

## Analyze your first packet with Wireshark

### Scenario

In this scenario, you're a security analyst investigating traffic to a website. You'll analyze a network packet capture file that contains traffic data related to a user connecting to an internet site. The ability to filter network traffic using packet sniffers to gather relevant information is an essential skill as a security analyst.

You must filter the data in order to:

- identify the source and destination IP addresses involved in this web browsing session,
- examine the protocols that are used when the user makes the connection to the website, and
- analyze some of the data packets to identify the type of information sent and received by the systems that connect to each other when the network data is captured.

### Task 1. Apply a basic Wireshark filter and inspect a packet

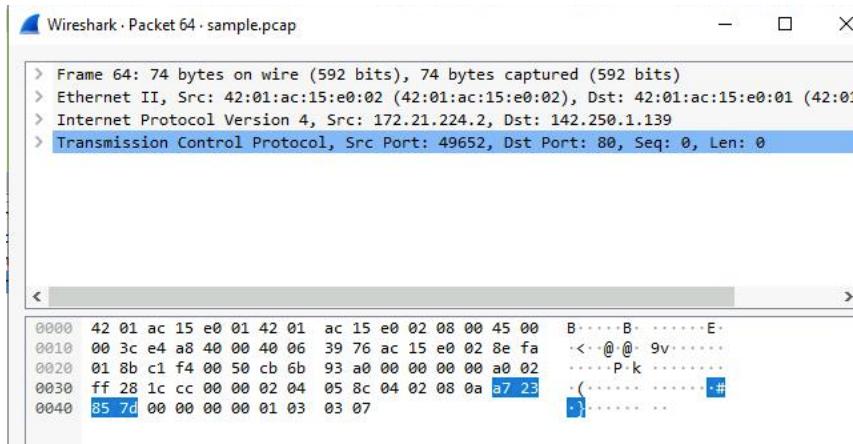
#### 1.1. Filter for traffic associated with a specific IP address.

ip.addr == 142.250.1.139

The screenshot shows the Wireshark interface with a packet list window. The filter bar at the top displays "ip.addr == 142.250.1.139". The packet list shows several frames, mostly ICMP Echo requests and replies between two hosts (172.21.224.2 and 142.250.1.139). A green highlight covers the entire list of frames. Below the list, the packet details and bytes panes show the structure of frame 16, which is an ICMP request. The details pane shows fields like Source MAC, Destination MAC, Source IP, Destination IP, Protocol, Length, and Info. The bytes pane shows the raw hex and ASCII data of the frame.

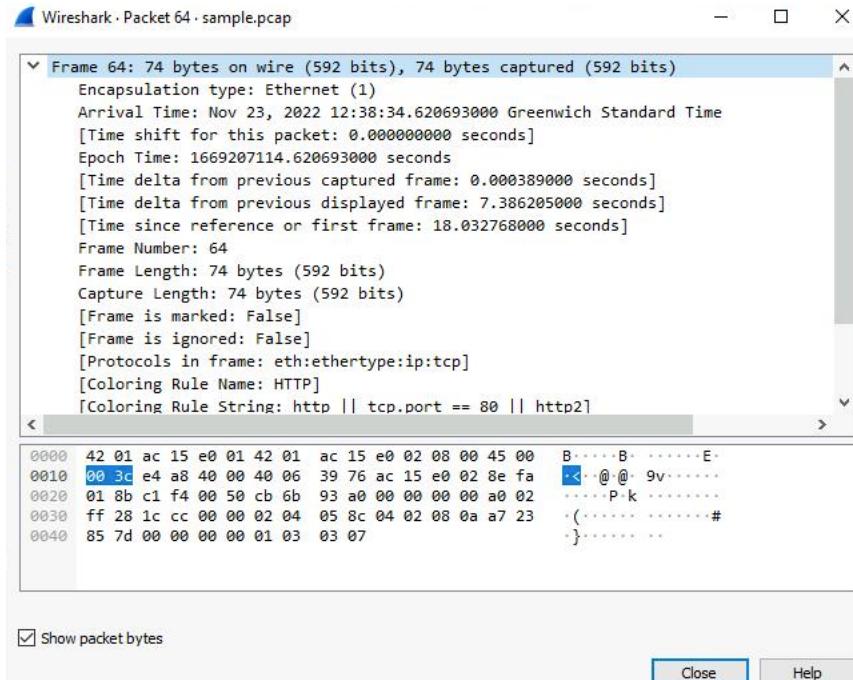
No.	Time	Source	Destination	Protocol	Length	Info
16	8.642690	172.21.224.2	142.250.1.139	ICMP	98	Echo (ping) request id=0x6831
18	8.643923	142.250.1.139	172.21.224.2	ICMP	98	Echo (ping) reply id=0x6831
25	9.644712	172.21.224.2	142.250.1.139	ICMP	98	Echo (ping) request id=0x6831
26	9.645078	142.250.1.139	172.21.224.2	ICMP	98	Echo (ping) reply id=0x6831
31	10.646049	172.21.224.2	142.250.1.139	ICMP	98	Echo (ping) request id=0x6831
32	10.646563	142.250.1.139	172.21.224.2	ICMP	98	Echo (ping) reply id=0x6831
64	18.032768	172.21.224.2	142.250.1.139	TCP	74	49652 → 80 [SYN] Seq=0 Win=653
65	18.034210	142.250.1.139	172.21.224.2	TCP	74	80 → 49652 [SYN, ACK] Seq=0 Ack=1
66	18.034238	172.21.224.2	142.250.1.139	TCP	66	49652 → 80 [ACK] Seq=1 Ack=1 W
67	18.034291	172.21.224.2	142.250.1.139	HTTP	151	GET / HTTP/1.1
68	18.034724	142.250.1.139	172.21.224.2	TCP	66	80 → 49652 [ACK] Seq=1 Ack=86
69	18.036927	142.250.1.139	172.21.224.2	HTTP	648	HTTP/1.1 301 Moved Permanently
70	18.036941	172.21.224.2	142.250.1.139	TCP	66	49652 → 80 [ACK] Seq=86 Ack=58

## 1.2.Double-click the first packet that lists **TCP** as the protocol.



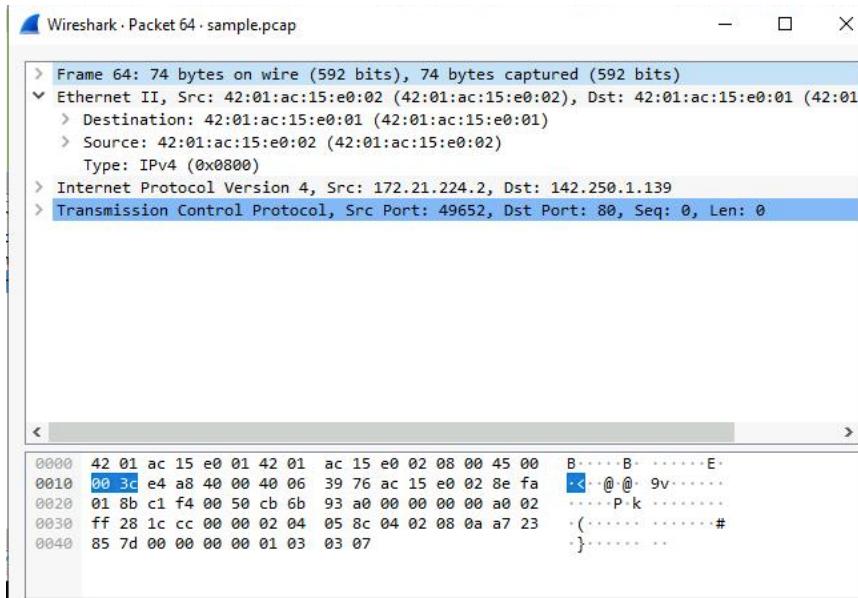
The upper section of this window contains subtrees where Wireshark will provide you with an analysis of the various parts of the network packet. The lower section of the window contains the raw packet data displayed in hexadecimal and ASCII text. There is also placeholder text for fields where the character data does not apply, as indicated by the dot (".").

## 1.3.Double-click the first subtree in the upper section. This starts with the word **Frame**.



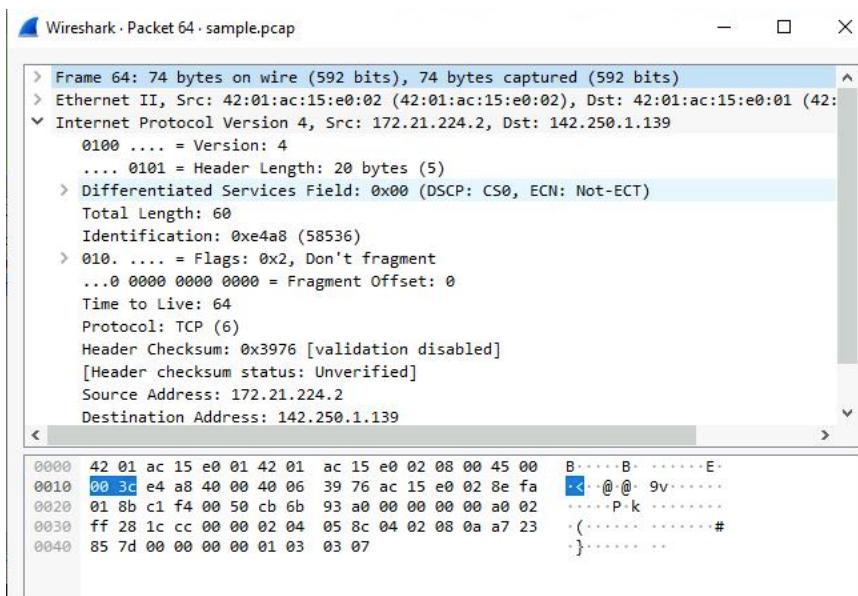
This provides you with details about the overall network packet, or frame, including the frame length and the arrival time of the packet. At this level, you're viewing information about the entire packet of data.

1.4. Double-click **Frame** again to collapse the subtree and then double-click the **Ethernet II** subtree.



This item contains details about the packet at the Ethernet level, including the source and destination MAC addresses and the type of internal protocol that the Ethernet packet contains.

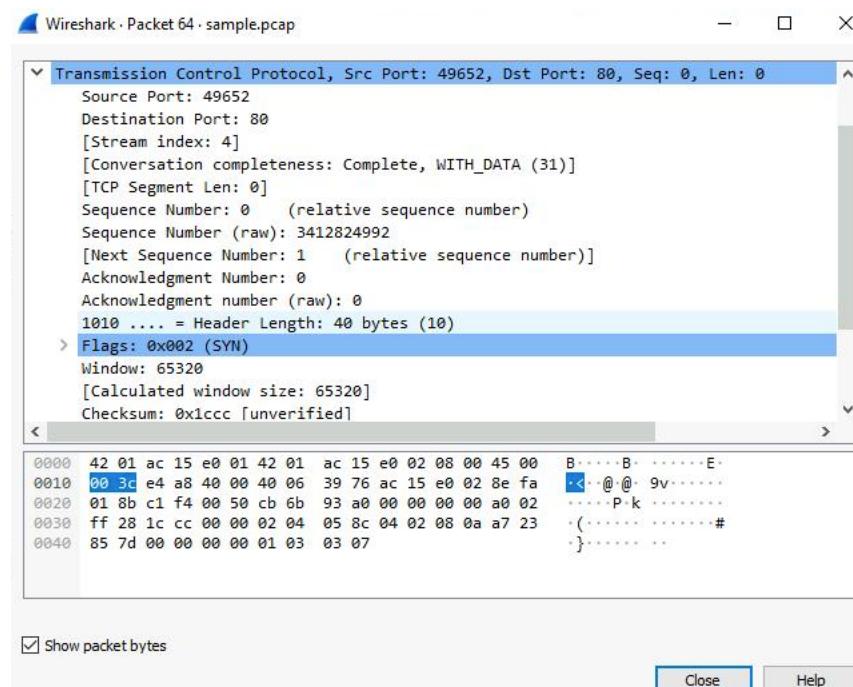
1.5. Double-click **Ethernet II** again to collapse that subtree and then double-click the **Internet Protocol Version 4** subtree.



This provides packet data about the Internet Protocol (IP) data contained in the Ethernet packet. It contains information such as the source and destination IP addresses and the Internal Protocol (for example, TCP or UDP), which is carried inside the IP packet.

The source and destination IP addresses shown here match the source and destination IP addresses in the summary display for this packet in the main Wireshark window.

1.6. Double-click **Internet Protocol Version 4** again to collapse that subtree and then double-click the **Transmission Control Protocol** subtree.

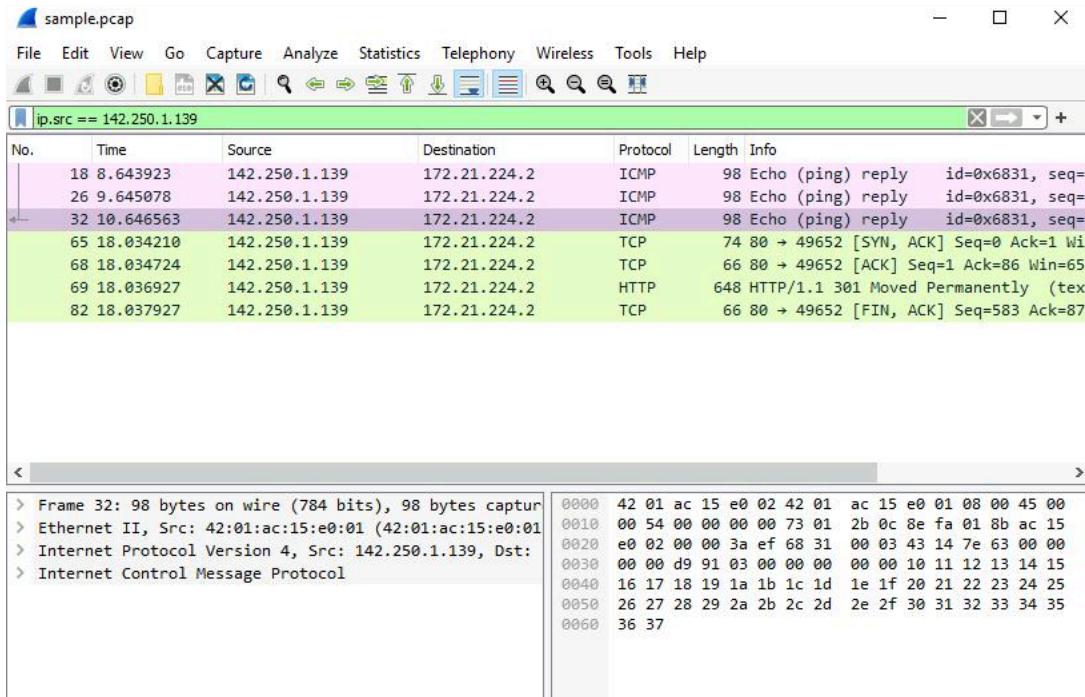


This provides detailed information about the TCP packet, including the source and destination TCP ports, the TCP sequence numbers, and the TCP flags.

## Task 2. Use filters to select packets

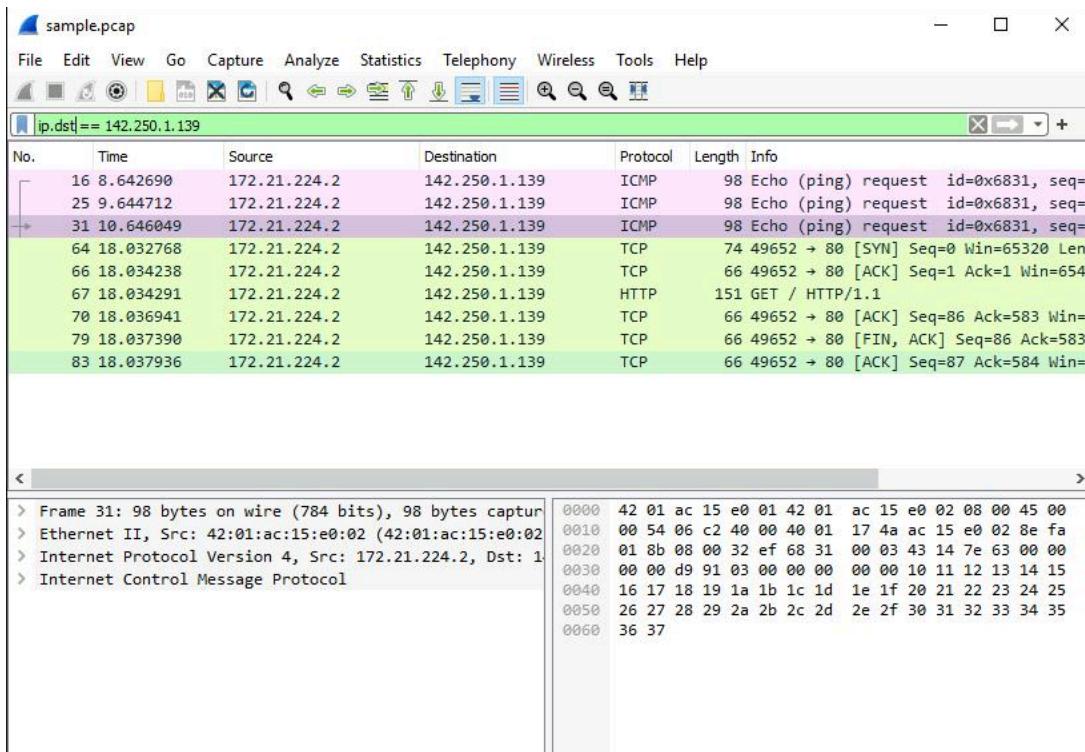
2.1. Filter to select traffic for a specific source IP address only.

ip.src == 142.250.1.139



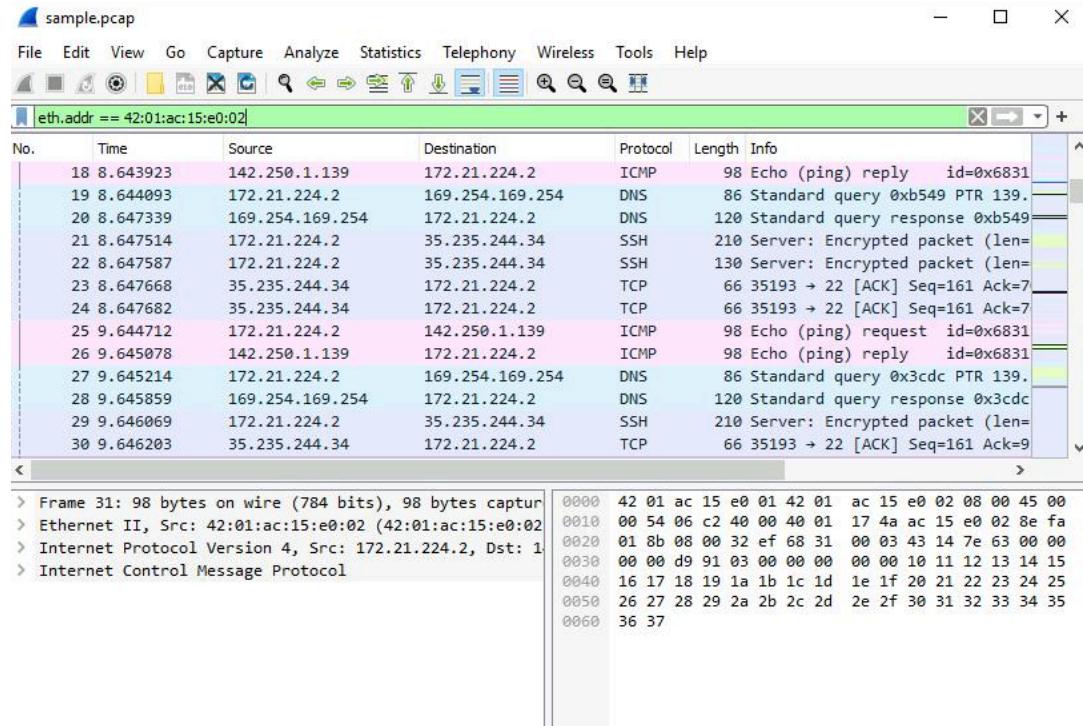
## 2.2. Filter to select traffic for a specific destination IP address only.

ip.dst == 142.250.1.139



2.3. Filter to select traffic to or from a specific Ethernet MAC address. This filters traffic related to one MAC address, regardless of the other protocols involved:

eth.addr == 42:01:ac:15:e0:02



2.4. Double-click the **Ethernet II** subtree if it is not already open.

The MAC address you specified in the filter is listed as either the source or destination address in the expanded Ethernet II subtree.

eth.addr == 42:01:ac:15:e0:02

No.	Time	Source	Destination	Protocol	Length	Info
18	8.643923	142.250.1.139	172.21.224.2	ICMP	98	Echo (ping) reply id=0x6831
19	8.644093	172.21.224.2	169.254.169.254	DNS	86	Standard query 0xb549 PTR 139.
20	8.647339	169.254.169.254	172.21.224.2	DNS	120	Standard query response 0xb549
21	8.647514	172.21.224.2	35.235.244.34	SSH	210	Server: Encrypted packet (len=
22	8.647587	172.21.224.2	35.235.244.34	SSH	130	Server: Encrypted packet (len=
23	8.647668	35.235.244.34	172.21.224.2	TCP	66	35193 → 22 [ACK] Seq=161 Ack=7
24	8.647682	35.235.244.34	172.21.224.2	TCP	66	35193 → 22 [ACK] Seq=161 Ack=7
25	9.644712	172.21.224.2	142.250.1.139	ICMP	98	Echo (ping) request id=0x6831
26	9.645078	142.250.1.139	172.21.224.2	ICMP	98	Echo (ping) reply id=0x6831
27	9.645214	172.21.224.2	169.254.169.254	DNS	86	Standard query 0x3cdc PTR 139.
28	9.645859	169.254.169.254	172.21.224.2	DNS	120	Standard query response 0x3cdc
29	9.646069	172.21.224.2	35.235.244.34	SSH	210	Server: Encrypted packet (len=
30	9.646203	35.235.244.34	172.21.224.2	TCP	66	35193 → 22 [ACK] Seq=161 Ack=9

```
> Frame 31: 98 bytes on wire (784 bits), 98 bytes captured
└─ Ethernet II, Src: 42:01:ac:15:e0:02 (42:01:ac:15:e0:02)
    └─ Destination: 42:01:ac:15:e0:01 (42:01:ac:15:e0:01)
    └─ Source: 42:01:ac:15:e0:02 (42:01:ac:15:e0:02)
        Type: IPv4 (0x0800)
    └─ Internet Protocol Version 4, Src: 172.21.224.2, Dst: 142.250.1.139
    └─ Internet Control Message Protocol
```

0000	42 01 ac 15 e0 01	42 01 ac 15 e0 02	08 00 45 00
0010	00 54 06 c2 40 00	40 01 17 4a ac 15 e0 02	8e fa
0020	01 8b 08 00 32 ef	68 31 00 03 43 14 7e 63 00 00	00
0030	00 00 d9 91 03 00 00 00	00 00 10 11 12 13 14 15	00
0040	16 17 18 19 1a 1b 1c 1d	1e 1f 20 21 22 23 24 25	25
0050	26 27 28 29 2a 2b 2c 2d	2e 2f 30 31 32 33 34 35	35
0060	36 37		

2.5. Double-click the **Internet Protocol Version 4** subtree to expand it and scroll down until the **Time to Live** and **Protocol** fields appear.

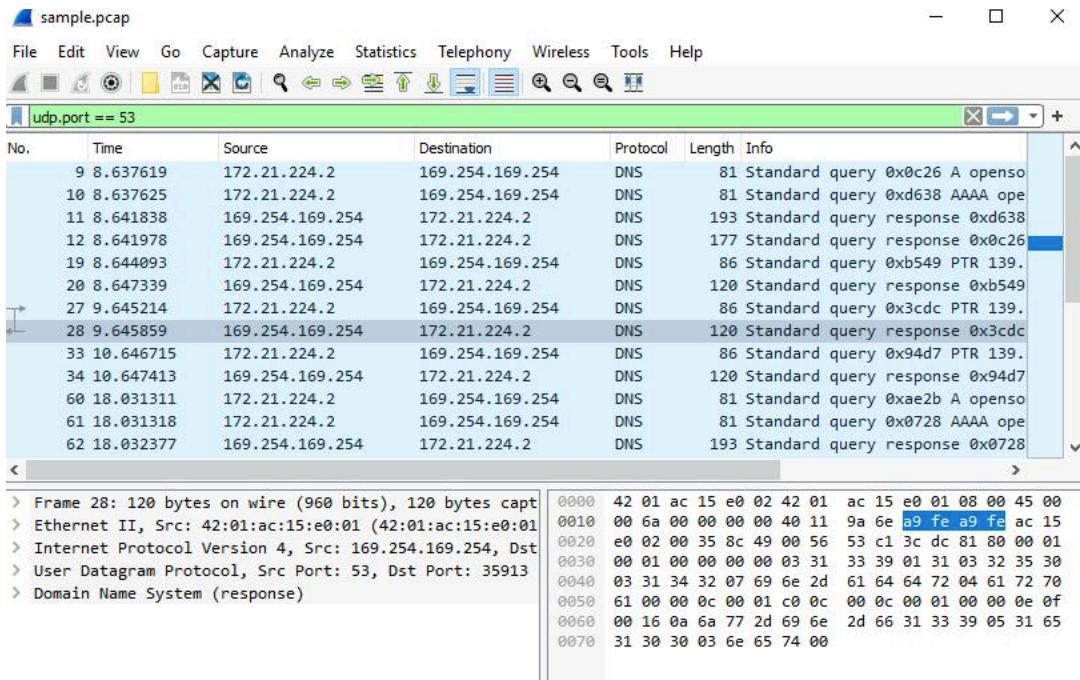
0100 .... = Version: 4	^
.... 0101 = Header Length: 20 bytes (5)	
> Differentiated Services Field: 0x00 (DSCP: CS0, E	
Total Length: 84	
Identification: 0x06c2 (1730)	
010. .... = Flags: 0x2, Don't fragment	
...0 0000 0000 0000 = Fragment Offset: 0	
Time to Live: 64	
Protocol: ICMP (1)	
Header Checksum: 0x174a [validation disabled]	
[Header checksum status: Unverified]	
Source Address: 172.21.224.2	
Destination Address: 142.250.1.139	
> Internet Control Message Protocol	v

0000	42 01 ac 15 e0 01	42 01 ac 15 e0 02	08 00 45 00
0010	00 54 06 c2 40 00	40 01 17 4a ac 15 e0 02	8e fa
0020	01 8b 08 00 32 ef	68 31 00 03 43 14 7e 63 00 00	00
0030	00 00 d9 91 03 00 00 00	00 00 10 11 12 13 14 15	00
0040	16 17 18 19 1a 1b 1c 1d	1e 1f 20 21 22 23 24 25	25
0050	26 27 28 29 2a 2b 2c 2d	2e 2f 30 31 32 33 34 35	35
0060	36 37		

## Task 3. Use filters to explore DNS packets

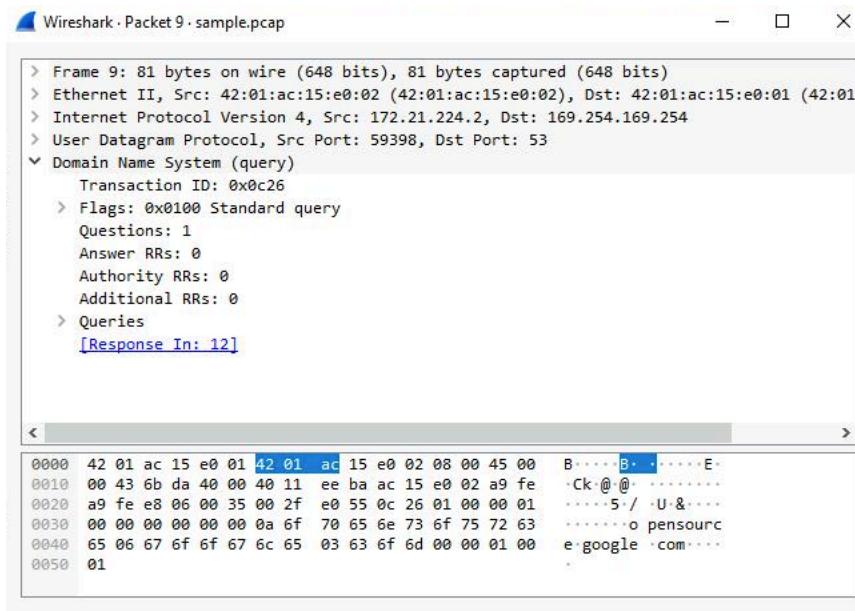
3.1. Filter to select UDP port 53 traffic. DNS traffic uses UDP port 53, so this will list traffic related to DNS queries and responses only

udp.port == 53

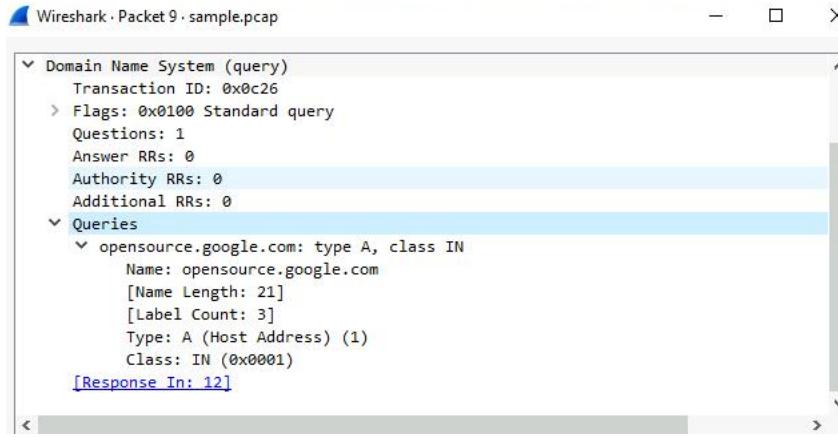


3.2. Double-click the first packet in the list to open the detailed packet window.

3.3. Scroll down and double-click the **Domain Name System (query)** subtree to expand it.



3.4. Scroll down and double-click **Queries**.



## Task 4. Use filters to explore TCP packets

4.1. Filter to select TCP port 80 traffic. TCP port 80 is the default port that is associated with web traffic:

tcp.port == 80

File	Edit	View	Go	Capture	Analyze	Statistics	Telephony	Wireless	Tools	Help
tcp.port == 80										
No.	Time	Source	Destination	Protocol	Length	Info				
37	10.799238	172.21.224.2	169.254.169.254	TCP	54	56664 → 80 [ACK] Seq=1 Ack=1 W				
38	10.799668	169.254.169.254	172.21.224.2	TCP	54	[TCP ACKed unseen segment] 80				
52	16.943231	172.21.224.2	169.254.169.254	TCP	54	41208 → 80 [ACK] Seq=1 Ack=1 W				
53	16.943758	169.254.169.254	172.21.224.2	TCP	54	[TCP ACKed unseen segment] 80				
64	18.032768	172.21.224.2	142.250.1.139	TCP	74	49652 → 80 [SYN] Seq=0 Win=653				
65	18.034210	142.250.1.139	172.21.224.2	TCP	74	80 → 49652 [SYN, ACK] Seq=0 Ac				
66	18.034238	172.21.224.2	142.250.1.139	TCP	66	49652 → 80 [ACK] Seq=1 Ack=1 W				
67	18.034291	172.21.224.2	142.250.1.139	HTTP	151	GET / HTTP/1.1				
68	18.034724	142.250.1.139	172.21.224.2	TCP	66	80 → 49652 [ACK] Seq=1 Ack=86				
69	18.036927	142.250.1.139	172.21.224.2	HTTP	648	HTTP/1.1 301 Moved Permanently				
70	18.036941	172.21.224.2	142.250.1.139	TCP	66	49652 → 80 [ACK] Seq=86 Ack=58				
79	18.037390	172.21.224.2	142.250.1.139	TCP	66	49652 → 80 [FIN, ACK] Seq=86 A				
82	18.037927	142.250.1.139	172.21.224.2	TCP	66	80 → 49652 [FIN, ACK] Seq=583				

Frame 37: 54 bytes on wire (432 bits), 54 bytes captured  
 Ethernet II, Src: 42:01:ac:15:e0:02 (42:01:ac:15:e0:02) | 0000 42 01 ac 15 e0 01 42 01 ac 15 e0 02 08 00 45 00  
 | 0010 00 28 89 a2 40 00 40 06 d1 18 ac 15 e0 02 a9 fe

4.2. Filter to select TCP packet data that contains specific text data.

tcp contains "curl"

