ECSE 308, Winter 2024
Introduction to Communication Systems and Networks
Laboratory L4 – Report

STUDENT 1
Name: Feiyang Huang
ID: 261022835

STUDENT 2
Name: Jingxuan Li
ID: 261013860

INSTRUCTIONS:

- Student(s) need to upload one report per team on *myCourses* before the due date.
- Upload a single, clearly readable pdf file, including this cover page plus answers.
- DUE DATE: Monday March 11, 5pm.

REPORT:

Part	Question	Mark
1	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
	11	
	12	
	13	
	14	
	15	
	16	

Part	Question	Mark
2	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
	11	
	12	
	13	
	14	
	15	

TOTAL:

	Student 1	Student 2
Participation	/20	/20
Report	/80	/80
TOTAL	/100	/100

McGill University
Montreal, Canada

IP & MAC

***Abstract*— This lab guides us to understanding and utilizing Wireshark for packet analysis, with a focus on Internet Protocol (IP) and Ethernet frameworks. It starts with a basic introduction to Wireshark, highlighting its role as a powerful tool for capturing and analyzing network packets, thus offering insights into the sequence of messages exchanged between protocols. It covers essential Wireshark functionalities, including the user interface, packet capturing process, and detailed examination of IP and ICMP protocols. The tutorial also delves into checksum calculations, echo/reply commands, route tracing, and the Address Resolution Protocol (ARP), providing a practical approach to network troubleshooting and analysis.**

Keywords—Wireshark, ICMP, IP, MAC, ARP

INTRODUCTION

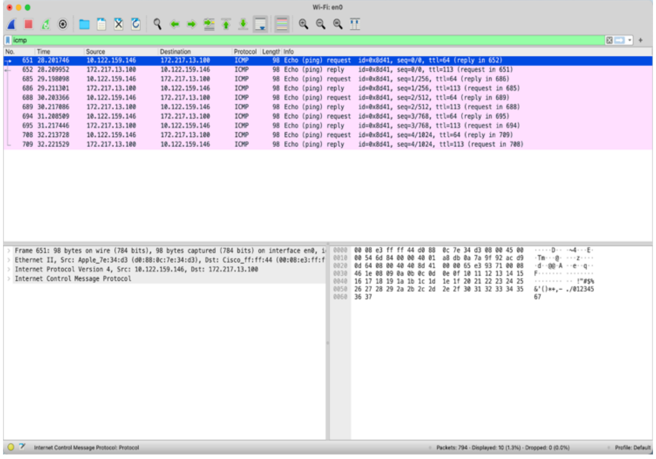
Wireshark is heralded as a pivotal tool in network analysis, offering users the capability to capture and meticulously examine the data traversing their networks. This lab introduces Wireshark, beginning with its user interface components, which are instrumental in navigating and utilizing the software effectively. It underscores the significance of the packet-listing window, packet-header details, and packet-content window, each serving a unique purpose in the analysis process. The tutorial progresses to outline the procedural steps for starting packet capture, emphasizing the application's versatility across different operating systems. The introduction to Internet Protocol (IP) and the subsequent exploration of ICMP protocols within the lab offers foundational knowledge, paving the way for detailed investigations into network behaviors, including routing path tracing with utilities like tracert/traceroute and ping. Through practical exercises and instructions, the lab aims to equip readers with the skills needed to dissect network activities, enhancing their understanding of network protocols and troubleshooting techniques.

Part 1: Internet Protocol (IP)

A. PING

After applying all the commands in the terminal and Wireshark, I can obtain the following figures and results.

Q1. What is the IP address of your host? What is the IP address of the destination host?



```
No.   Time       Source          Destination     Protocol Length Info
  651 28.201746  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=0/0, ttl=64 (reply in 652)
  652 28.209952  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=0/0, ttl=113 (request in 651)
  685 29.198098  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=1/256, ttl=64 (reply in 686)
  686 29.211301  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=1/256, ttl=113 (request in 685)
  688 30.203366  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=2/512, ttl=64 (reply in 689)
  689 30.217086  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=2/512, ttl=113 (request in 688)
  694 31.208509  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=3/768, ttl=64 (reply in 695)
  695 31.217446  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=3/768, ttl=113 (request in 694)
  708 32.213728  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=4/1024, ttl=64 (reply in 709)
  709 32.221529  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=4/1024, ttl=113 (request in 708)
```

```
> Frame 651: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, i
> Ethernet II, Src: Apple_7e:34:d3 (d0:88:0c:7e:34:d3), Dst: Cisco_ff:ff:44 (00:08:e3:ff:f
> Internet Protocol Version 4, Src: 10.122.159.146, Dst: 172.217.13.100
> Internet Control Message Protocol
```

```
0000  00 08 e3 ff ff 44 d0 88  0c 7e 34 d3 08 00 45 00   ·····D·· ·~4···E·
0010  00 54 6d 84 00 00 40 01  a8 db 0a 7a 9f 92 ac d9   ·Tm···@· ···z····
0020  0d 64 08 00 40 40 8d 41  00 00 65 e3 93 71 00 08   ·d··@@·A ··e··q··
0030  46 1e 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   F······· ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

Internet Control Message Protocol: Protocol — Packets: 794 · Displayed: 10 (1.3%) · Dropped: 0 (0.0%) — Profile: Default

Figure 1: Screenshot of Wireshark (PING)

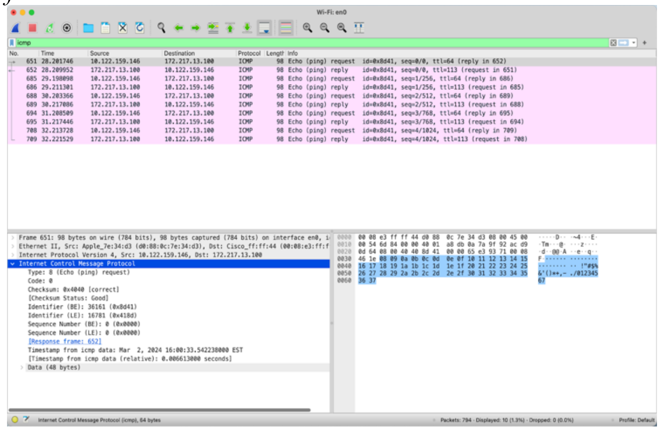
The IP address of my host is 10.122.159.146 (McGill Wi-Fi), and the IP address of the destination host is 172.217.13.100.

Q2. Why is it that an ICMP packet does not have source and destination port numbers?

The Internet Control Message Protocol (ICMP) does not have source and destination port numbers because it operates at the network layer of the OSI model, not the transport layer. Port numbers are a concept used in the transport layer to differentiate between different processes or services running on a computer.

ICMP is used for error reporting and diagnostic functions related to the network layer. It is used to send messages which are usually processed by the operating system or network device itself, rather than by a particular application. Therefore, the concept of port numbers is not applicable to ICMP packets.

Q3. Examine one of the ping request packets sent by your host. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?



```
No.   Time       Source          Destination     Protocol Length Info
  651 28.201746  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=0/0, ttl=64 (reply in 652)
  652 28.209952  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=0/0, ttl=113 (request in 651)
  685 29.198098  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=1/256, ttl=64 (reply in 686)
  686 29.211301  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=1/256, ttl=113 (request in 685)
  688 30.203366  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=2/512, ttl=64 (reply in 689)
  689 30.217086  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=2/512, ttl=113 (request in 688)
  694 31.208509  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=3/768, ttl=64 (reply in 695)
  695 31.217446  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=3/768, ttl=113 (request in 694)
  708 32.213728  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=4/1024, ttl=64 (reply in 709)
  709 32.221529  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=4/1024, ttl=113 (request in 708)
```

```
> Frame 651: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, i
> Ethernet II, Src: Apple_7e:34:d3 (d0:88:0c:7e:34:d3), Dst: Cisco_ff:ff:44 (00:08:e3:ff:f
> Internet Protocol Version 4, Src: 10.122.159.146, Dst: 172.217.13.100
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x4040 [correct]
    [Checksum Status: Good]
    Identifier (BE): 36161 (0x8d41)
    Identifier (LE): 16781 (0x418d)
    Sequence Number (BE): 0 (0x0000)
    Sequence Number (LE): 0 (0x0000)
    [Response frame: 652]
    Timestamp from icmp data: Mar  2, 2024 16:00:33.542238000 EST
    [Timestamp from icmp data (relative): 0.006613000 seconds]
  > Data (48 bytes)
```

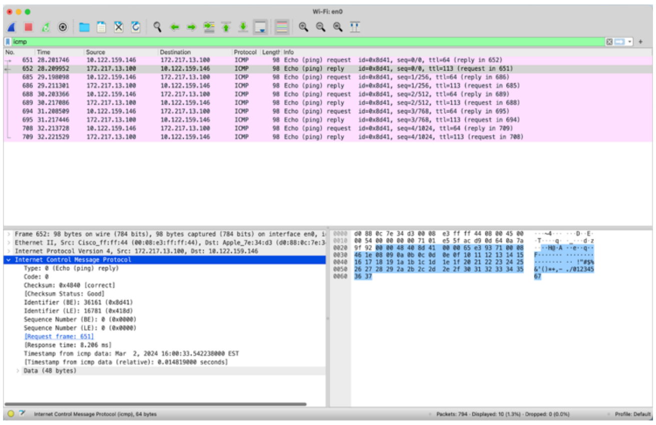
```
0000  00 08 e3 ff ff 44 d0 88  0c 7e 34 d3 08 00 45 00   ·····D·· ·~4···E·
0010  00 54 6d 84 00 00 40 01  a8 db 0a 7a 9f 92 ac d9   ·Tm···@· ···z····
0020  0d 64 08 00 40 40 8d 41  00 00 65 e3 93 71 00 08   ·d··@@·A ··e··q··
0030  46 1e 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   F······· ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

Internet Control Message Protocol (icmp), 64 bytes — Packets: 794 · Displayed: 10 (1.3%) · Dropped: 0 (0.0%) — Profile: Default

Figure 2: Screenshot of Wireshark (PING request)

The ICMP type number is 8, and the code number of it is 0. This ICMP packet also has the fields including Checksum, Identifier, and Sequence Number. The checksum, sequence number and identifier fields have the same size of fields which is 2 bytes.

Q4. Examine the corresponding ping reply packet. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?



```
Wi-Fi: en0
icmp
No.  Time       Source          Destination     Protocol Length Info
651  28.201746  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=0/0, ttl=64 (reply in 652)
652  28.209952  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=0/0, ttl=113 (request in 651)
685  29.198098  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=1/256, ttl=64 (reply in 686)
686  29.211301  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=1/256, ttl=113 (request in 685)
688  30.203366  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=2/512, ttl=64 (reply in 689)
689  30.217086  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=2/512, ttl=113 (request in 688)
694  31.208509  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=3/768, ttl=64 (reply in 695)
695  31.217446  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=3/768, ttl=113 (request in 694)
708  32.213728  10.122.159.146  172.217.13.100  ICMP     98 Echo (ping) request  id=0x8d41, seq=4/1024, ttl=64 (reply in 709)
709  32.221529  172.217.13.100  10.122.159.146  ICMP     98 Echo (ping) reply    id=0x8d41, seq=4/1024, ttl=113 (request in 708)

> Frame 652: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, i
> Ethernet II, Src: Cisco_ff:ff:44 (00:08:e3:ff:ff:44), Dst: Apple_7e:34:d3 (d0:88:0c:7e:3
> Internet Protocol Version 4, Src: 172.217.13.100, Dst: 10.122.159.146
v Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x4840 [correct]
    [Checksum Status: Good]
    Identifier (BE): 36161 (0x8d41)
    Identifier (LE): 16781 (0x418d)
    Sequence Number (BE): 0 (0x0000)
    Sequence Number (LE): 0 (0x0000)
    [Request frame: 651]
    [Response time: 8.206 ms]
    Timestamp from icmp data: Mar  2, 2024 16:00:33.542238000 EST
    [Timestamp from icmp data (relative): 0.014819000 seconds]
  > Data (48 bytes)

0000  d0 88 0c 7e 34 d3 00 08  e3 ff ff 44 08 00 45 00   ···~4··· ···D··E·
0010  00 54 00 00 00 00 71 01  e5 5f ac d9 0d 64 0a 7a   ·T····q· ·_···d·z
0020  9f 92 00 00 48 40 8d 41  00 00 65 e3 93 71 00 08   ····H@·A ··e··q··
0030  46 1e 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   F······· ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67

Internet Control Message Protocol (icmp), 64 bytes    Packets: 794 · Displayed: 10 (1.3%) · Dropped: 0 (0.0%)    Profile: Default
```

Figure 3: Screenshot of Wireshark (PING reply)

The ICMP type number is 0, and its code number is also 0. Additionally, this ICMP packet comprises fields such as Checksum, Identifier, and Sequence Number. Each of these fields, namely checksum, sequence number, and identifier, has a size of 2 bytes.

Q5. Examine the corresponding ping reply packet. What is the IP address of your host? What is the IP address of the target destination host?

My host's IP address is 10.122.159.146 (McGill Wi-Fi), and the destination host's IP address is 172.217.13.100.

B. Tracert

After executing all the commands in the terminal and analyzing the data with Wireshark, I obtained the following figures and results.

Q6. How many ICMP packets are in the list plane?

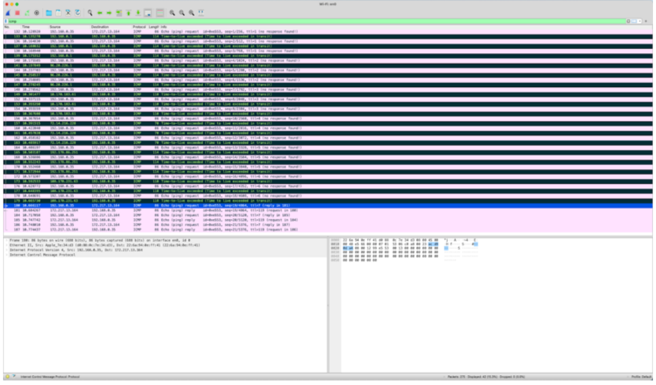


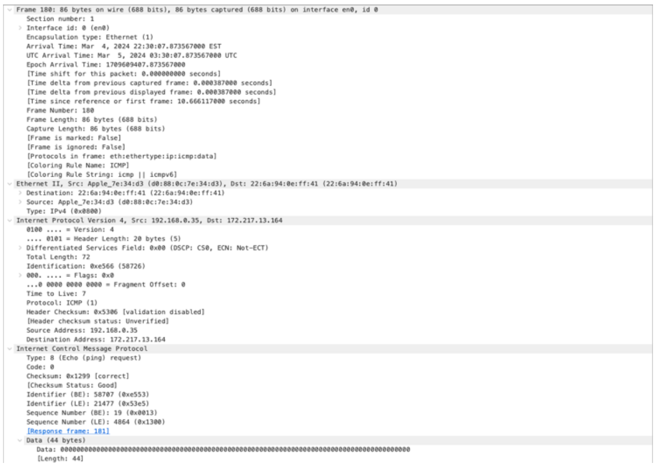
Figure 4: Screenshot of Wireshark (Tracert)

There are 42 packets in total in the list plane.

Q7. How many probe packets are sent from the source to the destination for each TTL?

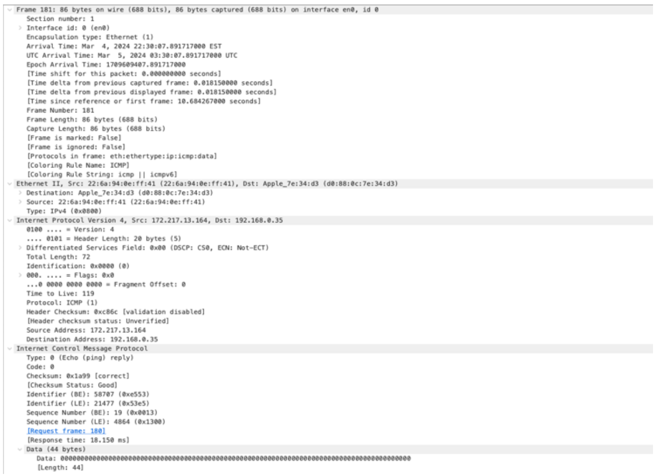
For each TTL value ranging from 1 to 7, there are three probe packets.

Q8. The last few echo request ICMP packets are followed by the echo reply ICMP packets. Compare one of them with the corresponding reply. Determine which fields are similar and which fields are different? Explain the reason.



```
v Frame 180: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface en0, id 0
    Section number: 1
  > Interface id: 0 (en0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Mar  4, 2024 22:30:07.873567000 EST
    UTC Arrival Time: Mar  5, 2024 03:30:07.873567000 UTC
    Epoch Arrival Time: 1709609407.873567000
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 0.000387000 seconds]
    [Time delta from previous displayed frame: 0.000387000 seconds]
    [Time since reference or first frame: 10.666117000 seconds]
    Frame Number: 180
    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:ip:icmp:data]
    [Coloring Rule Name: ICMP]
    [Coloring Rule String: icmp || icmpv6]
v Ethernet II, Src: Apple_7e:34:d3 (d0:88:0c:7e:34:d3), Dst: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41)
  > Destination: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41)
  > Source: Apple_7e:34:d3 (d0:88:0c:7e:34:d3)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 192.168.0.35, Dst: 172.217.13.164
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 72
    Identification: 0xe566 (58726)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 7
    Protocol: ICMP (1)
    Header Checksum: 0x5306 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.0.35
    Destination Address: 172.217.13.164
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x1299 [correct]
    [Checksum Status: Good]
    Identifier (BE): 58707 (0xe553)
    Identifier (LE): 21477 (0x53e5)
    Sequence Number (BE): 19 (0x0013)
    Sequence Number (LE): 4864 (0x1300)
    [Response frame: 181]
  v Data (44 bytes)
      Data: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
      [Length: 44]
```

Figure 5: Screenshot of Wireshark (Tracert request)



```
v Frame 181: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface en0, id 0
    Section number: 1
  > Interface id: 0 (en0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Mar  4, 2024 22:30:07.891717000 EST
    UTC Arrival Time: Mar  5, 2024 03:30:07.891717000 UTC
    Epoch Arrival Time: 1709609407.891717000
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 0.018150000 seconds]
    [Time delta from previous displayed frame: 0.018150000 seconds]
    [Time since reference or first frame: 10.684267000 seconds]
    Frame Number: 181
    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:ip:icmp:data]
    [Coloring Rule Name: ICMP]
    [Coloring Rule String: icmp || icmpv6]
v Ethernet II, Src: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41), Dst: Apple_7e:34:d3 (d0:88:0c:7e:34:d3)
  > Destination: Apple_7e:34:d3 (d0:88:0c:7e:34:d3)
  > Source: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 172.217.13.164, Dst: 192.168.0.35
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 72
    Identification: 0x0000 (0)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 119
    Protocol: ICMP (1)
    Header Checksum: 0xc86c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.217.13.164
    Destination Address: 192.168.0.35
v Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x1a99 [correct]
    [Checksum Status: Good]
    Identifier (BE): 58707 (0xe553)
    Identifier (LE): 21477 (0x53e5)
    Sequence Number (BE): 19 (0x0013)
    Sequence Number (LE): 4864 (0x1300)
    [Request frame: 180]
    [Response time: 18.150 ms]
  v Data (44 bytes)
      Data: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
      [Length: 44]
```

Figure 6: Screenshot of Wireshark (Tracert reply)

Common elements between ICMP Echo Request and Reply packets include the Identifier and Sequence Number, which help match replies to requests, and both have a Checksum and Data Payload, though Checksum values differ due to content changes. Differently, Echo Request packets have a Type of 8, while Echo Replies are Type 0. The TTL value decreases as packets traverse routers; Source and Destination IP Addresses are reversed in replies to route the response correctly. These differences facilitate accurate network communication, diagnostics, and ensure data integrity by recalculating the checksum based on packet alterations.

Q9. What are the TTL values for these last few packets? Determine the number of routers between the source and destination based on these TTL values.

The TTL value from the source is 7, while the TTL value from the destination is 119. Therefore, the number of routers between the source and destination is 8, calculated as the difference between the TTL values (128 - 119).

Q10. Examine the IP packet header of the last echo request ICMP packet, what is the value in the "Protocol" field? What does this field indicate?

The value in the "Protocol" field is 1, indicating ICMP (Internet Control Message Protocol).

Q11. How many bytes are in this IP header? How many bytes are in the payload of this IP packet? Explain how you determined the number of payload bytes.

In this IP packet, there are 20 bytes in the IP header and 66 bytes in the payload. The calculation involves subtracting the 20 bytes of the IP header from the total of 86 bytes, resulting in 66 bytes for the payload.

Q12. Has this IP packet been fragmented? Explain how you determined whether or not the packet has been fragmented.

This IP packet has not been fragmented. With a Flags field of 0x0 and a Fragment Offset of 0, I can determine that this IP packet has not been fragmented and represents a complete, unfragmented packet.

Q13. How the IP address of www.google.com can be found? Determine the packet and the field in the packet that contains this information.

To find www.google.com's IP address, DNS translates domain names to IP addresses. A DNS Query packet, sent to a DNS server, requests www.google.com's IP, containing the domain name. The DNS Response packet, returning from the server, carries the IP address. In Wireshark, this information could be found in the destination address field from the echo request packet or the source address field from the echo reply packet. In my case, the IP address of www.google.com is 172.217.13.164.

Q14. Examine the ICMP echo packet in your screenshot. Is this different from the ICMP ping query packets in the first half of this lab? If yes, how so?

This is indeed different from the ICMP ping query packets in the first half of this lab. ICMP Echo Request packets have a type of 8 and a code of 0, while ICMP Echo Reply packets have a type of 0 and a code of 0. In addition, the TTL value in the reply is different from the request due to the number of hops the packet has taken in transit. Furthermore, the source and destination addresses will be reversed in the echo reply compared to the echo request.

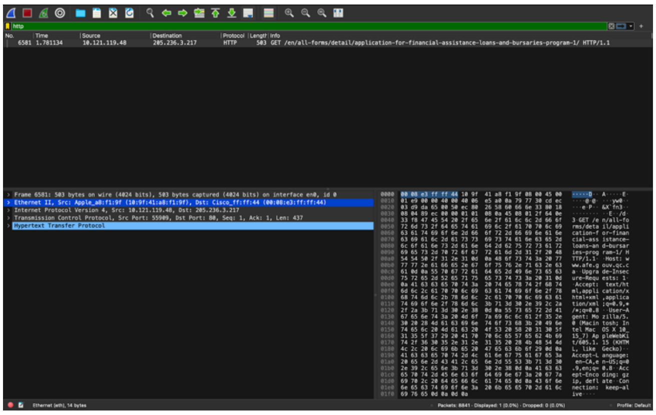
Q15. Examine the ICMP error packet in your screenshot. It has more fields than the ICMP echo packet. What is included in those fields?

Those fields include Unused, Additional data, and a brand-new checksum field.

Q16. Examine the last three ICMP packets received by the source host. How are these packets different from the ICMP error packets (time-to-live exceeded)? Why are they different?

The last three ICMP packets, which are Echo Replies, differ from "Time-to-live Exceeded" error packets mainly in their purpose and content. Echo Replies (Type 0) serve as direct responses to Echo Requests, indicating successful communication, whereas "Time-to-live Exceeded" messages (Type 11) notify about packets dropped due to TTL depletion, aiding in routing issue diagnosis. Echo Replies contain matching identifiers and sequence numbers for request correlation, lacking the original packet's IP header and initial data bytes included in TTL exceeded messages. These distinctions reflect their different roles in network diagnostics and communication efficiency.

Part 2: Ethernet



```
http
No.   Time      Source          Destination    Protocol Length Info
 6581 1.781134  10.121.119.48   205.236.3.217  HTTP     503 GET /en/all-forms/detail/application-for-financial-assistance-loans-and-bursaries-program-1/ HTTP/1.1

> Frame 6581: 503 bytes on wire (4024 bits), 503 bytes captured (4024 bits) on interface en0, id 0
> Ethernet II, Src: Apple_a8:f1:9f (10:9f:41:a8:f1:9f), Dst: Cisco_ff:ff:44 (00:08:e3:ff:ff:44)
> Internet Protocol Version 4, Src: 10.121.119.48, Dst: 205.236.3.217
> Transmission Control Protocol, Src Port: 55909, Dst Port: 80, Seq: 1, Ack: 1, Len: 437
> Hypertext Transfer Protocol

0000  00 08 e3 ff ff 44 10 9f  41 a8 f1 9f 08 00 45 00   ·····D·· A·····E·
0010  01 e9 00 00 40 00 40 06  e5 a0 0a 79 77 30 cd ec   ····@·@· ···yw0··
0020  03 d9 da 65 00 50 ec 80  26 58 60 66 6e 33 80 18   ···e·P·· &X`fn3··
0030  08 04 89 ec 00 00 01 01  08 0a 45 08 01 2f 64 0e   ········ ··E··/d·
0040  33 f8 47 45 54 20 2f 65  6e 2f 61 6c 6c 2d 66 6f   3·GET /e n/all-fo
0050  72 6d 73 2f 64 65 74 61  69 6c 2f 61 70 70 6c 69   rms/deta il/appli
0060  63 61 74 69 6f 6e 2d 66  6f 72 2d 66 69 6e 61 6e   cation-f or-finan
0070  63 69 61 6c 2d 61 73 73  69 73 74 61 6e 63 65 2d   cial-ass istance-
0080  6c 6f 61 6e 73 2d 61 6e  64 2d 62 75 72 73 61 72   loans-an d-bursar
0090  69 65 73 2d 70 72 6f 67  72 61 6d 2d 31 2f 20 48   ies-prog ram-1/ H
00a0  54 54 50 2f 31 2e 31 0d  0a 48 6f 73 74 3a 20 77   TTP/1.1· ·Host: w
00b0  77 77 2e 61 66 65 2e 67  6f 75 76 2e 71 63 2e 63   ww.afe.g ouv.qc.c
00c0  61 0d 0a 55 70 67 72 61  64 65 2d 49 6e 73 65 63   a··Upgra de-Insec
00d0  75 72 65 2d 52 65 71 75  65 73 74 73 3a 20 31 0d   ure-Requ ests: 1·
00e0  0a 41 63 63 65 70 74 3a  20 74 65 78 74 2f 68 74   ·Accept:  text/ht
00f0  6d 6c 2c 61 70 70 6c 69  63 61 74 69 6f 6e 2f 78   ml,appli cation/x
0100  68 74 6d 6c 2b 78 6d 6c  2c 61 70 70 6c 69 63 61   html+xml ,applica
0110  74 69 6f 6e 2f 78 6d 6c  3b 71 3d 30 2e 39 2c 2a   tion/xml ;q=0.9,*
0120  2f 2a 3b 71 3d 30 2e 38  0d 0a 55 73 65 72 2d 41   /*;q=0.8 ··User-A
0130  67 65 6e 74 3a 20 4d 6f  7a 69 6c 6c 61 2f 35 2e   gent: Mo zilla/5.
0140  30 20 28 4d 61 63 69 6e  74 6f 73 68 3b 20 49 6e   0 (Macin tosh; In
0150  74 65 6c 20 4d 61 63 20  4f 53 20 58 20 31 30 5f   tel Mac  OS X 10_
0160  31 35 5f 37 29 20 41 70  70 6c 65 57 65 62 4b 69   15_7) Ap pleWebKi
0170  74 2f 36 30 35 2e 31 2e  31 35 20 28 4b 48 54 4d   t/605.1. 15 (KHTM
0180  4c 2c 20 6c 69 6b 65 20  47 65 63 6b 6f 29 0d 0a   L, like  Gecko)··
0190  41 63 63 65 70 74 2d 4c  61 6e 67 75 61 67 65 3a   Accept-L anguage:
01a0  20 65 6e 2d 43 41 2c 65  6e 2d 55 53 3b 71 3d 30    en-CA,e n-US;q=0
01b0  2e 39 2c 65 6e 3b 71 3d  30 2e 38 0d 0a 41 63 63   .9,en;q= 0.8··Acc
01c0  65 70 74 2d 45 6e 63 6f  64 69 6e 67 3a 20 67 7a   ept-Enco ding: gz
01d0  69 70 2c 20 64 65 66 6c  61 74 65 0d 0a 43 6f 6e   ip, defl ate··Con
01e0  6e 65 63 74 69 6f 6e 3a  20 6b 65 65 70 2d 61 6c   nection:  keep-al
01f0  69 76 65 0d 0a 0d 0a                               ive····

Ethernet (eth), 14 bytes          Packets: 8841 · Displayed: 1 (0.0%) · Dropped: 0 (0.0%)          Profile: Default
```

Figure 7: Screenshot of Wireshark (Ethernet)

Q1. What is the 48-bit Ethernet address of your computer?

10:9f:41:a8:f1:9f

Q2. What is the 48-bit destination address in the Ethernet frame?

00:08:e3:ff:ff:44

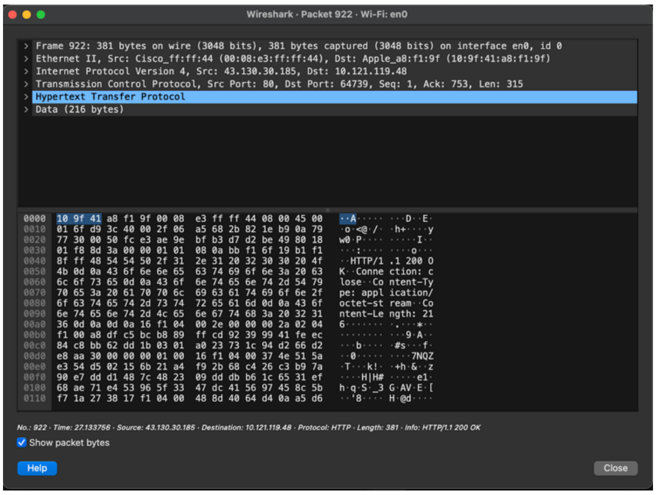
Q3. Is this (destination address) the Ethernet address of https://wiki.wireshark.org/Ethernet?

No.

Q4. What device has this as its Ethernet address?

My router which is Cisco Systems has this as its Ethernet address.

Q5. What is the value of the Ethernet source address? Is this the address of your computer, or of www.wireshark ? What device has this as its Ethernet address?



```
Wireshark · Packet 922 · Wi-Fi: en0

> Frame 922: 381 bytes on wire (3048 bits), 381 bytes captured (3048 bits) on interface en0, id 0
> Ethernet II, Src: Cisco_ff:ff:44 (00:08:e3:ff:ff:44), Dst: Apple_a8:f1:9f (10:9f:41:a8:f1:9f)
> Internet Protocol Version 4, Src: 43.130.30.185, Dst: 10.121.119.48
> Transmission Control Protocol, Src Port: 80, Dst Port: 64739, Seq: 1, Ack: 753, Len: 315
> Hypertext Transfer Protocol
> Data (216 bytes)

0000  10 9f 41 a8 f1 9f 00 08  e3 ff ff 44 08 00 45 00   ··A····· ···D··E·
0010  01 6f d9 3c 40 00 2f 06  a5 68 2b 82 1e b9 0a 79   ·o·<@·/· ·h+····y
0020  77 30 00 50 fc e3 ae 9e  bf b3 d7 d2 be 49 80 18   w0·P···· ·····I··
0030  01 f8 8d 3a 00 00 01 01  08 0a bb f1 6f 19 b1 f1   ···:···· ····o···
0040  8f ff 48 54 54 50 2f 31  2e 31 20 32 30 30 20 4f   ··HTTP/1 .1 200 O
0050  4b 0d 0a 43 6f 6e 6e 65  63 74 69 6f 6e 3a 20 63   K··Conne ction: c
0060  6c 6f 73 65 0d 0a 43 6f  6e 74 65 6e 74 2d 54 79   lose··Co ntent-Ty
0070  70 65 3a 20 61 70 70 6c  69 63 61 74 69 6f 6e 2f   pe: appl ication/
0080  6f 63 74 65 74 2d 73 74  72 65 61 6d 0d 0a 43 6f   octet-st ream··Co
0090  6e 74 65 6e 74 2d 4c 65  6e 67 74 68 3a 20 32 31   ntent-Le ngth: 21
00a0  36 0d 0a 0d 0a 16 f1 04  00 2e 00 00 00 2a 02 04   6······· ·.···*··
00b0  f1 00 a8 df c5 bc b8 89  ff cd 92 39 99 41 fe ec   ········ ···9·A··
00c0  84 c8 bb 62 dd 1b 03 01  a0 23 73 1c 94 d2 66 d2   ···b···· ·#s···f·
00d0  e8 aa 30 00 00 00 01 00  16 f1 04 00 37 4e 51 5a   ··0····· ····7NQZ
00e0  e3 54 d5 02 15 6b 21 a4  f9 2b 68 c4 26 c3 b9 7a   ·T···k!· ·+h·&··z
00f0  90 e7 dd d1 48 7c 48 23  09 dd db b6 1c 65 31 ef   ····H|H# ·····e1·
0100  68 ae 71 e4 53 96 5f 33  47 dc 41 56 97 45 8c 5b   h·q·S·_3 G·AV·E·[
0110  f7 1a 27 38 17 f1 04 00  48 8d 40 64 d4 0a a5 d6   ··'8···· H·@d····

No.: 922 · Time: 27.133756 · Source: 43.130.30.185 · Destination: 10.121.119.48 · Protocol: HTTP · Length: 381 · Info: HTTP/1.1 200 OK
☑ Show packet bytes
```

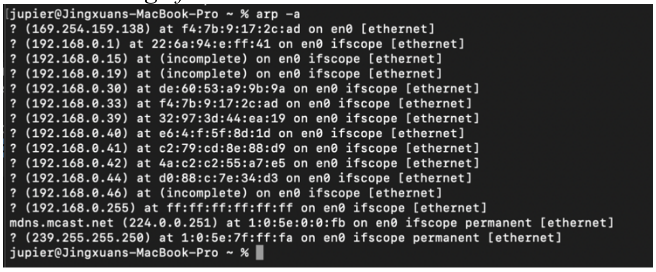
Figure 8: Screenshot of Wireshark (Ethernet)

The Ethernet source address is Cisco_ff:ff:44 (00:08:e3:ff:ff:44), which is the MAC address of the network interface of the computer or device that initiated the packet capture. This address is not the address of www.wireshark.org, but rather the device used to capture the packets, which in this case is a Cisco device with this as its Ethernet address.

Q6. What is the destination address in the Ethernet frame? Is this the Ethernet address of your computer?

The destination address in the Ethernet frame is "Apple_a8:f1:9f ". This is the Ethernet address of my computer.

Q7. Take a screenshot of your computer's ARP cache. What is the meaning of each column value?

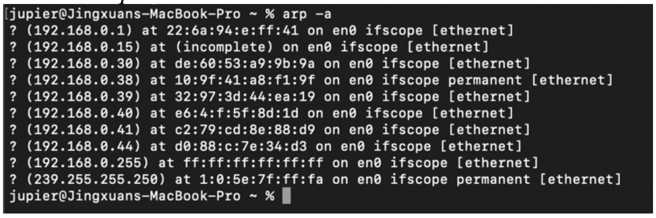


```
jupier@Jingxuans-MacBook-Pro ~ % arp -a
? (169.254.159.138) at f4:7b:9:17:2c:ad on en0 [ethernet]
? (192.168.0.1) at 22:6a:94:e:ff:41 on en0 ifscope [ethernet]
? (192.168.0.15) at (incomplete) on en0 ifscope [ethernet]
? (192.168.0.19) at (incomplete) on en0 ifscope [ethernet]
? (192.168.0.30) at de:60:53:a9:9b:9a on en0 ifscope [ethernet]
? (192.168.0.33) at f4:7b:9:17:2c:ad on en0 ifscope [ethernet]
? (192.168.0.39) at 32:97:3d:44:ea:19 on en0 ifscope [ethernet]
? (192.168.0.40) at e6:4:f:5f:8d:1d on en0 ifscope [ethernet]
? (192.168.0.41) at c2:79:cd:8e:88:d9 on en0 ifscope [ethernet]
? (192.168.0.42) at 4a:c2:c2:55:a7:e5 on en0 ifscope [ethernet]
? (192.168.0.44) at d0:88:c:7e:34:d3 on en0 ifscope [ethernet]
? (192.168.0.46) at (incomplete) on en0 ifscope [ethernet]
? (192.168.0.255) at ff:ff:ff:ff:ff:ff on en0 ifscope [ethernet]
mdns.mcast.net (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
? (239.255.255.250) at 1:0:5e:7f:ff:fa on en0 ifscope permanent [ethernet]
jupier@Jingxuans-MacBook-Pro ~ %
```

Figure 9: Screenshot of Wireshark (Ethernet)

The first column shows the IP address of devices on the local network.
The second column displays the corresponding MAC address for each IP address, which is the physical hardware address of the network interface on that device.
The third column indicates the network interface (in this case `en0`) on which the device is reachable.
"Permanent" entries are statically configured, usually by the user, while dynamic entries have been learned by the ARP protocol.
Entries marked as "incomplete" indicate that the ARP resolution process has been initiated for the IP address, but the corresponding MAC address has not been received yet.

Q8. Run arp -a command and compare the list with your answer to question 7.

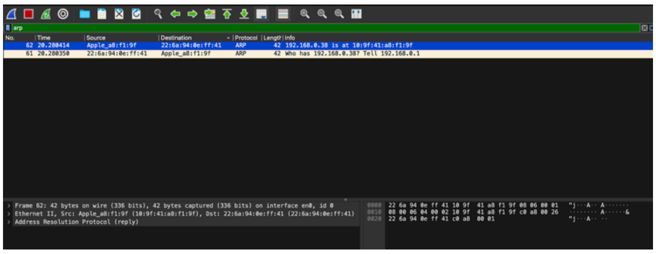


```
jupier@Jingxuans-MacBook-Pro ~ % arp -a
? (192.168.0.1) at 22:6a:94:e:ff:41 on en0 ifscope [ethernet]
? (192.168.0.15) at (incomplete) on en0 ifscope [ethernet]
? (192.168.0.30) at de:60:53:a9:9b:9a on en0 ifscope [ethernet]
? (192.168.0.38) at 10:9f:41:a8:f1:9f on en0 ifscope permanent [ethernet]
? (192.168.0.39) at 32:97:3d:44:ea:19 on en0 ifscope [ethernet]
? (192.168.0.40) at e6:4:f:5f:8d:1d on en0 ifscope [ethernet]
? (192.168.0.41) at c2:79:cd:8e:88:d9 on en0 ifscope [ethernet]
? (192.168.0.44) at d0:88:c:7e:34:d3 on en0 ifscope [ethernet]
? (192.168.0.255) at ff:ff:ff:ff:ff:ff on en0 ifscope [ethernet]
? (239.255.255.250) at 1:0:5e:7f:ff:fa on en0 ifscope permanent [ethernet]
jupier@Jingxuans-MacBook-Pro ~ %
```

Figure 10: Screenshot of Wireshark (Ethernet)

Upon running the `arp -a` command, the ARP cache displays the current IP-to-MAC address mappings on your machine. Comparing the ARP cache from question 7 to question 8, it looks like they are identical. This indicates that no new devices have connected to the network and the ARP cache hasn't changed between the two commands, meaning your machine has not discovered any new hardware addresses or removed any old ones in that time. Your machine's IP address `192.168.0.38` is still associated with the MAC address `10:9f:41:a8:f1:9f` and is marked as "permanent," as expected. Entries labelled "incomplete" suggest that the machine has not received a response to ARP requests sent to those IP addresses.

Q9. How many bytes from the very beginning of the Ethernet frame does the ARP opcode field begin?



```
No.  Time       Source              Destination         Protocol Length Info
  62 20.280414  Apple_a8:f1:9f      22:6a:94:0e:ff:41   ARP          42 192.168.0.38 is at 10:9f:41:a8:f1:9f
  61 20.280350  22:6a:94:0e:ff:41   Apple_a8:f1:9f      ARP          42 Who has 192.168.0.38? Tell 192.168.0.1

> Frame 62: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface en0, id 0
> Ethernet II, Src: Apple_a8:f1:9f (10:9f:41:a8:f1:9f), Dst: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41)
> Address Resolution Protocol (reply)

0000  22 6a 94 0e ff 41 10 9f  41 a8 f1 9f 08 06 00 01   "j···A·· A·······
0010  08 00 06 04 00 02 10 9f  41 a8 f1 9f c0 a8 00 26   ········ A······&
0020  22 6a 94 0e ff 41 c0 a8  00 01                     "j···A·· ··
```

Figure 11: Screenshot of Wireshark (Ethernet)

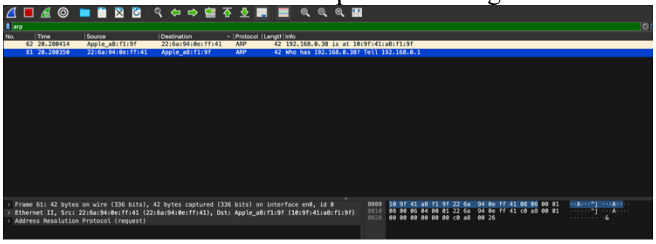
6 (destination MAC) + 6 (source MAC) + 2 (EtherType) + 2 (hardware type) + 2 (protocol type) +1 (hardware size) + 1 (protocol size) = 20 bytes

Q10. What is the value of the opcode field within the ARP-payload part of the Ethernet frame in which an ARP request is made?
The value of the opcode field within the ARP payload part of the Ethernet frame for an ARP request is 1.

Q11. Does the ARP message contain the IP address of the sender?
Yes, the ARP message does contain the IP address of the sender which is 192.168.0.38.

Q12. Where in the ARP request does the "question" appear – the Ethernet address of the machine whose corresponding IP address is being queried?
In the ARP request, the "question" appears towards the end of the ARP packet. After the source hardware and protocol addresses, the target hardware address (which will be blank for a request), and finally the target protocol address, which is the IP address being queried.

Q13. How many bytes from the very beginning of the Ethernet frame does the ARP opcode field begin?



```
No.  Time       Source              Destination         Protocol Length Info
  62 20.280414  Apple_a8:f1:9f      22:6a:94:0e:ff:41   ARP          42 192.168.0.38 is at 10:9f:41:a8:f1:9f
  61 20.280350  22:6a:94:0e:ff:41   Apple_a8:f1:9f      ARP          42 Who has 192.168.0.38? Tell 192.168.0.1

> Frame 61: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface en0, id 0
> Ethernet II, Src: 22:6a:94:0e:ff:41 (22:6a:94:0e:ff:41), Dst: Apple_a8:f1:9f (10:9f:41:a8:f1:9f)
> Address Resolution Protocol (request)

0000  10 9f 41 a8 f1 9f 22 6a  94 0e ff 41 08 06 00 01   ··A···"j ···A····
0010  08 00 06 04 00 01 22 6a  94 0e ff 41 c0 a8 00 01   ······"j ···A····
0020  00 00 00 00 00 00 c0 a8  00 26                     ········ ·&
```

Figure 12: Screenshot of Wireshark (Ethernet)

Ethernet header: 14 bytes (6 bytes for the destination MAC, 6 bytes for the source MAC, 2 bytes for the EtherType)
ARP hardware type: 2 bytes
ARP protocol type: 2 bytes
ARP hardware size: 1 byte
ARP protocol size: 1 byte
Ans: 14+2+2+1+1 = 20 bytes

Q14. What is the value of the opcode field within the ARP-payload part of the Ethernet frame in which an ARP response is made?
The value of the opcode field within the ARP-payload part of the Ethernet frame for an ARP response is 2. This value indicates that the packet is an ARP reply, which is sent in response to an ARP request.

Q15. Where in the ARP message does the "answer" to the earlier ARP request appear – the IP address of the machine having the Ethernet address whose corresponding IP address is being queried?
In an ARP reply message, the "answer" to the ARP request appears in the sender's address fields.

CONCLUSION

This lab comprehensively covers the utilization of Wireshark for network analysis, focusing on IP and Ethernet protocols. It details the practical use of Wireshark, from capturing and analyzing packets to understanding the intricacies of network communication through ICMP and ARP protocols. Through exercises and instructions, it aims to enhance the reader's skills in network troubleshooting and analysis, showcasing the significance of tools like ping and traceroute for diagnosing network issues. Ultimately, the lab serves as a foundational guide for effectively navigating and applying Wireshark in the context of network engineering and troubleshooting, emphasizing the importance of protocol analysis in maintaining and securing network operations.

REFERENCES

[1] "L4: IP & MAC" McGill University. https://mycourses2.mcgill.ca/d2l/le/lessons/688501/topics/7676417 (accessed Feb. 12, 2024)

Experiment was taken in the laboratory section of ECSE 308 at McGill University, Montreal, QC, H3A 0E9, Canada.