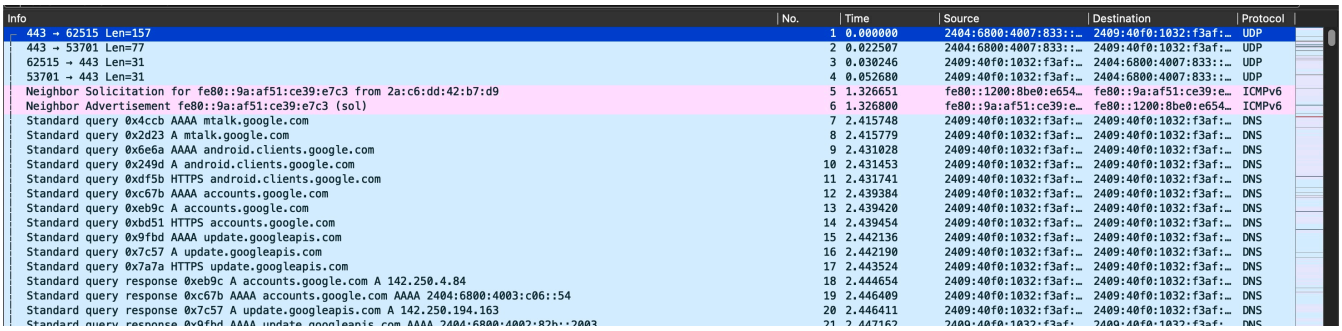
Capturing and analyzing IP, TCP protocols using Wireshark

Report by : Bhargav Surisetti
Date : July-02,2025

Activity:

Part A: Filtering and analyzing PCAP file of website access to **"cisco.com"**

Step 1: Filtering cisco.com website packets

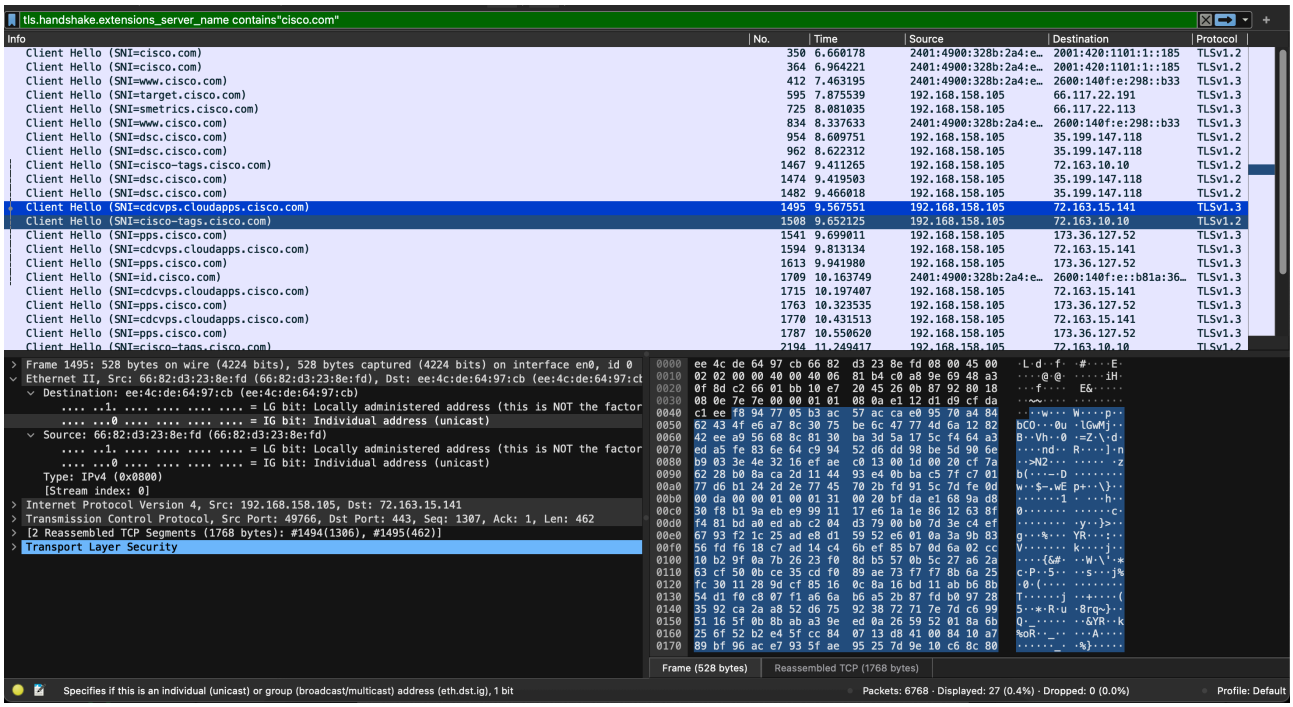


Info	No.	Time	Source	Destination	Protocol
443 → 62515 Len=157	1	0.000000	2404:6800:4007:833::…	2409:40f0:1032:f3af:…	UDP
443 → 53701 Len=77	2	0.022507	2404:6800:4007:833::…	2409:40f0:1032:f3af:…	UDP
62515 → 443 Len=31	3	0.030246	2409:40f0:1032:f3af:…	2404:6800:4007:833::…	UDP
53701 → 443 Len=31	4	0.052680	2409:40f0:1032:f3af:…	2404:6800:4007:833::…	UDP
Neighbor Solicitation for fe80::9a:af51:ce39:e7c3 from 2a:c6:dd:42:b7:d9	5	1.326651	fe80::1200:8be0:e654…	fe80::9a:af51:ce39:e…	ICMPv6
Neighbor Advertisement fe80::9a:af51:ce39:e7c3 (sol)	6	1.326800	fe80::9a:af51:ce39:e…	fe80::1200:8be0:e654…	ICMPv6
Standard query 0x4ccb AAAA mtalk.google.com	7	2.415748	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x2d23 A mtalk.google.com	8	2.415779	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x6e6a AAAA android.clients.google.com	9	2.431028	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x249d A android.clients.google.com	10	2.431453	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0xdf5b HTTPS android.clients.google.com	11	2.431741	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0xc67b AAAA accounts.google.com	12	2.439384	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0xeb9c A accounts.google.com	13	2.439420	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0xbd51 HTTPS accounts.google.com	14	2.439454	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x9fbd AAAA update.googleapis.com	15	2.442136	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x7c57 A update.googleapis.com	16	2.442190	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query 0x7a7a HTTPS update.googleapis.com	17	2.443524	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query response 0xeb9c A accounts.google.com A 142.250.4.84	18	2.444654	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query response 0xc67b AAAA accounts.google.com AAAA 2404:6800:4003:c06::54	19	2.446409	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS
Standard query response 0x7c57 A update.googleapis.com A 142.250.194.163	20	2.446411	2409:40f0:1032:f3af:…	2409:40f0:1032:f3af:…	DNS

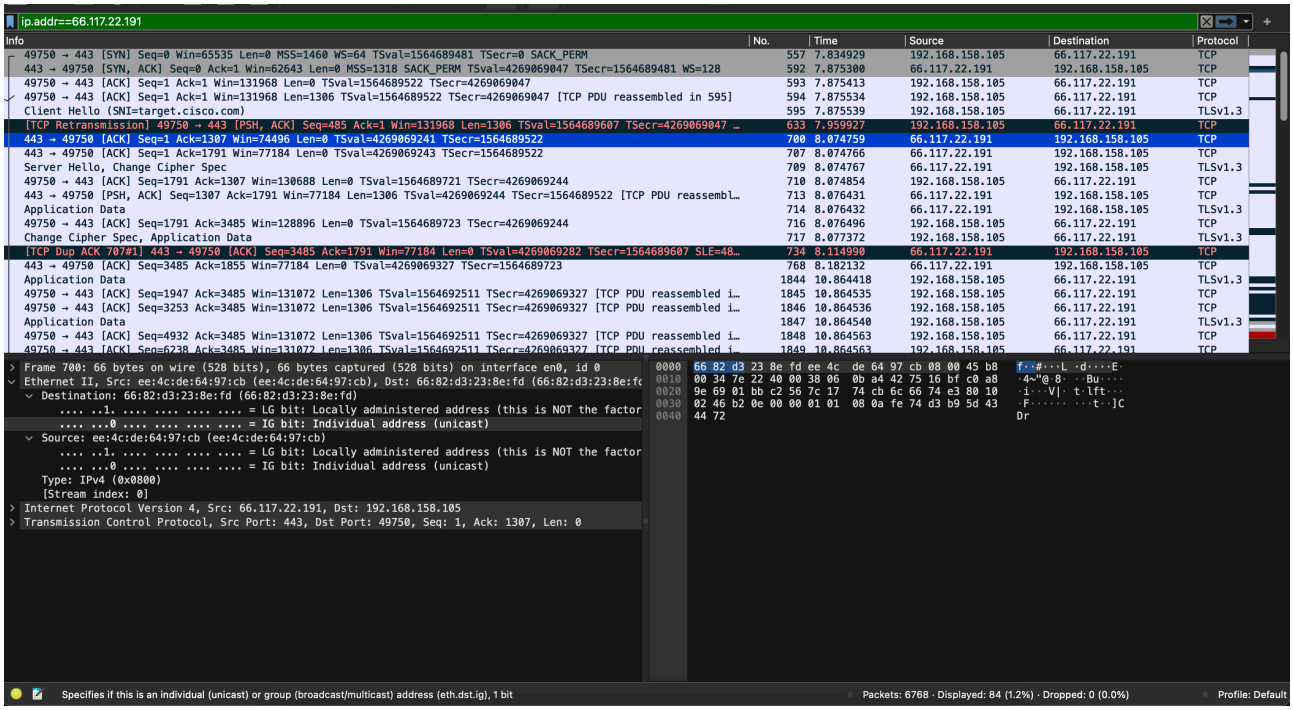
Total Packets	6768
displayed packets	6768
Time stamp of the first packet	0 sec
Time stamp of the last packet	22.50 sec
Average packet rate(packets/sec)	300.8

- To filter cisco.com website traffic apply the filter: **tls.handshake.extensions_server_name contains "cisco.com"**

Packet Number	**595**
IP Address of your PC	**192.168.158.105**
IP Address of cisco.com	**66.117.22.191**

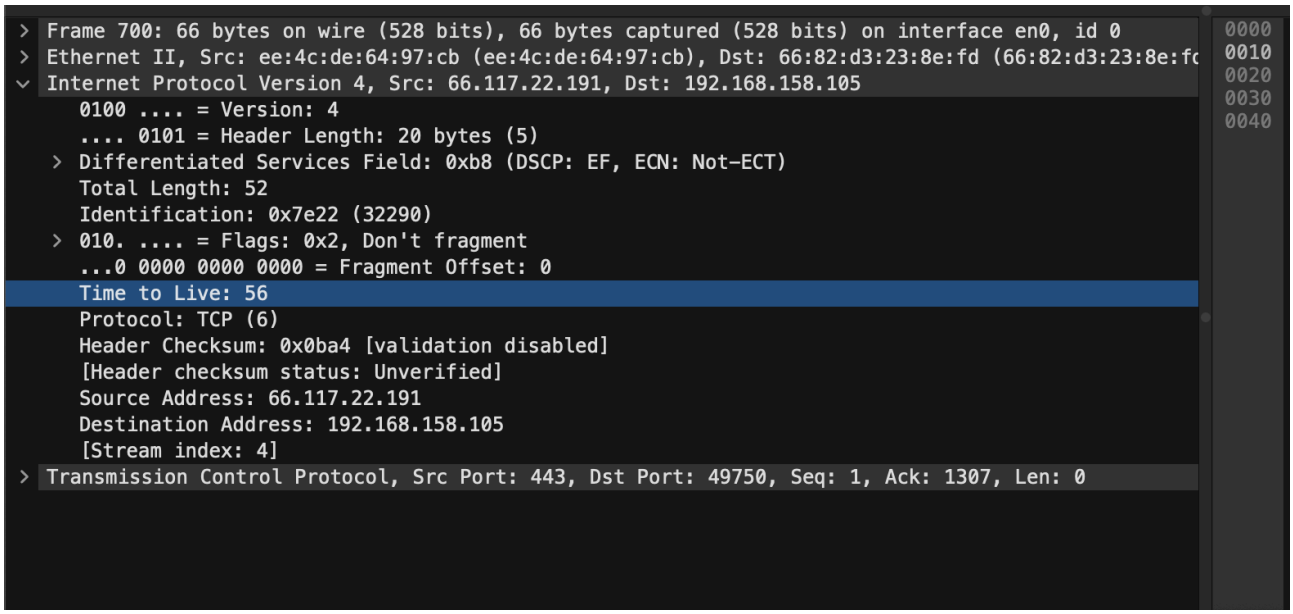


- Filter all the traffic of cisco.com ip with command: **ip.addr== 66.117.22.191**



Step 2: Determining the number of networks between your PC and cisco.com by observing the TTL field value of IP header.

- Open IP header of packet with cisco.com as source and your PC as destination.
- Observe the Time to Live field value.



- Assuming the field value is X.

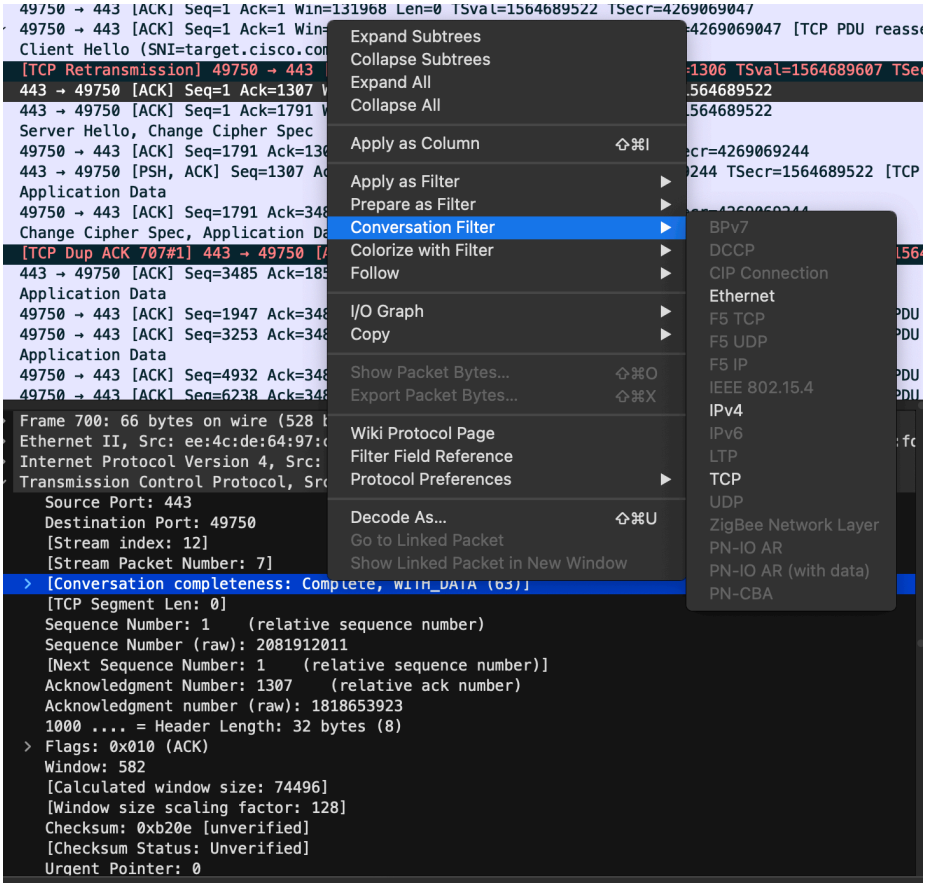
1. If X<64: subtract 64-X.
2. If 64< X<128: subtract 128-X.
3. If X>128: subtract 256-X.

Number of networks: 64-X or 128-X or 256-X.	64-56=8

- Verify the same value by pinging cisco.com in Command Prompt.

Step 3: Analyzing the TCP Header and Understanding the TCP Three-Way Handshake.

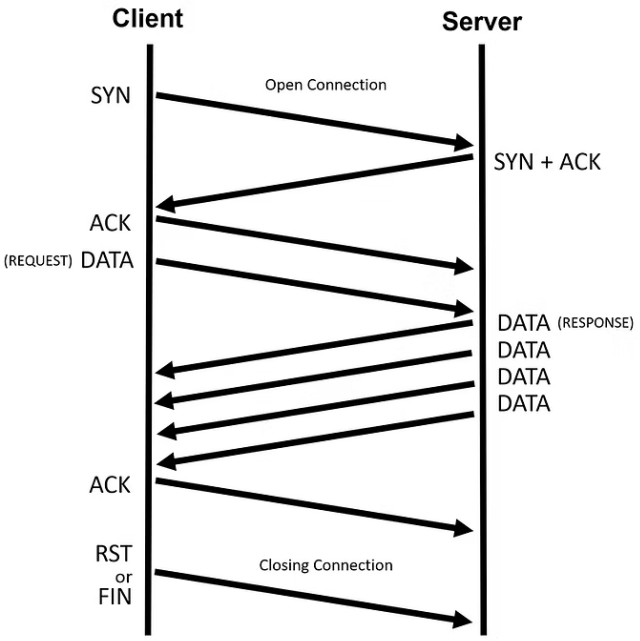
- Expand TCP header of the first packet with cisco.com IP address.
- Right click on conversation completeness field- apply as filter-selected-tcp.



- Note down the number of displayed packets in the bottom section.

Displayed packets	84
Time stamp of the first packet in the conversation	7.8 sec
Time stamp of the last packet in the conversation	19.5 sec
Average packet rate (packets/ sec)	3.36sec

- These are the total number packets transferred when accessing the website.
- Determining **TCP Conversation Completeness:** A full TCP conversation has the following scenario.



- Wireshark assigns a value to each of the following packet types:

1 : SYN
2 : SYN-ACK

4 : ACK
8 : DATA
16 : FIN
32 : RST

- Observe the numeric value corresponding to the conversation completeness field of the first TCP packet.

```
> Frame 592: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface en0, id 0
> Ethernet II, Src: ee:4c:de:64:97:cb (ee:4c:de:64:97:cb), Dst: 66:82:d3:23:8e:fd (66:82:d3:23:8e:fd)
> Internet Protocol Version 4, Src: 66.117.22.191, Dst: 192.168.158.105
v Transmission Control Protocol, Src Port: 443, Dst Port: 49750, Seq: 0, Ack: 1, Len: 0
    Source Port: 443
    Destination Port: 49750
    [Stream index: 12]
    [Stream Packet Number: 2]
  > [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 2081912010
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1818652617
    1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x012 (SYN, ACK)
    Window: 62643
    [Calculated window size: 62643]
    Checksum: 0x9767 [unverified]
    [Checksum Status: Unverified]
```

- Equate the value observed as sum of all or any of the values assigned to the packet types above.

Conversation Completeness	Equated value:	Packets in the Current conversation	Is the conversation completed or not
	63=1+2+4+8+16+32	SYN, SYN-ACK,ACK,DATA,FIN,RST	completed

- From the above value we can determine if the conversation is complete or not.
- Filter the SYN packet using the filter: **(ip.addr eq 192.168.158.105 and ip.addr eq 66.117.22.191) and (tcp.port eq 49750 and tcp.port eq 443) && tcp.flags.syn == 1 && tcp.flags.ack == 0.**

Packet no	**557**
Source IP	192.168.158.105
Destination IP	66.117.22.191
Source port number	49750
Destination Port number	443
Relative sequence number	1
Relative Acknowledgement number	0
Syn flag	1
Ack flag	0
Push flag	0
Fin flag	0
Reset flag	0
Packet Type	SYN

- Filter the SYN-ACK packet using the filter: **(ip.addr eq 192.168.158.105 and ip.addr eq 66.117.22.191) and (tcp.port eq 49750 and tcp.port eq 443) && tcp.flags.syn == 1 && tcp.flags.ack == 1.**

Packet no	**592**
Source IP	66.117.22.191
Destination IP	192.168.158.105
Source port number	443
Destination Port number	49750
Relative sequence number	1
Relative Acknowledgement number	1
Syn flag	1
Ack flag	1
Push flag	0
Fin flag	0
Reset flag	0
Packet Type	SYN-ACK

- Filter the FIN packet using the filter: **(ip.addr eq 192.168.158.105 and ip.addr eq 66.117.22.191) and (tcp.port eq 49750 and tcp.port eq 443) && tcp.flags.fin==1.**

Packet no	6359
Source IP	192.168.158.105
Destination IP	66.117.22.191
Source port number	49750
Destination Port number	443
Relative sequence number	24385
Relative Acknowledgement number	7434
Syn flag	0
Ack flag	1
Push flag	0
Fin flag	1
Reset flag	0
Packet Type	FIN,ACK

Conclusion:

- In this activity Wireshark skills of packet capturing and analysis are performed using TCP and IP protocol headers.
- The IP address and distance to the server are determined by filtering and analyzing the IP header.
- The parameters of conversation with server including time duration of conversation, port numbers, flag fields which determine TCP three way handshake are analyzed using TCP header.