Analyze TCP, UDP and IPV4 Header using Wireshark

TCP Analysis using Wireshark

TCP, or Transmission Control Protocol, is one of the main protocols in IP networking. Wireshark, being a potent network protocol analyzer, provides the ability to dissect the details of TCP communication. It can be utilized to analyze TCP connections in-depth, offering valuable insights about the network traffic.

In performing TCP analysis, Wireshark enables users to visualize and examine the intricate details of TCP packets. These details may include source and destination ports, sequence and acknowledgement numbers, flags, and more. It also allows users to investigate the mechanisms of TCP flow control, error control, and congestion control. Users can view the TCP window size, identify retransmissions, and observe how TCP responds to network congestion.

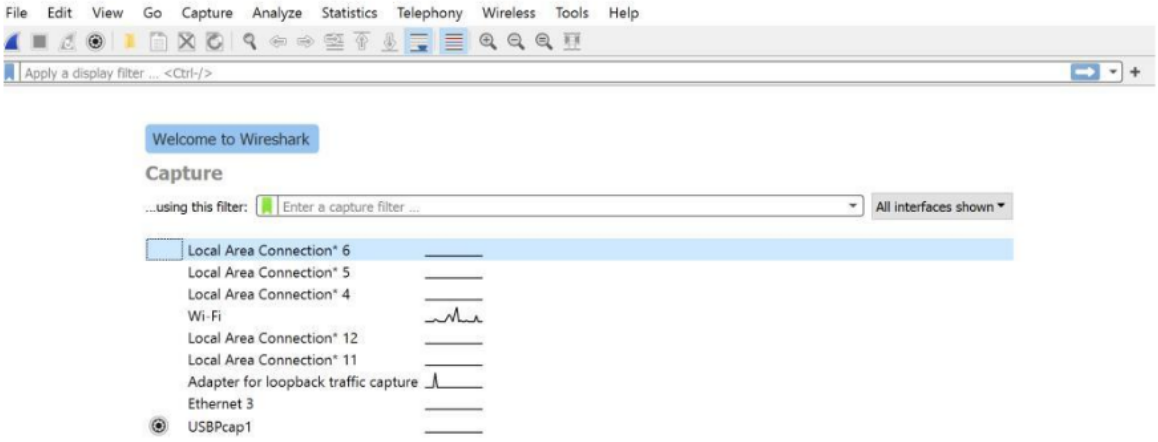
In addition to these, Wireshark provides a graphical representation of the communication flow between systems. This feature proves vital in identifying potential issues or anomalies with the TCP connections. It empowers users with the ability to perform deep packet inspection, which can provide detailed information like packet timing, TCP handshake information, and even the payload data itself. By leveraging these data, users can gain important insights into their network's behavior, aiding them to troubleshoot issues effectively and optimize network performance efficiently.

Furthermore, Wireshark provides the functionality to filter the TCP segments based on specific criteria. This includes filtering by source and destination IP addresses, ports, and specific flags in the TCP header. This greatly enhances the efficiency of analyzing TCP traffic, allowing users to focus on particular aspects or incidents in the network traffic.

To launch Wireshark, follow these steps:

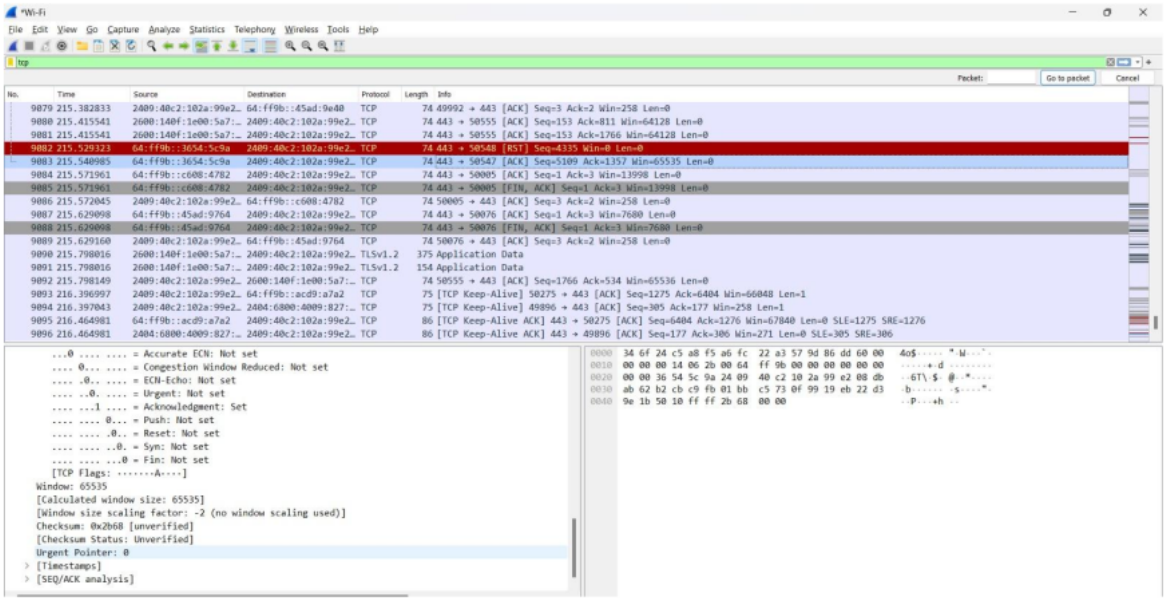
- Open the start menu or application directory on your computer.

- Locate and click on the Wireshark application to open it. If it's not there, you might need to install it first.
- Once the application is open, you'll see an interface with a list of available network connections.
- Select the network connection you wish to monitor. Usually, this will be the connection you're currently using to connect to the internet.
- Click on the 'Start' button to begin capturing packets on that network.
- You are now running Wireshark and can begin your analysis.

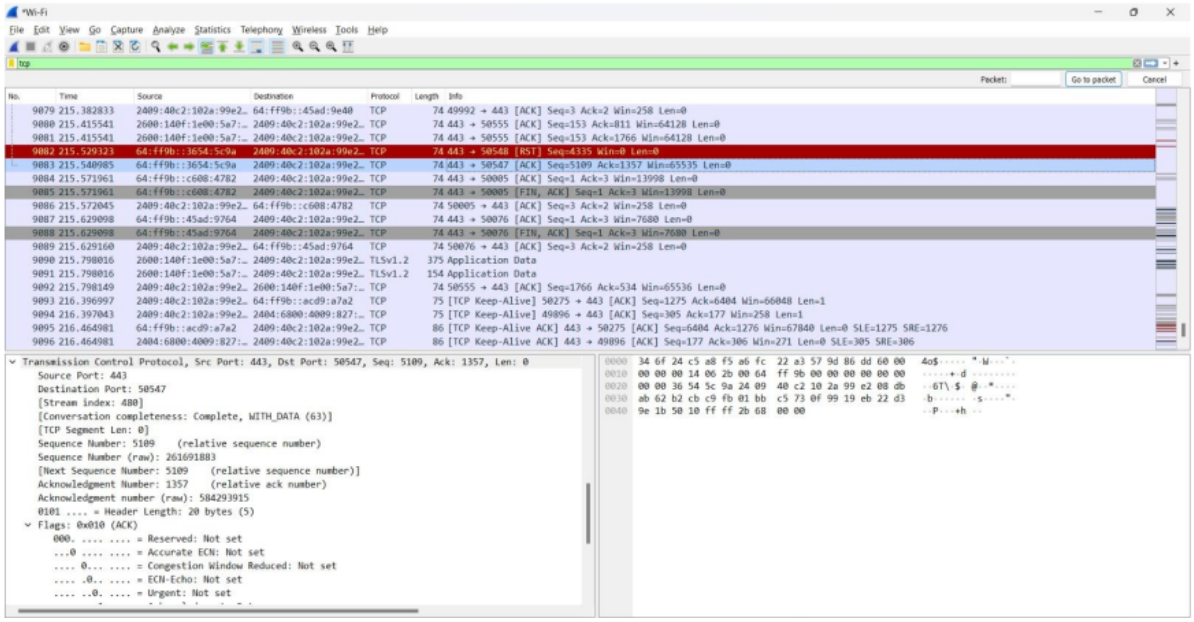


Now we have the captured packets and you will be having the captured packet list on the screen. Since we are concerned here with only TCP packets as we are doing TCP analysis, we shall be filtering out TCP packets from the packet pool.

In the display filter bar on the screen, enter TCP and apply the filter.



From analyzing the menu in the menu bar select display filters or from capture select capture filters and then TCP only and ok



Below, you will find a list of TCP packets. The first three packets in the list are part of the three-way handshake mechanism of TCP, which is used to establish a connection.

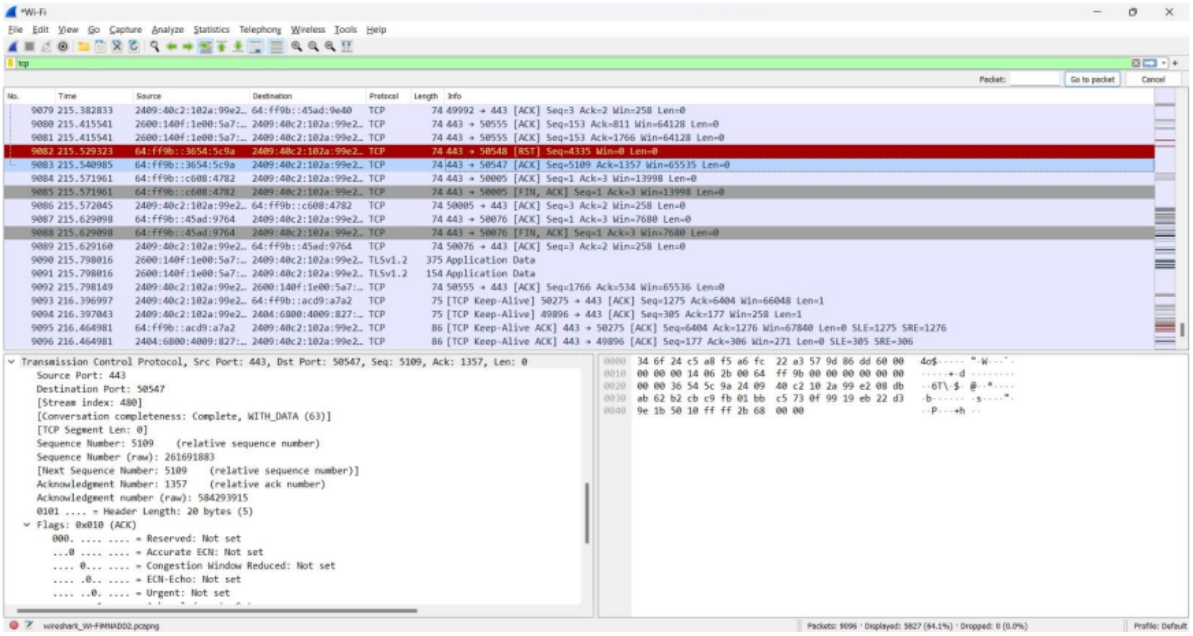
Here's a brief overview of this mechanism and the three steps involved:

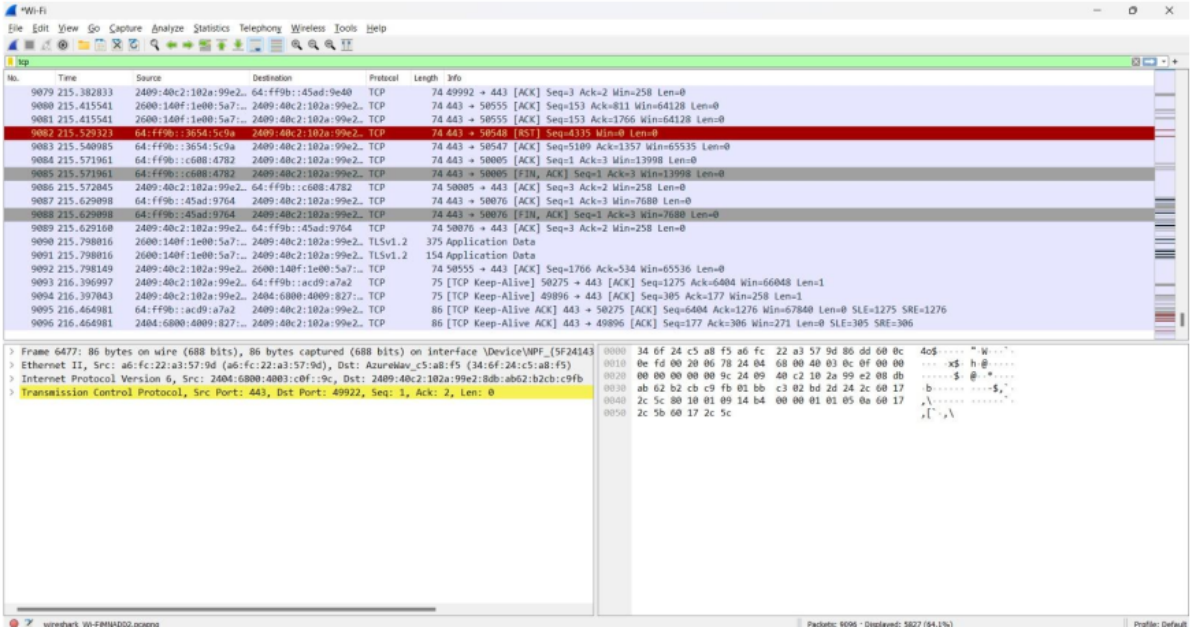
- Your local host sends a synchronization packet (SYN) to the server it wants to connect to.
- The server responds by sending an acknowledgment packet (ACK) to your local host, indicating that it has received the SYN request and also sends a synchronization packet (SYN) to confirm the connection. This is known as an SYN+ACK packet.
- Your local host responds to this request by sending an ACK after receiving the SYN from the server.

The following parameters are important when analyzing TCP packets:

- **Source port**: The port of your host network used for communication.
- **Destination port**: The port of the destination server.
- **TCP segment length**: Represents the data length in the selected packet.
- **Sequence number:** A method used by Wireshark to give each packet a particular index for tracking purposes. This index starts at 0.
- **Next sequence numbe**r: The sum of the sequence number and the segment length of the current packet.
- **Acknowledgment number**: Contains the byte length of data received.
- **Header length**: The length of the TCP header, which can vary from 20 to 60.

The first three packets of the TCP list exhibit the three connection establishment steps, with each packet type (ACK, SYN, SYN-ACK) listed on their respective sides. To examine a packet more closely, you can select it and view the TCP parameters in the expert view in the packet detail section just below the packet list.





UDP Analysis using Wireshark

User Datagram Protocol (UDP) is another core communication protocol used in IP networking. Unlike TCP, it is a connectionless protocol with no guarantee of data

delivery, making it faster but less reliable. Wireshark can be used to analyze UDP traffic just as effectively as TCP.

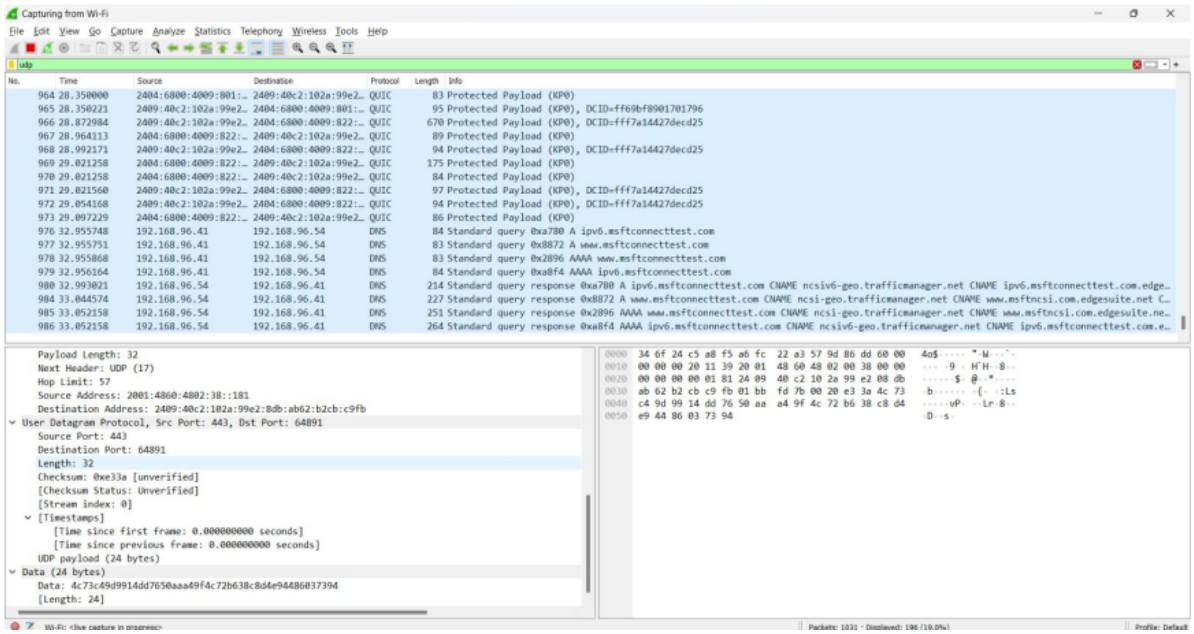
To filter and analyze UDP packets, you can follow the same initial steps as for TCP analysis. However, in the display filter bar, enter `udp` instead of `tcp` to display only UDP packets.

UDP analysis in Wireshark can provide valuable information about the network traffic, such as the source and destination ports, length of the data, and the payload data itself. Additionally, Wireshark allows you to view any potential issues with the UDP traffic, such as errors or packet loss.

Wireshark also provides a graphical representation of the communication flow between systems for UDP traffic, similar to TCP. This can help identify potential issues or anomalies in the UDP traffic.

To examine a packet more closely, you can select it and view the UDP parameters in the expert view in the packet detail section just below the packet list. Here, you will find the details of the UDP packet, including the source port, destination port, and length of data.

Please note that, unlike TCP, UDP does not have sequence or acknowledgment numbers, and there is no three-way handshake mechanism. This is because, as a connectionless protocol, UDP does not establish a connection before data transmission. As a result, the analysis of UDP packets is somewhat more straightforward than that of TCP packets.



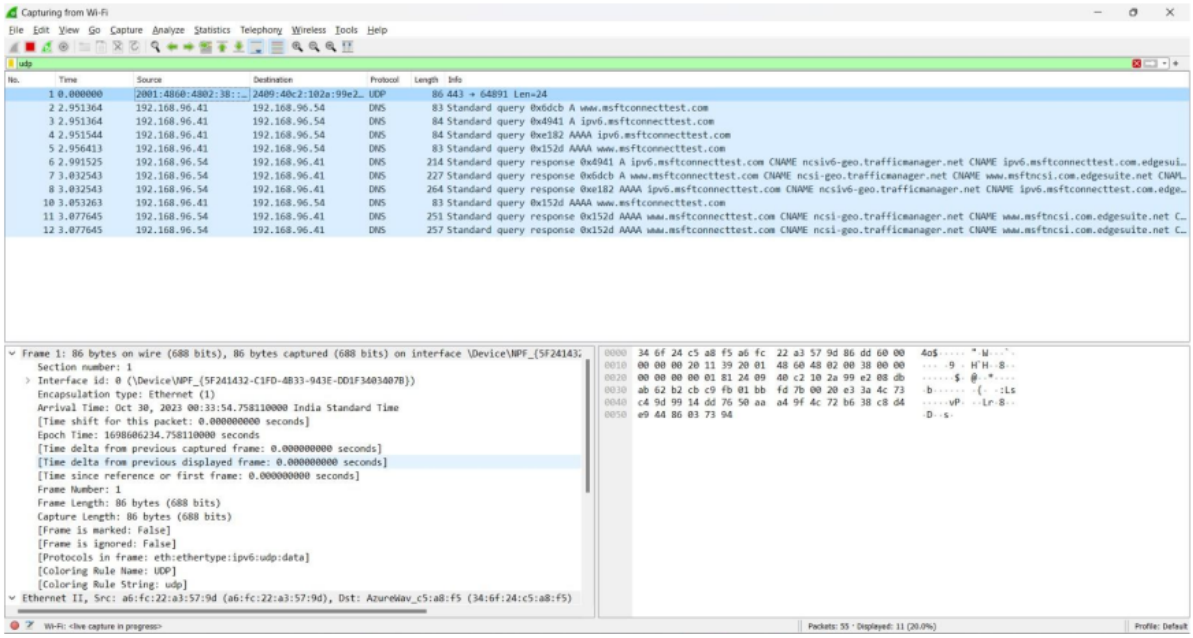
UDP HEADER:

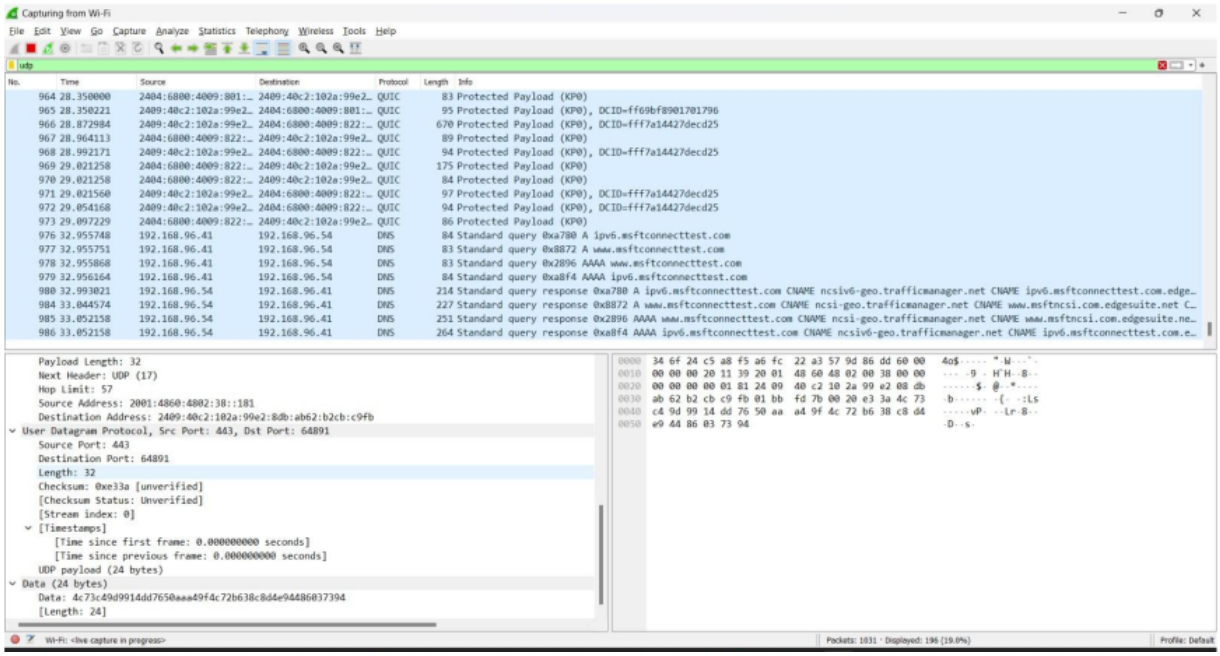
The UDP header is much simpler than the TCP header. It has only four fields: Source Port, Destination Port, Length, and Checksum.

- **Source Port**: This field identifies the sending port.
- **Destination Port**: This is the port to which the packet is sent.
- **Length**: This field specifies the length in bytes of the UDP header and the encapsulated data. The minimum length is 8 bytes, the length of the header.
- **Checksum**: This is used for error-checking of the header and data.

These fields can be analyzed directly in Wireshark by selecting a UDP packet and examining the details in the packet details pane. The Source Port and Destination Port fields can give you an understanding of the endpoints of the communication. The Length field can tell you the size of the data being transmitted, and the Checksum field can help you verify the integrity of the data.

Please note that because UDP is a connectionless protocol, these fields are the only ones available in the UDP header. There is no sequence or acknowledgment information, as there is in TCP, because UDP does not guarantee delivery or require a connection setup phase.


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
udp
No. Time Source Destination Protocol Length Info
1 0.000000 2001:4860:4802:38::... 2409:40c2:102a:99e2... UDP 86 443 → 64891 Len=24
2 2.951364 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x6dcb A www.msftconnecttest.com
3 2.951364 192.168.96.41 192.168.96.54 DNS 84 Standard query 0x4941 A ipv6.msftconnecttest.com
4 2.951544 192.168.96.41 192.168.96.54 DNS 84 Standard query 0xe182 AAAA ipv6.msftconnecttest.com
5 2.956413 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x152d AAAA www.msftconnecttest.com
6 2.991525 192.168.96.54 192.168.96.41 DNS 214 Standard query response 0x4941 A ipv6.msftconnecttest.com CNAME ncsiv6-geo.trafficmanager.net CNAME ipv6.msftconnecttest.com.edgesui...
7 3.032543 192.168.96.54 192.168.96.41 DNS 227 Standard query response 0x6dcb A www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.net CNAM...
8 3.032543 192.168.96.54 192.168.96.41 DNS 264 Standard query response 0xe182 AAAA ipv6.msftconnecttest.com CNAME ncsiv6-geo.trafficmanager.net CNAME ipv6.msftconnecttest.com.edge...
10 3.053263 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x152d AAAA www.msftconnecttest.com
11 3.077645 192.168.96.54 192.168.96.41 DNS 251 Standard query response 0x152d AAAA www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.net C...
12 3.077645 192.168.96.54 192.168.96.41 DNS 257 Standard query response 0x152d AAAA www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.net C...
Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface \Device\NPF_{5F24143;
Section number: 1
Interface id: 0 (\Device\NPF_{5F241432-C1FD-4B33-943E-DD1F3403407B})
Encapsulation type: Ethernet (1)
Arrival Time: Oct 30, 2023 00:33:54.758110000 India Standard Time
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1698606234.758110000 seconds
[Time delta from previous captured frame: 0.000000000 seconds]
[Time delta from previous displayed frame: 0.000000000 seconds]
[Time since reference or first frame: 0.000000000 seconds]
Frame Number: 1
Frame Length: 86 bytes (688 bits)
Capture Length: 86 bytes (688 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ethertype:ipv6:udp:data]
[Coloring Rule Name: UDP]
[Coloring Rule String: udp]
Ethernet II, Src: a6:fc:22:a3:57:9d (a6:fc:22:a3:57:9d), Dst: AzureWav_c5:a8:f5 (34:6f:24:c5:a8:f5)
0000 34 6f 24 c5 a8 f5 a6 fc 22 a3 57 9d 86 dd 60 00 4o$ ···· "·W···`·
0010 00 00 00 20 11 39 20 01 48 60 48 02 00 38 00 00 ··· ·9 · H`H··8··
0020 00 00 00 00 01 81 24 09 40 c2 10 2a 99 e2 08 db ·····$· @··*····
0030 ab 62 b2 cb c9 fb 01 bb fd 7b 00 20 e3 3a 4c 73 ·b····· ·{· ·:Ls
0040 c4 9d 99 14 dd 76 50 aa a4 9f 4c 72 b6 38 c8 d4 ·····vP· ··Lr·8··
0050 e9 44 86 03 73 94 ·D··s·
Wi-Fi: <live capture in progress>
Packets: 55 · Displayed: 11 (20.0%)
Profile: Default


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
udp
No. Time Source Destination Protocol Length Info
964 28.350000 2404:6800:4009:801:... 2409:40c2:102a:99e2... QUIC 83 Protected Payload (KP0)
965 28.350221 2409:40c2:102a:99e2... 2404:6800:4009:801:... QUIC 95 Protected Payload (KP0), DCID=ff69bf8901701796
966 28.872984 2409:40c2:102a:99e2... 2404:6800:4009:822:... QUIC 670 Protected Payload (KP0), DCID=fff7a14427decd25
967 28.964113 2404:6800:4009:822:... 2409:40c2:102a:99e2... QUIC 89 Protected Payload (KP0)
968 28.992171 2409:40c2:102a:99e2... 2404:6800:4009:822:... QUIC 94 Protected Payload (KP0), DCID=fff7a14427decd25
969 29.021258 2404:6800:4009:822:... 2409:40c2:102a:99e2... QUIC 175 Protected Payload (KP0)
970 29.021258 2404:6800:4009:822:... 2409:40c2:102a:99e2... QUIC 84 Protected Payload (KP0)
971 29.021560 2409:40c2:102a:99e2... 2404:6800:4009:822:... QUIC 97 Protected Payload (KP0), DCID=fff7a14427decd25
972 29.054168 2409:40c2:102a:99e2... 2404:6800:4009:822:... QUIC 94 Protected Payload (KP0), DCID=fff7a14427decd25
973 29.097229 2404:6800:4009:822:... 2409:40c2:102a:99e2... QUIC 86 Protected Payload (KP0)
976 32.955748 192.168.96.41 192.168.96.54 DNS 84 Standard query 0xa780 A ipv6.msftconnecttest.com
977 32.955751 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x8872 A www.msftconnecttest.com
978 32.955868 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x2896 AAAA www.msftconnecttest.com
979 32.956164 192.168.96.41 192.168.96.54 DNS 84 Standard query 0xa8f4 AAAA ipv6.msftconnecttest.com
980 32.993021 192.168.96.54 192.168.96.41 DNS 214 Standard query response 0xa780 A ipv6.msftconnecttest.com CNAME ncsiv6-geo.trafficmanager.net CNAME ipv6.msftconnecttest.com.edge...
984 33.044574 192.168.96.54 192.168.96.41 DNS 227 Standard query response 0x8872 A www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.net C...
985 33.052158 192.168.96.54 192.168.96.41 DNS 251 Standard query response 0x2896 AAAA www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.ne...
986 33.052158 192.168.96.54 192.168.96.41 DNS 264 Standard query response 0xa8f4 AAAA ipv6.msftconnecttest.com CNAME ncsiv6-geo.trafficmanager.net CNAME ipv6.msftconnecttest.com.e...
Payload Length: 32
Next Header: UDP (17)
Hop Limit: 57
Source Address: 2001:4860:4802:38::181
Destination Address: 2409:40c2:102a:99e2:8db:ab62:b2cb:c9fb
User Datagram Protocol, Src Port: 443, Dst Port: 64891
Source Port: 443
Destination Port: 64891
Length: 32
Checksum: 0xe33a [unverified]
[Checksum Status: Unverified]
[Stream index: 0]
[Timestamps]
[Time since first frame: 0.000000000 seconds]
[Time since previous frame: 0.000000000 seconds]
UDP payload (24 bytes)
Data (24 bytes)
Data: 4c73c49d9914dd7650aaa49f4c72b638c8d4e94486037394
[Length: 24]
0000 34 6f 24 c5 a8 f5 a6 fc 22 a3 57 9d 86 dd 60 00 4o$ ···· "·W···`·
0010 00 00 00 20 11 39 20 01 48 60 48 02 00 38 00 00 ··· ·9 · H`H··8··
0020 00 00 00 00 01 81 24 09 40 c2 10 2a 99 e2 08 db ·····$· @··*····
0030 ab 62 b2 cb c9 fb 01 bb fd 7b 00 20 e3 3a 4c 73 ·b····· ·{· ·:Ls
0040 c4 9d 99 14 dd 76 50 aa a4 9f 4c 72 b6 38 c8 d4 ·····vP· ··Lr·8··
0050 e9 44 86 03 73 94 ·D··s·
Wi-Fi: <live capture in progress>
Packets: 1031 · Displayed: 196 (19.0%)
Profile: Default

IPv4 Analysis using Wireshark

Internet Protocol version 4 (IPv4) is the fourth version of the Internet Protocol (IP). It is one of the core protocols of standards-based internetworking methods on the Internet. Wireshark can be used to dissect and analyze IPv4 packets effectively, providing valuable insights about the network traffic.

To analyze IPv4 packets, follow the same initial steps as for TCP or UDP analysis, but in the display filter bar, enter `ip` to display only IPv4 packets.

The IPv4 header contains crucial information about the packet, including the source and destination IP addresses, the protocol used (TCP, UDP, ICMP, etc.), and other parameters like TTL (Time to Live), flags, and header checksum.

The source and destination IP addresses provide information about who is sending the packet and to whom. The protocol field indicates the protocol used by the data portion of the IP packet. The TTL field is used to avoid packet loops in the network, decrementing the value by one for each router crossed by the packet. The flag field is used for fragmentation and reassembly of packets, and the header checksum is used for error checking of the header.

By analyzing these fields, users can gain insights into their network's behavior, aiding them to troubleshoot issues effectively and optimize network performance efficiently. For a more detailed view of an IPv4 packet, select a packet from the packet list to display the packet details below. This will provide a comprehensive view of the IPv4 header fields.

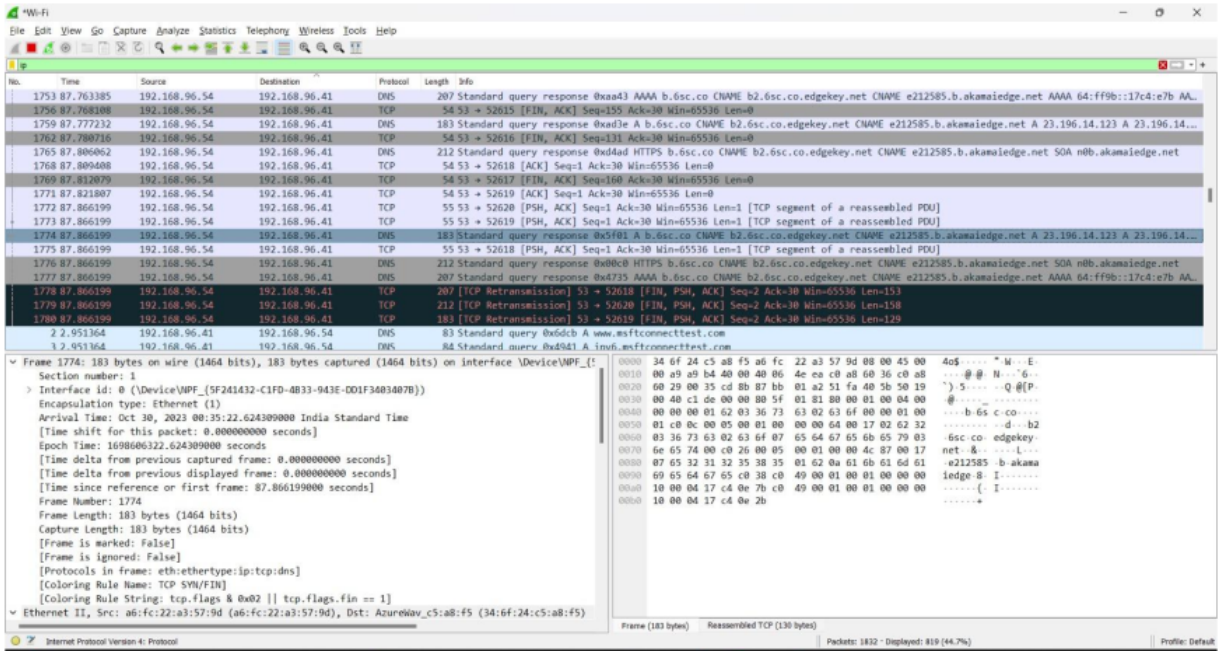
Please note that unlike TCP and UDP, IP is a network layer protocol and does not have port numbers. Therefore, the analysis of IP packets focuses on network layer information. However, the encapsulated transport layer protocol (TCP, UDP, etc.) can be further analyzed by digging into the packet content.

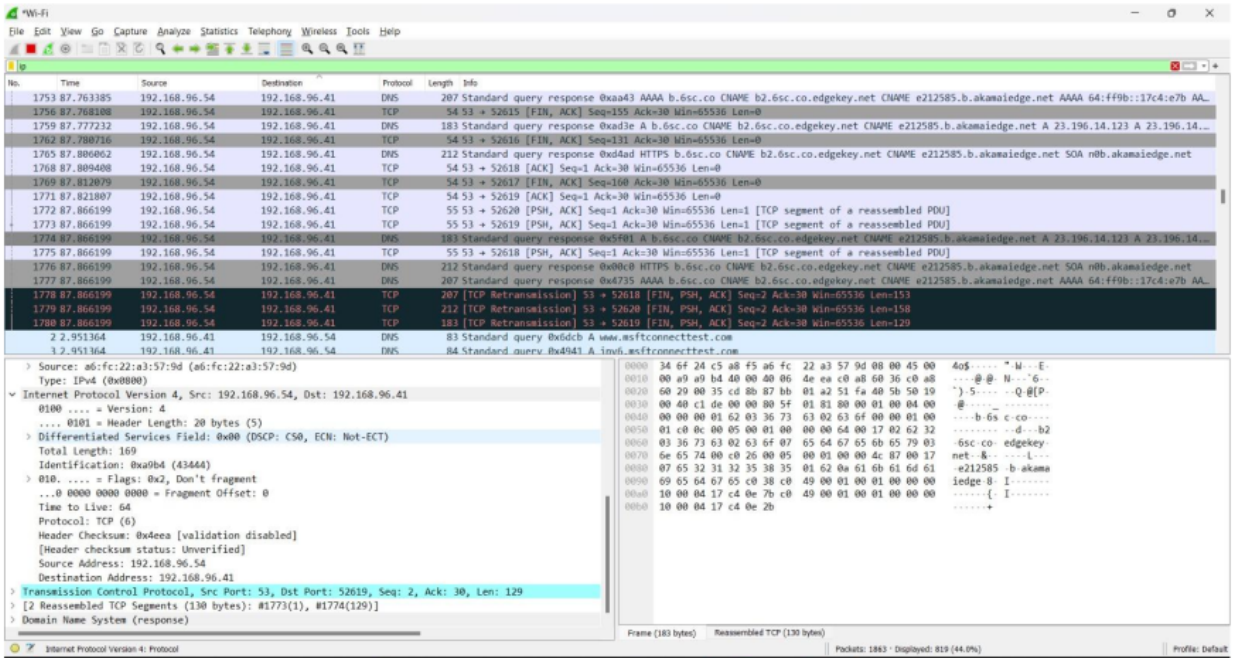
Wireshark allows you to view any potential issues with the IPv4 traffic, such as errors or packet loss. It also provides a graphical representation of the communication flow between systems, which can help identify potential issues or anomalies in the IPv4 traffic.

IPv4, or Internet Protocol Version 4, Its characteristics include:

- It uses a 32-bit address scheme allowing for a total of just over 4 billion addresses.

- Addresses are divided into five classes: A, B, C, D, and E.
- It uses a dotted decimal notation, meaning addresses appear as four sets of numbers separated by periods.
- It supports unicast, broadcast, and multicast operations.
- It includes fields for options such as security, record route, timestamp, and more.
- IPv4 headers include a checksum.
- It is connectionless, where each packet is handled independently from others.
- It allows for fragmentation of packets, which can then be reassembled.




*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
No. Time Source Destination Protocol Length Info
1753 87.763385 192.168.96.54 192.168.96.41 DNS 207 Standard query response 0xaa43 AAAA b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net AAAA 64:ff9b::17c4:e7b AA…
1756 87.768108 192.168.96.54 192.168.96.41 TCP 54 53 → 52615 [FIN, ACK] Seq=155 Ack=30 Win=65536 Len=0
1759 87.777232 192.168.96.54 192.168.96.41 DNS 183 Standard query response 0xad3e A b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net A 23.196.14.123 A 23.196.14.…
1762 87.780716 192.168.96.54 192.168.96.41 TCP 54 53 → 52616 [FIN, ACK] Seq=131 Ack=30 Win=65536 Len=0
1765 87.806062 192.168.96.54 192.168.96.41 DNS 212 Standard query response 0xd4ad HTTPS b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net SOA n0b.akamaiedge.net
1768 87.809408 192.168.96.54 192.168.96.41 TCP 54 53 → 52618 [ACK] Seq=1 Ack=30 Win=65536 Len=0
1769 87.812079 192.168.96.54 192.168.96.41 TCP 54 53 → 52617 [FIN, ACK] Seq=160 Ack=30 Win=65536 Len=0
1771 87.821807 192.168.96.54 192.168.96.41 TCP 54 53 → 52619 [ACK] Seq=1 Ack=30 Win=65536 Len=0
1772 87.866199 192.168.96.54 192.168.96.41 TCP 55 53 → 52620 [PSH, ACK] Seq=1 Ack=30 Win=65536 Len=1 [TCP segment of a reassembled PDU]
1773 87.866199 192.168.96.54 192.168.96.41 TCP 55 53 → 52619 [PSH, ACK] Seq=1 Ack=30 Win=65536 Len=1 [TCP segment of a reassembled PDU]
1774 87.866199 192.168.96.54 192.168.96.41 DNS 183 Standard query response 0x5f01 A b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net A 23.196.14.123 A 23.196.14.…
1775 87.866199 192.168.96.54 192.168.96.41 TCP 55 53 → 52618 [PSH, ACK] Seq=1 Ack=30 Win=65536 Len=1 [TCP segment of a reassembled PDU]
1776 87.866199 192.168.96.54 192.168.96.41 DNS 212 Standard query response 0x00c0 HTTPS b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net SOA n0b.akamaiedge.net
1777 87.866199 192.168.96.54 192.168.96.41 DNS 207 Standard query response 0x4735 AAAA b.6sc.co CNAME b2.6sc.co.edgekey.net CNAME e212585.b.akamaiedge.net AAAA 64:ff9b::17c4:e7b AA…
1778 87.866199 192.168.96.54 192.168.96.41 TCP 207 [TCP Retransmission] 53 → 52618 [FIN, PSH, ACK] Seq=2 Ack=30 Win=65536 Len=153
1779 87.866199 192.168.96.54 192.168.96.41 TCP 212 [TCP Retransmission] 53 → 52620 [FIN, PSH, ACK] Seq=2 Ack=30 Win=65536 Len=158
1780 87.866199 192.168.96.54 192.168.96.41 TCP 183 [TCP Retransmission] 53 → 52619 [FIN, PSH, ACK] Seq=2 Ack=30 Win=65536 Len=129
2 2.951364 192.168.96.41 192.168.96.54 DNS 83 Standard query 0x6dcb A www.msftconnecttest.com
3 2.951364 192.168.96.41 192.168.96.54 DNS 84 Standard query 0x4941 A ipv6.msftconnecttest.com
Source: a6:fc:22:a3:57:9d (a6:fc:22:a3:57:9d)
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.96.54, Dst: 192.168.96.41
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 169
Identification: 0xa9b4 (43444)
010. .... = Flags: 0x2, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: TCP (6)
Header Checksum: 0x4eea [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.96.54
Destination Address: 192.168.96.41
Transmission Control Protocol, Src Port: 53, Dst Port: 52619, Seq: 2, Ack: 30, Len: 129
[2 Reassembled TCP Segments (130 bytes): #1773(1), #1774(129)]
Domain Name System (response)
0000 34 6f 24 c5 a8 f5 a6 fc 22 a3 57 9d 08 00 45 00 4o$····· "·W···E·
0010 00 a9 a9 b4 40 00 40 06 4e ea c0 a8 60 36 c0 a8 ····@·@· N···`6··
0020 60 29 00 35 cd 8b 87 bb 01 a2 51 fa 40 5b 50 19 `).5···· ··Q·@[P·
0030 00 40 c1 de 00 00 80 5f 01 81 80 00 01 00 04 00 ·@····· _·······
0040 00 00 00 01 62 03 36 73 63 02 63 6f 00 00 01 00 ····b·6s c·co····
0050 01 c0 0c 00 05 00 01 00 00 00 64 00 17 02 62 32 ·····d··b2
0060 03 36 73 63 02 63 6f 07 65 64 67 65 6b 65 79 03 ·6sc·co· edgekey·
0070 6e 65 74 00 c0 26 00 05 00 01 00 00 4c 87 00 17 net··&·· ····L···
0080 07 65 32 31 32 35 38 35 01 62 0a 61 6b 61 6d 61 ·e212585 ·b·akama
0090 69 65 64 67 65 c0 38 c0 49 00 01 00 01 00 00 00 iedge·8· I·······
00a0 10 00 04 17 c4 0e 7b c0 49 00 01 00 01 00 00 00 ·····{· I·······
00b0 10 00 04 17 c4 0e 2b ·····+
Frame (183 bytes)
Reassembled TCP (130 bytes)
Internet Protocol Version 4: Protocol
Packets: 1863 · Displayed: 819 (44.0%)
Profile: Default