Wireshark – Packet Analyzer

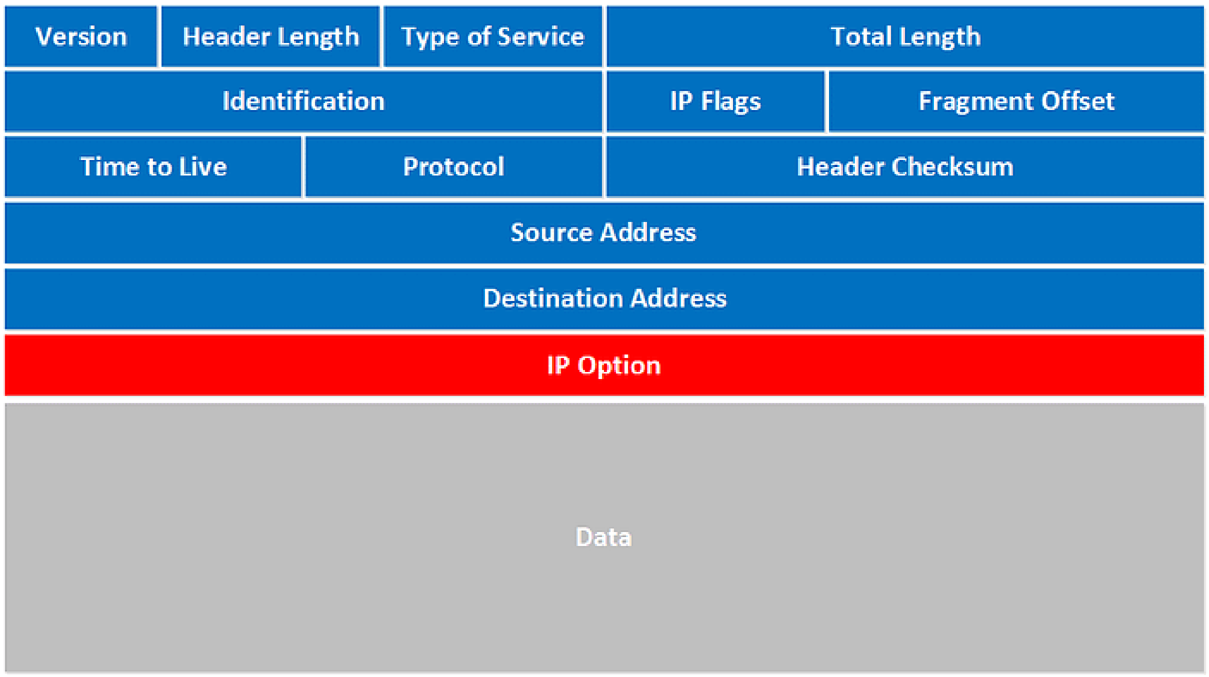
Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues.

Wireshark is also completely open-source, thanks to the community of network engineers around the world. While most security tools are CLI based, Wireshark comes with a fantastic user interface.

Packets

When data is transferred from one computer to another, the data stream consists of smaller units called packets.

When you download a file from the internet, the data is sent from the server as packets are re-assembled by your computer to give you the original file.



A packet can contain the following data:

- Source and destination IP addresses
- Protocol
- Source and destination ports
- Data

- Length, Flags, TTL, etc.

Each packet contains valuable information about the devices involved in a packet transfer. Each data transfer involves thousands or even millions of these packets of data being sent between the source and the destination devices.

Now you would have understood the importance of Wireshark. Wireshark lets you capture each of these packets and inspect them for data.

Wireshark, to a network engineer, is similar to a microscope for a biologist. Wireshark lets you 'listen' to a live network (after you establish a connection to it), capture and inspect packets on the fly.

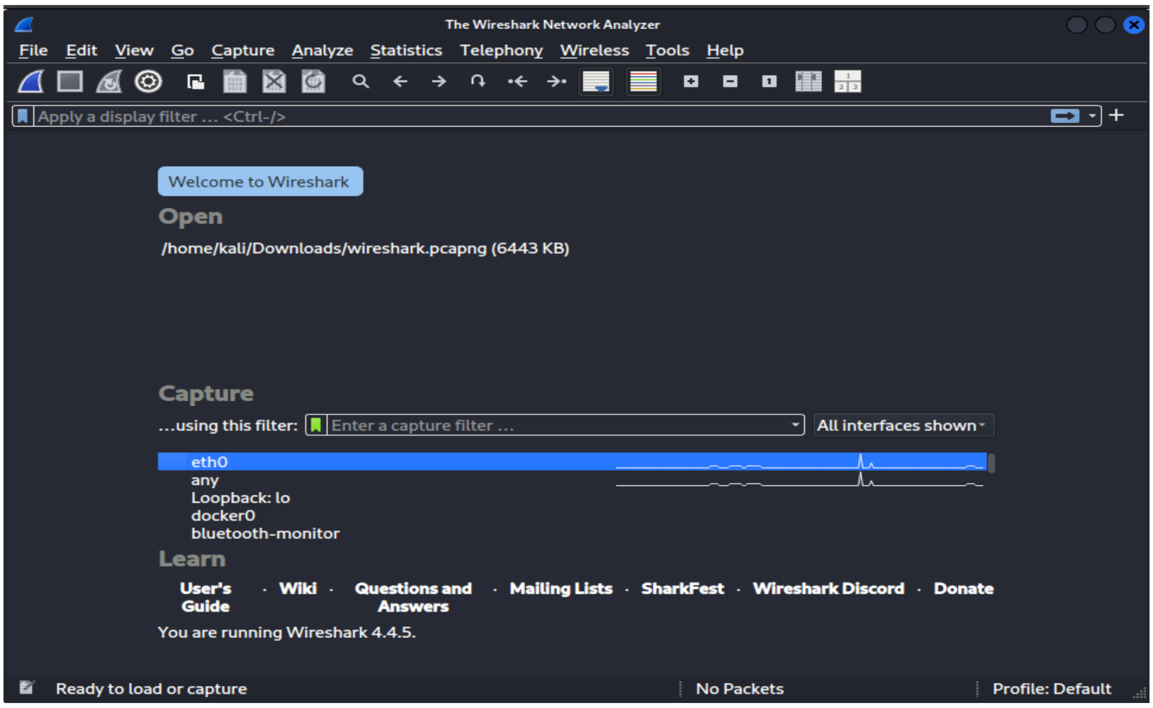
As a network engineer or ethical hacker, you can use Wireshark to debug and secure your networks. As a bad guy (which I don't recommend), you can 'sniff' packets in the network and capture information like credit card transactions.

This is why it is unwise to connect to a public network like Starbucks and perform financial transactions or access private data. Even though sites with HTTPS can encrypt your packets, it is still visible over the network. If someone really wants to crack it, they can.

Wireshark Basics

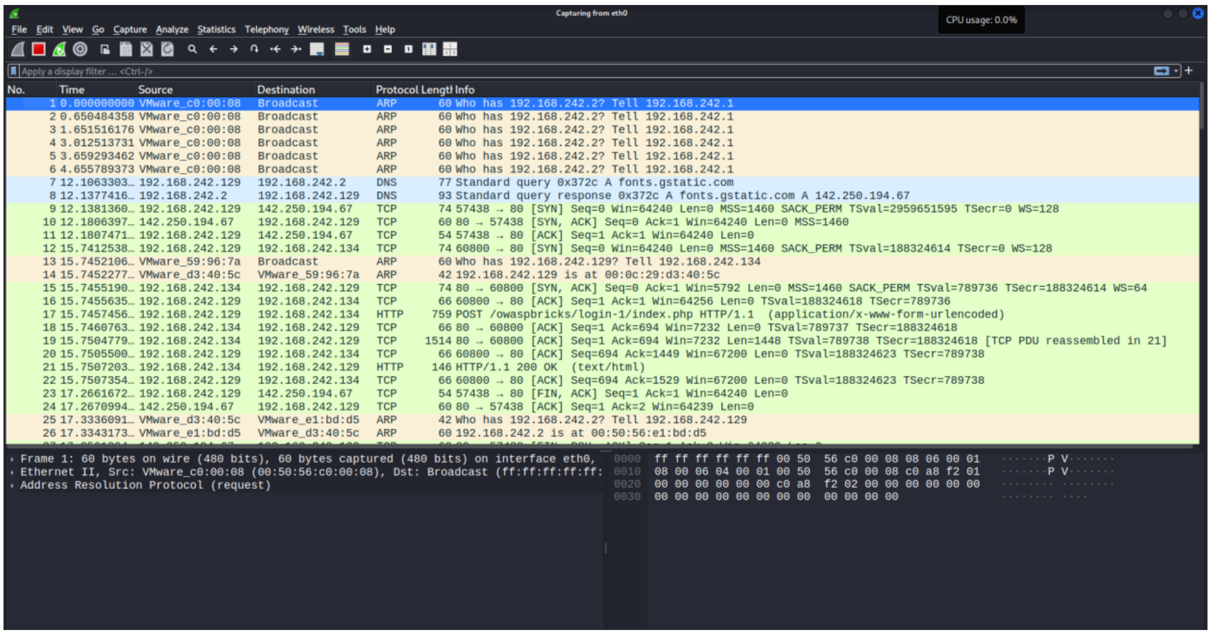
Now let's look at how you can play with wireshark. Download and install wireshark from here.

Wireshark has an awesome GUI, unlike most penetration testing tools. Here's how wireshark looks when you load it.



```
The Wireshark Network Analyzer
File  Edit  View  Go  Capture  Analyze  Statistics  Telephony  Wireless  Tools  Help
Apply a display filter … <Ctrl-/>

Welcome to Wireshark
Open
/home/kali/Downloads/wireshark.pcapng (6443 KB)

Capture
…using this filter: Enter a capture filter …          All interfaces shown
eth0
any
Loopback: lo
docker0
bluetooth-monitor
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Donate
You are running Wireshark 4.4.5.

Ready to load or capture          No Packets          Profile: Default
```

Wireshark lists out the networks you are connected to and you can choose one of them and start listening to the network.

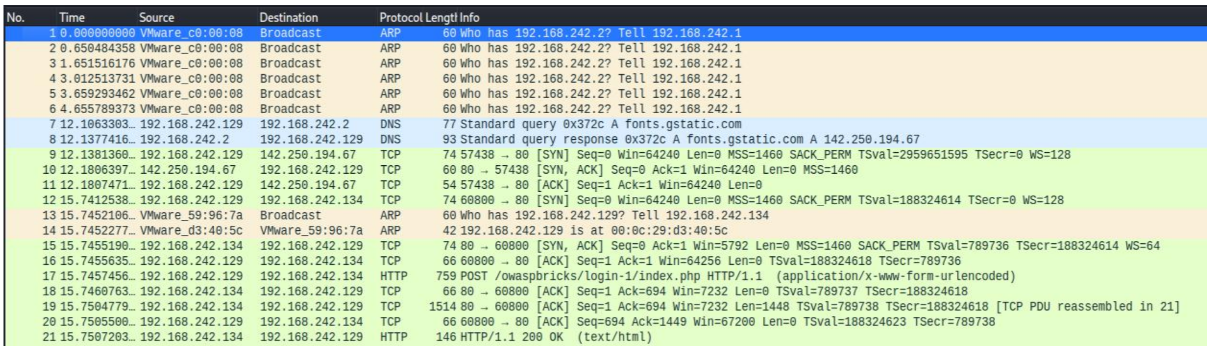


```
Capturing from eth0                                                  CPU usage: 0.0%
File  Edit  View  Go  Capture  Analyze  Statistics  Telephony  Wireless  Tools  Help
Apply a display filter … <Ctrl-/>
No.  Time          Source            Destination       Protocol Length Info
   1 0.000000000   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   2 0.650484358   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   3 1.651516176   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   4 3.012513731   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   5 3.659293462   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   6 4.655789373   VMware_c0:00:08   Broadcast         ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   7 12.1063303…   192.168.242.129   192.168.242.2     DNS        77 Standard query 0x372c A fonts.gstatic.com
   8 12.1377416…   192.168.242.2     192.168.242.129   DNS        93 Standard query response 0x372c A fonts.gstatic.com A 142.250.194.67
   9 12.1381360…   192.168.242.129   142.250.194.67    TCP        74 57438 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2959651595 TSecr=0 WS=128
  10 12.1806397…   142.250.194.67    192.168.242.129   TCP        60 80 → 57438 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
  11 12.1807471…   192.168.242.129   142.250.194.67    TCP        54 57438 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  12 15.7412538…   192.168.242.129   192.168.242.134   TCP        74 60800 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=188324614 TSecr=0 WS=128
  13 15.7452106…   VMware_59:96:7a   Broadcast         ARP        60 Who has 192.168.242.129? Tell 192.168.242.134
  14 15.7452277…   VMware_d3:40:5c   VMware_59:96:7a   ARP        42 192.168.242.129 is at 00:0c:29:d3:40:5c
  15 15.7455190…   192.168.242.134   192.168.242.129   TCP        74 80 → 60800 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM TSval=789736 TSecr=188324614 WS=64
  16 15.7455635…   192.168.242.129   192.168.242.134   TCP        66 60800 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=188324618 TSecr=789736
  17 15.7457456…   192.168.242.129   192.168.242.134   HTTP      759 POST /owaspbricks/login-1/index.php HTTP/1.1  (application/x-www-form-urlencoded)
  18 15.7460763…   192.168.242.134   192.168.242.129   TCP        66 80 → 60800 [ACK] Seq=1 Ack=694 Win=7232 Len=0 TSval=789737 TSecr=188324618
  19 15.7504779…   192.168.242.134   192.168.242.129   TCP      1514 80 → 60800 [ACK] Seq=1 Ack=694 Win=7232 Len=1448 TSval=789738 TSecr=188324618 [TCP PDU reassembled in 21]
  20 15.7505500…   192.168.242.129   192.168.242.134   TCP        66 60800 → 80 [ACK] Seq=694 Ack=1449 Win=67200 Len=0 TSval=188324623 TSecr=789738
  21 15.7507203…   192.168.242.134   192.168.242.129   HTTP      146 HTTP/1.1 200 OK  (text/html)
  22 15.7507354…   192.168.242.129   192.168.242.134   TCP        66 60800 → 80 [ACK] Seq=694 Ack=1529 Win=67200 Len=0 TSval=188324623 TSecr=789738
  23 17.2661672…   192.168.242.129   142.250.194.67    TCP        54 57438 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64240 Len=0
  24 17.2670994…   142.250.194.67    192.168.242.129   TCP        60 80 → 57438 [ACK] Seq=1 Ack=2 Win=64239 Len=0
  25 17.3336091…   VMware_d3:40:5c   VMware_e1:bd:d5   ARP        42 Who has 192.168.242.2? Tell 192.168.242.129
  26 17.3343173…   VMware_e1:bd:d5   VMware_d3:40:5c   ARP        60 192.168.242.2 is at 00:50:56:e1:bd:d5

› Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0,
› Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: Broadcast (ff:ff:ff:ff:ff:
› Address Resolution Protocol (request)

0000  ff ff ff ff ff ff 00 50  56 c0 00 08 08 06 00 01   ·······P V·······
0010  08 00 06 04 00 01 00 50  56 c0 00 08 c0 a8 f2 01   ·······P V·······
0020  00 00 00 00 00 00 c0 a8  f2 02 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 00 00 00 00  00 00 00 00               ········ ····
```

Total there are three panes in wireshark.

Packet List pane

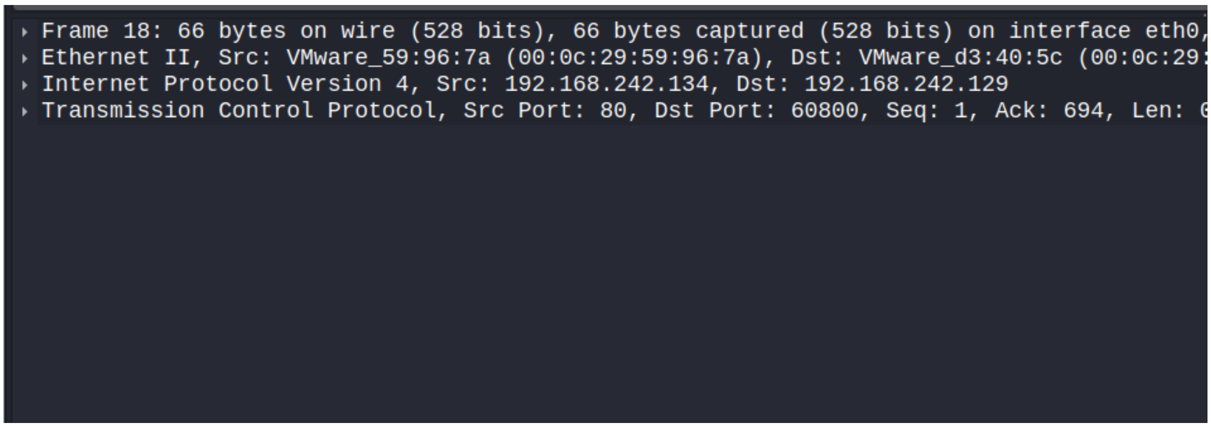
In this packet list pane there are several types of information like Timing, destination and Source IP , What types of protocol used , what was the length of the packet, and some detailed information about the packet.



```
No.  Time          Source             Destination      Protocol Lengtl Info
   1 0.000000000   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   2 0.650484358   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   3 1.651516176   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   4 3.012513731   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   5 3.659293462   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   6 4.655789373   VMware_c0:00:08    Broadcast        ARP        60 Who has 192.168.242.2? Tell 192.168.242.1
   7 12.1063303…   192.168.242.129    192.168.242.2    DNS        77 Standard query 0x372c A fonts.gstatic.com
   8 12.1377416…   192.168.242.2      192.168.242.129  DNS        93 Standard query response 0x372c A fonts.gstatic.com A 142.250.194.67
   9 12.1381360…   192.168.242.129    142.250.194.67   TCP        74 57438 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2959651595 TSecr=0 WS=128
  10 12.1806397…   142.250.194.67     192.168.242.129  TCP        60 80 → 57438 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
  11 12.1807471…   192.168.242.129    142.250.194.67   TCP        54 57438 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  12 15.7412538…   192.168.242.129    192.168.242.134  TCP        74 60800 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=188324614 TSecr=0 WS=128
  13 15.7452106…   VMware_59:96:7a    Broadcast        ARP        60 Who has 192.168.242.129? Tell 192.168.242.134
  14 15.7452277…   VMware_d3:40:5c    VMware_59:96:7a  ARP        42 192.168.242.129 is at 00:0c:29:d3:40:5c
  15 15.7455190…   192.168.242.134    192.168.242.129  TCP        74 80 → 60800 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM TSval=789736 TSecr=188324614 WS=64
  16 15.7455635…   192.168.242.129    192.168.242.134  TCP        66 60800 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=188324618 TSecr=789736
  17 15.7457456…   192.168.242.129    192.168.242.134  HTTP      759 POST /owaspbricks/login-1/index.php HTTP/1.1  (application/x-www-form-urlencoded)
  18 15.7460763…   192.168.242.134    192.168.242.129  TCP        66 80 → 60800 [ACK] Seq=1 Ack=694 Win=7232 Len=0 TSval=789737 TSecr=188324618
  19 15.7504779…   192.168.242.134    192.168.242.129  TCP      1514 80 → 60800 [ACK] Seq=1 Ack=694 Win=7232 Len=1448 TSval=789738 TSecr=188324618 [TCP PDU reassembled in 21]
  20 15.7505500…   192.168.242.129    192.168.242.134  TCP        66 60800 → 80 [ACK] Seq=694 Ack=1449 Win=67200 Len=0 TSval=188324623 TSecr=789738
  21 15.7507203…   192.168.242.134    192.168.242.129  HTTP      146 HTTP/1.1 200 OK  (text/html)
```

In this pane display the packets captured. Each line represents an individual packet that you can click and analyse in detail using the other two panes.

Packet Details Pane



```
› Frame 18: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0,
› Ethernet II, Src: VMware_59:96:7a (00:0c:29:59:96:7a), Dst: VMware_d3:40:5c (00:0c:29:
› Internet Protocol Version 4, Src: 192.168.242.134, Dst: 192.168.242.129
› Transmission Control Protocol, Src Port: 80, Dst Port: 60800, Seq: 1, Ack: 694, Len: 0
```

You can select a packet and then look at the packet information in more detail using the Packet Details pane. It displays information such as IP addresses,Ports,and other imformation contained within the packet.

After selecting a packet from the packet list, the **Packet Details** pane allows you to view an in-depth breakdown of the packet's contents. This includes structured information about each protocol layer—such as:

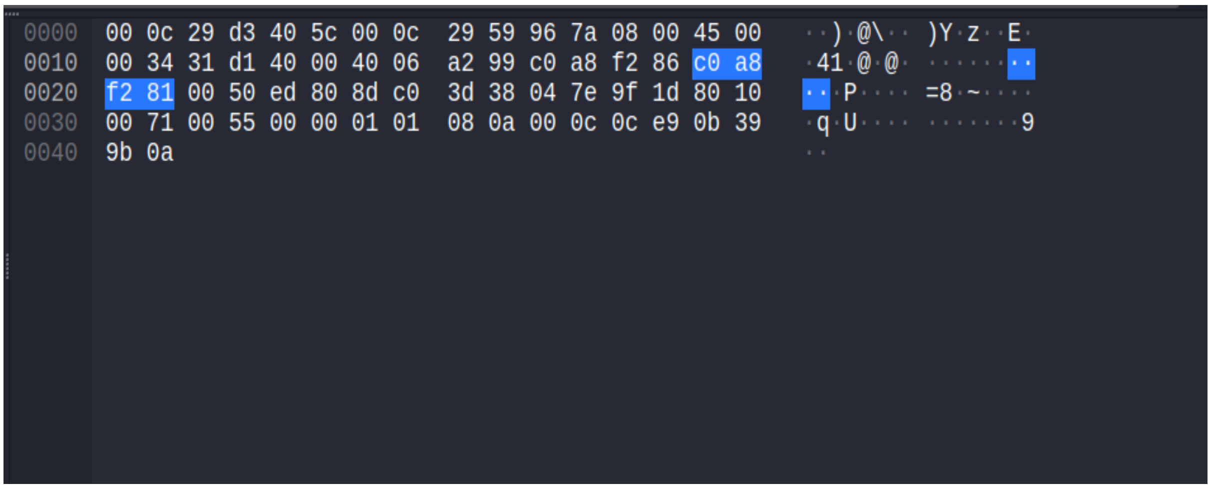
- **Ethernet Header**: MAC addresses (source and destination)

- **IP Header**: Source and destination IP addresses, TTL, fragmentation info, etc.
- **Transport Layer**: Port numbers (TCP/UDP), flags (SYN, ACK), sequence numbers
- **Application Layer**: HTTP, DNS, TLS, etc., depending on the protocol used
- **Other Information**: Checksums, packet lengths, and additional metadata

You can expand each protocol layer to view individual fields and their decoded values, making it easier to analyze or troubleshoot specific aspects of network traffic.

Packet Bytes Pane

This pane give the raw data of the selected packet in bytes. The data is displayed as a hex dump, which is displaying binary data in hexadecimal.



```
0000  00 0c 29 d3 40 5c 00 0c  29 59 96 7a 08 00 45 00   ··)·@\··  )Y·z··E·
0010  00 34 31 d1 40 00 40 06  a2 99 c0 a8 f2 86 c0 a8   ·41·@·@·  ········
0020  f2 81 00 50 ed 80 8d c0  3d 38 04 7e 9f 1d 80 10   ···P····  =8·~····
0030  00 71 00 55 00 00 01 01  08 0a 00 0c 0c e9 0b 39   ·q·U····  ·······9
0040  9b 0a                                              ··
```

Filters

Wireshark has filters that help you to narrow down the type of data you are looking for. There are two types of Filters: Capture Filter and Display Filter.

Capture Filter

You can set a capture filter before starting to analyse a network. When you set a capture filter, it only capture the packet that match the capture filter.

- **Purpose:** Limits the traffic that Wireshark collects during a live capture.
- **Set Before Capture:** Must be configured **before** starting the packet capture.
- **Efficient:** Useful when you're only interested in specific traffic, which reduces memory usage and noise.

Example:

If you want to capture only the traffic to and form a specific ip address

Host 192.198.242.134

- Capture only HTTP traffic:

Port 80

- Capture only HTTP traffic:

src host 192.168.242.129

Display Filter

- **Purpose:** Allows you to filter **after** the capture, based on various packet fields.
- **Real-time or Post-Capture:** Can be used while capturing or after the capture is complete.
- **Very Precise:** Supports complex expressions and protocol-level filters.

Example:

To display packets where the source IP is 192.168.242.134.

ip.scr == 192.168.242.134

Show only HTTP packets:

http

Show TCP packets with destination port 443 (HTTPS):

tcp.dstport == 443

Filter only DNS queries:

dns.flags.response == 0