, 'Hello, I accept your request and will use this encryption method."

After that, they exchange keys to create a **secure, encrypted communication channel**, so no one else can read the data. This is how websites keep our passwords and banking information safe."

for example if you get command like (**tcp.stream eq 2)**

==**tcp.stream eq 2**==

This is a **Wireshark display filter**.
It means:

"Show only the packets that belong to **TCP connection number 2**."

Each TCP connection (or "stream") between your computer and a server gets a number (0, 1, 2, etc.).
So when you type tcp.stream eq 2, Wireshark only shows packets from that conversation — between your system (client) and the website (server, e.g. Amazon).

 It helps you **focus on one secure session** instead of seeing all network traffic.

While Running On My Machine I found following details which may will get change as per machine varies
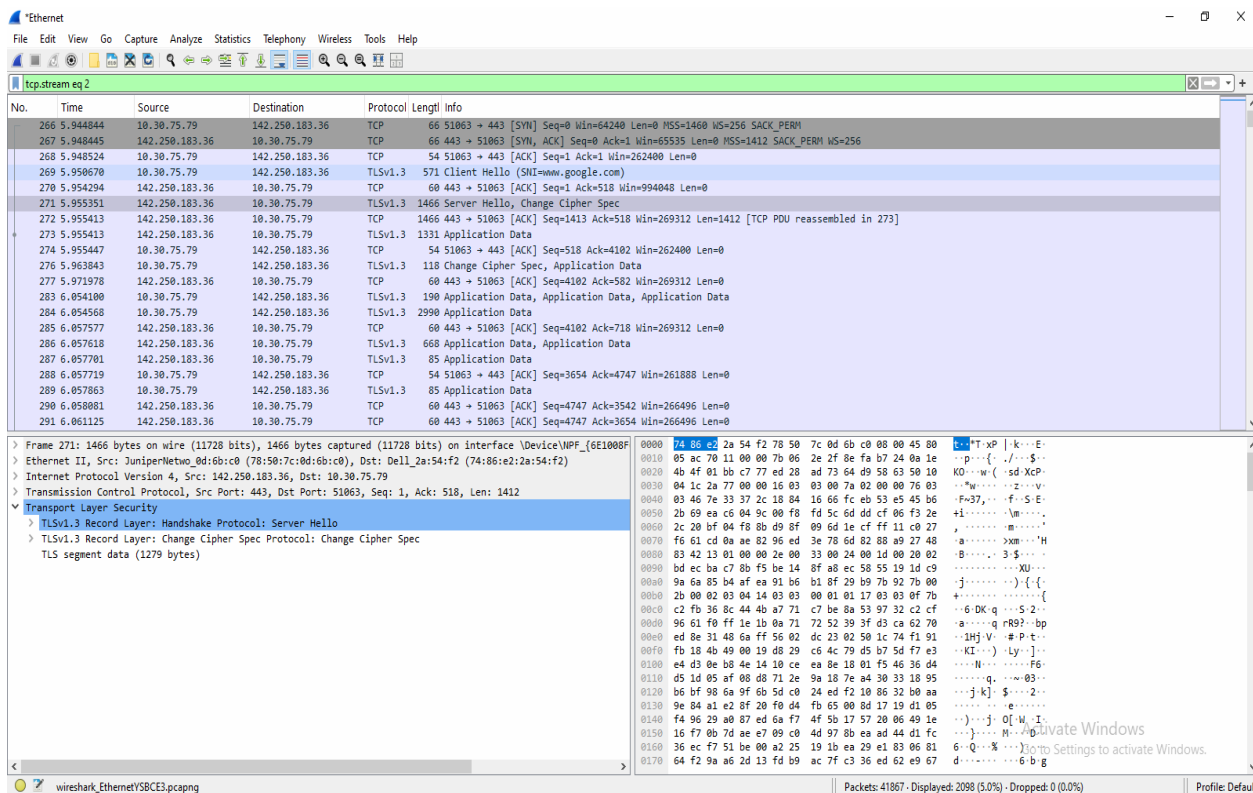
the fields you shown in Wireshark window you must know

1. Frame : This line tells us about the **whole packet** Wireshark captured.

   Example:
   Frame 271: 1466 bytes on wire
   means this packet is 1466 bytes in total size.

2. Ethernet II: This is the **Data Link Layer (Layer 2)** of the OSI model.
   It shows information like the **MAC addresses** of sender and receiver devices.
   Example:
   Src: → the source device (your computer)
   Dst: → the destination (router or server)
   So, **Ethernet II = physical network address info**.

3. Internet Protocol Version 4 (IPv4)

This is the **Network Layer (Layer 3)**.It shows the **IP addresses** of the source and destination.

Example:

    a. `Src: 142.250.183.36` → the server (like Amazon or Google)
    b. `Dst: 10.30.75.79` → your computer's IP

So, **IPv4 = shows who is talking to whom over the internet**.

4. Transmission Control Protocol (TCP) :
    This is the **Transport Layer (Layer 4)**.It handles reliable delivery of data.
    Example:
- Src Port: 443 → server port for HTTPS
- Dst Port: 51063 → your computer's port

TCP = ensures safe and ordered data delivery.

5. Transport Layer Security (TLS)

TLSv1.3 Record Layer: Handshake Protocol: Server Hello

- This is part of the **SSL/TLS handshake**.

- It means the **server is replying** to the client's "Hello".

- The server tells:

    o "Yes, I accept secure connection."

    o "Here's the encryption method I'll use."

  So, **Server Hello = Server's response to create a secure connection.**

6. **TLSv1.3 Record Layer: Change Cipher Spec Protocol**
    This means both sides are **switching from plain text to encrypted communication**.
    After this message, all data becomes **encrypted** (cannot be read in Wireshark).
    So, **Change Cipher Spec = Start of encryption**
    In short you must keep in your mind
    Frame → Whole captured packet

    Ethernet II → Device (MAC) addresses

IPv4 → IP addresses of sender and receiver

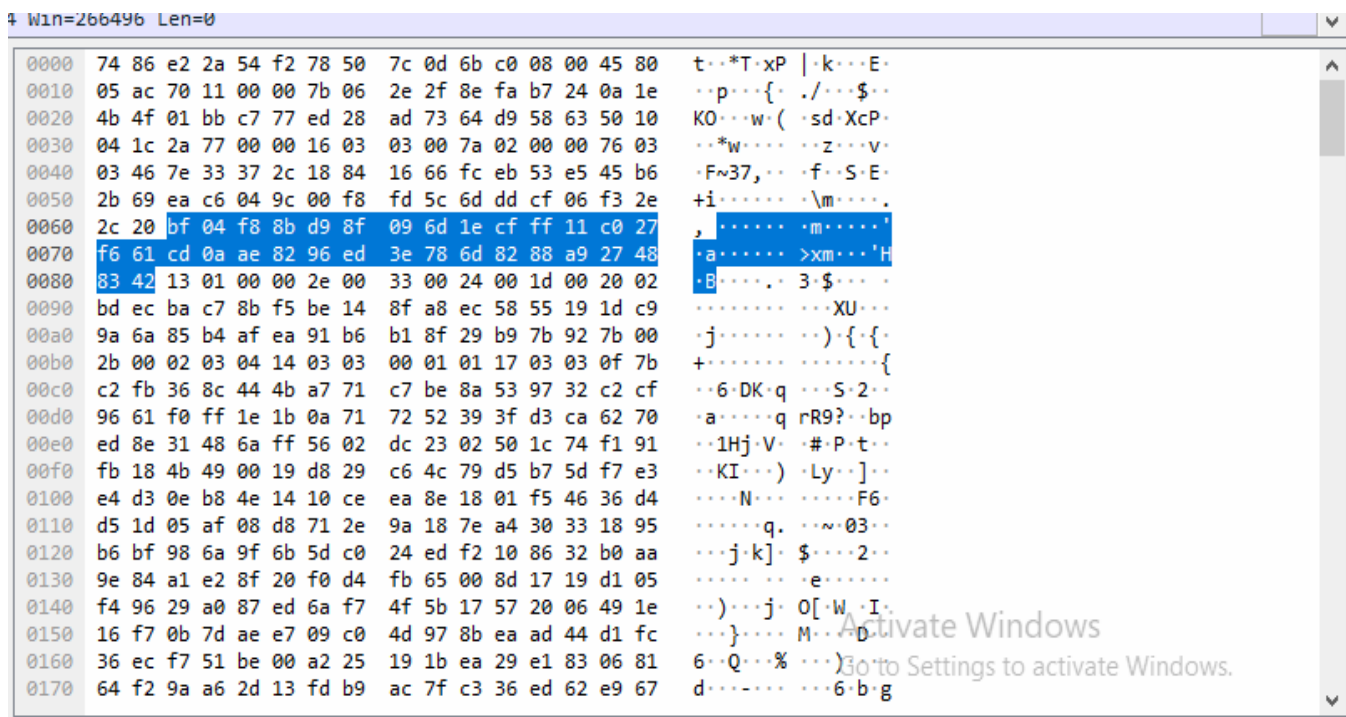TCP → Transport control (port numbers, sequence)

TLS → Security layer (handshake and encryption)

— just the roles are reversed.

TLSv1.3 Record Layer: Handshake Protocol: Client Hello

This packet is sent **from your computer (the client)** to the **server (like Amazon or Google)** when you first open a secure website.

==How to Read This==

Part	Description	Example from your image

Left side	Memory offset (line number in hex).	0000, 0010, 0020 ...
Middle part	Actual data in **hexadecimal**. Each pair (like 74 86) = 1 byte.	74 86 e2 e2 54 f2 78 50 ...
Right side	ASCII (text) view of that data, if printable.	You see things like t..T.xP..

This is the raw packet data. Wireshark shows it in hexadecimal form.

Each pair of numbers represents one byte of data.

After encryption, this data looks like random letters and numbers — this is how SSL keeps our information safe."