

Assignment 2: Use Wireshark to Examine Network Traffic

RENE OLUOCH
CS-CNS09-25030

Contents

Introduction	2
Local ICMP Traffic Analysis	2
Experiment Setup	2
Observations	4
Interpretation and Key Learnings	4
Remote ICMP Traffic Analysis	5
Experiment Setup	5
Observations	5
Interpretation and Key Learnings	8
ARP and MAC Resolution	8
Experiment Setup	8
Observations	9
Interpretation and Key Learnings	9
Appendix A and Beyond – What I Did	10
Conclusion	10

Introduction

In this assignment, I used Wireshark, a network protocol analyzer, to study ICMP traffic. The aim was to gain hands-on understanding of how ICMP packets behave during both local and remote communication. My analysis involved capturing ICMP packets generated by the ping command and examining their structure in different networking scenarios.

I began by performing ping tests to my default gateway to observe how communication happens within a local network. I then proceeded to ping several external websites, including Google, Cisco, and Yahoo, to analyze how packets are handled in remote communication. Throughout these experiments, I closely monitored the packet flows and headers using Wireshark, focusing on how MAC and IP addresses are used and how the Address Resolution Protocol (ARP) resolves MAC addresses.

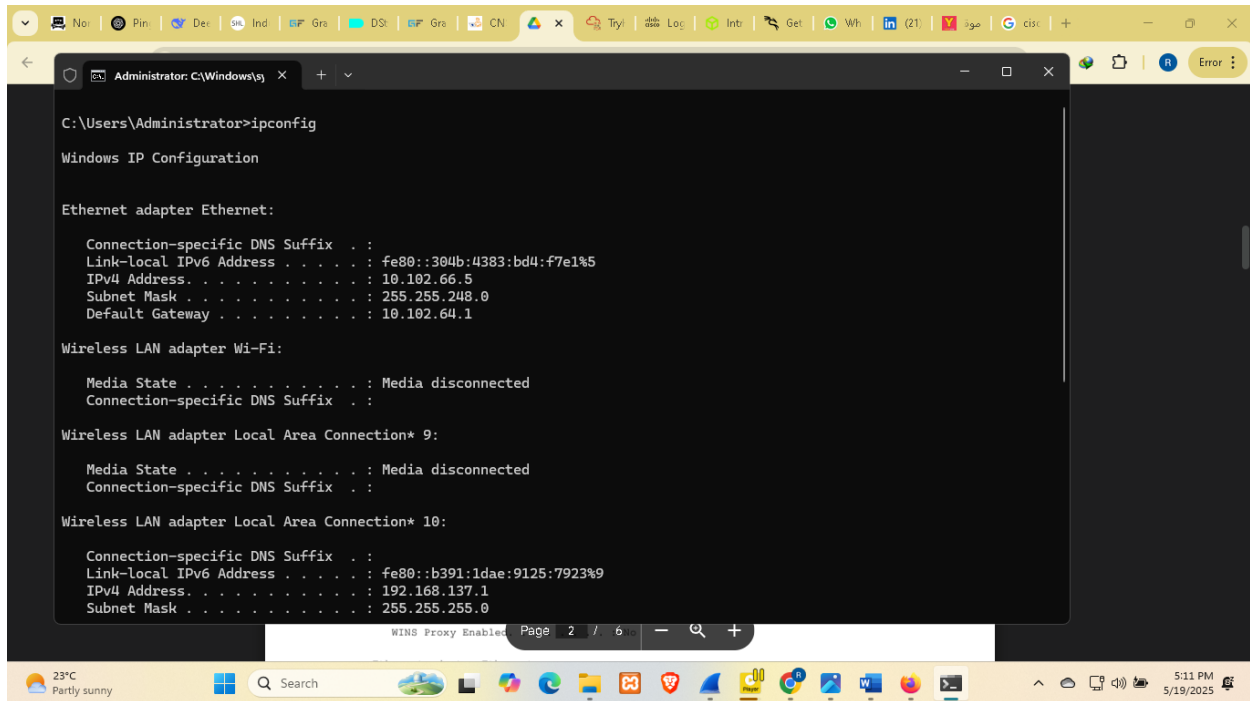
This activity helped me better understand the relationship between Layer 2 (Data Link Layer) and Layer 3 (Network Layer) protocols, the behavior of ICMP, and the role of ARP in resolving addresses within the LAN.

Local ICMP Traffic Analysis

Experiment Setup

To begin the local traffic analysis, I first retrieved the IP and MAC addresses of my personal computer. I did this by executing the `ipconfig` command in the command prompt. The IP address I obtained was 10.102.66.5, and the MAC address was B0-5C-DA-A1-97-DF. I then used the `ping` command to send ICMP Echo Requests to my default gateway, which had the IP address 10.102.64.1.

While these ping requests were running, I launched Wireshark to capture the traffic and inspect the resulting ICMP packets, along with the Ethernet headers and any accompanying ARP packets.



```
C:\Users\Administrator>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::304b:4383:bd4:f7e1%5
    IPv4 Address. . . . . : 10.102.66.5
    Subnet Mask . . . . . : 255.255.248.0
    Default Gateway . . . . . : 10.102.64.1

Wireless LAN adapter Wi-Fi:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

Wireless LAN adapter Local Area Connection* 9:

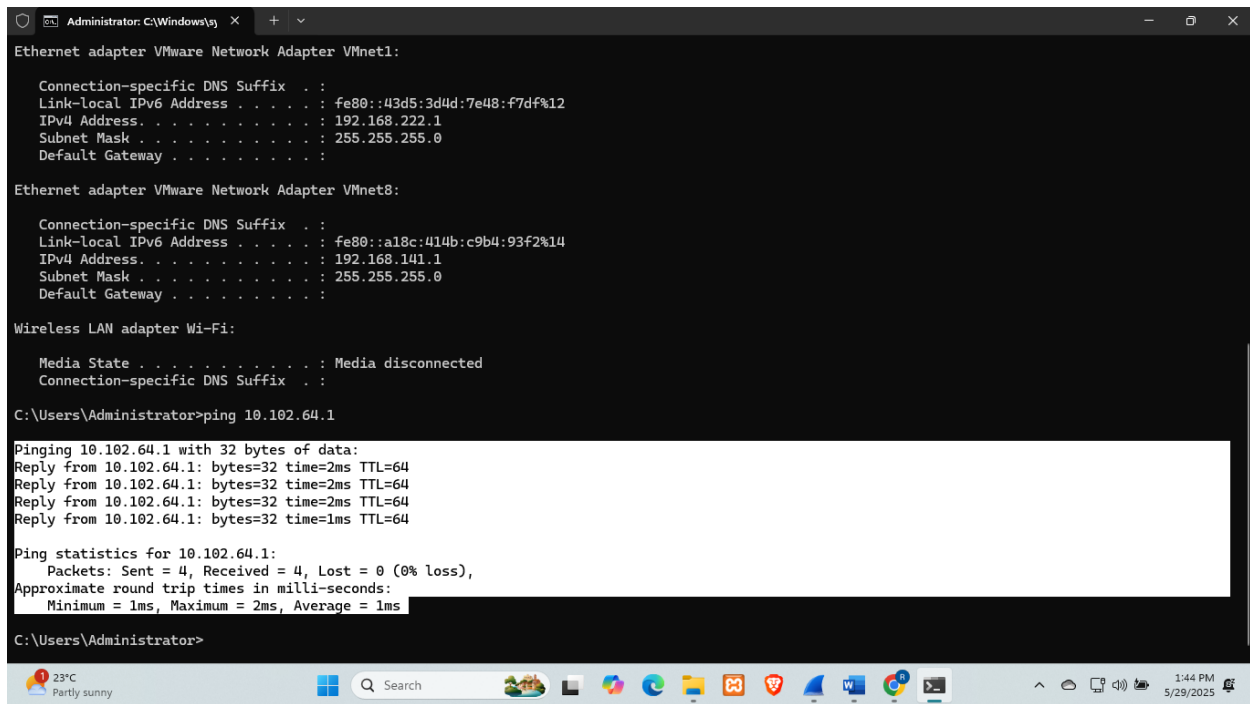
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

Wireless LAN adapter Local Area Connection* 10:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::b391:1dae:9125:7923%9
    IPv4 Address. . . . . : 192.168.137.1
    Subnet Mask . . . . . : 255.255.255.0
```

Retrieving my ip and mac address

I then pinged my router, whose IP I also got from ipconfig command above while retrieving my ip and captured the traffic in wireshark. I knew it was my router as it was labeled default gateway. I got replies from the default gateway as shown in the screenshot below



```
Ethernet adapter VMware Network Adapter VMnet1:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::43d5:3d4d:7e48:f7df%12
    IPv4 Address. . . . . : 192.168.222.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 

Ethernet adapter VMware Network Adapter VMnet8:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::a18c:414b:c9b4:93f2%14
    IPv4 Address. . . . . : 192.168.141.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 

Wireless LAN adapter Wi-Fi:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

C:\Users\Administrator>ping 10.102.64.1

Pinging 10.102.64.1 with 32 bytes of data:
Reply from 10.102.64.1: bytes=32 time=2ms TTL=64
Reply from 10.102.64.1: bytes=32 time=2ms TTL=64
Reply from 10.102.64.1: bytes=32 time=2ms TTL=64
Reply from 10.102.64.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.102.64.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 2ms, Average = 1ms

C:\Users\Administrator>
```

After capturing the traffic in Wireshark, I observed a clear exchange of ICMP Echo Request and Echo Reply packets between my PC and the gateway. The Ethernet frames of these packets included my computer's MAC address as the source and the gateway's MAC address as the destination. Prior to the ICMP traffic, I also noticed ARP packets. These were used to resolve the gateway's IP address to its MAC address, allowing my PC to deliver the ICMP packet on the local network.



From this experiment, I learned that when communicating within a local network, both the source and destination MAC addresses are included in the Ethernet frame. Since the gateway was on the same subnet as my PC, ARP was necessary to resolve its MAC address before the ICMP packet could be sent.

This experiment reinforced my understanding of how devices communicate on a local network and how ARP facilitates that communication by mapping IP addresses to MAC addresses.

Remote ICMP Traffic Analysis

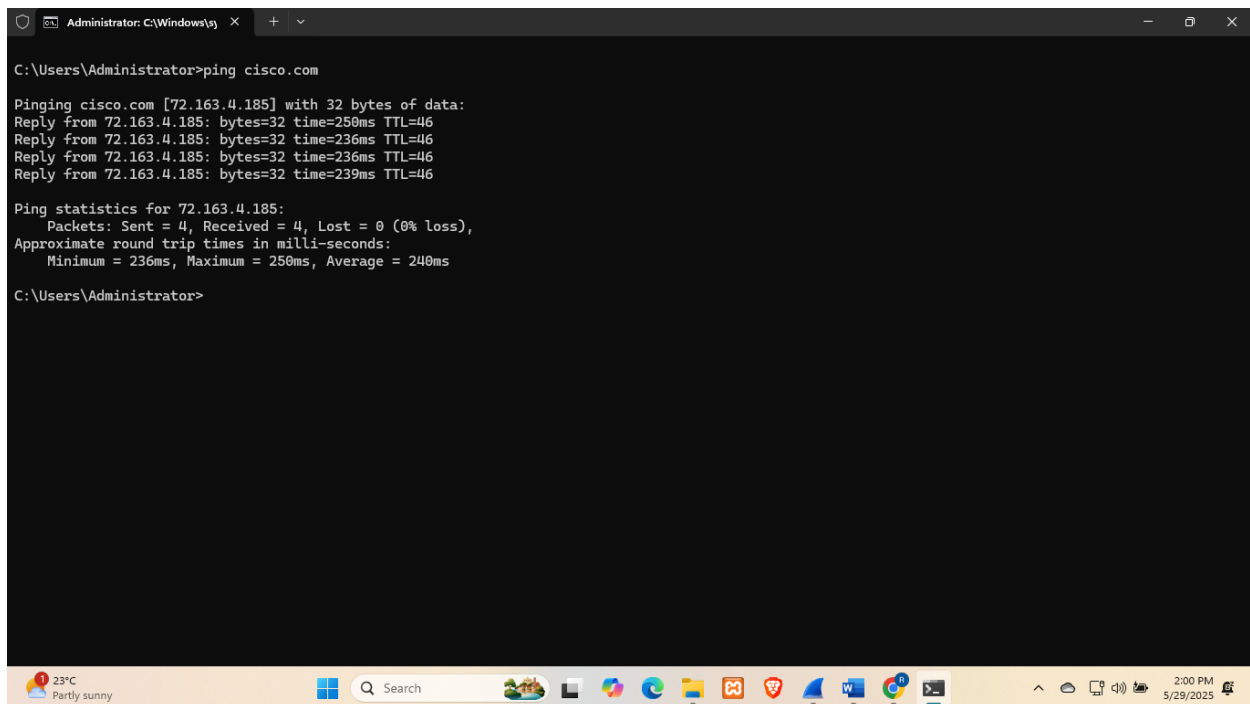
Experiment Setup

To analyze remote ICMP traffic, I decided to ping a set of external websites. I chose www.google.com, www.cisco.com, and www.yahoo.com for this test. Each website resolved to a different public IP address. While the pings were running, I captured the network traffic using Wireshark to study how ICMP packets behave when they traverse beyond the local network.

Observations

In Wireshark, I saw the ICMP Echo Request packets being sent from my PC to the external IP addresses. However, unlike the local ping test, the Ethernet frames only showed my MAC address as the source and my gateway's MAC address as the destination. The MAC addresses of the remote servers were not visible at all.

Additionally, I noticed that the Time To Live (TTL) values in the IP headers decreased with each response, indicating that the packets had passed through multiple routers or hops before reaching the destination.



```
C:\Users\Administrator>ping cisco.com

Pinging cisco.com [72.163.4.185] with 32 bytes of data:
Reply from 72.163.4.185: bytes=32 time=250ms TTL=46
Reply from 72.163.4.185: bytes=32 time=236ms TTL=46
Reply from 72.163.4.185: bytes=32 time=236ms TTL=46
Reply from 72.163.4.185: bytes=32 time=239ms TTL=46

Ping statistics for 72.163.4.185:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 236ms, Maximum = 250ms, Average = 240ms

C:\Users\Administrator>
```

Pinging cisco.com

The screenshot shows a Wireshark packet capture on the Ethernet interface. The packet list on the left shows several ICMP Echo (ping) requests and replies. The selected packet (No. 376) is an ICMP Echo (ping) request from 10.102.66.5 to 72.163.4.185. The packet details pane shows the Internet Protocol Version 4 header and the ICMP Echo (ping) request structure. The packet bytes pane shows the raw data of the packet, including the IP header and the ICMP Echo (ping) request data.

No.	Time	Source	Destination	Protocol	Length	Info
376	4.056997	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=30/7680, ttl=128 (reply in 376)
426	4.822908	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) reply id=0x0100, seq=30/7680, ttl=46 (request in 376)
447	5.059003	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=31/7936, ttl=128 (reply in 447)
519	5.841592	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=31/7936, ttl=46 (request in 447)
542	6.077530	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=32/8192, ttl=128 (reply in 542)
600	6.860148	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=32/8192, ttl=46 (request in 542)
630	7.099247	10.102.66.5	72.163.4.185	ICMP	74	Echo (ping) request id=0x0100, seq=33/8448, ttl=128 (reply in 630)

Packet details for selected packet (No. 376):

- Frame 355: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF{...}
- Ethernet II, Src: HP_al197:df (b0:5c:da:a1:97:df), Dst: TpLinkTechno_4f:66:61 (34:f7:16:4f:66:61)
- Internet Protocol Version 4, Src: 10.102.66.5, Dst: 72.163.4.185
- 0100 = Version: 4
- 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 60
- Identification: 0x0b7d (27517)
- 0000 = Flags: 0x0
- ... 0 0000 0000 0000 = Fragment Offset: 0
- Time to Live: 128
- Protocol: ICMP (1)
- Header Checksum: 0x0000 [validation disabled]
- [Header checksum status: Unverified]
- Source Address: 10.102.66.5
- Destination Address: 72.163.4.185
- Internet Control Message Protocol: Protocol

Packet bytes (0000-0040):

```

0000 34 f7 16 4f 66 61 b0 5c da a1 97 df 08 00 45 00 4-Ofa\ .....E
0010 00 3c 68 7d 00 00 80 01 00 00 66 42 05 48 a3 4d}.... .fbH+
0020 04 b9 06 00 4c 3e 01 00 00 1e 61 62 63 64 65 66 ..L... .abcdef
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklm opqrstuv
0040 77 61 62 63 64 65 66 67 68 69 wabdefgh i
  
```

Capturing cisco icmp packets

The screenshot shows a Windows command prompt window with the following output:

```

C:\Users\Administrator>ping yahoo.com

Pinging yahoo.com [74.6.143.25] with 32 bytes of data:
Reply from 74.6.143.25: bytes=32 time=211ms TTL=47
Reply from 74.6.143.25: bytes=32 time=213ms TTL=47
Reply from 74.6.143.25: bytes=32 time=210ms TTL=47
Reply from 74.6.143.25: bytes=32 time=211ms TTL=47

Ping statistics for 74.6.143.25:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 210ms, Maximum = 213ms, Average = 211ms

C:\Users\Administrator>
  
```

Pinging yahoo.com

Wireshark interface showing ICMP traffic. The packet list displays several ICMP Echo (ping) requests and replies between 10.102.66.5 and 74.6.143.25. The packet details pane shows the structure of an ICMP Echo request (Frame 300).

No.	Time	Source	Destination	Protocol	Length	Info
145	1.999088	10.102.66.5	74.6.143.25	ICMP	74	Echo (ping) request id=0x0100, seq=26/6656, ttl=128 (reply in 159)
159	2.209930	74.6.143.25	10.102.66.5	ICMP	74	Echo (ping) reply id=0x0100, seq=26/6656, ttl=47 (request in 145)
226	3.018610	10.102.66.5	74.6.143.25	ICMP	74	Echo (ping) request id=0x0100, seq=27/6912, ttl=128 (reply in 242)
242	3.230961	74.6.143.25	10.102.66.5	ICMP	74	Echo (ping) reply id=0x0100, seq=27/6912, ttl=47 (request in 226)
300	4.035655	10.102.66.5	74.6.143.25	ICMP	74	Echo (ping) request id=0x0100, seq=28/7168, ttl=128 (reply in 315)
315	4.245523	74.6.143.25	10.102.66.5	ICMP	74	Echo (ping) reply id=0x0100, seq=28/7168, ttl=47 (request in 300)
371	5.054387	10.102.66.5	74.6.143.25	ICMP	74	Echo (ping) request id=0x0100, seq=29/7424, ttl=128 (reply in 392)
392	5.265094	74.6.143.25	10.102.66.5	ICMP	74	Echo (ping) reply id=0x0100, seq=29/7424, ttl=47 (request in 371)

Frame 300: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF{...} Ethernet II, Src: HP_a1:97:df (b0:5c:da:a1:97:df), Dst: TpLinkTechno_4f:66:61 (34:f7:16:4f:66:61) Internet Protocol Version 4, Src: 10.102.66.5, Dst: 74.6.143.25

0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 60
Identification: 0x2b06 (11014)
> 000. = Flags: 0x0
... 0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: ICMP (1)
Header Checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.102.66.5
Destination Address: 74.6.143.25

0000 34 f7 16 4f 66 61 b0 5c da a1 97 df 08 00 45 00 4..Ofa\E
0010 00 3c 2b 06 00 00 80 01 00 00 0a 66 42 05 4a 06 -c+... ..fbj
0020 8f 19 08 00 4c 40 01 00 00 1c 61 62 63 64 65 66L@...abedef
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklm opqrstuv
0040 77 61 62 63 64 65 66 67 68 69 wabdefgh i

Capturing yahoo icmp packets

Windows Command Prompt showing the results of a ping command to google.com.

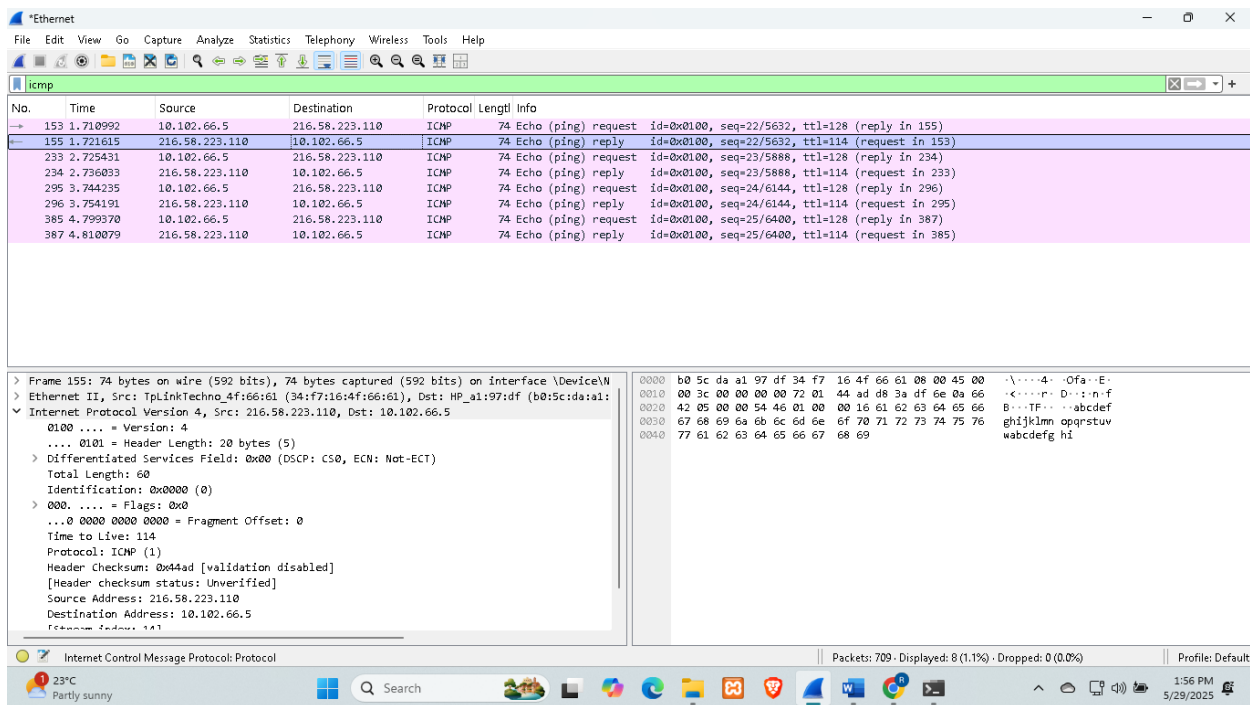
```
C:\Users\Administrator>ping google.com

Pinging google.com [216.58.223.110] with 32 bytes of data:
Reply from 216.58.223.110: bytes=32 time=11ms TTL=114
Reply from 216.58.223.110: bytes=32 time=11ms TTL=114
Reply from 216.58.223.110: bytes=32 time=10ms TTL=114
Reply from 216.58.223.110: bytes=32 time=11ms TTL=114

Ping statistics for 216.58.223.110:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 10ms, Maximum = 11ms, Average = 10ms

C:\Users\Administrator>
```

Pinging google.com



Capturing google icmp packets

Interpretation and Key Learnings

This experiment made it clear that MAC addresses are only relevant within a single network segment. Since remote servers are not on the same local network, their MAC addresses are not included in the Ethernet frames. Instead, all ICMP traffic is sent to the default gateway, which then forwards the packets using IP routing.

I also learned that each router along the path replaces the MAC address in the Ethernet frame with one suitable for the next link in the route. This confirms that MAC addressing is only used between adjacent devices on a single link, while IP addressing is used for end-to-end communication.

Furthermore, by examining the TTL values, I understood how many hops the ICMP packets had taken. This insight helped me understand how tools like traceroute use TTL to discover the path that packets take through a network.

ARP and MAC Resolution

Experiment Setup

To explore how ARP works as extra work, I executed the `arp -a` command on my computer. This command displays the ARP cache, which contains IP-to-MAC mappings that my PC has recently resolved.

```
C:\Users\Administrator>arp -a

Interface: 10.102.66.5 --- 0x5
Internet Address      Physical Address      Type
10.102.64.1           34-f7-16-4f-66-61    dynamic
10.102.65.71          48-f1-7f-b6-f0-5d    dynamic
10.102.71.255         ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static

Interface: 192.168.137.1 --- 0x9
Internet Address      Physical Address      Type
192.168.137.229       fa-87-e2-cc-03-b4    static
192.168.137.249       7e-ae-8d-d9-1f-07    static
192.168.137.255       ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static

Interface: 192.168.222.1 --- 0xc
Internet Address      Physical Address      Type
192.168.222.254       00-50-56-e2-5b-e5    dynamic
192.168.222.255       ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static

Interface: 192.168.141.1 --- 0xe
```

Arp – a command

Observations

The ARP cache showed entries for devices on my local network, including the default gateway. For example, I saw an entry showing that the IP address 10.102.64.1 was mapped to the physical address of the gateway. However, there were no entries for the IP addresses of Google, Cisco, or Yahoo, even though I had recently pinged those servers.

Interpretation and Key Learnings

Through this activity, I learned that ARP is limited to the local area network. It only resolves MAC addresses for devices within the same subnet. When I pinged external websites, my computer sent the ICMP packets to the gateway's MAC address, and the gateway took over the responsibility of forwarding the packets to the wider internet.

This experiment demonstrated that MAC resolution does not occur across the internet. Instead, routers handle IP-level forwarding, and each router only resolves the MAC address of the next device in the path. This understanding clarified the role and limitations of ARP as a Layer 2 protocol.

Appendix A and Beyond – What I Did

After completing the earlier parts of the lab, I moved on to the section that deals with ICMP traffic and potential firewall issues. According to the lab, if other devices couldn't ping my PC, it might be due to the Windows Firewall blocking ICMP requests. The appendix provided a detailed guide on how to create a new inbound rule to allow ICMP traffic through the firewall.

However, in my case, I didn't need to go through those steps.

To verify whether ICMP traffic was being blocked, I simply tried pinging my **default gateway**—and the pings were successful. This confirmed that ICMP traffic was already allowed on my machine and that there were no firewall restrictions interfering with ping operations.

Since everything worked as expected and there were no connectivity issues, I didn't have to create or modify any firewall rules. I was able to move forward with the lab without making changes to my system's firewall settings.

Conclusion

This assignment was highly informative and helped me understand the differences in network behavior when dealing with local versus remote traffic. By using Wireshark, I was able to visualize how ICMP packets are structured and how they are transmitted through the network.

From the local ICMP analysis, I learned how Ethernet frames include actual MAC addresses and how ARP is used to resolve them within the LAN. The remote ICMP analysis taught me that MAC addresses are only applicable within a local link and that routing beyond the gateway is purely IP-based. The TTL values offered further insight into how far packets travel, providing a way to estimate the number of network hops.

Finally, by executing the `arp -a` command, I observed how ARP maintains a table of resolved MAC addresses only for local devices, reinforcing the idea that MAC addressing is confined to the LAN.

Overall, this hands-on activity greatly improved my understanding of fundamental networking protocols such as ICMP, ARP, and IP. It also enhanced my ability to use tools like Wireshark for real-time traffic inspection and protocol analysis—skills that are essential for both network administration and cybersecurity analysis.