IT – 304 | CN | LAB-9

(Harsh Gajjar – 202201140)

Part A: Exploring ICMP with Ping and Traceroute

"ping -n 10 gaia.cs.umass.edu" command:

```
C:\Users\Harsh>ping -n 10 gaia.cs.umass.edu

Pinging gaia.cs.umass.edu [128.119.245.12] with 32 bytes of data:
Reply from 128.119.245.12: bytes=32 time=240ms TTL=43
Reply from 128.119.245.12: bytes=32 time=360ms TTL=43
Reply from 128.119.245.12: bytes=32 time=263ms TTL=43
Reply from 128.119.245.12: bytes=32 time=237ms TTL=43
Reply from 128.119.245.12: bytes=32 time=280ms TTL=43
Reply from 128.119.245.12: bytes=32 time=301ms TTL=43
Reply from 128.119.245.12: bytes=32 time=299ms TTL=43
Reply from 128.119.245.12: bytes=32 time=317ms TTL=43
Reply from 128.119.245.12: bytes=32 time=315ms TTL=43
Reply from 128.119.245.12: bytes=32 time=334ms TTL=43

Ping statistics for 128.119.245.12:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 237ms, Maximum = 360ms, Average = 294ms

C:\Users\Harsh>
```

Wireshark Capture:

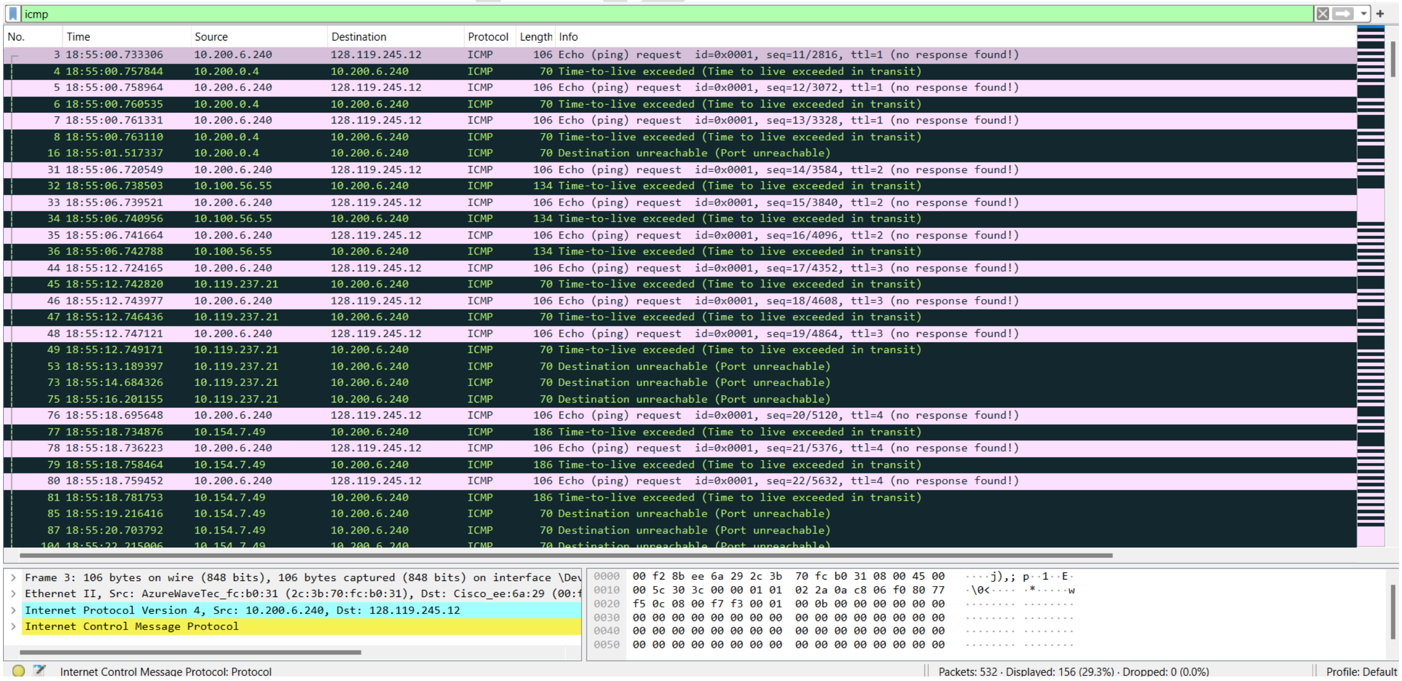
icmp

No.	Time	Source	Destination	Protocol	Length	Info
56	18:49:47.907390	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=1/256, ttl=128 (reply in 57)
57	18:49:48.148175	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=1/256, ttl=43 (request in 56)
58	18:49:48.923788	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=2/512, ttl=128 (reply in 59)
59	18:49:49.284304	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=2/512, ttl=43 (request in 58)
61	18:49:49.936620	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=3/768, ttl=128 (reply in 62)
62	18:49:50.200169	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=3/768, ttl=43 (request in 61)
64	18:49:50.950778	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=4/1024, ttl=128 (reply in 65)
65	18:49:51.187957	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=4/1024, ttl=43 (request in 64)
66	18:49:51.967887	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=128 (reply in 67)
67	18:49:52.248296	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=5/1280, ttl=43 (request in 66)
69	18:49:52.977853	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=6/1536, ttl=128 (reply in 70)
70	18:49:53.278751	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=6/1536, ttl=43 (request in 69)
71	18:49:53.991423	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=7/1792, ttl=128 (reply in 72)
72	18:49:54.290368	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=7/1792, ttl=43 (request in 71)
73	18:49:55.008527	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=8/2048, ttl=128 (reply in 74)
74	18:49:55.326104	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=8/2048, ttl=43 (request in 73)
77	18:49:56.025077	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (reply in 78)
78	18:49:56.340029	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=9/2304, ttl=43 (request in 77)
79	18:49:57.040847	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (reply in 80)
80	18:49:57.375015	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=10/2560, ttl=43 (request in 79)

"tracert gaia.cs.umass.edu" command:

```
C:\Users\Harsh>tracert gaia.cs.umass.edu

Tracing route to gaia.cs.umass.edu [128.119.245.12]
over a maximum of 30 hops:

  1    24 ms     1 ms     1 ms  10.200.0.4
  2    18 ms     1 ms     1 ms  10.100.56.55
  3    18 ms     2 ms     2 ms  10.119.237.21
  4    39 ms    22 ms    22 ms  10.154.7.49
  5    31 ms    15 ms    14 ms  10.255.236.161
  6    31 ms    14 ms    14 ms  180.149.48.18
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10   310 ms   312 ms   297 ms  fourhundredge-0-0-0-1.4079.core1.hart2.net.internet2.edu [163.253.1.228]
 11   249 ms   247 ms   230 ms  fourhundredge-0-0-0-2.4079.core1.bost2.net.internet2.edu [163.253.2.168]
 12   348 ms   317 ms   300 ms  69.16.3.250
 13   359 ms   315 ms   314 ms  69.16.0.8
 14   313 ms   296 ms   233 ms  69.16.1.0
 15   328 ms   294 ms   312 ms  core2-rt-et-8-3-0.gw.umass.edu [192.80.83.113]
 16   238 ms   235 ms   235 ms  n1-rt-1-1-et-10-0-0.gw.umass.edu [128.119.0.120]
 17   234 ms   351 ms   316 ms  128.119.7.74
 18   345 ms   314 ms   297 ms  128.119.7.66
 19   345 ms   314 ms   298 ms  core2-rt-et-7-2-1.gw.umass.edu [128.119.0.121]
 20   372 ms   244 ms   235 ms  n5-rt-1-1-xe-2-1-0.gw.umass.edu [128.119.3.33]
 21   248 ms   276 ms   311 ms  cics-rt-xe-0-0-0.gw.umass.edu [128.119.3.32]
 22   281 ms   297 ms   309 ms  nscs1bbs1.cs.umass.edu [128.119.240.253]
 23   376 ms   294 ms   329 ms  gaia.cs.umass.edu [128.119.245.12]

Trace complete.

C:\Users\Harsh>
```

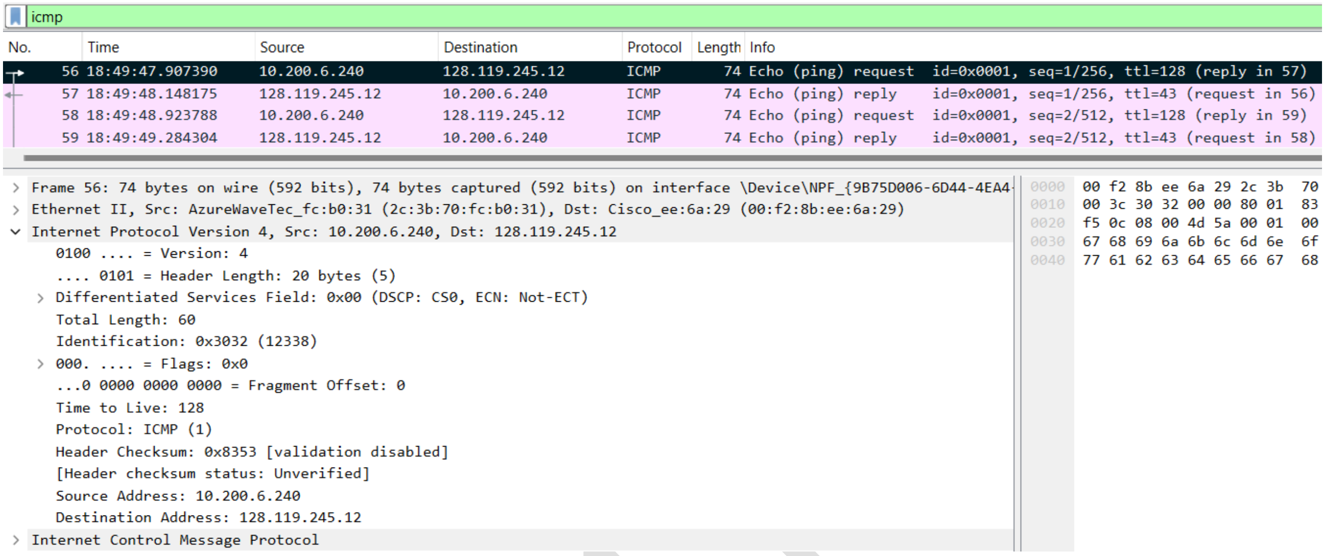
Wireshark Capture:


icmp

No.	Time	Source	Destination	Protocol	Length	Info
3	18:55:00.733306	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=11/2816, ttl=1 (no response found!)
4	18:55:00.757844	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
5	18:55:00.758964	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=12/3072, ttl=1 (no response found!)
6	18:55:00.760535	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
7	18:55:00.761331	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=13/3328, ttl=1 (no response found!)
8	18:55:00.763110	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
16	18:55:01.517337	10.200.0.4	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
31	18:55:06.720549	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=14/3584, ttl=2 (no response found!)
32	18:55:06.738503	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
33	18:55:06.739521	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=15/3840, ttl=2 (no response found!)
34	18:55:06.740956	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
35	18:55:06.741664	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=16/4096, ttl=2 (no response found!)
36	18:55:06.742788	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
44	18:55:12.724165	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=17/4352, ttl=3 (no response found!)
45	18:55:12.742820	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
46	18:55:12.743977	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=18/4608, ttl=3 (no response found!)
47	18:55:12.746436	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
48	18:55:12.747121	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=19/4864, ttl=3 (no response found!)
49	18:55:12.749171	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
53	18:55:13.189397	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
73	18:55:14.684326	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
75	18:55:16.201155	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
76	18:55:18.695648	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=20/5120, ttl=4 (no response found!)
77	18:55:18.734876	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
78	18:55:18.736223	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=21/5376, ttl=4 (no response found!)
79	18:55:18.758464	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
80	18:55:18.759452	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=22/5632, ttl=4 (no response found!)
81	18:55:18.781753	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
85	18:55:19.216416	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
87	18:55:20.703792	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
104	18:55:22.215006	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)

Questions:

1. What is the IP address of your local host and the destination host?



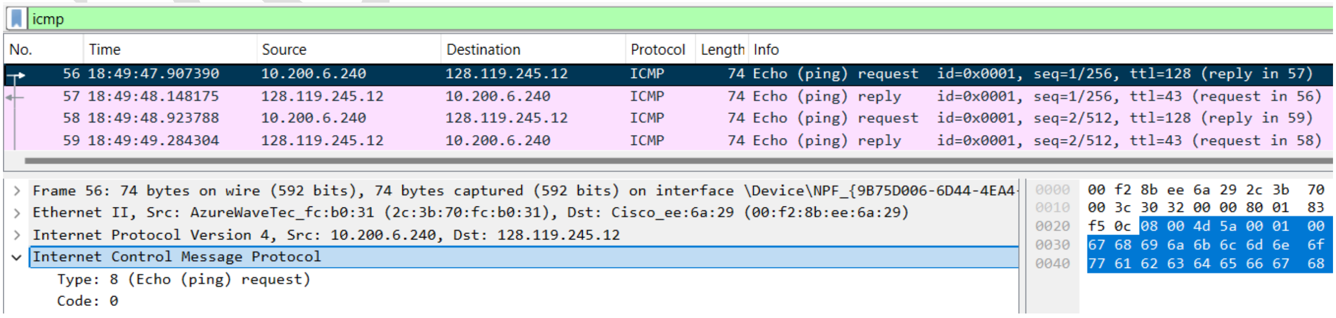
Source Address: 10.200.6.240

Destination Address: 128.119.245.12

2. Why do ICMP packets do not contain source and destination port numbers?

ICMP operates at the network layer (Layer 3 of the OSI model). Since ICMP is used for diagnostic purposes (like ping), it doesn't involve port-based communication, unlike TCP or UDP.

3. Examine one of the captured ICMP echo request packets. What are the `Type` and `Code` values, and what do they signify?



<u>ICMP Echo Request (Type 8, Code 0)</u>

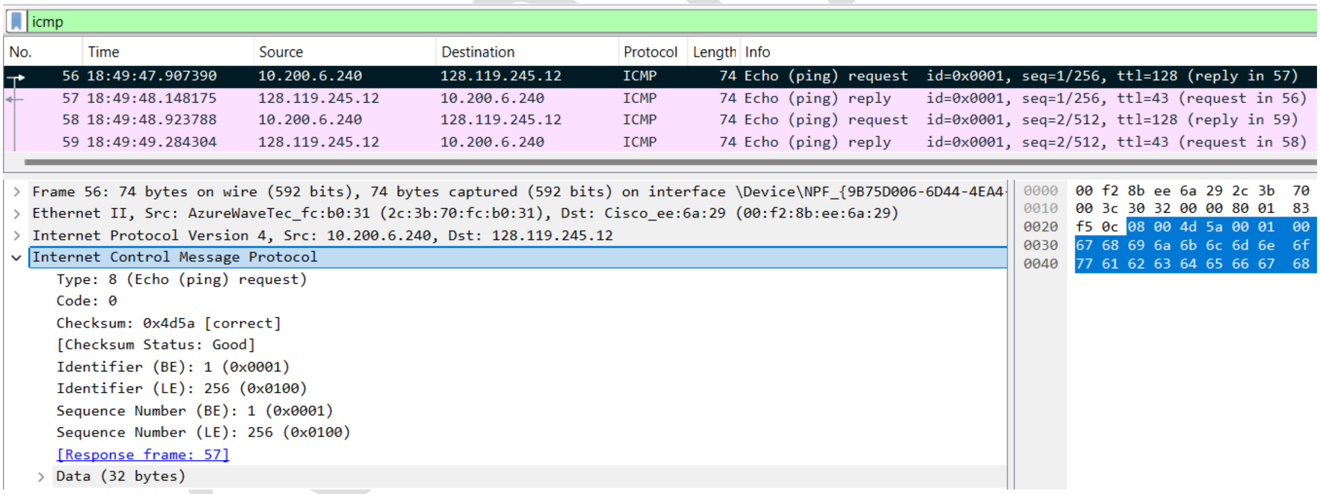
- This type of ICMP packet is used to send a ping from one host to another to test if the target is reachable and measure latency.
    - **Type 8**: This identifies the packet as an Echo Request.

        - Purpose: To initiate a ping query.
    - **Code 0**: This code provides additional detail about the type.
        - For Echo Request, the Code is always 0 (indicating there are no extra qualifiers).

<u>ICMP Echo Reply (Type 0, Code 0)</u>

- The corresponding reply to an Echo Request is an ICMP Echo Reply.
    - **Type 0**: Identifies the packet as an Echo Reply.
        - Purpose: To respond to the request and confirm that the host is reachable.
    - **Code 0**: Again, the code value is 0, indicating that there are no additional qualifiers for this reply.

4. **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?**



Type 8/Type 0 and Code 0 indicate an Echo Request/Reply (response to the ping).

Other fields are:

- Identifier: Typically, 2 bytes.
- Sequence Number: 2 bytes.
- Checksum: 2 bytes.

5. **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?**

icmp

No.	Time	Source	Destination	Protocol	Length	Info
56	18:49:47.907390	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=1/256, ttl=128 (reply in 57)
57	18:49:48.148175	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=1/256, ttl=43 (request in 56)
58	18:49:48.923788	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=2/512, ttl=128 (reply in 59)
59	18:49:49.284304	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=2/512, ttl=43 (request in 58)
61	18:49:49.936620	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=3/768, ttl=128 (reply in 62)
62	18:49:50.200169	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=3/768, ttl=43 (request in 61)
64	18:49:50.950778	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=4/1024, ttl=128 (reply in 65)
65	18:49:51.187957	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=4/1024, ttl=43 (request in 64)
66	18:49:51.967887	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=128 (reply in 67)
67	18:49:52.248296	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=5/1280, ttl=43 (request in 66)
69	18:49:52.977853	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=6/1536, ttl=128 (reply in 70)
70	18:49:53.278751	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=6/1536, ttl=43 (request in 69)
71	18:49:53.991423	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=7/1792, ttl=128 (reply in 72)
72	18:49:54.290368	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=7/1792, ttl=43 (request in 71)
73	18:49:55.008527	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=8/2048, ttl=128 (reply in 74)
74	18:49:55.326104	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=8/2048, ttl=43 (request in 73)
77	18:49:56.025077	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (reply in 78)
78	18:49:56.340029	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=9/2304, ttl=43 (request in 77)
79	18:49:57.040847	10.200.6.240	128.119.245.12	ICMP	74	Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (reply in 80)
80	18:49:57.375015	128.119.245.12	10.200.6.240	ICMP	74	Echo (ping) reply id=0x0001, seq=10/2560, ttl=43 (request in 79)

TTL Values Difference:

- Echo Requests sent from 10.200.6.240 have a TTL of 128.
- Echo Replies from 128.119.245.12 have a TTL of 43.

ICMP Sequence Number Increasing Gradually:

- The sequence numbers increase incrementally with each new request. For example:
    - seq=1 for the first request and reply pair.
    - seq=2, 3, …, 10 for subsequent pairs.
    - This shows the packets are correctly ordered.

6. **Compare the captured ICMP echo request packets to the responses. Identify any differences in the packet details (such as identifiers and sequence numbers).**

icmp

No.	Time	Source	Destination	Protocol	Length	Info
3	18:55:00.733306	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=11/2816, ttl=1 (no response found!)
4	18:55:00.757844	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
5	18:55:00.758964	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=12/3072, ttl=1 (no response found!)
6	18:55:00.760535	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
7	18:55:00.761331	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=13/3328, ttl=1 (no response found!)
8	18:55:00.763110	10.200.0.4	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
16	18:55:01.517337	10.200.0.4	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
31	18:55:06.720549	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=14/3584, ttl=2 (no response found!)
32	18:55:06.738503	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
33	18:55:06.739521	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=15/3840, ttl=2 (no response found!)
34	18:55:06.740956	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
35	18:55:06.741664	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=16/4096, ttl=2 (no response found!)
36	18:55:06.742788	10.100.56.55	10.200.6.240	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
44	18:55:12.724165	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=17/4352, ttl=3 (no response found!)
45	18:55:12.742820	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
46	18:55:12.743977	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=18/4608, ttl=3 (no response found!)
47	18:55:12.746436	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
48	18:55:12.747121	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=19/4864, ttl=3 (no response found!)
49	18:55:12.749171	10.119.237.21	10.200.6.240	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
53	18:55:13.189397	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
73	18:55:14.684326	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
75	18:55:16.201155	10.119.237.21	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
76	18:55:18.695648	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=20/5120, ttl=4 (no response found!)
77	18:55:18.734876	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
78	18:55:18.736223	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=21/5376, ttl=4 (no response found!)
79	18:55:18.758464	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
80	18:55:18.759452	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=22/5632, ttl=4 (no response found!)
81	18:55:18.781753	10.154.7.49	10.200.6.240	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
85	18:55:19.216416	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
87	18:55:20.703792	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
104	18:55:22.215006	10.154.7.49	10.200.6.240	ICMP	70	Destination unreachable (Port unreachable)
114	18:55:24.705353	10.200.6.240	128.119.245.12	ICMP	106	Echo (ping) request id=0x0001, seq=23/5888, ttl=5 (no response found!)

Identifiers and Sequence Numbers:

- All echo requests have the same identifier (id=0x0001).
- The sequence numbers increase incrementally (e.g., 11, 12, 13, ..., 23).

TTL Values:

- Requests from the source host (10.200.6.240) have low TTL values (starting from 1), which indicates that they are part of a traceroute operation. Each packet is sent with an increasing TTL value to explore successive hops in the route to the destination.

7. Why might some `traceroute` hops not return any response?

There are several reasons why some traceroute hops do not respond:

- ICMP filtering by routers: Some routers block or deprioritize ICMP packets to prevent excessive ping traffic.
- Firewalls: Security configurations may block ICMP packets for certain networks.
- Asymmetric routing: Responses may travel back through a different path, causing Wireshark to miss them.
- Router load or rate limiting: Some routers prioritize routing over ICMP replies, leading to timeouts.

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

ICMP error packets, such as **"Time Exceeded**" or **"Destination Unreachable"**, contain additional fields compared to regular Echo Request/Reply packets:

- Type and Code: Identifies the specific error (e.g., TTL expired, destination unreachable).
- Original IP Header: The packet includes a copy of the original IP header and the first 8 bytes of the original data. This allows the sender to identify which packet triggered the error.
- Checksum: Ensures the integrity of the error packet.

These fields make error packets more informative than standard ping packets.

9. Examine the last three ICMP packets received by the source host. How are these packets different from the ICMP error packets? Why are they different?

Looking at the last three ICMP packets received by the source host (packets 511, 513, and 515):

- These show as "Echo (ping) reply" packets
- They have normal 106-byte lengths
- They contain sequence numbers and IDs matching their corresponding requests
- They are direct responses from the destination

These are different from the ICMP error packets ("Time-to-live exceeded" messages) because:

- The error packets indicate routing/TTL issues along the path
- Error packets are generated by intermediate routers when TTL expires
- The reply packets represent successful end-to-end communication with the destination host
- Reply packets indicate the trace has reached its target destination

10. **Within the tracert measurements, is there a link whose delay is significantly longer than others? Refer to the screenshot in Figure 4, is there a link whose delay is significantly longer than others? On the basis of the router names, can you guess the location of the two routers on the end of this link?**

```
Command Prompt

C:\Users\yashs>tracert www.inria.fr

Tracing route to inria.fr [128.93.162.83]
over a maximum of 30 hops:

  1     2 ms     2 ms     2 ms  gpon.net [192.168.1.1]
  2     4 ms     4 ms     4 ms  10.230.192.1 [10.230.192.1]
  3     *        *        *     Request timed out.
  4     5 ms     5 ms     5 ms  136.232.112.109
  5    13 ms    14 ms    14 ms  172.16.25.116 [172.16.25.116]
  6    14 ms    14 ms    13 ms  172.16.1.220 [172.16.1.220]
  7    15 ms    15 ms    15 ms  103.198.140.176
  8   135 ms   134 ms   134 ms  103.198.140.54
  9   137 ms   172 ms   134 ms  103.198.140.45
 10     *        *        *     Request timed out.
 11   139 ms   139 ms   143 ms  be3671.ccr51.lhr01.atlas.cogentco.com [130.117.48.137]
 12   141 ms   144 ms   141 ms  be3487.ccr41.lon13.atlas.cogentco.com [154.54.60.5]
 13   151 ms   150 ms   150 ms  be2870.ccr22.lon01.atlas.cogentco.com [154.54.58.174]
 14   139 ms   139 ms   138 ms  ae5.cr12-lon1.ip4.gtt.net [154.14.40.57]
 15   146 ms   146 ms   146 ms  ae8.cr6-par11.ip4.gtt.net [141.136.110.229]
 16   145 ms   145 ms   145 ms  ip4.gtt.net [212.222.6.69]
 17   146 ms   145 ms   145 ms  hu0-4-0-0-ren-nr-orsay-rtr-091.noc.renater.fr [193.51.180.131]
 18   149 ms   147 ms   148 ms  te-0-1-0-13-ren-nr-jouy-rtr-091.noc.renater.fr [193.55.204.201]
 19   147 ms   147 ms   155 ms  te2-8-inria-rtr-021.noc.renater.fr [193.51.180.125]
 20   145 ms   145 ms   150 ms  inria-rocquencourt-vl1631-te1-4-inria-rtr-021.noc.renater.fr [193.51.184.177]
 21   147 ms   147 ms   147 ms  unit240-reth1-vfw-ext-dc1.inria.fr [192.93.122.19]
 22   146 ms   145 ms   145 ms  prod-inriafr-cms.inria.fr [128.93.162.83]

Trace complete.
```

Yes, there is a link with significantly longer delay. Looking at hops 7 to 8, there's a dramatic increase in latency:

- Hop 7: ~14 ms
- Hop 8: ~135 ms

This represents a jump of about 120ms in delay.

Based on the router names visible in the trace:

- The router at hop 11 shows "be3671.ccr51.lhr01.atlas.cogentco.com" which indicates London (lhr01)
- The subsequent routers show European locations, indicating this is likely a transatlantic link between North America and Europe

The large delay increases between hops 7 and 8 likely represents the undersea cable crossing between North America and Europe. This significant latency jump is typical for transoceanic links due to the physical distance the data must travel, with typical transatlantic round-trip times being around 80-100ms.

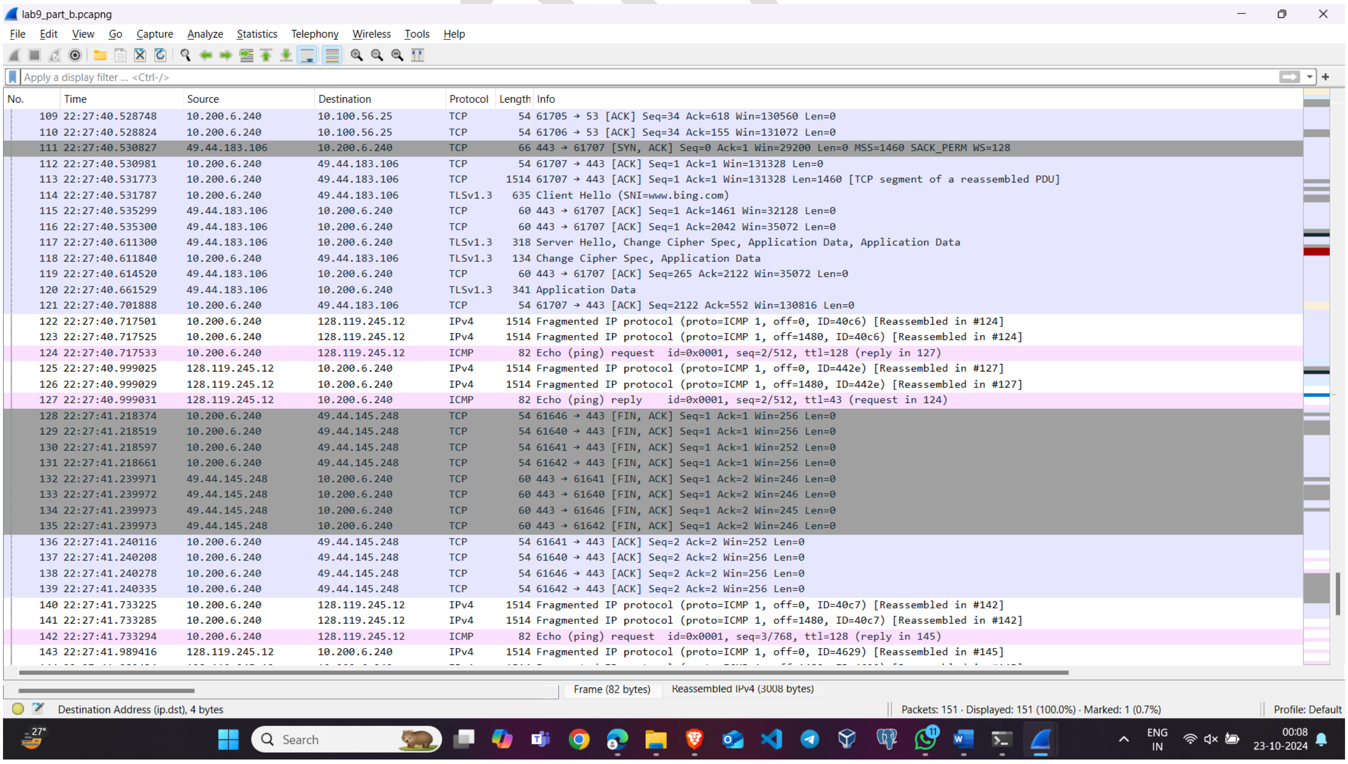
(PTO)

Part B: Analysing IP Protocol Behaviour

"ping gaia.cs.umass.edu -l 3000" command:

```
C:\Users\Harsh>ping gaia.cs.umass.edu -l 3000

Pinging gaia.cs.umass.edu [128.119.245.12] with 3000 bytes of data:
Reply from 128.119.245.12: bytes=3000 time=238ms TTL=43
Reply from 128.119.245.12: bytes=3000 time=281ms TTL=43
Reply from 128.119.245.12: bytes=3000 time=256ms TTL=43
Reply from 128.119.245.12: bytes=3000 time=301ms TTL=43

Ping statistics for 128.119.245.12:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 238ms, Maximum = 301ms, Average = 269ms

C:\Users\Harsh>
```

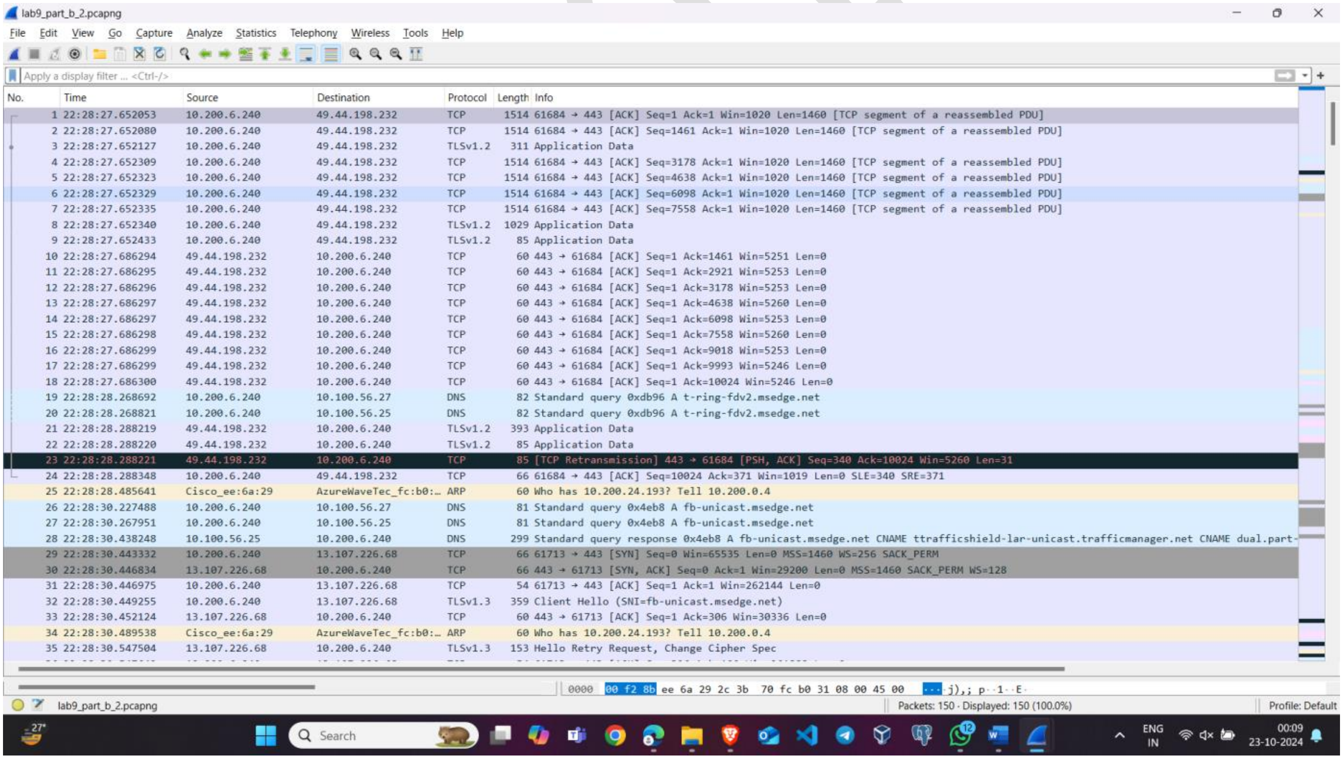
Wireshark Capture:



"ping youtube.com" command:

```
C:\Users\Harsh>ping youtube.com

Pinging youtube.com [216.58.203.46] with 32 bytes of data:
Request timed out.
Reply from 216.58.203.46: bytes=32 time=43ms TTL=117
Reply from 216.58.203.46: bytes=32 time=42ms TTL=117
Reply from 216.58.203.46: bytes=32 time=44ms TTL=117

Ping statistics for 216.58.203.46:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 42ms, Maximum = 44ms, Average = 43ms

C:\Users\Harsh>
```

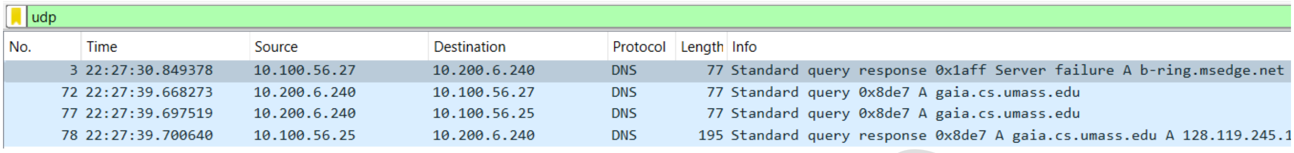
Wireshark Capture:



No.	Time	Source	Destination	Protocol	Length	Info
1	22:28:27.652053	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=1 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
2	22:28:27.652080	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=1461 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
3	22:28:27.652127	10.200.6.240	49.44.198.232	TLSv1.2	311	Application Data
4	22:28:27.652309	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=3178 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
5	22:28:27.652323	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=4638 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
6	22:28:27.652329	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=6098 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
7	22:28:27.652335	10.200.6.240	49.44.198.232	TCP	1514	61684 → 443 [ACK] Seq=7558 Ack=1 Win=1020 Len=1460 [TCP segment of a reassembled PDU]
8	22:28:27.652340	10.200.6.240	49.44.198.232	TLSv1.2	1029	Application Data
9	22:28:27.652433	10.200.6.240	49.44.198.232	TLSv1.2	85	Application Data
10	22:28:27.686294	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=1461 Win=5251 Len=0
11	22:28:27.686295	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=2921 Win=5253 Len=0
12	22:28:27.686296	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=3178 Win=5253 Len=0
13	22:28:27.686297	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=4638 Win=5260 Len=0
14	22:28:27.686297	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=6098 Win=5253 Len=0
15	22:28:27.686298	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=7558 Win=5260 Len=0
16	22:28:27.686299	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=9018 Win=5253 Len=0
17	22:28:27.686299	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=9993 Win=5246 Len=0
18	22:28:27.686300	49.44.198.232	10.200.6.240	TCP	60	443 → 61684 [ACK] Seq=1 Ack=10024 Win=5246 Len=0
19	22:28:28.268692	10.200.6.240	10.100.56.27	DNS	82	Standard query 0xdb96 A t-ring-fdv2.msedge.net
20	22:28:28.268821	10.200.6.240	10.100.56.25	DNS	82	Standard query 0xdb96 A t-ring-fdv2.msedge.net
21	22:28:28.288219	49.44.198.232	10.200.6.240	TLSv1.2	393	Application Data
22	22:28:28.288220	49.44.198.232	10.200.6.240	TLSv1.2	85	Application Data
23	22:28:28.288221	49.44.198.232	10.200.6.240	TCP	85	[TCP Retransmission] 443 → 61684 [PSH, ACK] Seq=340 Ack=10024 Win=5260 Len=31
24	22:28:28.288348	10.200.6.240	49.44.198.232	TCP	66	61684 → 443 [ACK] Seq=10024 Ack=371 Win=1019 Len=0 SLE=340 SRE=371
25	22:28:28.485641	Cisco_ee:6a:29	AzureWaveTec_fc:b0:…	ARP	60	Who has 10.200.24.193? Tell 10.200.0.4
26	22:28:30.227488	10.200.6.240	10.100.56.27	DNS	81	Standard query 0x4eb8 A fb-unicast.msedge.net
27	22:28:30.267951	10.200.6.240	10.100.56.25	DNS	81	Standard query 0x4eb8 A fb-unicast.msedge.net
28	22:28:30.438248	10.100.56.25	10.200.6.240	DNS	299	Standard query response 0x4eb8 A fb-unicast.msedge.net CNAME ttrafficshield-lar-unicast.trafficmanager.net CNAME dual.part-
29	22:28:30.443332	10.200.6.240	13.107.226.68	TCP	66	61713 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
30	22:28:30.446834	13.107.226.68	10.200.6.240	TCP	66	443 → 61713 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM WS=128
31	22:28:30.446975	10.200.6.240	13.107.226.68	TCP	54	61713 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
32	22:28:30.449255	10.200.6.240	13.107.226.68	TLSv1.3	359	Client Hello (SNI=fb-unicast.msedge.net)
33	22:28:30.452124	13.107.226.68	10.200.6.240	TCP	60	443 → 61713 [ACK] Seq=1 Ack=306 Win=30336 Len=0
34	22:28:30.489538	Cisco_ee:6a:29	AzureWaveTec_fc:b0:…	ARP	60	Who has 10.200.24.193? Tell 10.200.0.4
35	22:28:30.547504	13.107.226.68	10.200.6.240	TLSv1.3	153	Hello Retry Request, Change Cipher Spec

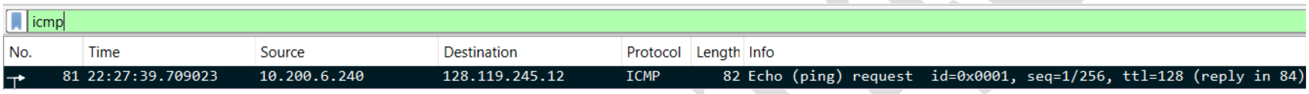
Questions:

1. **Select the first UDP segment sent by your computer via the traceroute command to gaia.cs.umass.edu. Expand the Internet Protocol part of the packet in the packet details window. What is the IP address of your computer?**



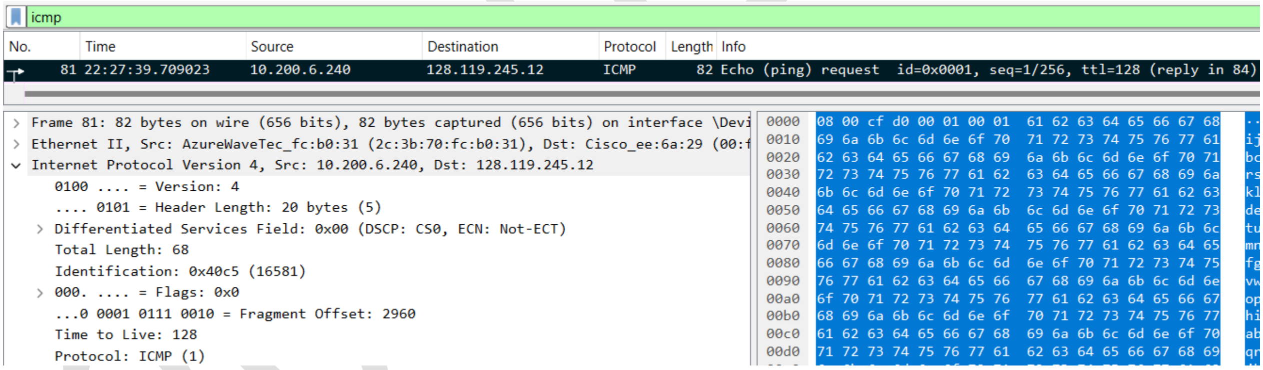
Source (My) IP: 10.100.56.27

2. **What is the value in the time-to-live (TTL) field in this IPv4 datagram's header? (search in 1st ICMP packet in trace).**



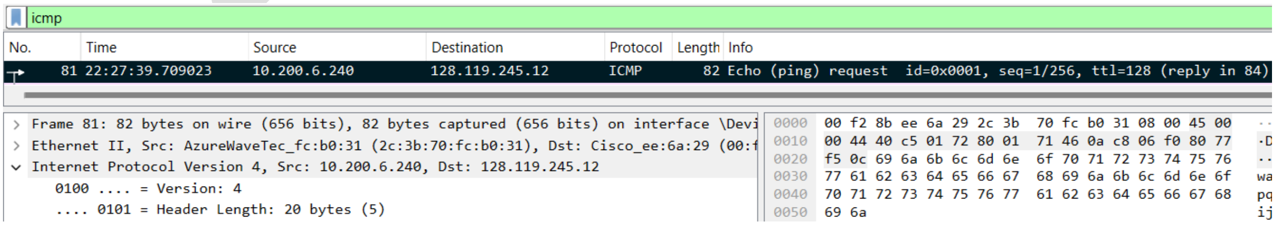
TTL field value (for 1st ICMP packet): 128

3. **What is the value in the upper layer protocol field in this IPv4 datagram's header? [Note: the answers for Linux/MacOS differ from Windows here].**



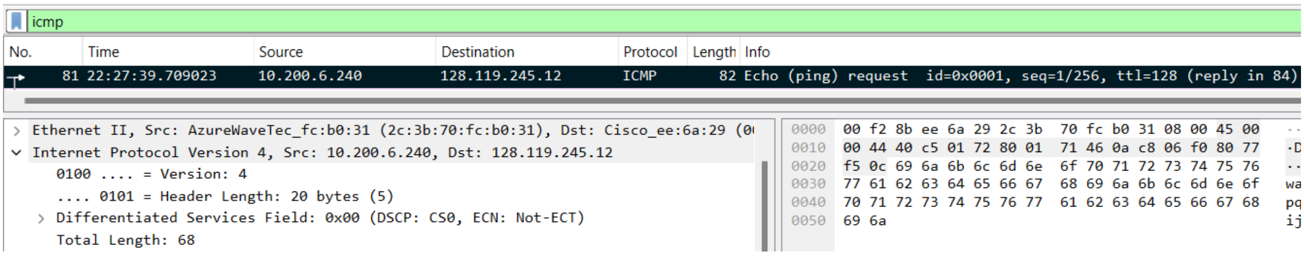
Upper layer protocol: ICMP

4. **How many bytes are in the IP header?**



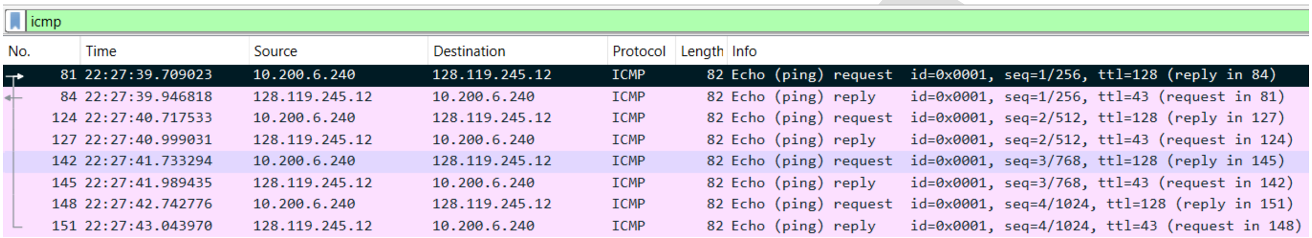
IP Header Size: 20 bytes

5. **How many bytes are in the payload of the IP datagram? Explain how you determined the number of payload bytes.**



IP Datagram Payload: 62 bytes (82 total – 20 header bytes)

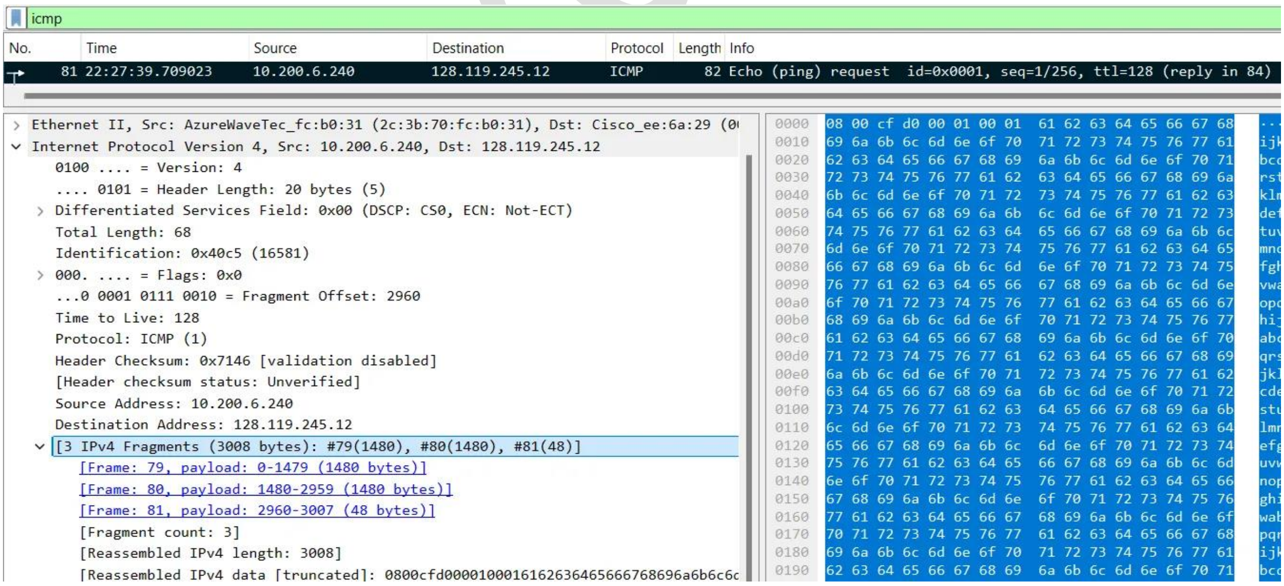
6. **Are the values of the TTL fields similar, across all of ICMP packets from all of the routers?**



No, the TTL values are not similar. There are two distinct patterns:

- For outgoing requests from 10.200.6.240, TTL = 128
- For incoming replies from 128.119.245.12, TTL = 43

7. **Has this IP datagram been fragmented? Explain how you determined whether the datagram has been fragmented.**



Fragment count: 3

- Frame 79: 0-1479 (1480 bytes)
- Frame 80: 1480-2959 (1480 bytes)
- Frame 81: 2960-3007 (48 bytes)

8. Describe the pattern you see in the values in the Identification field of the IP datagrams being sent by your computer.

_ws.col.protocol == "IPv4"

No.	Time	Source	Destination	Protocol	Length	Info
79	22:27:39.709000	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=40c5) [Reassembled in #81]
80	22:27:39.709018	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=40c5) [Reassembled in #81]
82	22:27:39.946800	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=425a) [Reassembled in #84]
83	22:27:39.946817	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=425a) [Reassembled in #84]
122	22:27:40.717501	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=40c6) [Reassembled in #124]
123	22:27:40.717525	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=40c6) [Reassembled in #124]
125	22:27:40.999025	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=442e) [Reassembled in #127]
126	22:27:40.999029	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=442e) [Reassembled in #127]
140	22:27:41.733225	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=40c7) [Reassembled in #142]
141	22:27:41.733285	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=40c7) [Reassembled in #142]
143	22:27:41.989416	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4629) [Reassembled in #145]
144	22:27:41.989434	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=4629) [Reassembled in #145]
146	22:27:42.742746	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=40c8) [Reassembled in #148]
147	22:27:42.742770	10.200.6.240	128.119.245.12	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=40c8) [Reassembled in #148]
149	22:27:43.043964	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4a04) [Reassembled in #151]
150	22:27:43.043969	128.119.245.12	10.200.6.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=4a04) [Reassembled in #151]

The ID field follows a sequential hexadecimal pattern: 40c5, 425a, 40c6, 442e, 40c7, 4629, 40c8, 4a04.

We can observe that the ID values increment sequentially for each new fragmented datagram set, ensuring that fragments of the same original datagram share the same ID.

9. Which fields in the IP datagram always change from one datagram to the next within this series of UDP segments sent by your computer destined to 128.119.245.12, via traceroute? Why?

Changing fields:

- ID field
- TTL
- Checksum
- Source/Destination IP/Ports

10. Which fields in this sequence of IP datagrams (containing UDP segments) stay constant? Why? (Note: if you find your packet has not been fragmented, you should download the zip file "http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces-8.1.zip" and extract the trace file ip-wireshark-trace1-1.pcapng . If your computer has an Ethernet or WiFi interface, a packet size of 3000 should cause fragmentation)

Constant fields:

- Version (4)
- Source IP

- Destination IP
- Protocol

11. Find the first IP datagram containing the first part of the segment sent to 128.119.245.12 sent by your computer via the traceroute command to gaia.cs.umass.edu, after you specified that the traceroute packet length should be 3000. (Hint: This is packet 179 in the ip-wireshark-trace1-1.pcapng trace file in footnote 2. Packets 179, 180, and 181 are three IP datagrams created by fragmenting the first single 3000-byte UDP segment sent to 128.119.145.12). Has that segment been fragmented across more than one IP datagram? (Hint: the answer is yes!)

For the 3000-byte traceroute packet:

Yes, the segment has been fragmented across multiple IP datagrams. This can be confirmed by the packet details which showed:

Total of 3 fragments:

- Frame 79: payload 0-1479 (1480 bytes)
- Frame 80: payload 1480-2959 (1480 bytes)
- Frame 81: payload 2960-3007 (48 bytes)

The total reassembled IPv4 length was 3008 bytes, and the fragmentation was necessary because the size exceeded the maximum transmission unit (MTU) of the network path.

13. What information in the IP header indicates that this datagram been fragmented?

Flags field showing fragmentation flags

Fragment offset values (0, 1480, 2960)

Same ID (40c5) across fragments

14. What information in the IP header for this packet indicates whether this is the first fragment versus a latter fragment?

Fragment offset = 0 indicates first fragment

Non-zero offset (1480, 2960) indicates latter fragments

More fragments (MF) flag is set for all except last fragment

15. How many bytes are there in is this IP datagram (header plus payload)?

Size of IP datagram (header + payload): 1514 bytes (as shown in the "Length" field of each fragment)

16. What fields change in the IP header between the first and second fragment?

Fragment offset (changes from 0 to 1480)

Header checksum

Total length (for last fragment)

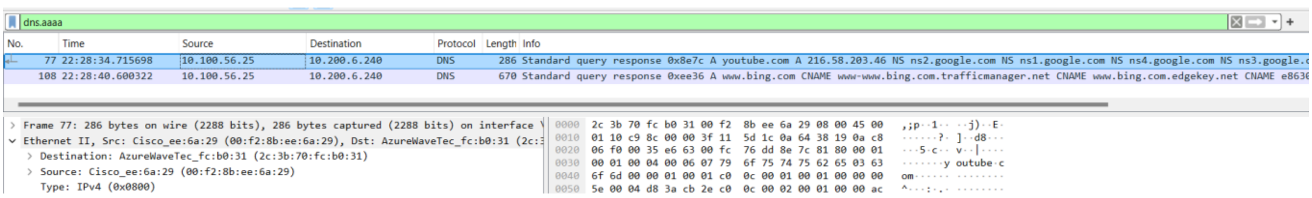
17. Now find the IP datagram containing the third fragment of the original UDP segment. What information in the IP header indicates that this is the last fragment of that segment? (The DNS AAAA request type is used to resolve names to IPv6 IP addresses.)

More Fragments (MF) flag is not set

Fragment offset is 2960

Smaller payload size (48 bytes vs 1480 bytes in earlier fragments)

18. What is the IPv6 address of the computer making the DNS AAAA request? Give the IPv6 source address for this datagram in the exact same form as displayed in the Wireshark window 13 (Recall that an IPv6 address is shown as 8 sets of 4 hexadecimal digits, with each set separated by colons, and with leading zeros omitted. If an IPv6 address has two colons in a row ('::'), this is shorthand meaning that all of the intervening bytes between the two colons are zero. Thus, for example, fe80::1085:6434:583:e79 is shorthand for fe80:0000:0000:0000:1085:6434:0583:0e79. Make sure you understand this example)



Source IP (IPv6): 2c:3b:70:fc:b0:31

19. What is the IPv6 destination address for this datagram? Give this IPv6 address in the exact same form as displayed in the Wireshark window.

Source Address (IPv6): 00:f2:8b:ee:6a:29

Destination Address (IPv6): 2c:3b:70:fc:b0:31

20. For IPv6 packets, describe any notable differences in structure compared to IPv4. How are headers formatted differently?

<u>No fragmentation field in IPv6</u>: IPv4 has fragmentation fields that are not present in IPv6 since IPv6 handles fragmentation differently.

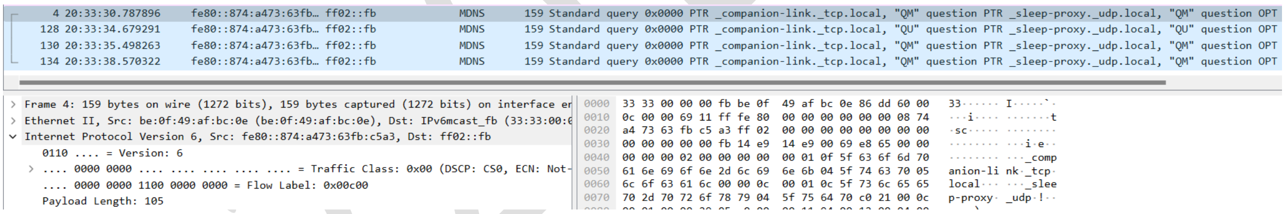
<u>No checksum in IPv6</u>: IPv4 headers have a checksum field, but IPv6 headers do not, which simplifies the packet processing.

<u>Larger addresses</u>: IPv4 uses 32-bit addresses, while IPv6 uses 128-bit addresses.

<u>Fixed header size</u>: IPv6 headers have a fixed size of 40 bytes, while IPv4 headers are variable in length (20–60 bytes).

<u>Flow label</u>: IPv6 includes a flow label for identifying packet flows requiring special handling, absent in IPv4.

21. How much payload data is carried in the 2nd IPv6 datagram?



Payload length: 105

22. What is the upper layer protocol to which 2nd datagram's payload will be delivered at the destination? (Lastly, find the IPv6 DNS response to the IPv6 DNS AAAA request made in the this trace. This DNS response contains IPv6 addresses for youtube.com)

UDP (17)

23. How many IPv6 addresses are returned in the response to this AAAA request?

2001:4860:4802:32::a, 2001:4860:4802:36::a

24. What is the first of the IPv6 addresses returned by the DNS for youtube? Give this IPv6 address in the exact same shorthand form as displayed in the Wireshark window.

2001:4860:4802:32::a