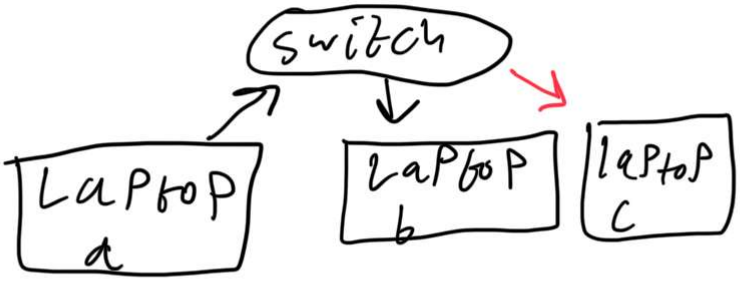
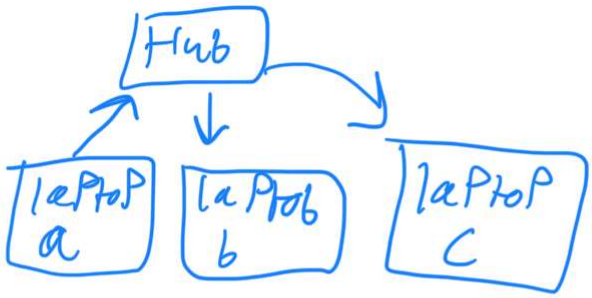
Capture network packet
↳ network Architecture

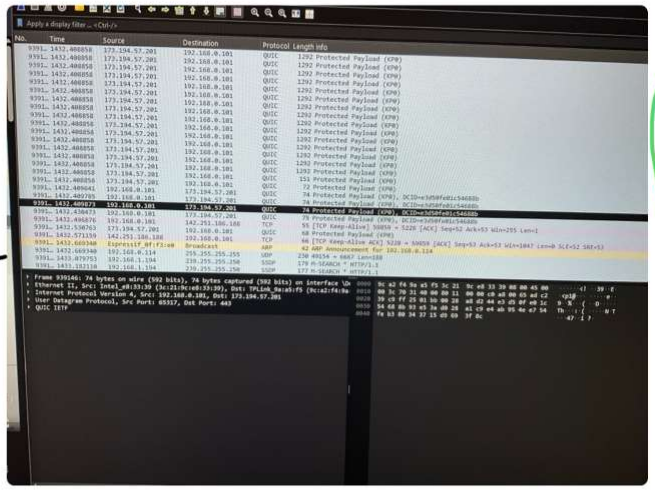




add a Port Spam
spoofing address

CAPTURING PACKETS

Frames

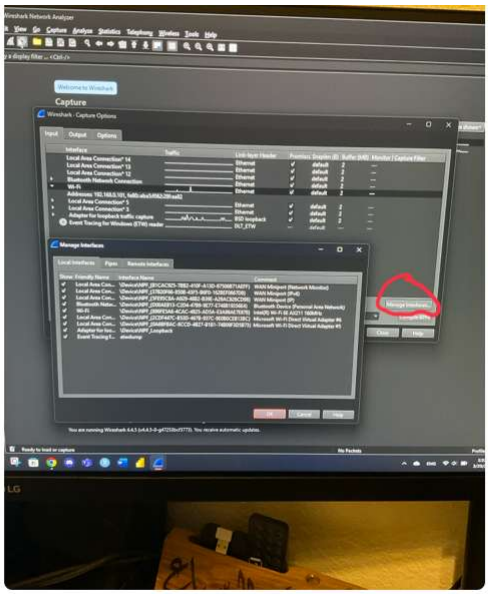


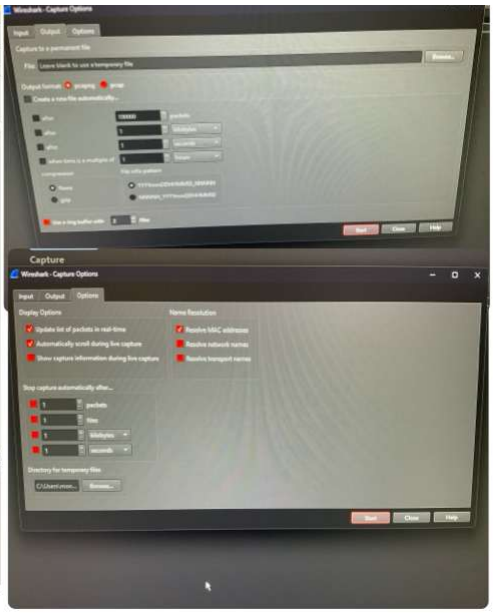
colors mean something

A packet is a group of data at 1 layer of the network stack!
Multiple Frames can go into 1 packet.
Frames come into the network interface ↓
messages going across the network interface
Frames can be aggregated up into packets in the IP layer.
Datagrams at UTP
Segments at TCP

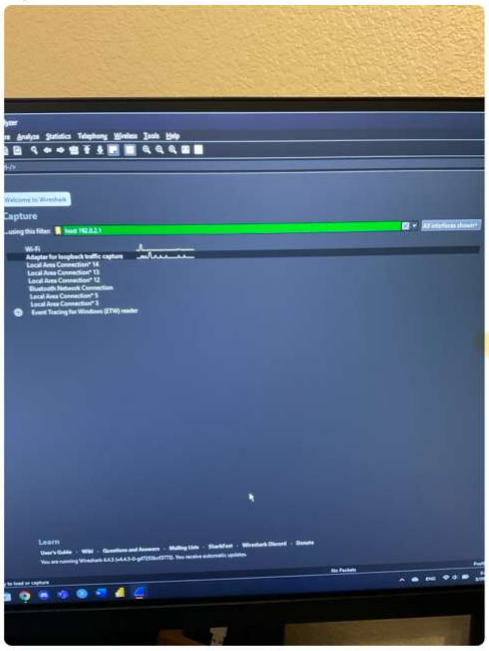
Starting a Packet Capture

Capturing Options:



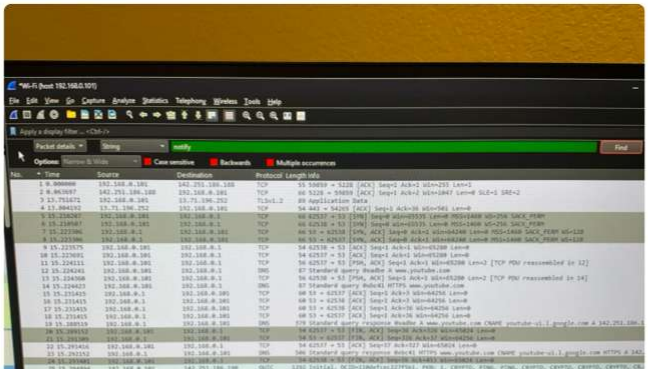


Filters:

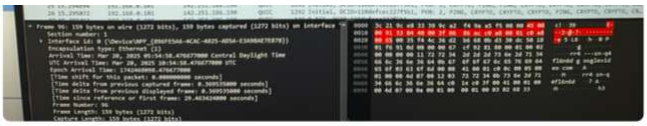


- use the filter bar to find something specefic.
- When the bar is red it didnt find anything. When the bar is green it found something.
- The filter will be in the source or Destination colomn

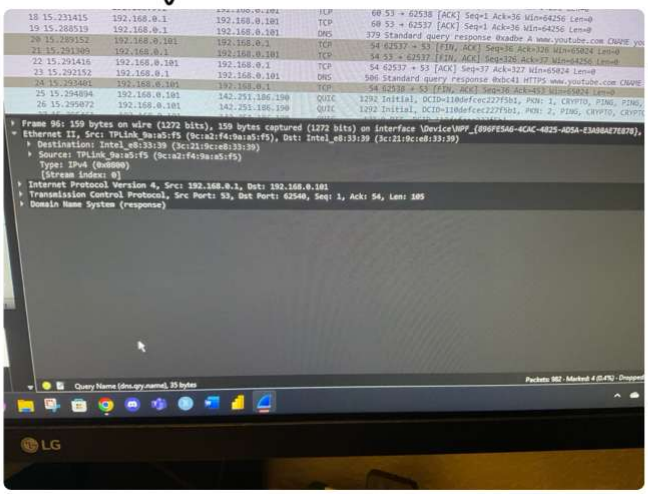
Sorting & Searching:



- Press edit → Find Packet.



Viewing Frame Data:

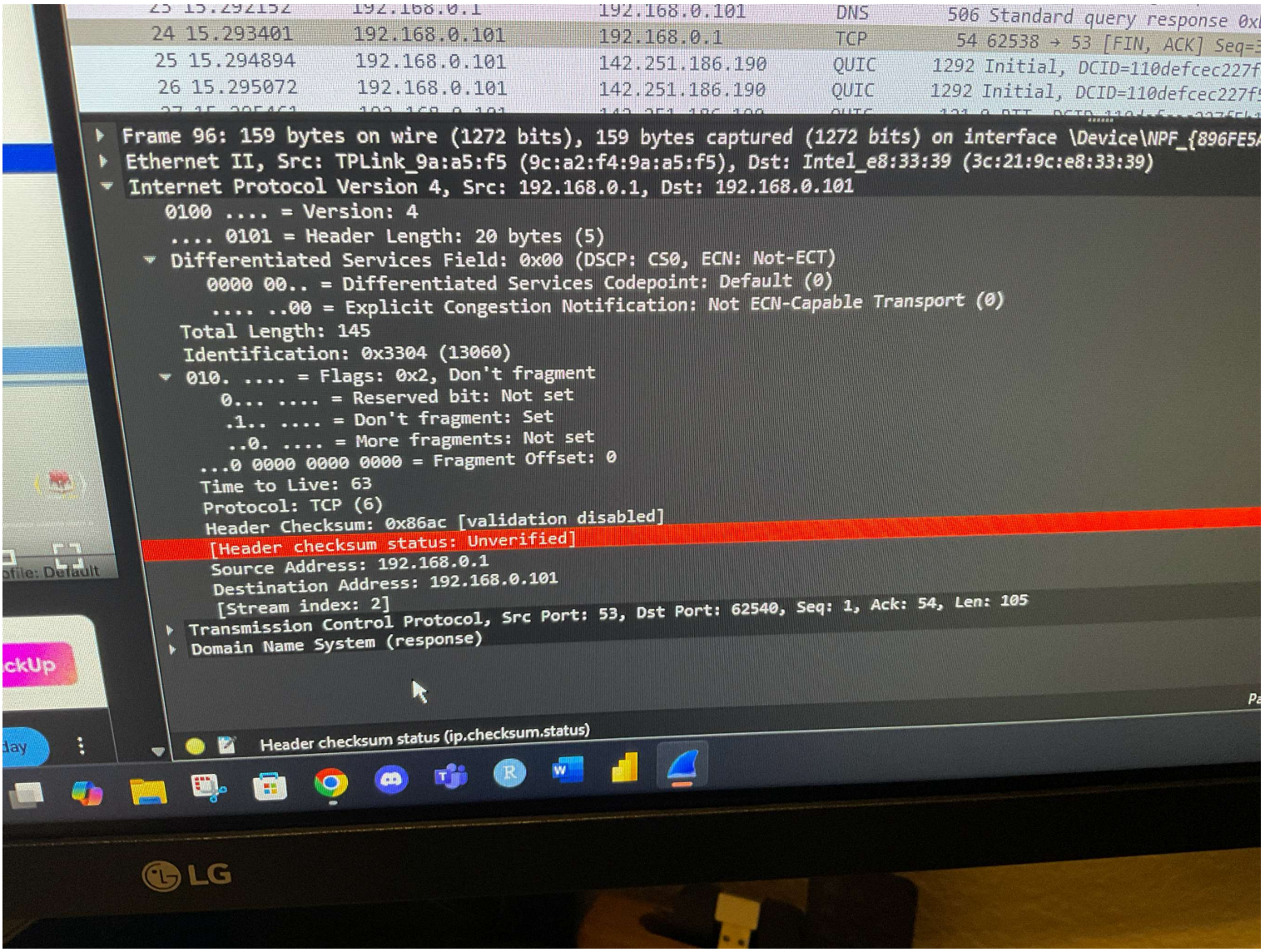


- layer 1 (line 1):
  - Frame # from the list
  - 159 bytes are on the wire which converts to 1272 bits
  - 159 bytes were captured

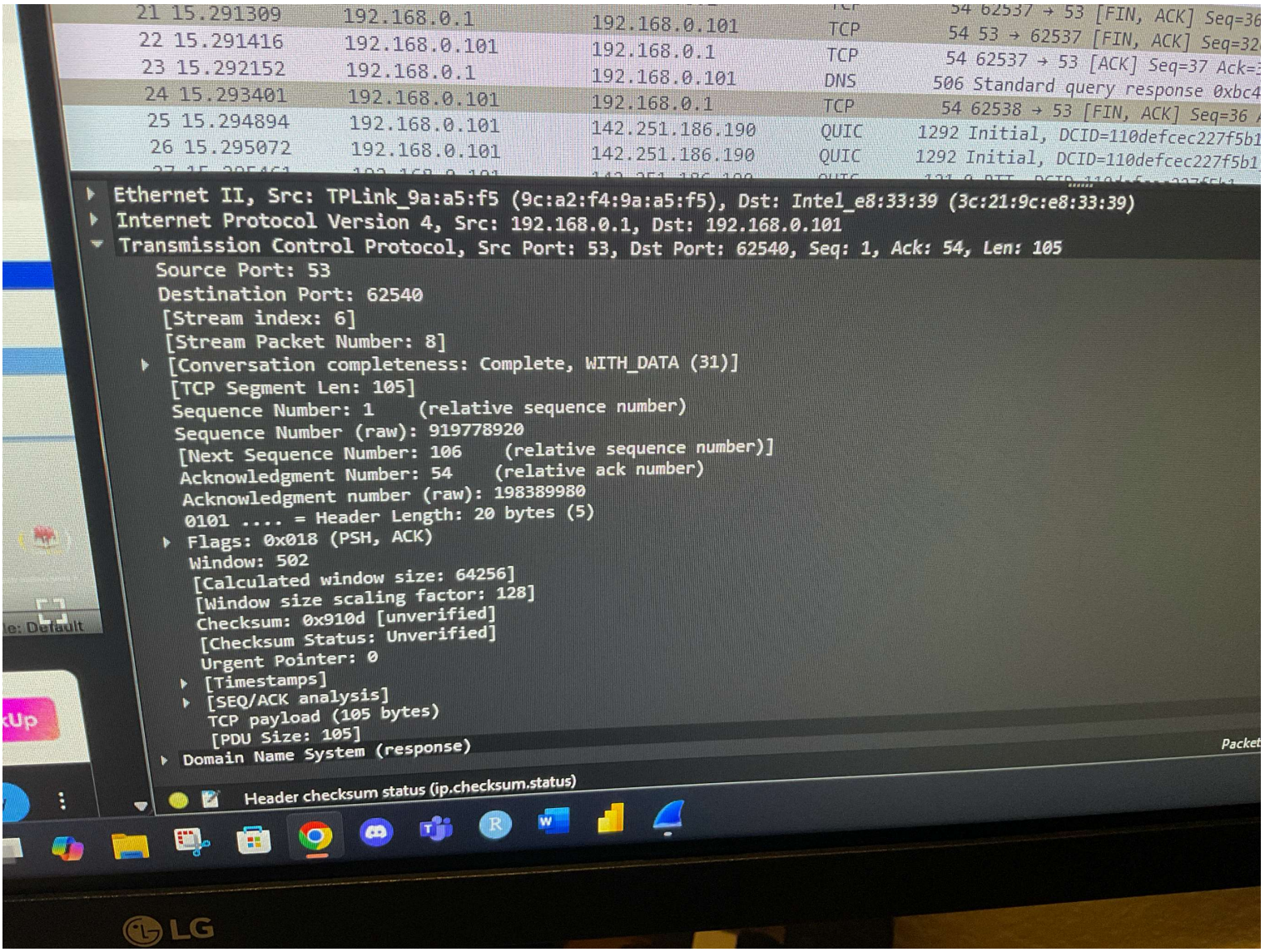
- layer 2 (Eth layer):
  - Source (src)
  - Destination (DST)
  - Type of packet

this refers to the next header. it's an indication in one header to tell the receiving device whats going to be in the next header. So it can figure out how to pull it apart.

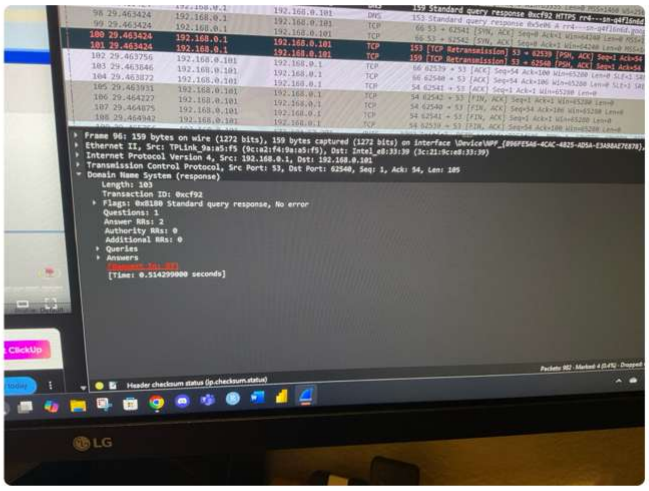
it shows we have an IPv4 message so the next header will have an IP header in the next line.

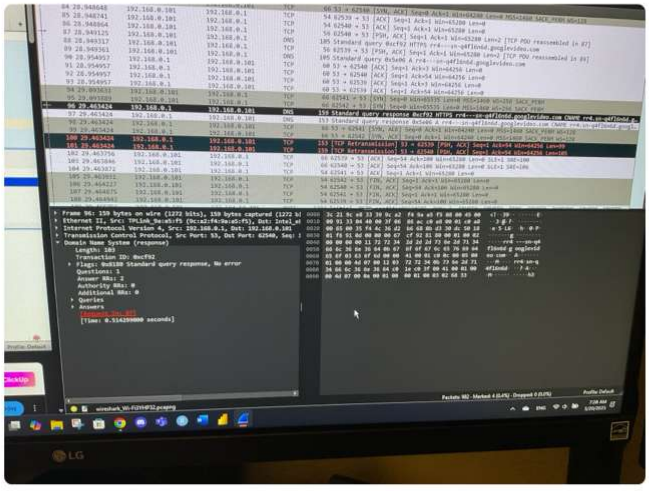

24 15.293401 192.168.0.101 192.168.0.1 TCP 54 62538 → 53 [FIN, ACK] Seq=
25 15.294894 192.168.0.101 142.251.186.190 QUIC 1292 Initial, DCID=110defcec227f
26 15.295072 192.168.0.101 142.251.186.190 QUIC 1292 Initial, DCID=110defcec227f
Frame 96: 159 bytes on wire (1272 bits), 159 bytes captured (1272 bits) on interface \Device\NPF_{896FE5
Ethernet II, Src: TPLink_9a:a5:f5 (9c:a2:f4:9a:a5:f5), Dst: Intel_e8:33:39 (3c:21:9c:e8:33:39)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.101
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
0000 00.. = Differentiated Services Codepoint: Default (0)
.... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
Total Length: 145
Identification: 0x3304 (13060)
010. .... = Flags: 0x2, Don't fragment
0... .... = Reserved bit: Not set
.1.. .... = Don't fragment: Set
..0. .... = More fragments: Not set
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 63
Protocol: TCP (6)
Header Checksum: 0x86ac [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.0.1
Destination Address: 192.168.0.101
[Stream index: 2]
Transmission Control Protocol, Src Port: 53, Dst Port: 62540, Seq: 1, Ack: 54, Len: 105
Domain Name System (response)
Header checksum status (ip.checksum.status)
LG

- layer 3
  - Src & Dst IPs
  - Identification: 0x3304 (13060) → helps identify messages that have been broken up into Fragments. Can identify different Frames that are associated with this the complete packet.
  - 0|0. .... = Flags: 0x2, Don't Fragment → Saying don't Fragment.
  - ...0 0000 0000 0000 = Fragment Offset: 0 → This is the First Fragment.
  - Protocol: TCP (6) → tells us what's the next layer going to be


21 15.291309 192.168.0.1 192.168.0.101 TCP 54 62537 → 53 [FIN, ACK] Seq=36
22 15.291416 192.168.0.101 192.168.0.1 TCP 54 53 → 62537 [FIN, ACK] Seq=32
23 15.292152 192.168.0.1 192.168.0.101 TCP 54 62537 → 53 [ACK] Seq=37 Ack=
24 15.293401 192.168.0.101 192.168.0.1 DNS 506 Standard query response 0xbc4
25 15.294894 192.168.0.101 192.168.0.1 TCP 54 62538 → 53 [FIN, ACK] Seq=36
26 15.295072 192.168.0.101 142.251.186.190 QUIC 1292 Initial, DCID=110defcec227f5b1
27 15.295461 192.168.0.101 142.251.186.190 QUIC 1292 Initial, DCID=110defcec227f5b1
Ethernet II, Src: TPLink_9a:a5:f5 (9c:a2:f4:9a:a5:f5), Dst: Intel_e8:33:39 (3c:21:9c:e8:33:39)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.101
Transmission Control Protocol, Src Port: 53, Dst Port: 62540, Seq: 1, Ack: 54, Len: 105
Source Port: 53
Destination Port: 62540
[Stream index: 6]
[Stream Packet Number: 8]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 105]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 919778920
[Next Sequence Number: 106 (relative sequence number)]
Acknowledgment Number: 54 (relative ack number)
Acknowledgment number (raw): 198389980
0101 .... = Header Length: 20 bytes (5)
Flags: 0x018 (PSH, ACK)
Window: 502
[Calculated window size: 64256]
[Window size scaling factor: 128]
Checksum: 0x910d [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
[Timestamps]
[SEQ/ACK analysis]
TCP payload (105 bytes)
[PDU Size: 105]
Domain Name System (response)
Header checksum status (ip.checksum.status)
Packet
LG

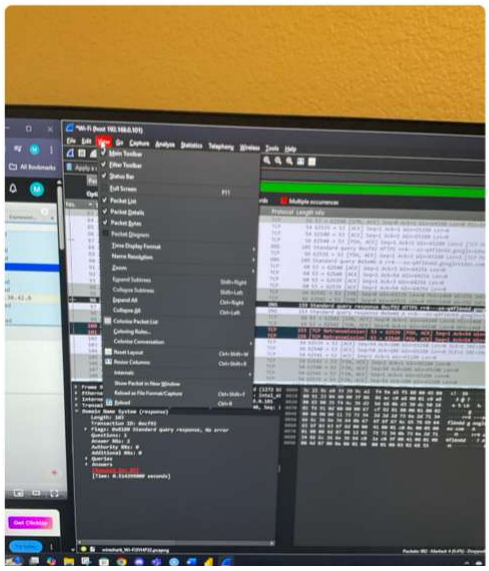
- TCP layer: Source Port: 53
Destination Port: 62540
length: 105



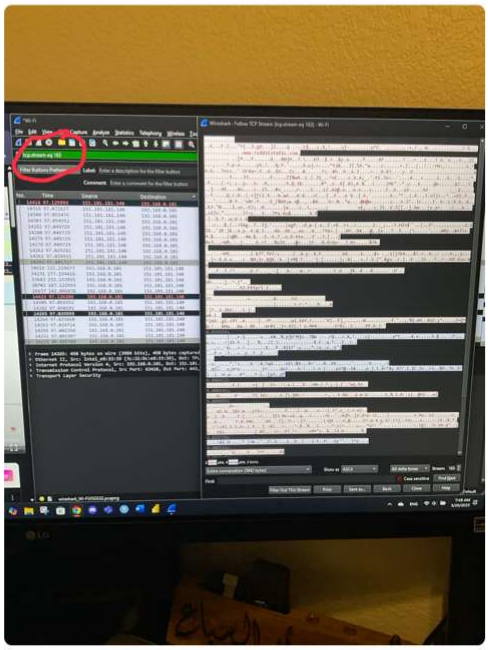


saying the same thing as the layers in machine language.

view tab

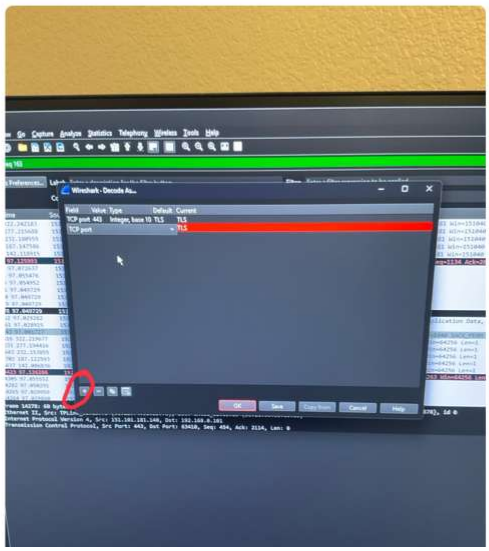


Streams:
communication with a website or system can be logged by wireshark.



- Step 1 → right click data frame
  Step 2 → Press "view"
  Step 3 → Press "TCP Stream"
- red is my laptop, blue is other system.
- shows its about reddit.
- This creates a capture filter

Dissectors:
Step 1: right click a frame
Step 2: press "Decode As..."
- you can add different fields & etc. it will filter



Name Resolutions:
Step 1: Press "view"
Step 2: "name resolution"

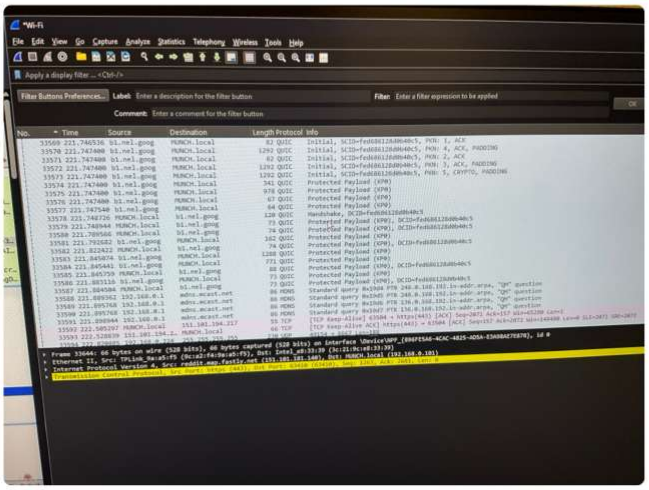
UDP

Physical, network, transport

Changes IP Addresses to known names

DNS lookup

Ports will also change



Saving: "Wireshark/tcpdump/... -pcap" is the best.

You can export specified packets ifs in the "File" tab

Capturing from other sources:
tCP dump on Linux, window, & Mac OS