Task 1: Ping PDU Capture

Step 2: Examine the Packet List pane.

What protocol is used by ping?

- ICMP

What is the full protocol name?

- Internet Protocol Version 4, Internet Control Message Protocol

What are the names of the two ping messages?

- *Echo (ping) request* and *Echo (ping) reply*

Are the listed source and destination IP addresses what you expected? Why?

- Yes, they are as I expected. The source IP address is similar to my IP address and the destination address is a public IP address.

Step 3: Select the first echo request packet on the list.

Locate the two different types of "Source" and "Destination". Why are there two types?

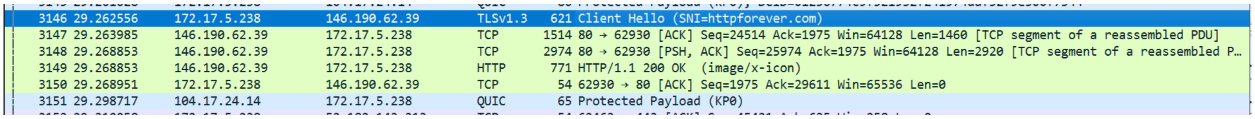
- The first Source and Destination is the IP Address, and the second is the MAC Address. IP addresses are for computers to communicate with each other while the MAC addresses are for communication in the local network.

What protocols are in the Ethernet frame?

- In the particular ping request that I am viewing, the protocol in the Ethernet frame is of type IPv4 (0x0800). Other data packets have ARP (0x0806).

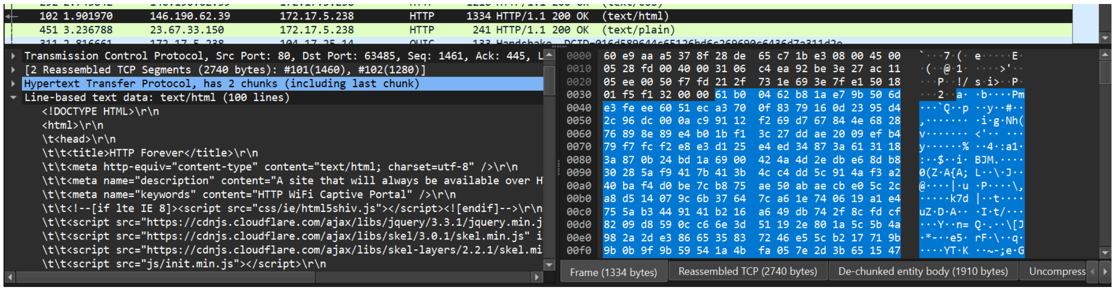
Task 2: HTTP PDU Capture

Step 2: Scroll through the PDUs listed.



```
3146 29.262556   172.17.5.238    146.190.62.39   TLSv1.3   621 Client Hello (SNI=httpforever.com)
3147 29.263985   146.190.62.39   172.17.5.238    TCP      1514 80 → 62930 [ACK] Seq=24514 Ack=1975 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
3148 29.268853   146.190.62.39   172.17.5.238    TCP      2974 80 → 62930 [PSH, ACK] Seq=25974 Ack=1975 Win=64128 Len=2920 [TCP segment of a reassembled P…
3149 29.268853   146.190.62.39   172.17.5.238    HTTP      771 HTTP/1.1 200 OK  (image/x-icon)
3150 29.268951   172.17.5.238    146.190.62.39   TCP        54 62930 → 80 [ACK] Seq=1975 Ack=29611 Win=65536 Len=0
3151 29.298717   104.17.24.14    172.17.5.238    QUIC       65 Protected Payload (KP0)
```

Step 3: In the Packet List pane, highlight an HTTP packet that has the notation "(text/html)" in the Info column.



```
 102 1.901970   146.190.62.39   172.17.5.238   HTTP   1334 HTTP/1.1 200 OK  (text/html)
 451 3.236788   23.67.33.150    172.17.5.238   HTTP    241 HTTP/1.1 200 OK  (text/plain)
```

```
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 63485, Seq: 1461, Ack: 445, L
▸ [2 Reassembled TCP Segments (2740 bytes): #101(1460), #102(1280)]
▸ Hypertext Transfer Protocol, has 2 chunks (including last chunk)
▾ Line-based text data: text/html (100 lines)
    <!DOCTYPE HTML>\r\n
    <html>\r\n
    \t<head>\r\n
    \t\t<title>HTTP Forever</title>\r\n
    \t\t<meta http-equiv="content-type" content="text/html; charset=utf-8" />\r\n
    \t\t<meta name="description" content="A site that will always be available over H
    \t\t<meta name="keywords" content="HTTP WiFi Captive Portal" />\r\n
    \t\t<!--[if lte IE 8]><script src="css/ie/html5shiv.js"></script><![endif]-->\r\n
    \t\t<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.j
    \t\t<script src="https://cdnjs.cloudflare.com/ajax/libs/skel/3.0.1/skel.min.js" i
    \t\t<script src="https://cdnjs.cloudflare.com/ajax/libs/skel-layers/2.2.1/skel.mi
    \t\t<script src="js/init.min.js"></script>\r\n
```

```
0000  60 e9 aa a5 37 8f 28 de  65 c7 1b e3 08 00 45 00   `···7·(· e·····E·
0010  05 28 fd 00 40 00 31 06  c4 ea 92 be 3e 27 ac 11   ·(··@·1· ····>'··
0020  05 ee 00 50 f7 fd 21 2f  73 1e 69 3e 7f e1 50 18   ···P··!/ s·i>··P·
0030  01 f5 f1 32 00 00 61 b0  04 62 b8 1a e7 9b 50 6d   ···2··a· ·b····Pm
0040  e3 fe ee 60 51 ec a3 70  0f 83 79 16 0d 23 95 d4   ···`Q··p ··y··#··
0050  2c 96 dc 00 0a c9 91 12  f2 69 d7 67 84 4e 68 28   ,······· ·i·g·Nh(
0060  76 89 8e 89 e4 b0 1b f1  3c 27 dd ae 20 09 ef b4   v······· <'·· ···
0070  79 f7 fc f2 e8 e3 d1 25  e4 ed 34 87 3a 61 31 18   y······% ··4·:a1·
0080  3a 87 0b 24 bd 1a 69 00  42 4a 4d 2e db e6 8d b8   :··$··i· BJM.····
0090  30 28 5a f9 41 7b 41 3b  4c c4 dd 5c 91 4a f3 a2   0(Z·A{A; L··\·J··
00a0  40 ba f4 d0 be 7c b8 75  ae 50 ab ae cb e0 5c 2c   @····|·u ·P····\,
00b0  a8 d5 14 07 9c 6b 37 64  7c a6 1e 74 06 19 a1 e4   ·····k7d |··t····
00c0  75 5a b3 44 91 41 b2 16  a6 49 db 74 2f 8c fd cf   uZ·D·A·· ·I·t/···
00d0  82 09 d8 59 0c c6 6e 3d  51 19 2e 80 1a 5c 5b 4a   ···Y··n= Q·.··\[J
00e0  98 2a 2d e3 86 65 35 83  72 46 e5 5c b2 17 71 9b   ·*-··e5· rF·\··q·
00f0  9b 0b 9f 9b 59 54 1a 4b  fa 05 7e 2d 3b 65 15 47   ····YT·K ··~-;e·G
```

Frame (1334 bytes)
Reassembled TCP (2740 bytes)
De-chunked entity body (1910 bytes)
Uncompress

Task 4: Reflection

- Using Wireshark has helped me understand the difference between the data link layer and the network layer, along with the different protocols that each layer handles. These are important to understand if ever I will need to troubleshoot or analyze network behavior in the future if ever I get into this field of work.

Task 5: Challenge

Discuss how protocol analyzer:

1. Troubleshoot the failure of a webpage to download successfully to a browser on a computer.
   Protocol analyzers like Wireshark help in figuring out why a webpage isn't loading by looking at the data traveling between your computer and the web server, checking for problems like slow connections, errors in communication, or issues with the website itself.
2. Identify data traffic on a network that is required by users.
   Wireshark helps us find and focus on the internet activities users do, like web browsing or email, by looking at the data flowing through the network. Through this, it can be analyzed by cross-referencing them with user expectations and organizational requirements to confirm whether the observed network activities align with the intended and necessary user behaviors, ensuring that the network traffic meets organizational needs.