LAB 5 - The TCP Connection, The 3-Way Handshake / FIN Packets and a deeper look into TCP Headers

TCP is a connection oriented protocol. The TCP connection is established through a 3-way handshake. Once this connection is established the TCP layer of one host can communicate directly to the TCP layer of the other connected host.

In this lab we will study the TCP 3-way handshake. We will view the 3-way handshake in wireshark, and will use the opportunity to learn more about the TCP packet header.

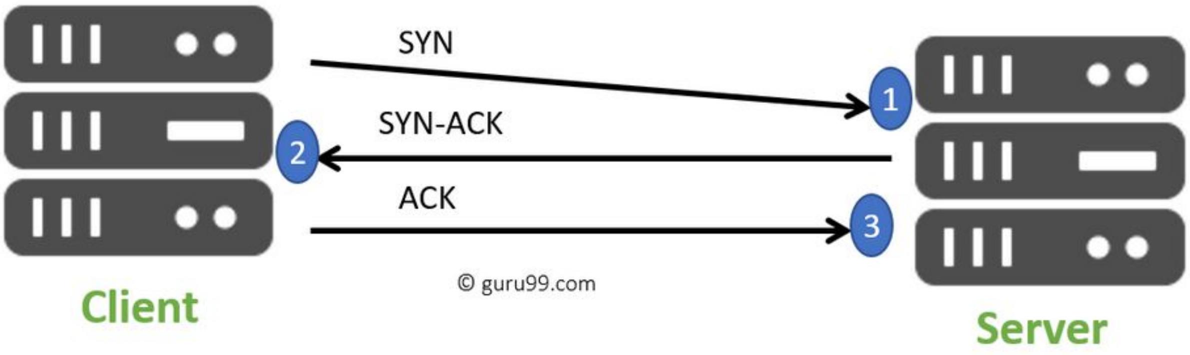
Prerequisites

Completion of Lab 4, or the installation of a Web Server on your Ubuntu Server.

The TCP 3-way Handshake

TCP uses a 3-way handshake to establish its connection:

1. The client initiates the connection by sending a TCP SYN packet to the server.
2. The server responds to the client with a TCP SYN-ACK packet.
3. Finally the client responds to the server with a TCP ACK packet. The 3-way handshake is complete and the connection is established.



© guru99.com

After the connection is established the client and the server can communicate over the network as though they were directly connected to each other.

A common question occurs here. Why does TCP use a 3-way handshake instead of a 2-way handshake? In order to establish a connection 4 facts need to be confirmed:

1. The server needs to know it can receive data from the client.
2. The server needs to know that the client can receive data from itself.
3. The client needs to know that it can receive data from the server.
4. The client needs to know that the server can receive data from itself.

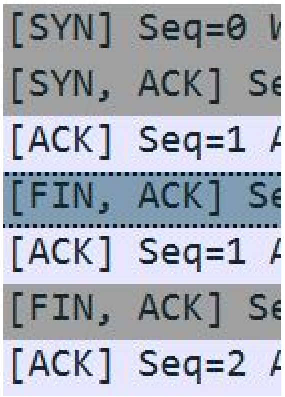
When the server receives the initial SYN packet, fact 1 is confirmed. When the client receives the SYN-ACK packet facts 3 & 4 are confirmed. When the server receives the final ACK packet fact 2 is confirmed. All four of these facts are confirmed by the 3-way handshake and the connection is established.

Using Wireshark to View the TCP 3-way Handshake and the TCP FIN Packet

We are going to use Wireshark to view the TCP 3-way handshake and the FIN packet that ends the TCP connection between the web browser and the web server on the Ubuntu Server. We want to capture the entire TCP connection from the 3-way Handshake all the way until the FIN packet.

1. Make sure your Ubuntu Server is running on Virtual Box.
2. Start Wireshark Capture on the appropriate interface.
3. Then visit your web server. If you completed Lab 4 your web server will be located at http://127.0.0.1:4995
4. Now, close the browser tab you had this page open in.
5. Stop the Wireshark Capture

Now filter by tcp.port == 4995 and you will see the entire TCP connection starting with the SYN packet and ending with a FIN packet:



```
[SYN] Seq=0 W
[SYN, ACK] Se
[ACK] Seq=1 A
[FIN, ACK] Se
[ACK] Seq=1 A
[FIN, ACK] Se
[ACK] Seq=2 A
```

[SYN], [SYN, ACK] and [ACK] are the 3-way Handshake

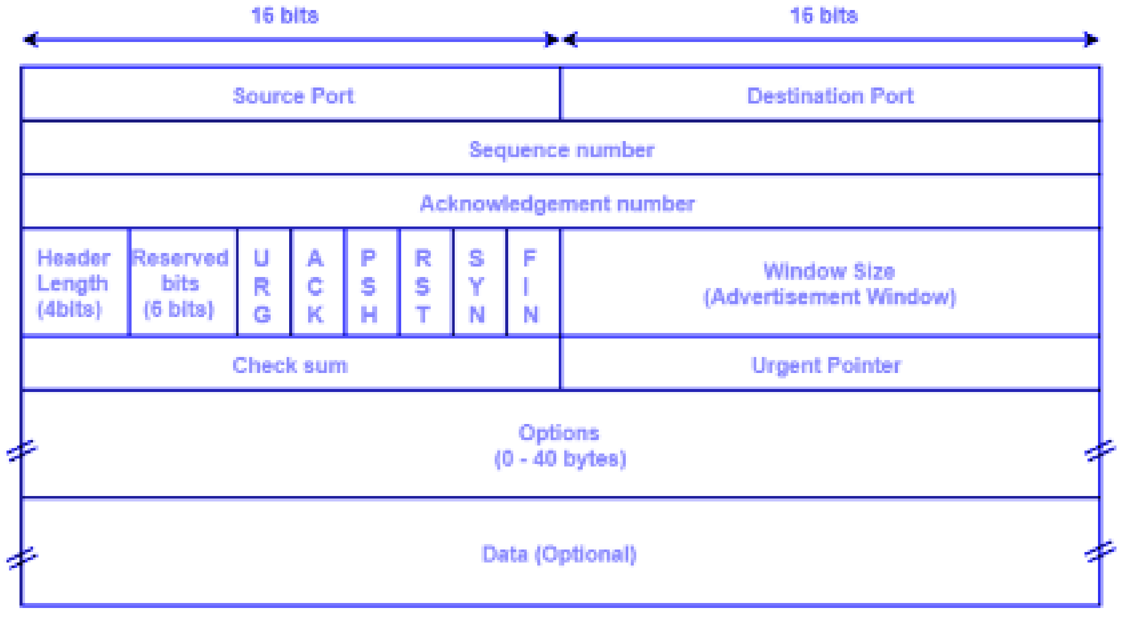
First[FIN, ACK] is where the client closed the connection

Second[FIN, ACK] is where the server closes its connection.

Constructing a TCP Header

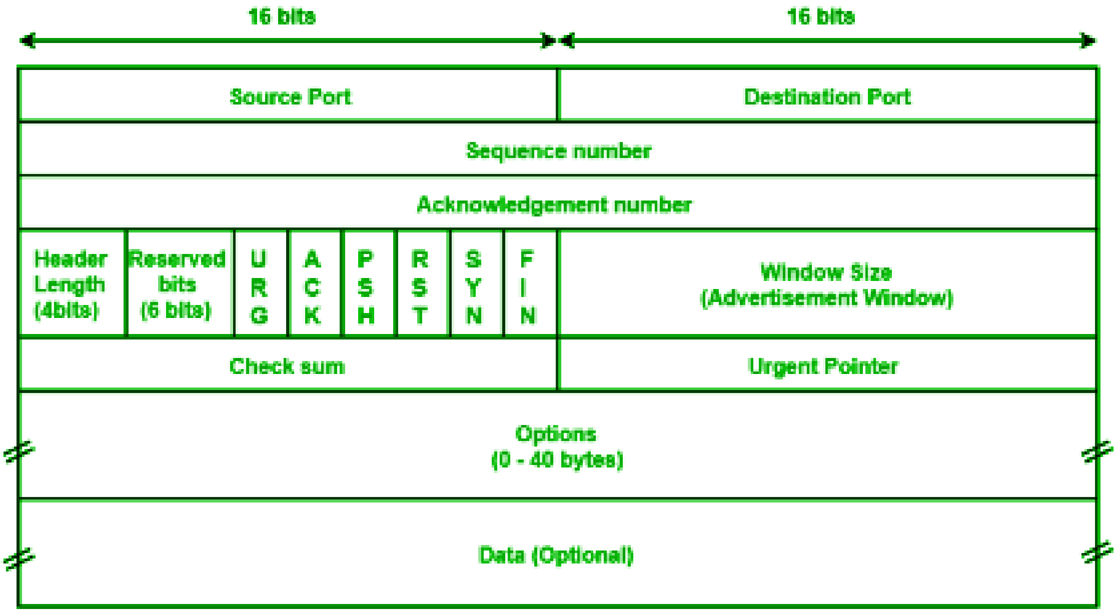
View the first TCP [FIN, ACK] packet in wireshark and build the following TCP Header. Find the Matching fields in Wireshark. If it gives you the choice between a “relative or raw” field, choose raw. Fill out all fields from “source port” to “urgent pointer”

Fill out the first (BLUE) header using **Decimal Numbers**.



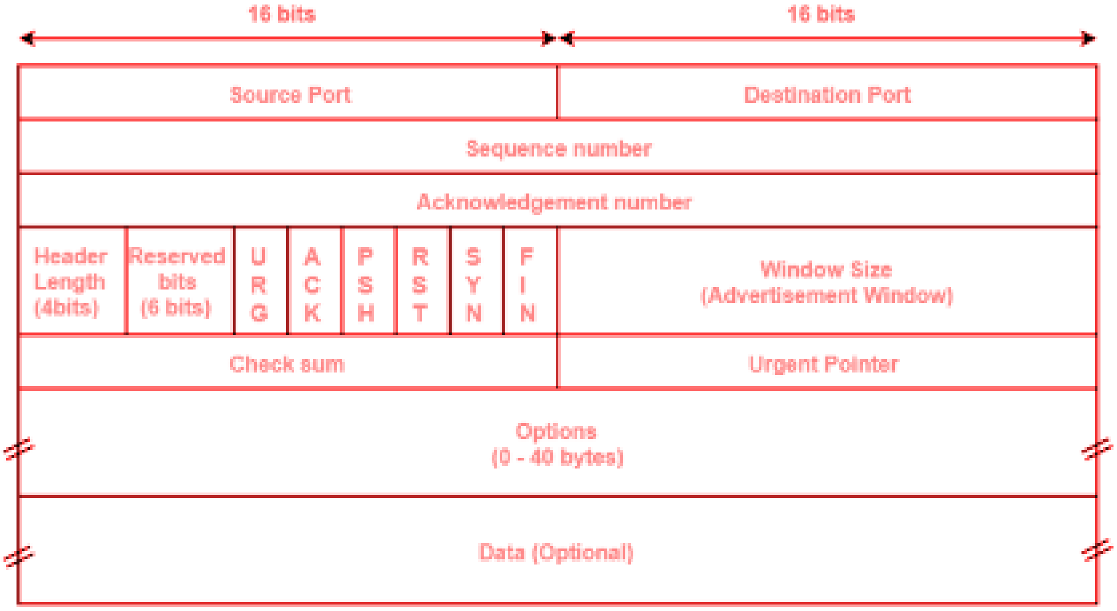
TCP Header

Fill out the second (Green Header) by converting all the Decimal Values in the first Header to **Binary Numbers** in this one. Make sure to pay attention to the number of bits in each field.



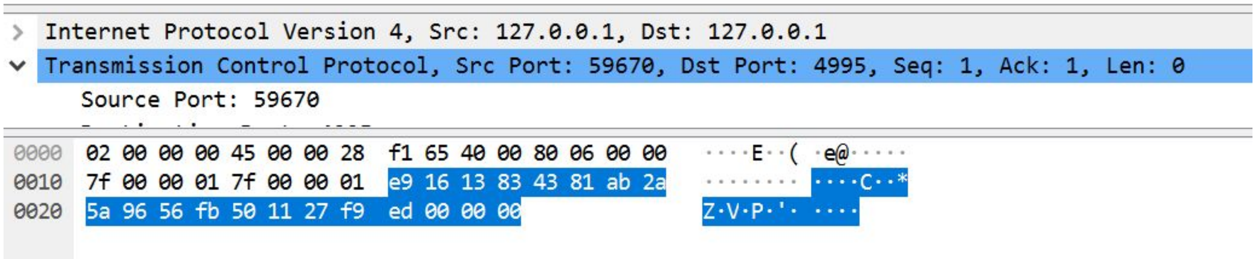
TCP Header

Fill out the 3rd (RED) Header using **Hexadecimal Numbers**. Group the Binary Numbers into groups of 4, and translate to HEX.



TCP Header

Notice that when you highlight the TCP Frame in the packet, the TCP Header data is also highlighted.



```
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
v Transmission Control Protocol, Src Port: 59670, Dst Port: 4995, Seq: 1, Ack: 1, Len: 0
     Source Port: 59670
0000  02 00 00 00 45 00 00 28  f1 65 40 00 80 06 00 00   ····E··( ·e@·····
0010  7f 00 00 01 7f 00 00 01  e9 16 13 83 43 81 ab 2a   ········ ····C··*
0020  5a 96 56 fb 50 11 27 f9  ed 00 00 00              Z·V·P·'· ····
```

You see the raw TCP Packet data in Hexadecimal form on the left. On the right you see the direct ASCII translation of this data. The highlighted data on the left should match the HEX data you translated in the 3rd (RED) header.

Deliverables:

- A pdf document with a screen shot of:
    1. Wireshark Screenshot of SYN, SYN-ACK, and ACK packets.
    2. Wireshark Screenshot of FIN packet. (Can be same screenshot)
    3. TCP Header Filled out for [FIN, ACK] packet.
        - i. Decimal (BLUE)
        - ii. Binary (GREEN)
        - iii. HEX (RED)

Instructions for Deliverables

Create a document with screenshots pasted in of the included deliverables. Make sure your name, date and lab number is included at the top of the document. Save your document as a **.pdf** and upload to **LAB 5** on Canvas.

BONUS Research Exercise (WORTH ¼ Extra Lab Credit)

TCP uses the 3-way handshake algorithm we covered in this lab to establish a connection between two hosts. There are definitely other algorithms that can also establish a successful connection between two hosts. Research one of these alternate connection algorithms and write a 250 word summary explaining how it works. Provide at least two online references.

Instructions for Bonus

Write the summary and submit your online references using the **text box** under **LAB 5 - BONUS** on Canvas.