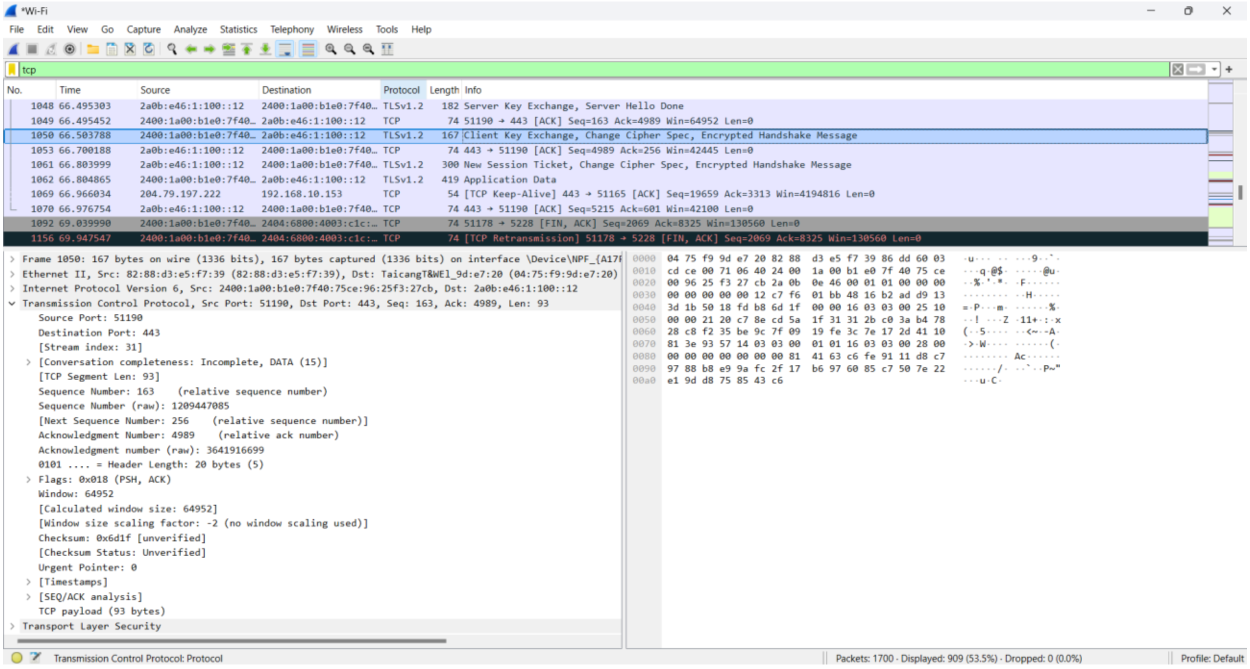
5. Locate a TCP package in Wireshark and explain why the field have the value that they have.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp

```
No.   Time      Source                 Destination            Protocol Length Info
 1048 66.495303 2a0b:e46:1:100::12     2400:1a00:b1e0:7f40…   TLSv1.2  182 Server Key Exchange, Server Hello Done
 1049 66.495452 2400:1a00:b1e0:7f40…   2a0b:e46:1:100::12     TCP       74 51190 → 443 [ACK] Seq=163 Ack=4989 Win=64952 Len=0
 1050 66.503788 2400:1a00:b1e0:7f40…   2a0b:e46:1:100::12     TLSv1.2  167 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
 1053 66.700188 2a0b:e46:1:100::12     2400:1a00:b1e0:7f40…   TCP       74 443 → 51190 [ACK] Seq=4989 Ack=256 Win=42445 Len=0
 1061 66.803999 2a0b:e46:1:100::12     2400:1a00:b1e0:7f40…   TLSv1.2  300 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
 1062 66.804865 2400:1a00:b1e0:7f40…   2a0b:e46:1:100::12     TLSv1.2  419 Application Data
 1069 66.966034 204.79.197.222         192.168.10.153         TCP       54 [TCP Keep-Alive] 443 → 51165 [ACK] Seq=19659 Ack=3313 Win=4194816 Len=0
 1070 66.976754 2a0b:e46:1:100::12     2400:1a00:b1e0:7f40…   TCP       74 443 → 51190 [ACK] Seq=5215 Ack=601 Win=42100 Len=0
 1092 69.039990 2400:1a00:b1e0:7f40…   2404:6800:4003:c1c:…   TCP       74 51178 → 5228 [FIN, ACK] Seq=2069 Ack=8325 Win=130560 Len=0
 1156 69.947547 2400:1a00:b1e0:7f40…   2404:6800:4003:c1c:…   TCP       74 [TCP Retransmission] 51178 → 5228 [FIN, ACK] Seq=2069 Ack=8325 Win=130560 Len=0
```

```
> Frame 1050: 167 bytes on wire (1336 bits), 167 bytes captured (1336 bits) on interface \Device\NPF_{A17F
> Ethernet II, Src: 82:88:d3:e5:f7:39 (82:88:d3:e5:f7:39), Dst: TaicangT&WEl_9d:e7:20 (04:75:f9:9d:e7:20)
> Internet Protocol Version 6, Src: 2400:1a00:b1e0:7f40:75ce:96:25f3:27cb, Dst: 2a0b:e46:1:100::12
v Transmission Control Protocol, Src Port: 51190, Dst Port: 443, Seq: 163, Ack: 4989, Len: 93
    Source Port: 51190
    Destination Port: 443
    [Stream index: 31]
  > [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 93]
    Sequence Number: 163    (relative sequence number)
    Sequence Number (raw): 1209447085
    [Next Sequence Number: 256    (relative sequence number)]
    Acknowledgment Number: 4989    (relative ack number)
    Acknowledgment number (raw): 3641916699
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
    Window: 64952
    [Calculated window size: 64952]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0x6d1f [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
    TCP payload (93 bytes)
> Transport Layer Security
```

```
0000  04 75 f9 9d e7 20 82 88  d3 e5 f7 39 86 dd 60 03   ·u··· ·· ···9··`·
0010  cd ce 00 71 06 40 24 00  1a 00 b1 e0 7f 40 75 ce   ···q·@$· ·····@u·
0020  00 96 25 f3 27 cb 2a 0b  0e 46 00 01 01 00 00 00   ··%·'·*· ·F······
0030  00 00 00 00 00 12 c7 f6  01 bb 48 16 b2 ad d9 13   ········ ··H·····
0040  3d 1b 50 18 fd b8 6d 1f  00 00 16 03 03 00 25 10   =·P···m· ······%·
0050  00 00 21 20 c7 8e cd 5a  1f 31 31 2b c0 3a b4 78   ··! ···Z ·11+·:·x
0060  28 c8 f2 35 be 9c 7f 09  19 fe 3c 7e 17 2d 41 10   (··5···· ··<~·-A·
0070  81 3e 93 57 14 03 03 00  01 01 16 03 03 00 28 00   ·>·W···· ······(·
0080  00 00 00 00 00 00 00 81  41 63 c6 fe 91 11 d8 c7   ········ Ac······
0090  97 88 b8 e9 9a fc 2f 17  b6 97 60 85 c7 50 7e 22   ······/· ··`··P~"
00a0  e1 9d d8 75 85 43 c6                               ···u·C·
```

Transmission Control Protocol: Protocol | Packets: 1700 · Displayed: 909 (53.5%) · Dropped: 0 (0.0%) | Profile: Default

- When analyzing a TCP packet in Wireshark, each field in the packet's header has a specific value that serves a particular function. Here’s a breakdown of why the fields in a TCP packet have the values they do:

TCP Header Fields and Their Values

1. **Source Port**:
   - **Value**: A number between 0 and 65535.
   - **Reason**: This identifies the sending port number. The source port is typically a dynamic (ephemeral) port number assigned by the operating system when a connection is initiated.

2. **Destination Port**:
   - **Value**: A number between 0 and 65535.
   - **Reason**: This identifies the receiving port number. The destination port is usually a well-known port number for standard services (e.g., 80 for HTTP, 443 for HTTPS).

3. **Sequence Number**:

- **Value**: A 32-bit number.
- **Reason**: This is used to keep track of the order of bytes sent from the sender to the receiver. The initial sequence number (ISN) is randomly chosen at the start of a connection and incremented for each byte of data sent.

4. **Acknowledgment Number**:
    - **Value**: A 32-bit number.
    - **Reason**: This indicates the next expected sequence number from the sender. It acknowledges receipt of the data up to this number minus one.

5. **Data Offset**:
    - **Value**: A 4-bit number.
    - **Reason**: This indicates where the data payload begins. It specifies the size of the TCP header in 32-bit words, allowing the receiver to locate the start of the actual data.

6. **Reserved**:
    - **Value**: Typically set to 0.
    - **Reason**: These bits are reserved for future use and should always be set to zero. They are not used in normal TCP operations.

7. **Flags** (Control Bits):
    - **Value**: 9 bits (each bit can be 0 or 1).
    - **Reason**: These bits control various aspects of the TCP connection. The common flags include:
        - **URG** (Urgent): Indicates urgent pointer field significant.
        - **ACK** (Acknowledgment): Acknowledgment field significant.
        - **PSH** (Push Function): Push function.
        - **RST** (Reset): Reset the connection.
        - **SYN** (Synchronize): Synchronize sequence numbers (used to initiate a connection).
        - **FIN** (Finish): No more data from sender (used to terminate a connection).

8. **Window Size**:
    - **Value**: A 16-bit number.
    - **Reason**: This indicates the size of the receive window, which is the amount of data (in bytes) that the sender is willing to accept. It's used for flow control to manage the rate of data transmission.

9. **Checksum**:
    - **Value**: A 16-bit number.
    - **Reason**: This is used for error-checking the header and data. The checksum is calculated by the sender and verified by the receiver to ensure data integrity.

10. **Urgent Pointer**:
    - **Value**: A 16-bit number.
    - **Reason**: This is used when the URG flag is set. It indicates the end of urgent data and points to the sequence number of the byte following the urgent data.

11. **Options**:
    - **Value**: Variable length.
    - **Reason**: Options provide additional functionality, such as maximum segment size (MSS), window scaling, and timestamps. They are used to negotiate various TCP parameters between the sender and receiver.