Network Security

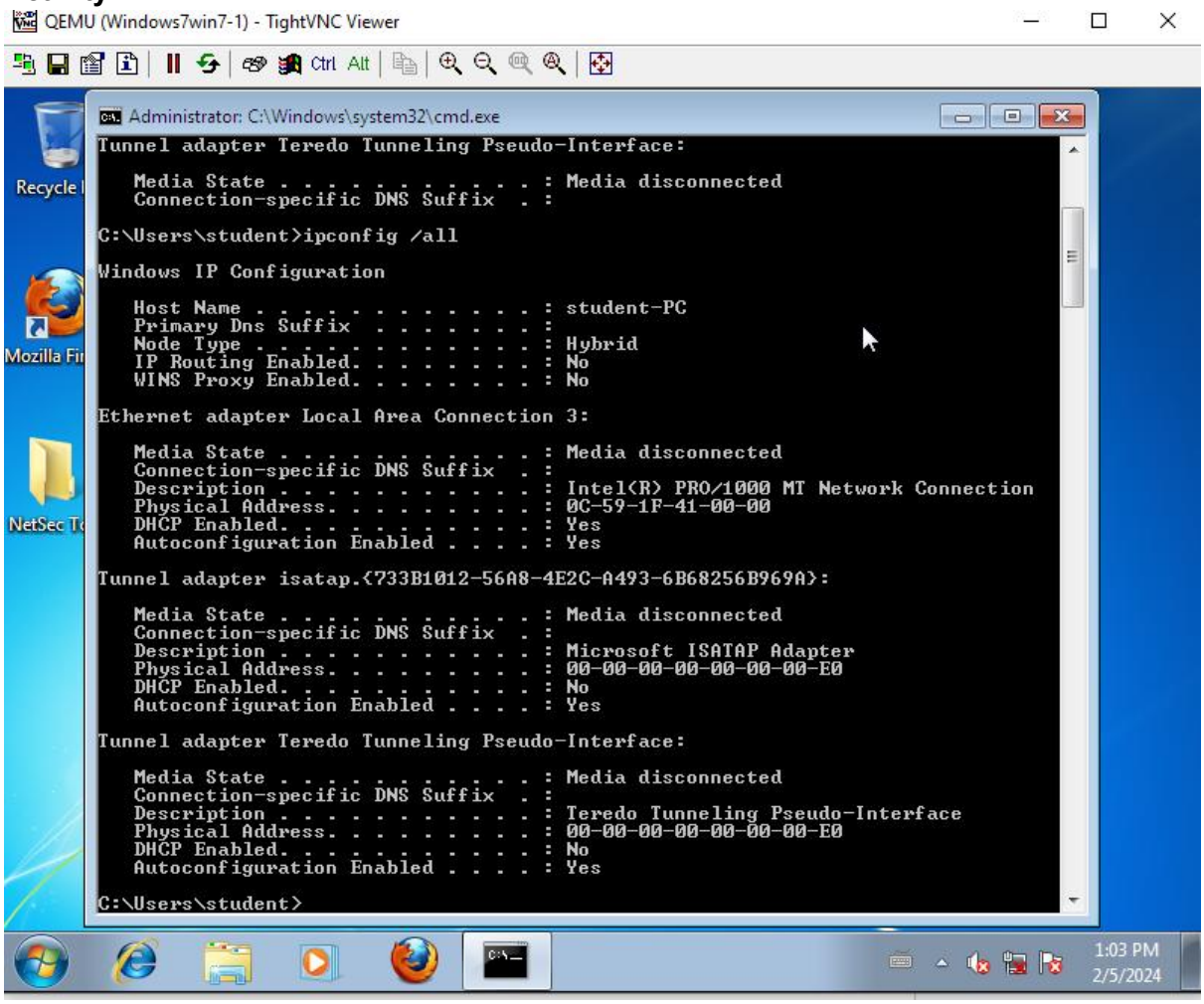
Laboratory Assignment I

Group 08
Yanxiang Du
Yongjie Hao
Siwei Zhang

Spring 2024
2024-03-01

Exercise 1. Ethernet adapters, MAC address, and Ethernet frame forwarding

Activity 1



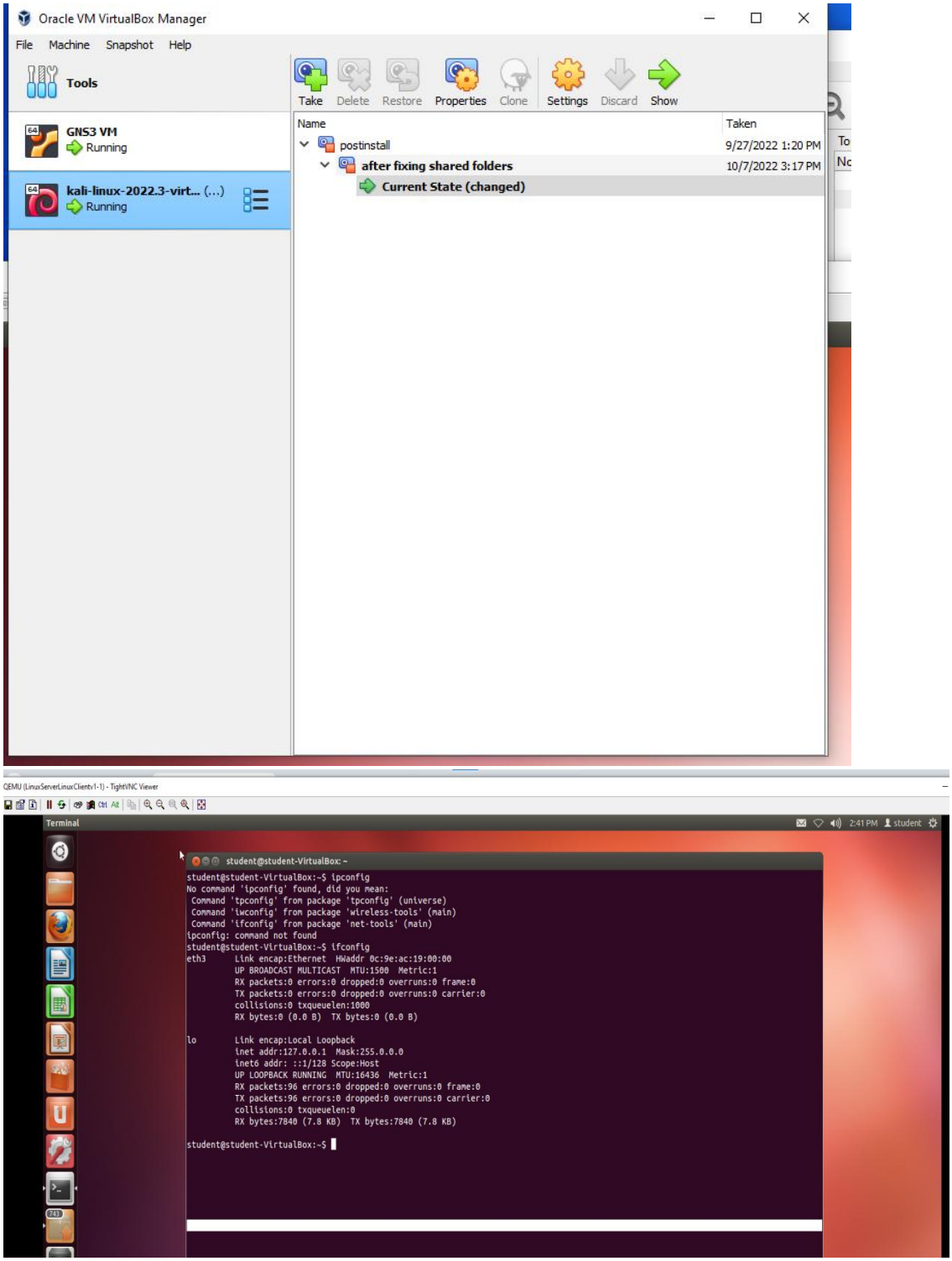
1. **How many Ethernet adapters does the Win7 system have? What are their name(s)/identifier(s)? What does a network adapter represent?**

   There are three Ethernet adapters in Win 7, it is Intel<R> PRO/1000 MT Network Connection, Microsoft ISATAP Adapter, and Teredo Tunneling Pseudo-Interface. A network adapter is a hardware component that allows a computer or other device to connect to a network. The network adapter serves as the interface between the computer and the underlying network infrastructure, facilitating communication with other devices on the same network.

2. **What is the MAC address of these adapter(s)? Based on the MAC value, can you identify the vendor of the network card(s)? What is a MAC address?**

   The manufacturer of network adapters can be determined by examining either the MAC address or the Organizationally Unique Identifier (OUI). Therefore, the OUI '00:00:00' is officially assigned to Xerox, indicating that network adapters with this

OUI are manufactured by Xerox. However, the vendor associated with the OUI '0C-59-1F' cannot be identified in the database.



3. **How many Ethernet adapters does the system have? What are their name(s)/identifier(s)?**

   The system has one Ethernet adapter, named eth3.

4. **What is the MAC address of the adapter(s)? Which is the vendor of the network card(s)?**

   The MAC address of the adapter(s) in Linux is "0C:9E: AC:19:00:00". Unfortunately, I couldn't find the vendor information for this MAC address on https://www.macvendorlookup.com/. This might be due to an incomplete database or because the MAC address belongs to a customized product from a specific vendor.

5. **How can the MAC address of an adapter be used in network security?**
   MAC addresses can be used in network security in the following ways:

   1. Authentication and Access Control: MAC addresses can serve as unique identifiers for network devices, thus they can be used for authentication and access control. Network administrators can configure network devices to only allow access to devices with specific MAC addresses, thereby enhancing network security.

   2. Intrusion Detection and Prevention: Monitoring MAC address activity on the network can help detect potential intrusion attempts. If unauthorized MAC addresses appear on the network, it may indicate someone is attempting unauthorized access. MAC address-based intrusion detection systems can issue alerts or block unauthorized devices from accessing the network.

   3. Network Traffic Analysis: Collecting and analyzing MAC address information in network traffic can help identify devices and user activity on the network. This aids in monitoring and investigating potential security incidents and identifying any unusual activity or potential security risks.

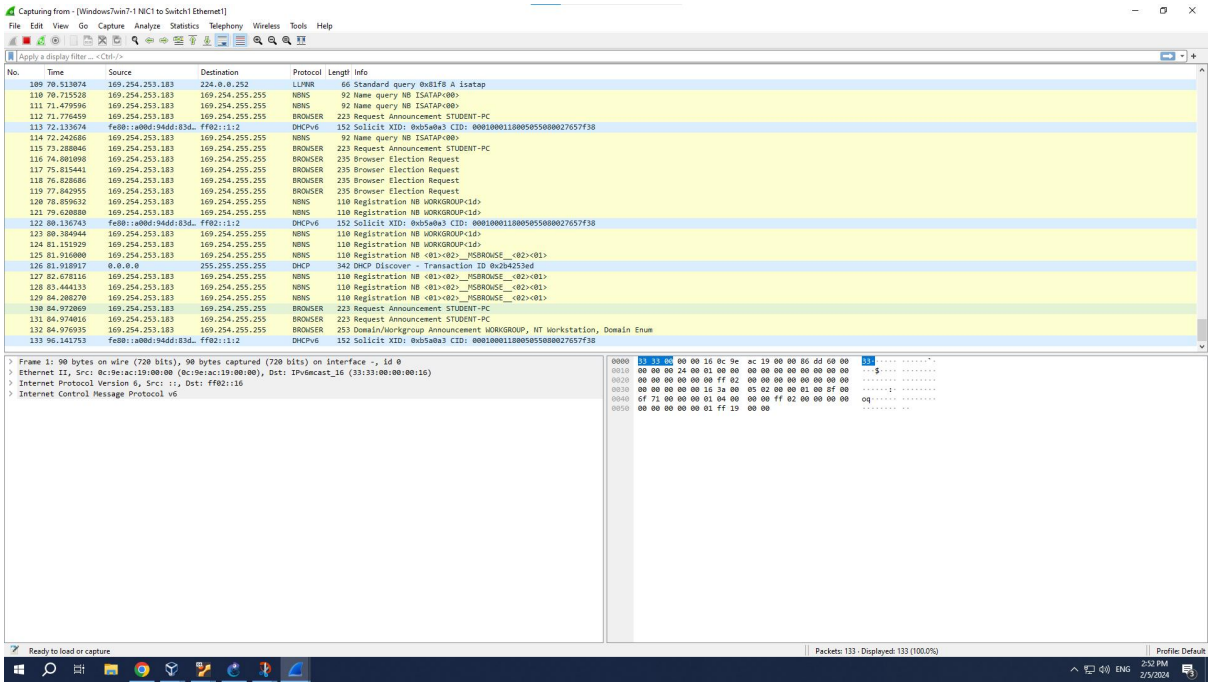
   4. Virtual Local Area Network (VLAN) Management: MAC addresses can be used to configure and manage Virtual Local Area Networks (VLANs). By using MAC addresses, network administrators can assign devices to different VLANs, thereby achieving network resource isolation and enhancing security.

      MAC addresses play a crucial role in network security by facilitating authentication, access control, intrusion detection and prevention, network traffic analysis, and VLAN management, ultimately enhancing network security and manageability.

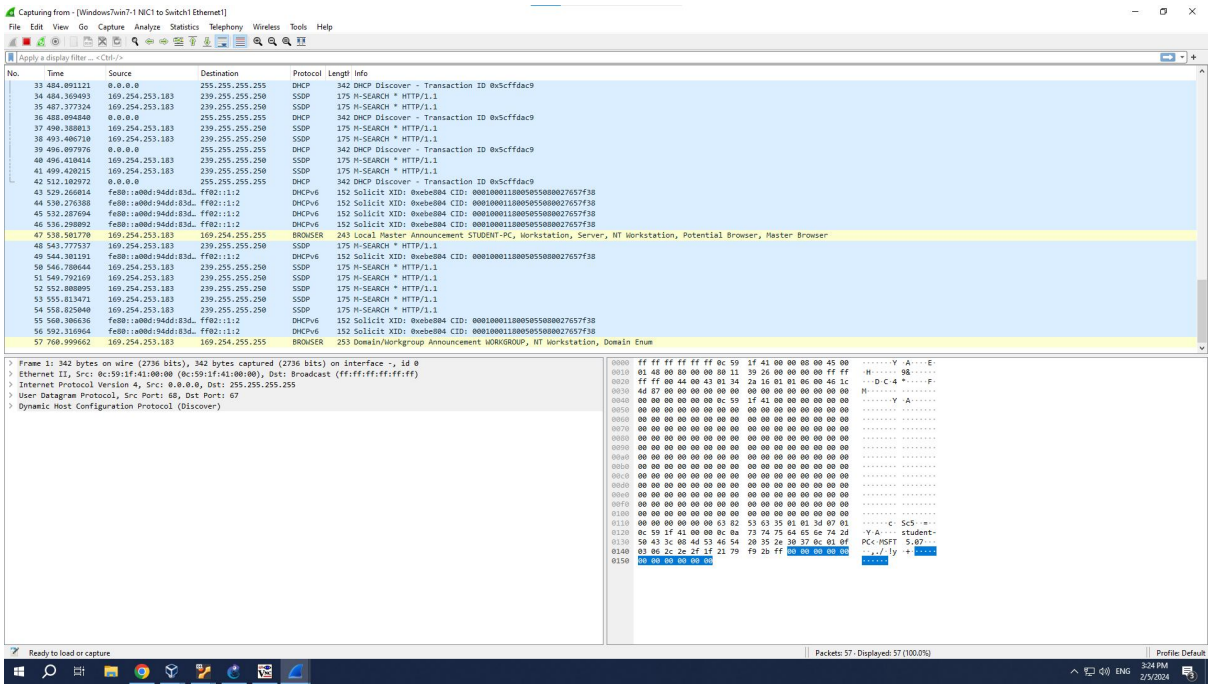
Activity 2

6. **“How many frames are sent? How many bytes are in total? What is the receiving MAC address? Is that the MAC address of the LinuxClient? Also, verify that the frames are sent with the MAC address of Win7’s interface.**

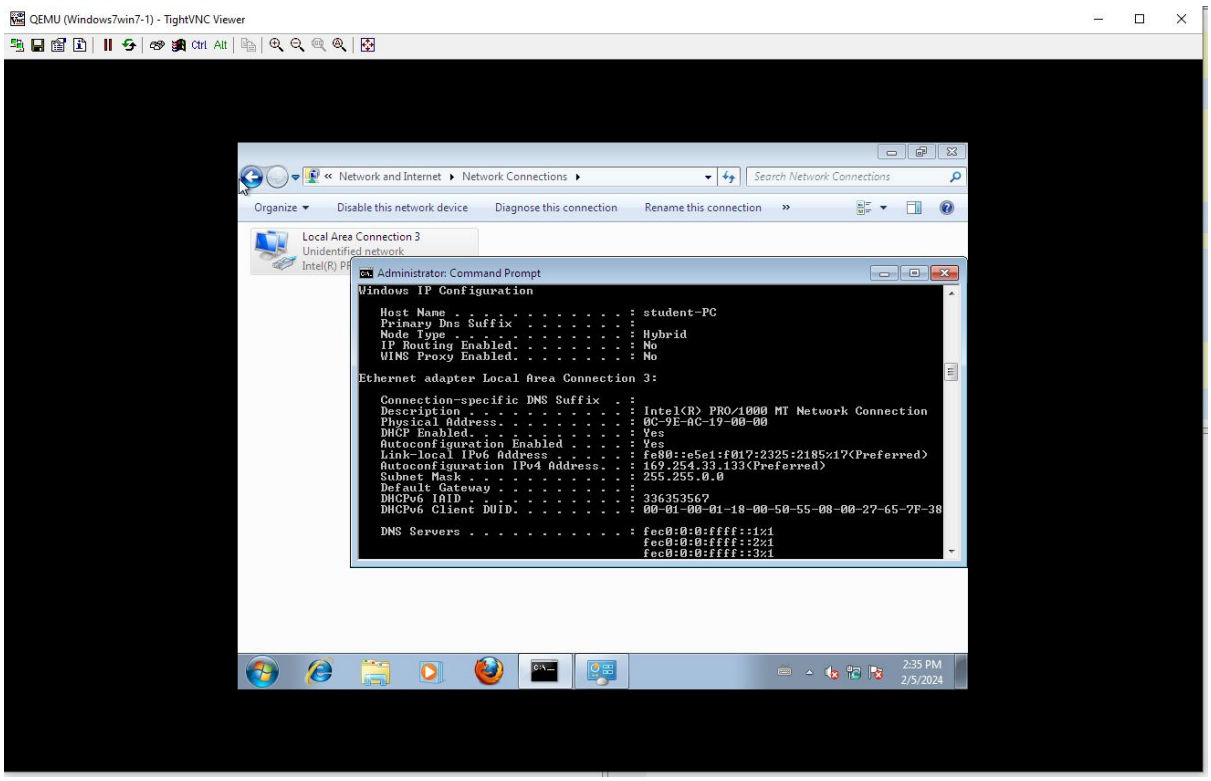
   In LinuxClient, there is one frame sent and there are 90 bytes in total, the receiving MAC address is 33:33:00:00:00:16, which is not the address of the LinuxClient.



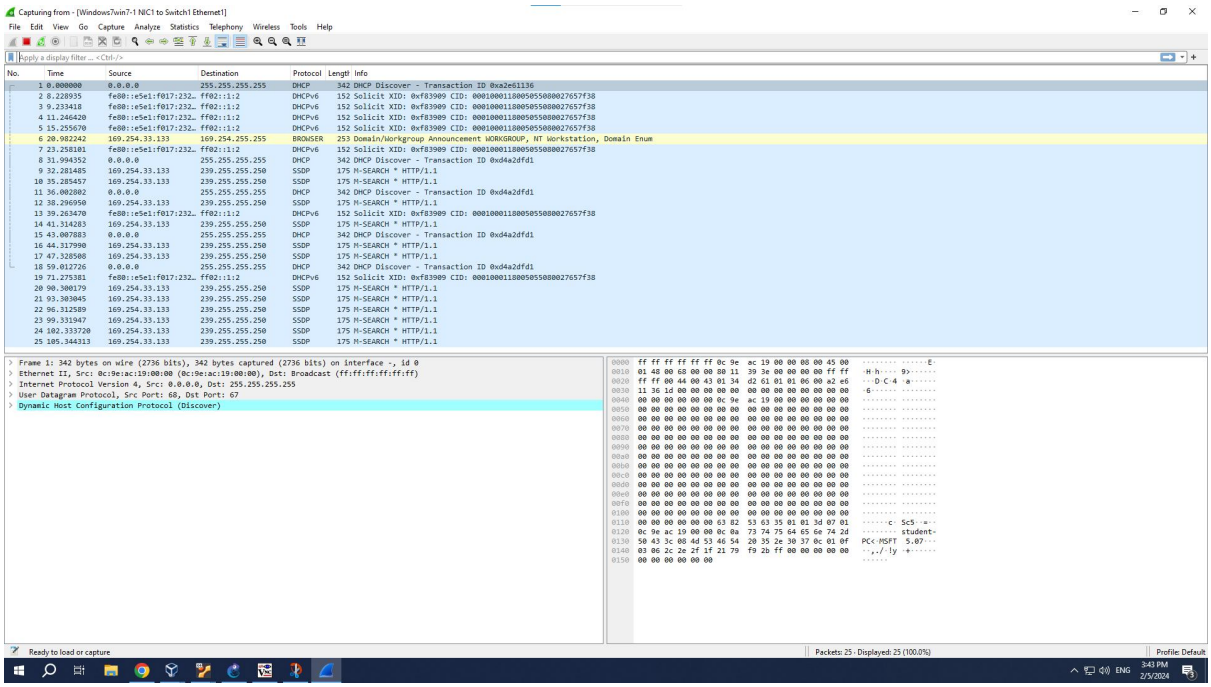
In Win7, there is one frame sent and there are 242 bytes in total, the receiving MAC address is ff:ff:ff:ff:ff:ff, which is not the address of the LinuxClient.



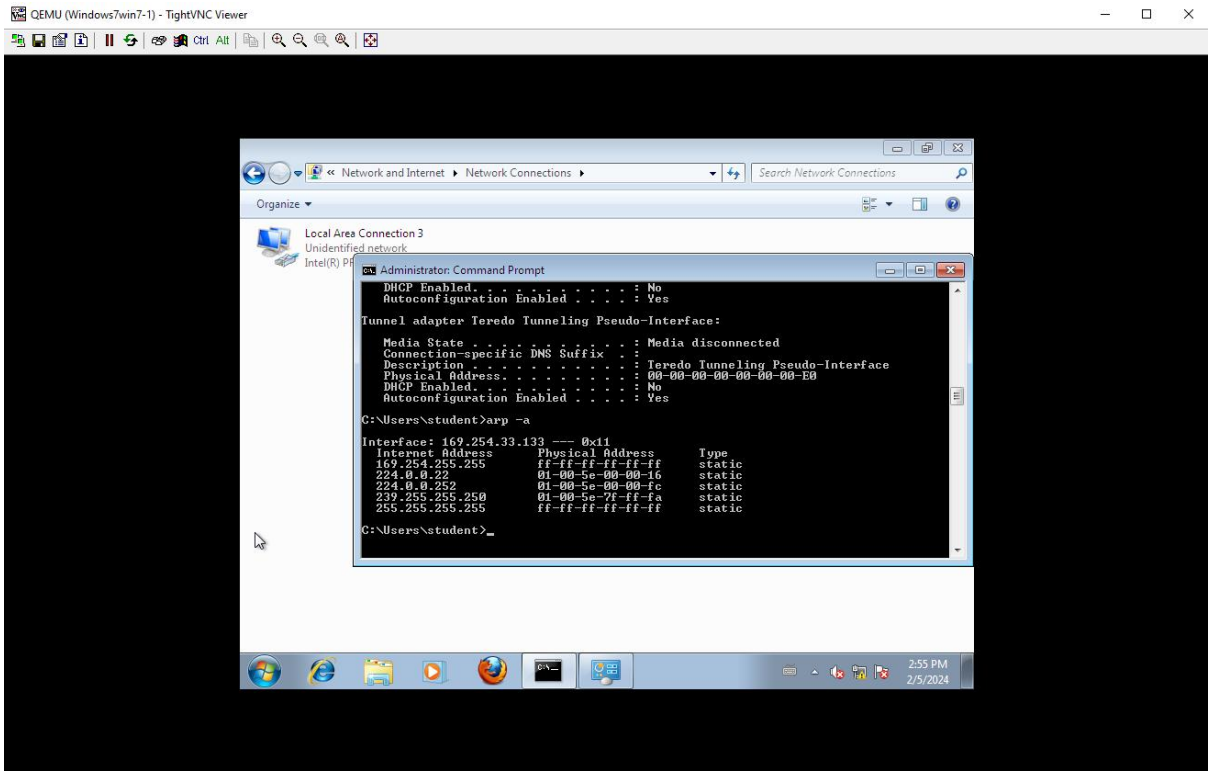
Activity 3



7. **What happens if two computers on the same network have the same MAC addresses?**



The MAC Address has been changed:



Address Resolution Protocol (ARP) conflicts can arise when multiple devices on a local network share the same MAC address. ARP plays a crucial role in mapping IP addresses to MAC addresses, facilitating communication within a network. However, when two devices possess identical MAC addresses, ARP tables can become confused, resulting in unpredictable behavior. This confusion can disrupt communication between devices and hinder network operations.

Switching and routing problems can also emerge due to MAC address conflicts. Network switches utilize MAC addresses to determine the appropriate path for forwarding network traffic. When multiple devices share a MAC address, the switch may struggle to accurately direct traffic, leading to inefficiencies or even network downtime. Routing decisions can also be affected if routers encounter conflicting MAC addresses, potentially disrupting the flow of data between different network segments.

8. **Although it is simple to forge a MAC address, it is difficult to determine the MAC address of a computer that is on a network that you have not already accessed. Why is this true?**

   Although it's easy to forge a MAC address, it's difficult to determine the MAC address of a computer on a network that you haven't already accessed. Why is this true?

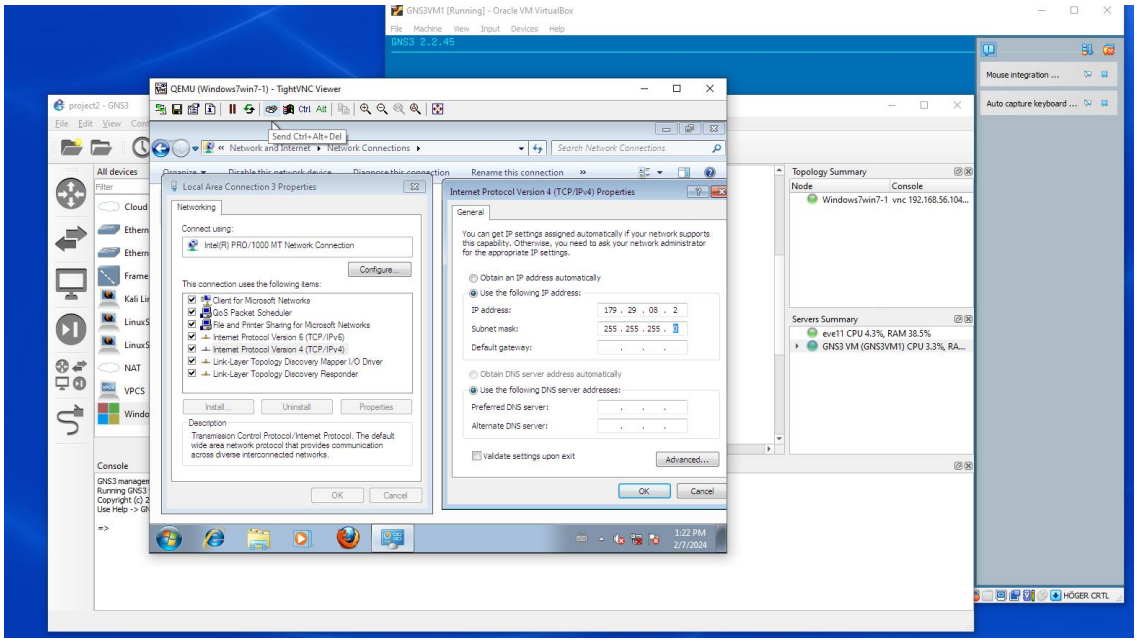
   This is because MAC addresses operate at the data link layer of the OSI model, which primarily functions within the local network segment. MAC addresses are used for communication within the local network segment and are typically not transmitted beyond it. Therefore, it's challenging to determine the MAC address of a computer on

a network you have yet to access because you would need access to the same local network segment to intercept or observe MAC address communication. Additionally, MAC addresses usually aren't routed across different network segments, so they can't be directly accessed from outside the local network. As a result, while it may be possible to forge a MAC address within the same local network segment, determining the MAC address of a computer on an unaccessed network is challenging due to the limitations of the data link layer and network segmentation.

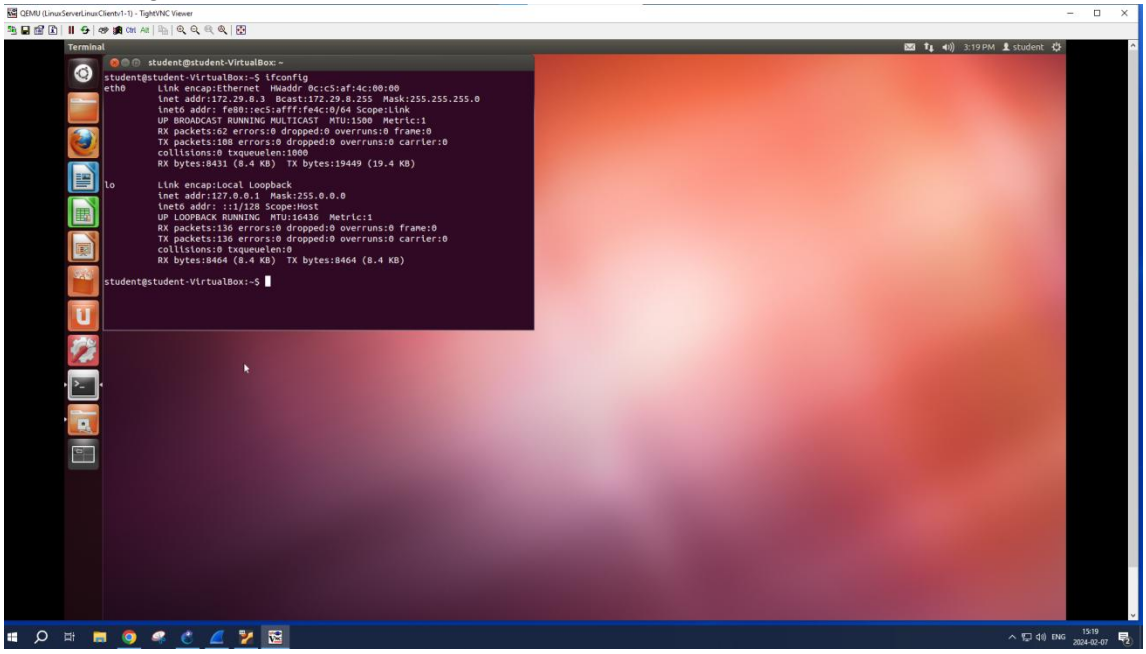
Exercise 2: Network layer packet forwarding

Activity 1

Windows changed ip & netmask



Linux changed ip&netmask



1. **What does the above (sudo ping -l 51020 -f -s 51020 -a 127.0.0.1) ping command do?**

   The sudo ping -l 51020 -f -s 51020 -a 127.0.0.1 command performs a ping operation with specific options and parameters.

sudo: Prefixing the command with sudo grants it superuser (root) privileges, allowing it to execute with elevated permissions. This might be necessary if the user doesn't have the necessary permissions to send ICMP packets.

-l 51020: Specifies the size of the ICMP data portion of the packet. In this case, it sets the data size to 51020 bytes. The -l option is often used to test the behavior of systems when dealing with large packets.

-f: This option sets the "Don't Fragment" flag in the packet's IP header. It ensures that the packet cannot be fragmented during transmission. This is useful for testing how the network handles large packets without fragmentation.

-s 51020: Specifies the size of the entire ICMP packet, including headers. This option sets the total packet size to 51020 bytes. It includes both the data portion specified by -l and additional packet overhead.

-a: This option resolves IP addresses to hostnames, if possible, before displaying them. In this case, it resolves 127.0.0.1 to its corresponding hostname before displaying it in the output.

127.0.0.1: The IP address of the target host. In this example, it's set to the loopback address (localhost), which represents the local machine itself. Testing with the loopback address helps verify that the network stack on the local machine is functioning correctly.

In summary, the sudo ping -l 51020 -f -s 51020 -a 127.0.0.1 command sends an ICMP Echo Request packet to the loopback address (127.0.0.1) with a total packet size of 51020 bytes, ensuring that it cannot be fragmented during transmission. It then resolves the IP address to its corresponding hostname (localhost) and displays the results. This command is often used for diagnostic purposes to test network behavior and performance.

2. **Which network protocol does the ping utility use to craft the ping request and response packets? On which layer of the TCP/IP stack does this network protocol belong?**

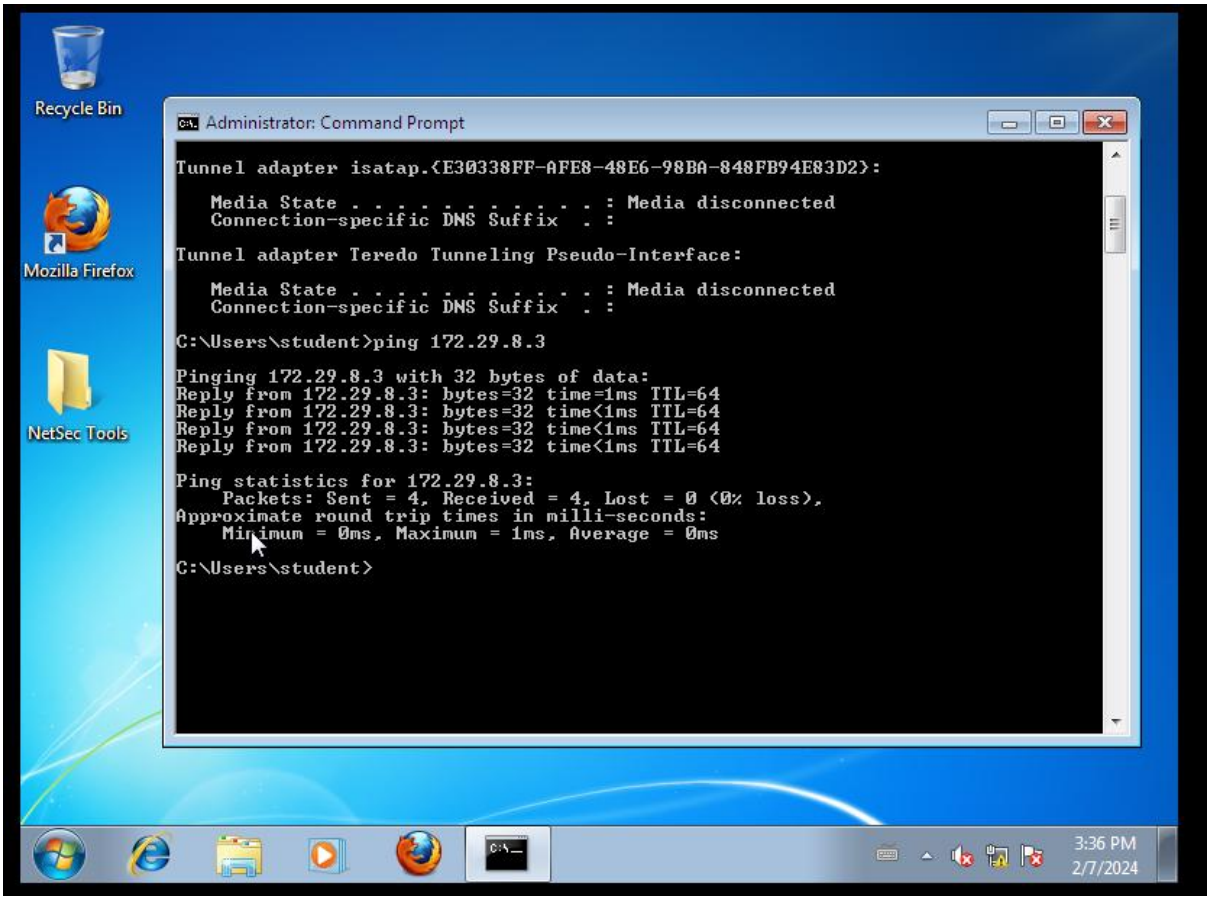
The ping utility uses the Internet Control Message Protocol (ICMP) to craft the ping request and response packets. ICMP is a network layer protocol (Layer 3 of the OSI model) and is an integral part of the TCP/IP protocol suite.

ICMP is primarily used for diagnostic and error-reporting purposes in IP networks. The ping utility sends ICMP Echo Request packets to a target host, and upon receiving the Echo Request, the target host sends back ICMP Echo Reply packets. These packets are used to

test the reachability of a host and measure round-trip time (RTT) between the source and destination.

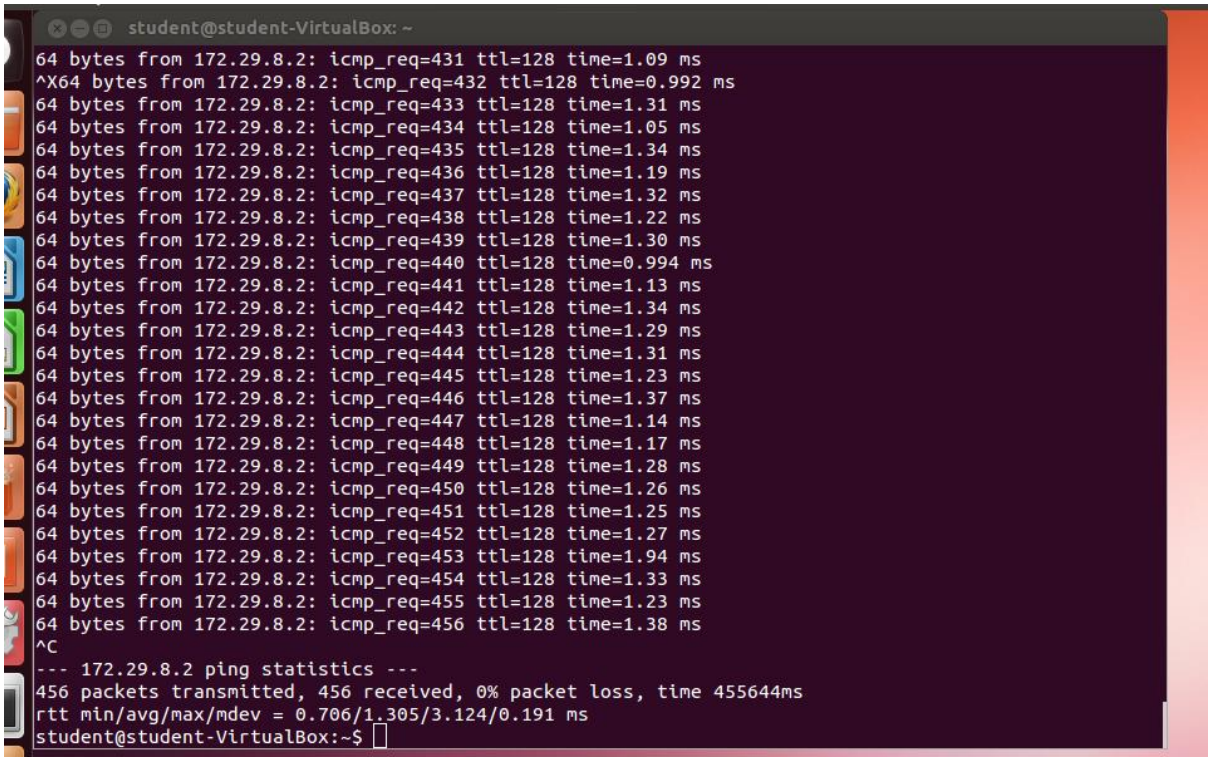
So, ICMP operates at the network layer (Layer 3) of the TCP/IP stack, which is responsible for routing packets between networks. It provides a mechanism for hosts and routers to communicate network-level information and diagnose network-related issues.

3. **How many ping requests did the 'Win7' system send? What options did you use? How many bytes of data did each request carry? What was the TTL value used, and what does the TTL abbreviation stand for? What was the packet loss rate? What was the average round trip time (RTT)?**



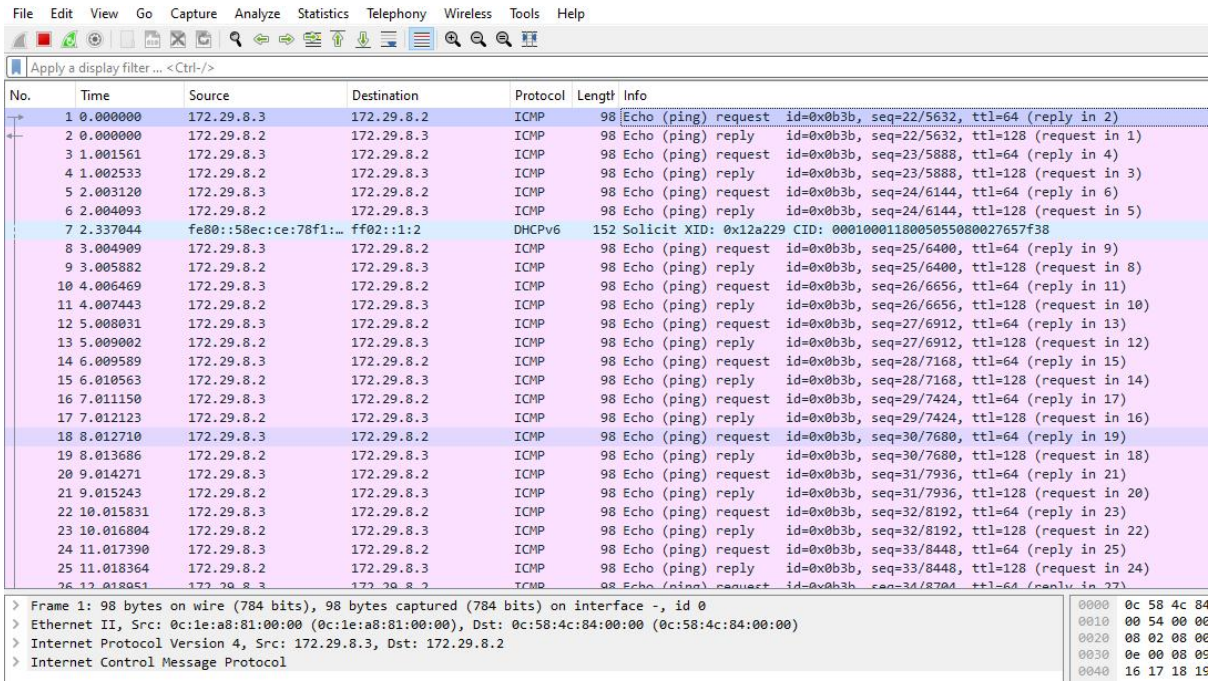
There are 4 requests, we use the ping command only with the IP address of Win7 without any additional options, each request carries 32 bytes of data. TTL was 64, which stands for time to live, the packet loss rate is 0%, and the average RTT is 0ms.

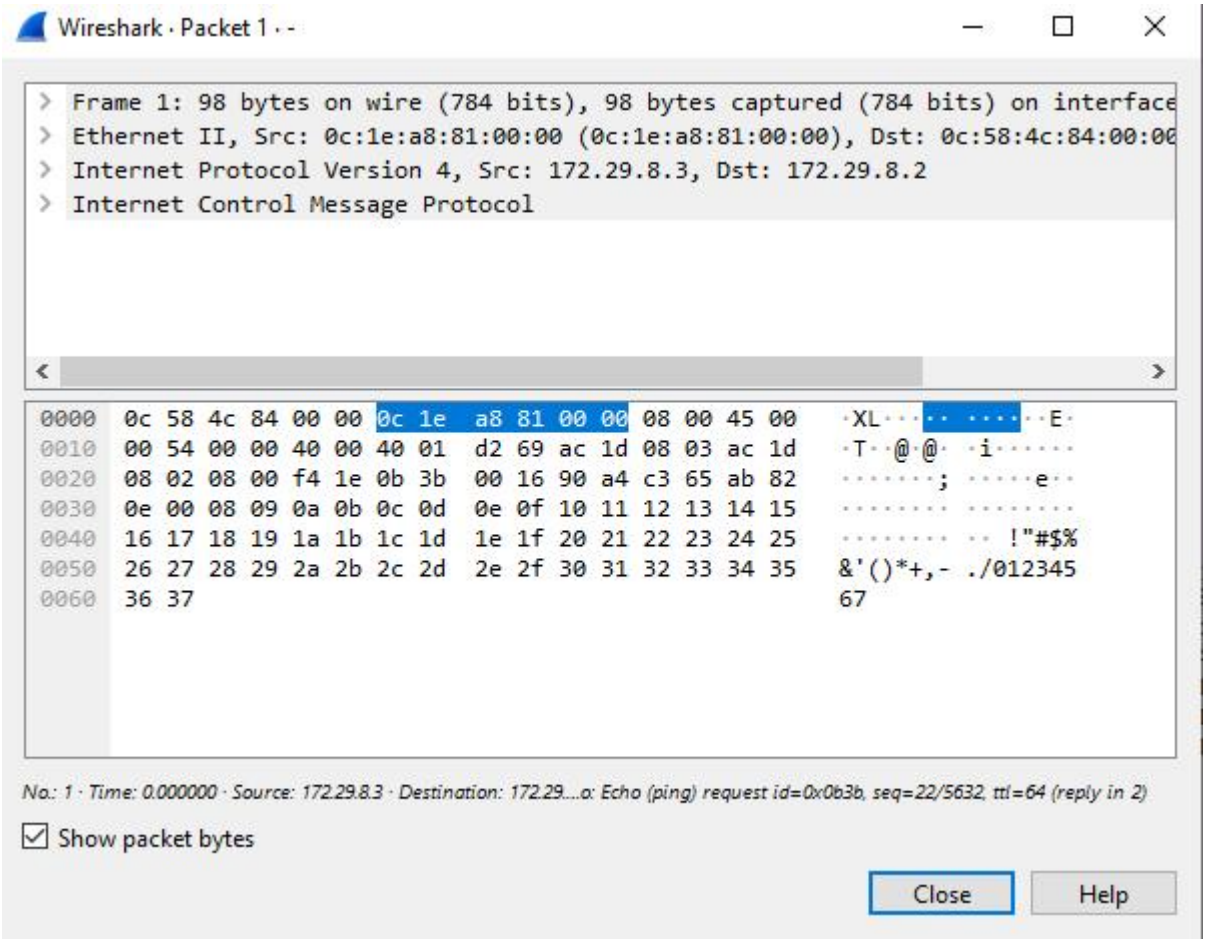
4. **How many ping requests did the 'LinuxcClient' system send (until you've decided to stop it)? What options did you use? How many bytes of data each request carried? What was the TTL value used? What were the packet loss rate and the average round trip time (RTT)?**



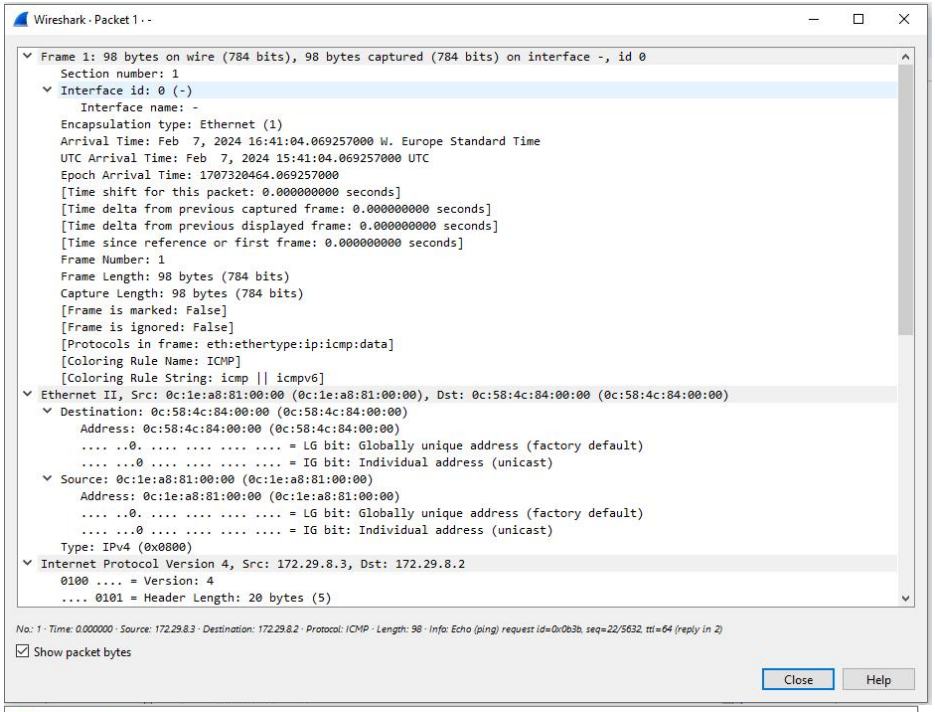
There are 4 requests, we use the ping command only with the IP address of Win7 without any additional options, each request carries 32 bytes of data. TTL was 64, which stands for time to live, the packet loss rate is 0%, and the average RTT is 0ms.

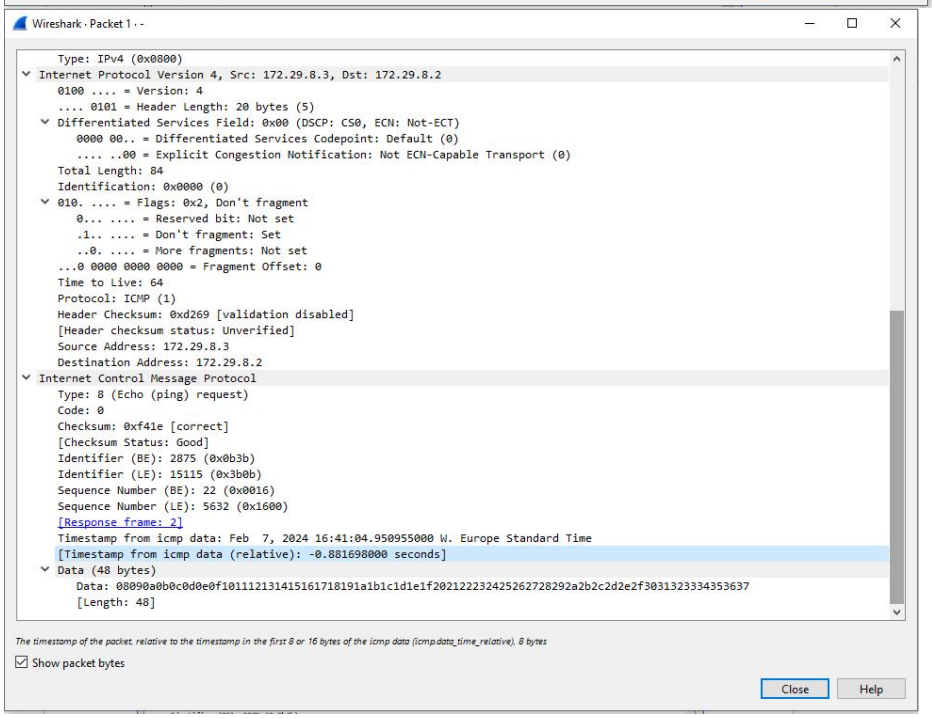
Activity 2





5. **Which protocols are employed for sending a ping request from one system to another? Can you describe the order of protocol encapsulation applied according to the TCP/IP or OSI model? What are the values for the type and code fields of the ping request? What is the ID value of the IP packet? What is the type of Ethernet frame?**





When sending a ping request from one system to another, the Internet Control Message Protocol (ICMP) is employed. ICMP is a network layer protocol (Layer 3 of the OSI model) and is used for diagnostic and error-reporting purposes in IP networks.

Here's the order of protocol encapsulation applied according to both the TCP/IP and OSI models:

Application Layer: The ping command is initiated by the user at the application layer.

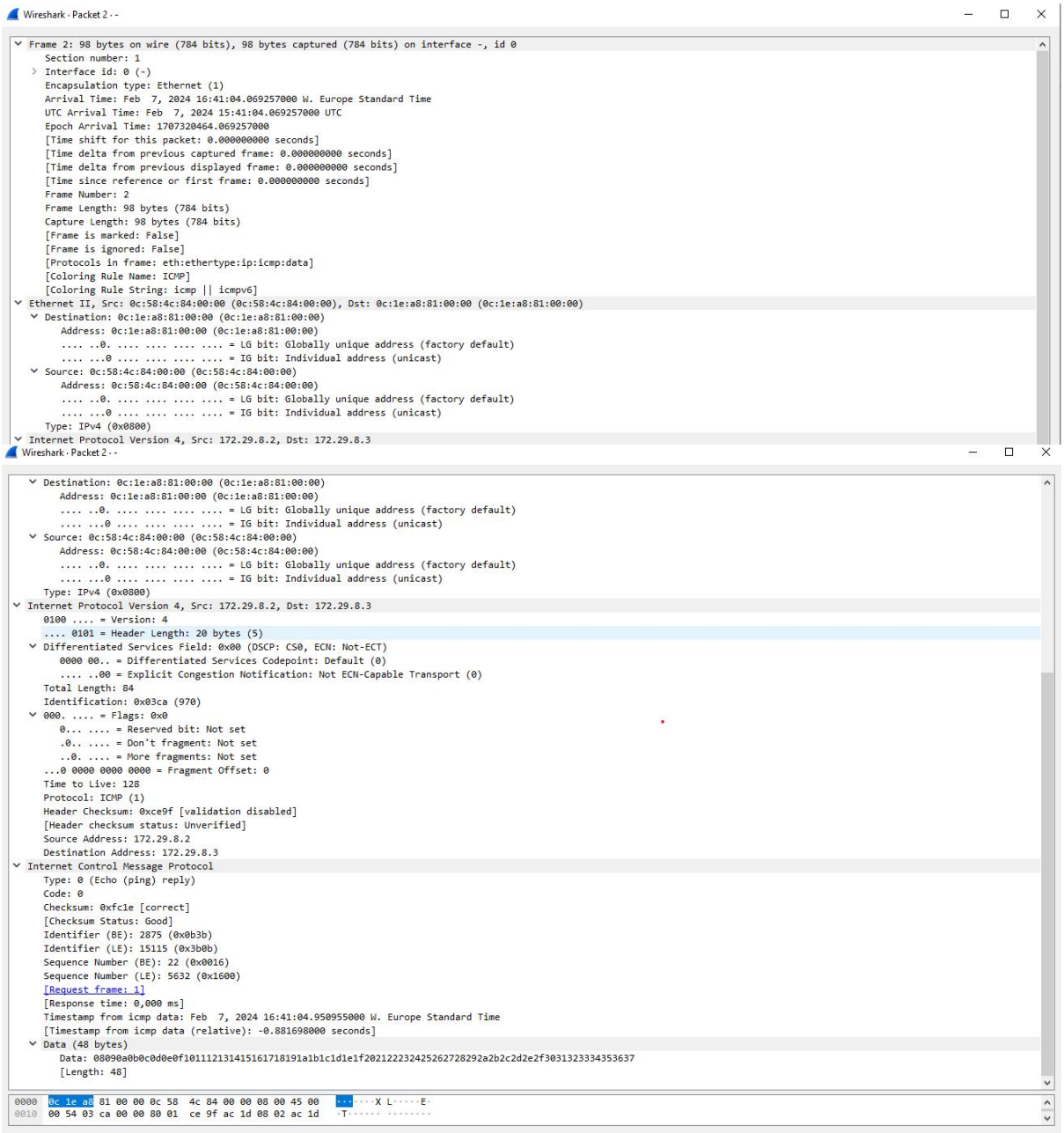
Transport Layer: The ping command uses the Internet Protocol (IP) at the transport layer (Layer 4 of the OSI model) to encapsulate the ICMP packets.

Network Layer: At the network layer (Layer 3 of the OSI model), the ICMP packets are encapsulated into IP packets. This includes adding IP headers such as source and destination IP addresses.

Data Link Layer: Finally, at the data link layer (Layer 2 of the OSI model), the IP packets are encapsulated into Ethernet frames for transmission over the local network. This includes adding MAC addresses (source and destination) and other Ethernet frame headers.

For a ping request, the ICMP type field is set to 8 (Echo Request) and the code field to 0. The ID value of the IP packet (identifier BE) is 2875. The Ethernet frame type for IPv4 packets (including ICMP ping requests) is 0x0800.

6. **Highlight a ping reply packet below the previously selected frame. Examine the values of all protocol fields and compare them with those of a ping request packet. Which fields have the same value between them?**



1. Ethernet layer: Packet 1 is sent, Packet 2 is received (destination and source addresses are reversed).

2. At the Internet Protocol Version layer:

For both ping request and ping reply packets, some identical field values include:

- Destination Address: Typically the IP address of the sender of the ping request.
- Source Address: Typically the IP address of the recipient of the ping request.
- Header Length: The header length of both ping request and ping reply packets is usually the same.

- Total Length: The total length of both ping request and ping reply packets is typically also the same.

Identification is 0x0000 with a TTL of 04 when sent, and 0x03ca (970) with a TTL of 128 when received. Header checksum also differs.

3. Internet Control Message Protocol:

Type 8 when sent, and Type 0 when received.

For the Internet Control Message Protocol (ICMP), the following fields have identical values between the send and receive packets:

- Code: Both send and receive packets have a code of 0.
- Checksum: The checksum field is the same in both send and receive packets.
- Checksum Status: The checksum status is identical in both send and receive packets.
- Identifier (BE): The identifier field in big-endian format is the same in both send and receive packets.
- Identifier (LE): Similarly, the identifier field in little-endian format is identical in both send and receive packets.
- Sequence Number (BE): The sequence number field in big-endian format is the same in both send and receive packets.
- Sequence Number (LE): Likewise, the sequence number field in little-endian format is identical in both send and receive packets.

7. **Notice that just before the ping requests and replies were exchanged, two frames using the ARP protocol have been captured. What does ARP stand for? In which layer of the TCP/IP stack does it belong? Briefly describe its main function. Briefly describe and explain why this protocol is important in a switched network.**

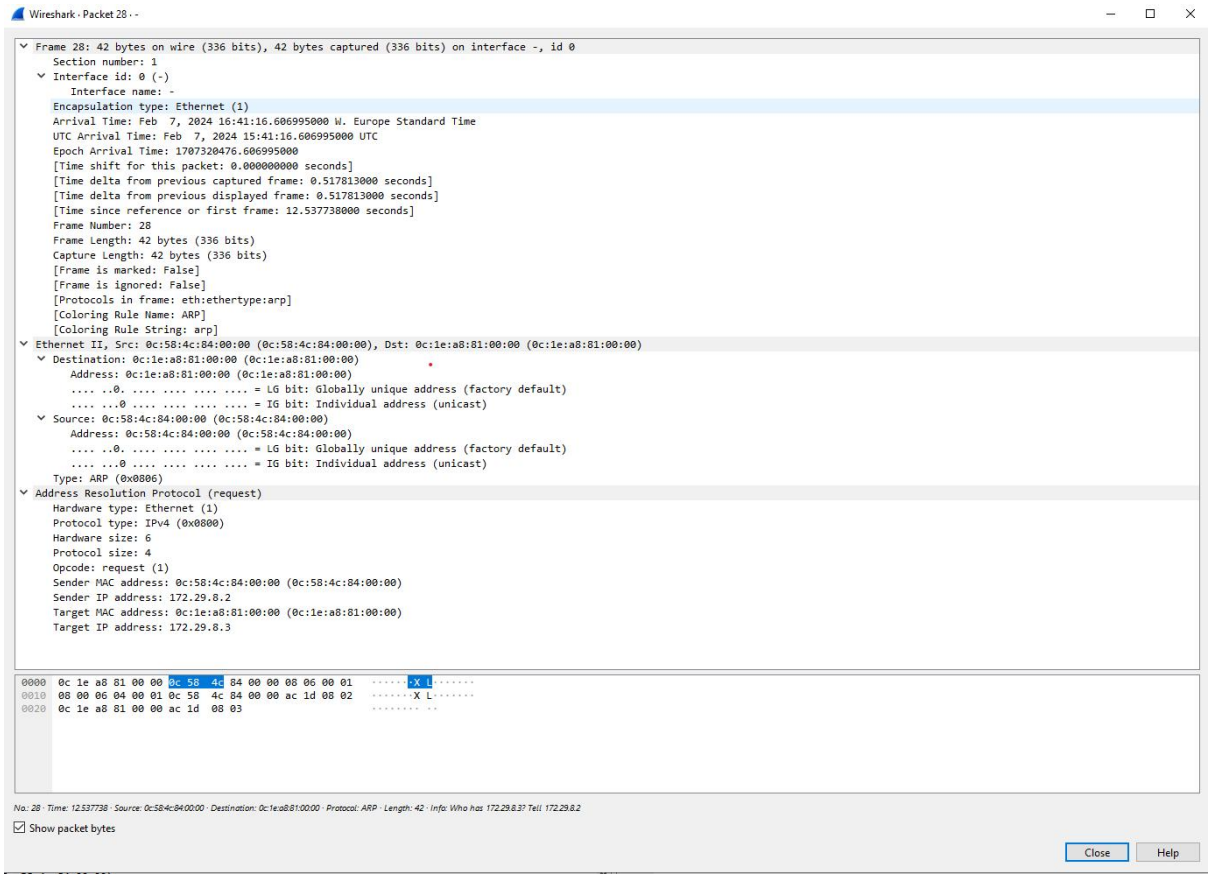
ARP stands for Address Resolution Protocol. It operates at the network layer (Layer 3) of the TCP/IP stack.

The main function of ARP is to map a known IP address to its corresponding MAC address in a local network. When a device wants to communicate with another device on the same network, it needs to know the MAC address of the destination device to frame the Ethernet frame correctly. ARP resolves this by broadcasting an ARP request packet containing the IP address of the target device. The device with the matching IP address responds with an ARP reply packet containing its MAC address. Once the sender receives the ARP reply, it can use the MAC address to send data packets to the destination device.

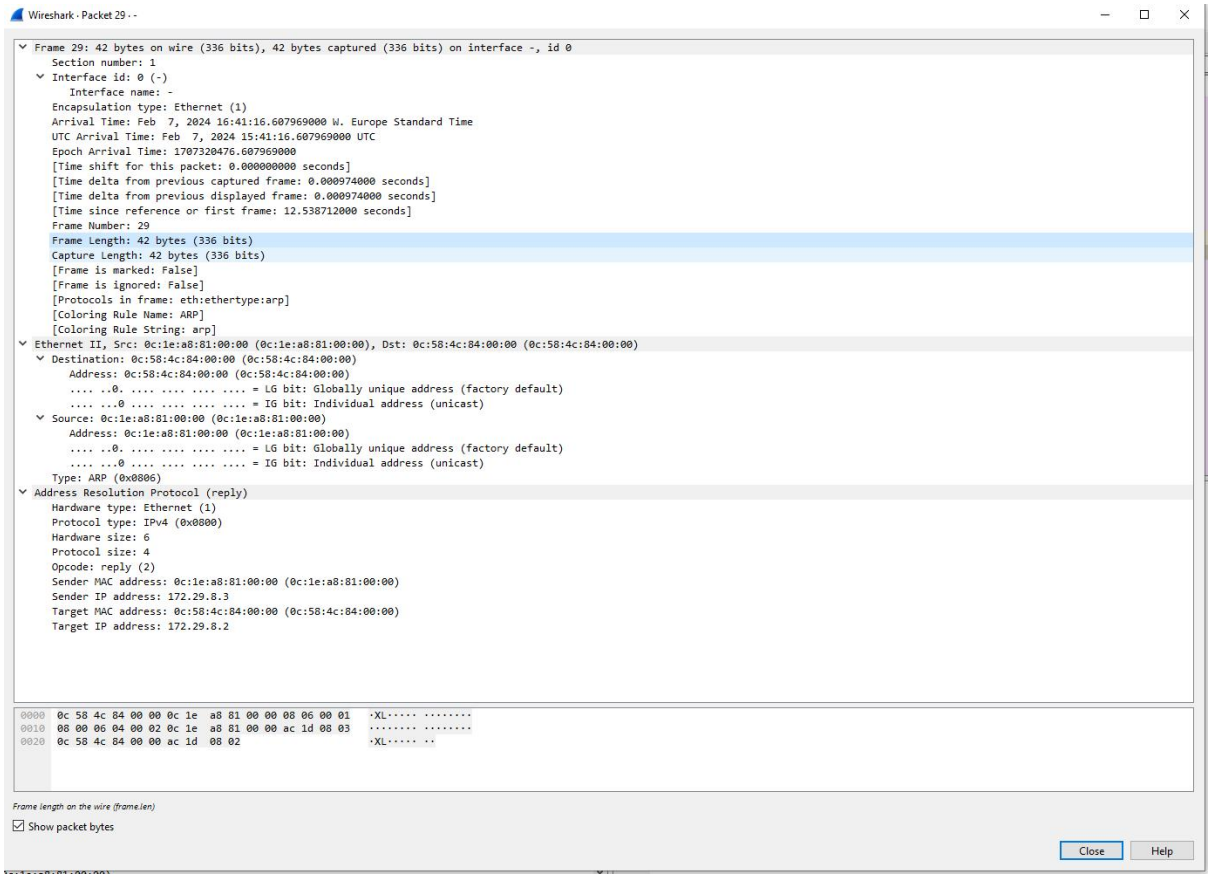
ARP is important in a switched network because it enables devices to communicate with each other at the data link layer (Layer 2) using MAC addresses. Switches use MAC addresses to forward frames within the local network. Without ARP, devices wouldn't be able to determine each other's MAC addresses, and communication within the network would not be possible. Therefore, ARP is crucial in facilitating communication within switched networks by dynamically mapping IP addresses to MAC addresses.

8. **Examine the ARP request packet. What is the destination MAC address? Briefly explain this.**

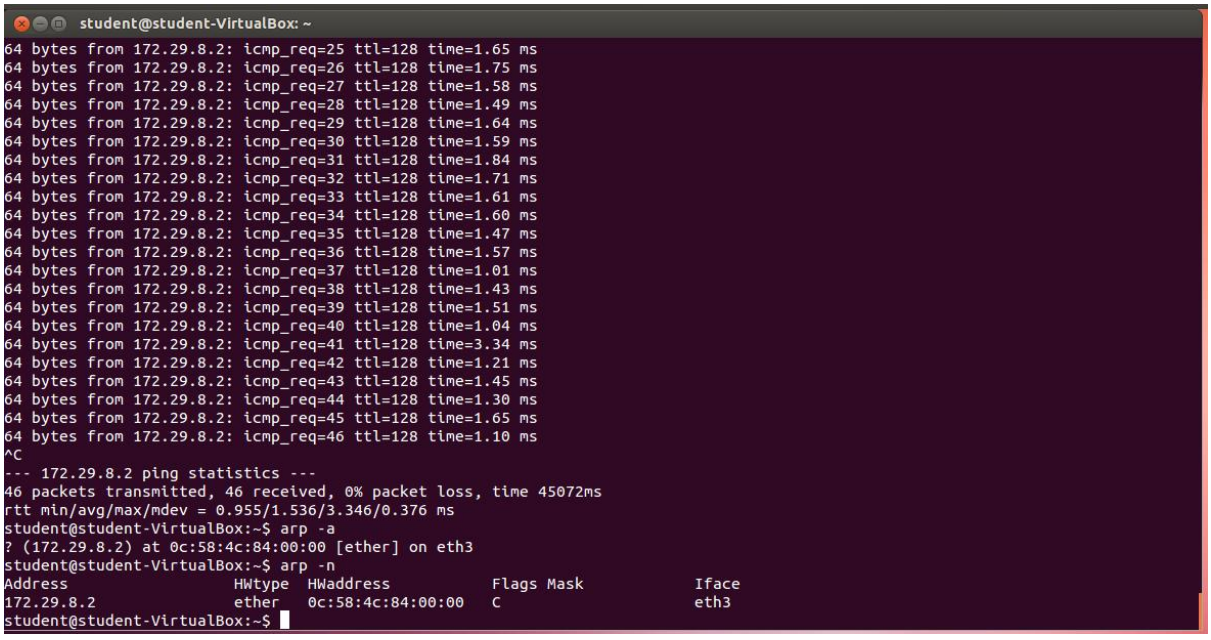
The destination MAC address is 0c:1e:a8:81:00:00, corresponding to the Linux Client's MAC address. In this context, Windows 7 firstly initiates an ARP request targeting the Linux Client.



9. **Examine the ARP reply packet. What is the source MAC address of the frame and to which network host does it belong? Does the ARP reply contain valid data?**



The source MAC address of the frame is 0c:1e:a8:81:00:00 which belongs to the host Linux Client.



The ARP reply contains valid data. When a device receives an ARP request for its IP address, it responds with an ARP reply packet containing its MAC address. When the ‘arp -a’ is executed in the Linux Client, there is a MAC address of the Win7 in the ARP cache, which means there is valid data in the ARP reply.

```
student@student-VirtualBox: ~
64 bytes from 172.29.8.2: icmp_req=25 ttl=128 time=1.65 ms
64 bytes from 172.29.8.2: icmp_req=26 ttl=128 time=1.75 ms
64 bytes from 172.29.8.2: icmp_req=27 ttl=128 time=1.58 ms
64 bytes from 172.29.8.2: icmp_req=28 ttl=128 time=1.49 ms
64 bytes from 172.29.8.2: icmp_req=29 ttl=128 time=1.64 ms
64 bytes from 172.29.8.2: icmp_req=30 ttl=128 time=1.59 ms
64 bytes from 172.29.8.2: icmp_req=31 ttl=128 time=1.84 ms
64 bytes from 172.29.8.2: icmp_req=32 ttl=128 time=1.71 ms
64 bytes from 172.29.8.2: icmp_req=33 ttl=128 time=1.61 ms
64 bytes from 172.29.8.2: icmp_req=34 ttl=128 time=1.60 ms
64 bytes from 172.29.8.2: icmp_req=35 ttl=128 time=1.47 ms
64 bytes from 172.29.8.2: icmp_req=36 ttl=128 time=1.57 ms
64 bytes from 172.29.8.2: icmp_req=37 ttl=128 time=1.01 ms
64 bytes from 172.29.8.2: icmp_req=38 ttl=128 time=1.43 ms
64 bytes from 172.29.8.2: icmp_req=39 ttl=128 time=1.51 ms
64 bytes from 172.29.8.2: icmp_req=40 ttl=128 time=1.04 ms
64 bytes from 172.29.8.2: icmp_req=41 ttl=128 time=3.34 ms
64 bytes from 172.29.8.2: icmp_req=42 ttl=128 time=1.21 ms
64 bytes from 172.29.8.2: icmp_req=43 ttl=128 time=1.45 ms
64 bytes from 172.29.8.2: icmp_req=44 ttl=128 time=1.30 ms
64 bytes from 172.29.8.2: icmp_req=45 ttl=128 time=1.65 ms
64 bytes from 172.29.8.2: icmp_req=46 ttl=128 time=1.10 ms
^C
--- 172.29.8.2 ping statistics ---
46 packets transmitted, 46 received, 0% packet loss, time 45072ms
rtt min/avg/max/mdev = 0.955/1.536/3.346/0.376 ms
student@student-VirtualBox:~$ arp -a
? (172.29.8.2) at 0c:58:4c:84:00:00 [ether] on eth3
student@student-VirtualBox:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.29.8.2               ether   0c:58:4c:84:00:00   C                     eth3
student@student-VirtualBox:~$
```

10: Reissue the ping command at the ‘LinuxClient’ to the ‘Win7’ one and examine the packet capture of the ‘LinuxClient’s link. Can you find any ARP request packets this time? Why or why not?

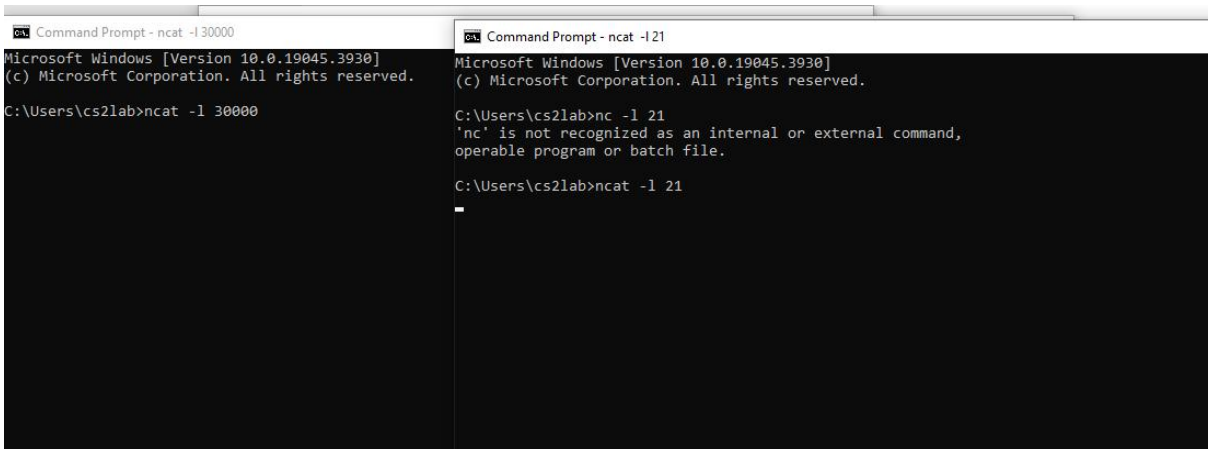
No.	Time	Source	Destination	Protocol	Length	Info
409	297.265586	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=151/38656, ttl=128 (request in 408)
410	298.266173	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=152/38912, ttl=64 (reply in 411)
411	298.267149	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=152/38912, ttl=128 (request in 410)
412	299.267733	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=153/39168, ttl=64 (reply in 413)
413	299.268706	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=153/39168, ttl=128 (request in 412)
414	300.268683	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=154/39424, ttl=64 (reply in 415)
415	300.269653	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=154/39424, ttl=128 (request in 414)
416	301.270438	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=155/39680, ttl=64 (reply in 417)
417	301.270438	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=155/39680, ttl=128 (request in 416)
418	302.271998	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=156/39936, ttl=64 (reply in 419)
419	302.271998	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=156/39936, ttl=128 (request in 418)
420	303.273559	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=157/40192, ttl=64 (reply in 421)
421	303.273559	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=157/40192, ttl=128 (request in 420)
422	304.274311	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=158/40448, ttl=64 (reply in 423)
423	304.275287	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=158/40448, ttl=128 (request in 422)
424	305.275873	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=159/40704, ttl=64 (reply in 425)
425	305.276847	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=159/40704, ttl=128 (request in 424)
426	306.277432	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=160/40960, ttl=64 (reply in 427)
427	306.277432	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=160/40960, ttl=128 (request in 426)
428	307.278996	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=161/41216, ttl=64 (reply in 429)
429	307.279966	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=161/41216, ttl=128 (request in 428)
430	308.280552	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=162/41472, ttl=64 (reply in 431)
431	308.281526	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=162/41472, ttl=128 (request in 430)
432	309.282113	172.29.8.3	172.29.8.2	ICMP	98	Echo (ping) request id=0x0c19, seq=163/41728, ttl=64 (reply in 433)
433	309.283086	172.29.8.2	172.29.8.3	ICMP	98	Echo (ping) reply id=0x0c19, seq=163/41728, ttl=128 (request in 432)

The ARP requests have not been found. The 'LinuxClient' has already communicated with the Win7 host recently, the ARP cache on the 'LinuxClient' may already contain an entry mapping the IP address of the 'Win7' host to its MAC address. In this case, the 'LinuxClient' wouldn't need to send an ARP request again because it already knows the MAC address of the 'Win7' host.

Exercise 3: Transport Layer

Activity 1

1. **Have the commands completed successfully? Why or why not? If a command has failed, how can the command be made to work? What does the 'l' parameter mean? (Note that the success/failure here may depend on the operating system you're using.)**



We entered nc -l 21 and nc -l 30000 without success.

The reason why nc -l 21 and nc -l 30000 did not succeed might be because the behavior of the nc command could vary depending on the operating system, or it could be because the version or configuration of the nc command on specific operating systems does not support the provided port numbers.

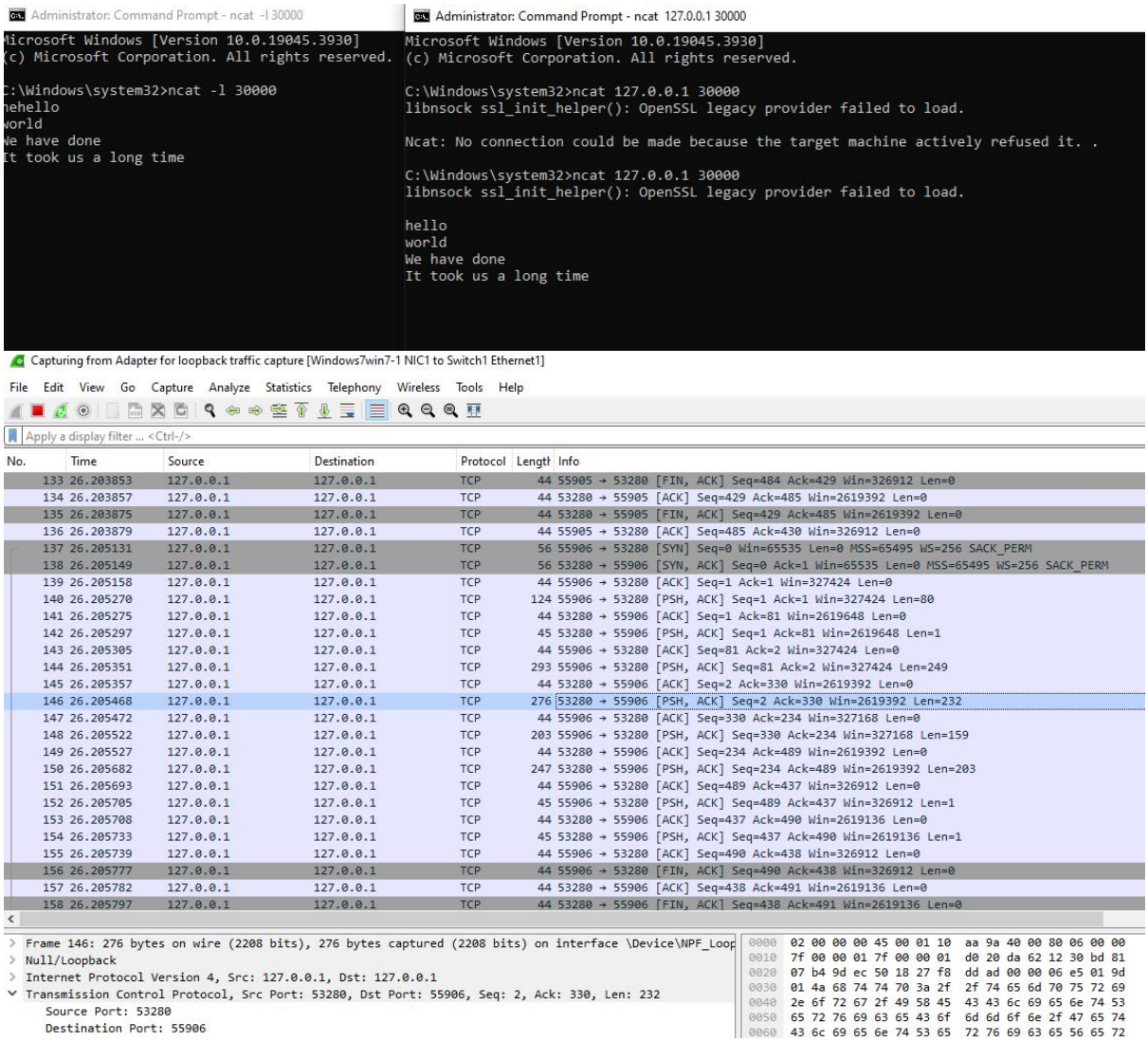
In contrast, ncat is an enhanced version of the nc command, offering additional features and improvements. In this case, ncat may have better compatibility and reliability, allowing it to successfully listen on the specified ports.

The solution, in this case, is to use the ncat command instead of the nc command to ensure successful listening on the specified ports.

The 'l' parameter means listen mode, when the -l is used, Netcat will start the listening service on the specified port and wait for the connection of other computers. 21 and 30000 are port numbers, indicating which port to start the listening service on.

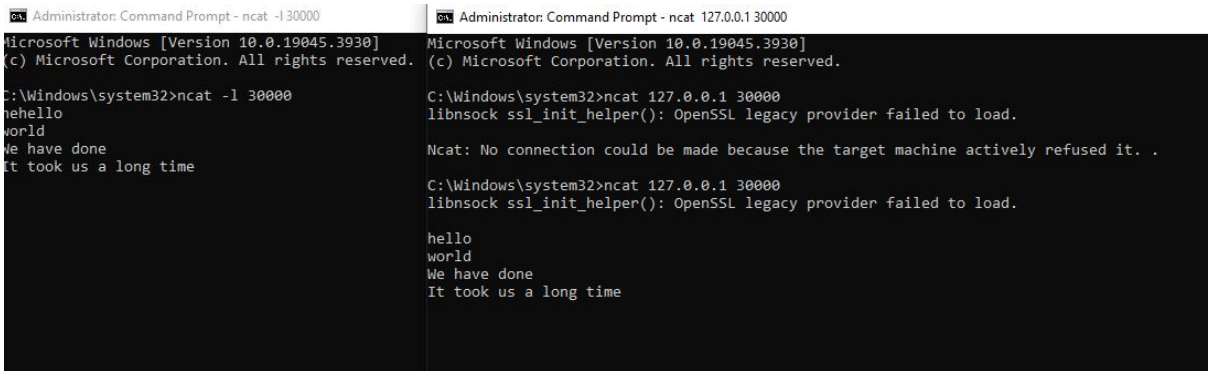
Activity 2

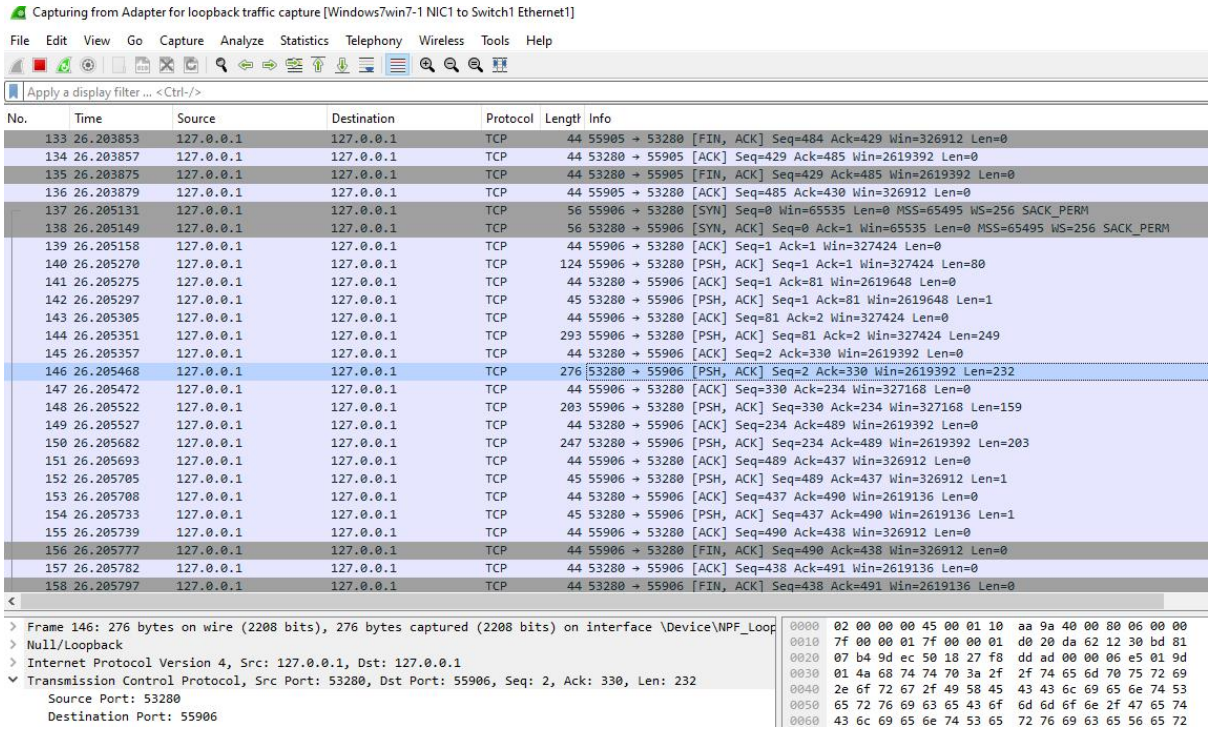
2. **View the traffic in Wireshark and answer the questions: "Which transport layer protocol has been used for the communication?**



Protocol-TCP

3. **Provide a screenshot showing that the communication has been properly established. What are the client and server ports?**

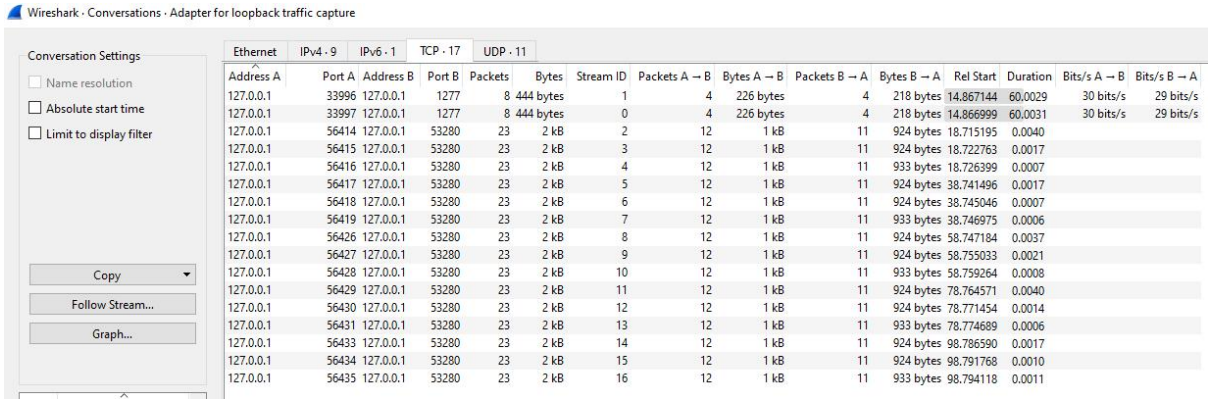




In the first screenshot, when we input text in one window, the same text appears in the other window as well. This indicates that communication has been properly established between the client and the server.

In the second screenshot, the source port is 53280 (client port) and the destination port is 55906 (server port).

4. **How many packets flow from client to server and how many vice-versa? How many bytes are sent in each direction and total?"**



Packets flow from client to server: 12
Packets flow from server to client: 11
Each byte: 2kB
Total bytes: 444 bytes

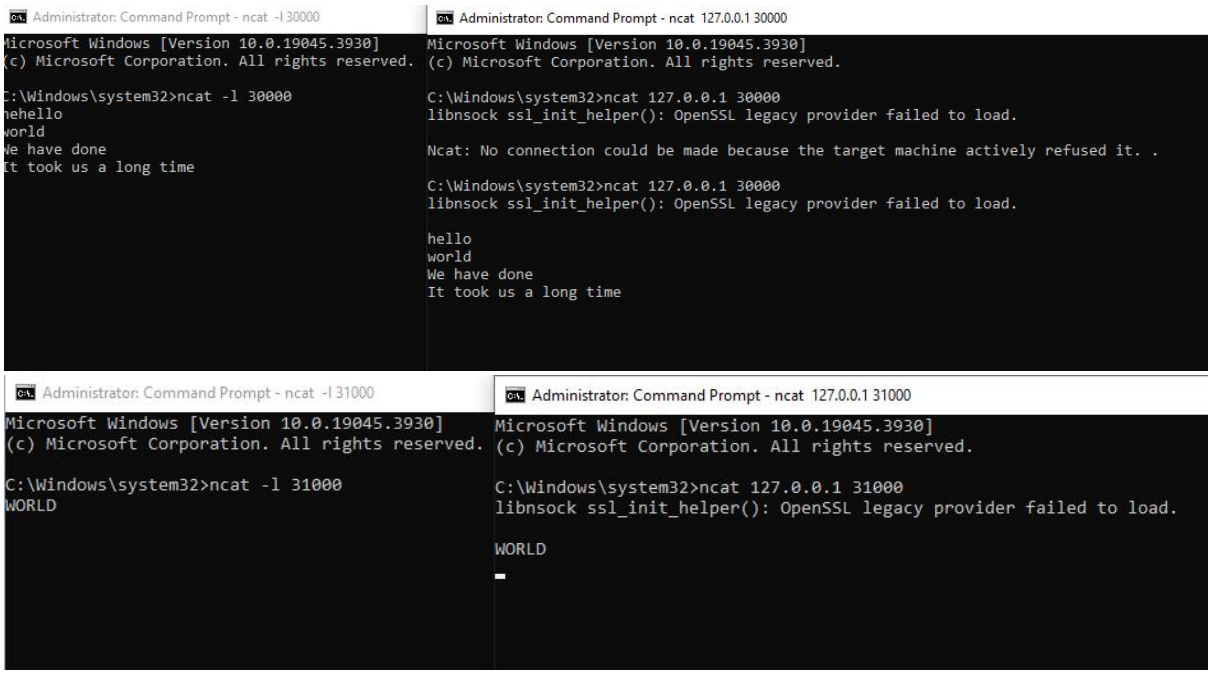
5. **What were the server and client ports that were used in this communication session?**

Client ports: 56414, 56415, 56416, 56417, 56418, 56419, 56426, 56427, 56428, 56429, 56430, 56431, 56433, 56434, 56435

Server port: 53280

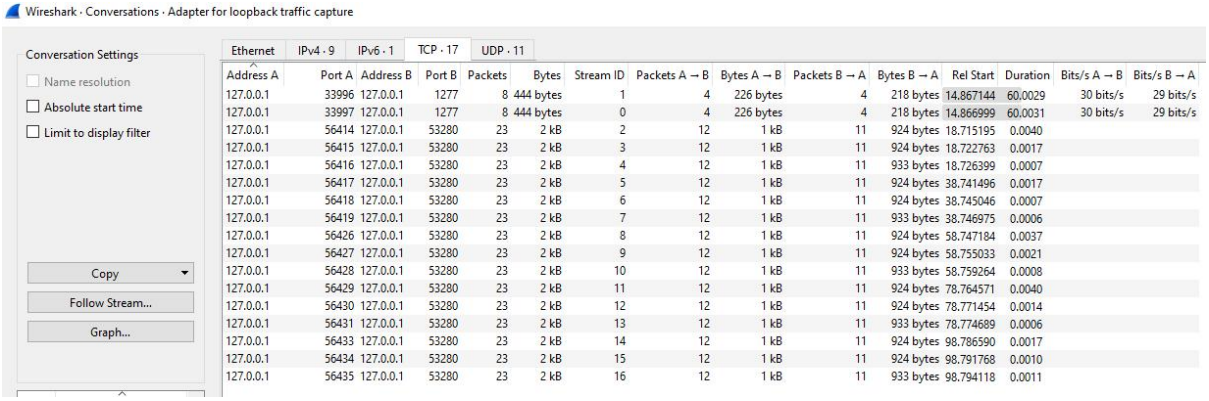
6. **Also, what does 127.0.0.1 mean in this context? What is special about this IP address? Retry the previous experiment but with two active parallel sessions. I.e., open four windows and start two nc instances listening on different ports; 30000 for the first, and 31000 for the second. To make it easy for you to identify them, I suggest you type" HELLO" in one conversation, and" WORLD" in the other.**

- 127.0.0.1 is the loopback address, often referred to as "localhost."
- It is a special IP address that points back to the same device from which the request originates.
- Communication to 127.0.0.1 is routed internally within the device, allowing processes and applications on the same device to communicate with each other via the network protocol stack.
- It is commonly used for testing network services on a local machine without the need for an actual network connection.



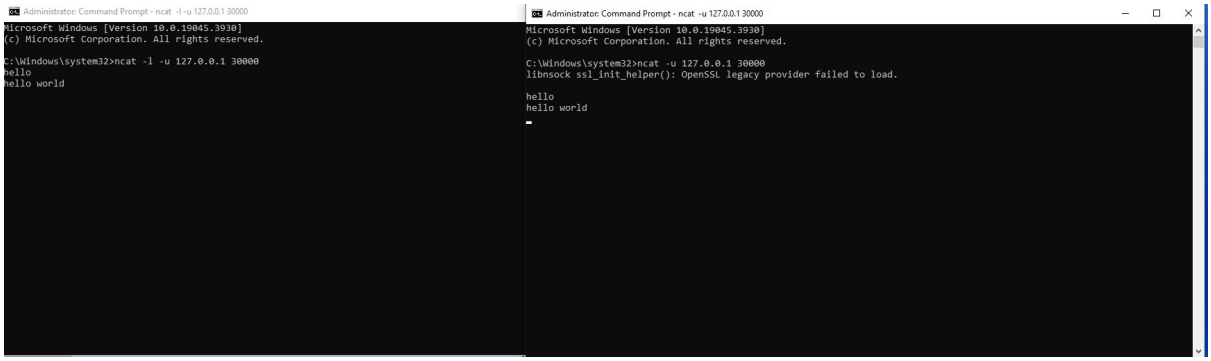
Each netcat instance will be in a listening state, waiting for connection requests from other hosts. When another host attempts to connect to a specific port, the netcat instance will accept the connection and initiate a communication session with that host. Because these two netcat instances are running on different ports and in separate windows, their communication is parallel, and there will be no interference between them.

7. **Which fields and values of the captured packets give you an idea of which is the client and which is the server in each individual session? Now we'll try to add the -u parameter to nc. Open two windows. Type 'nc -l -u 127.0.0.1 30000' in one and 'nc -u 127.0.0.1 30000' in the other. Try to communicate between the two instances as you're watching the traffic in Wireshark.**



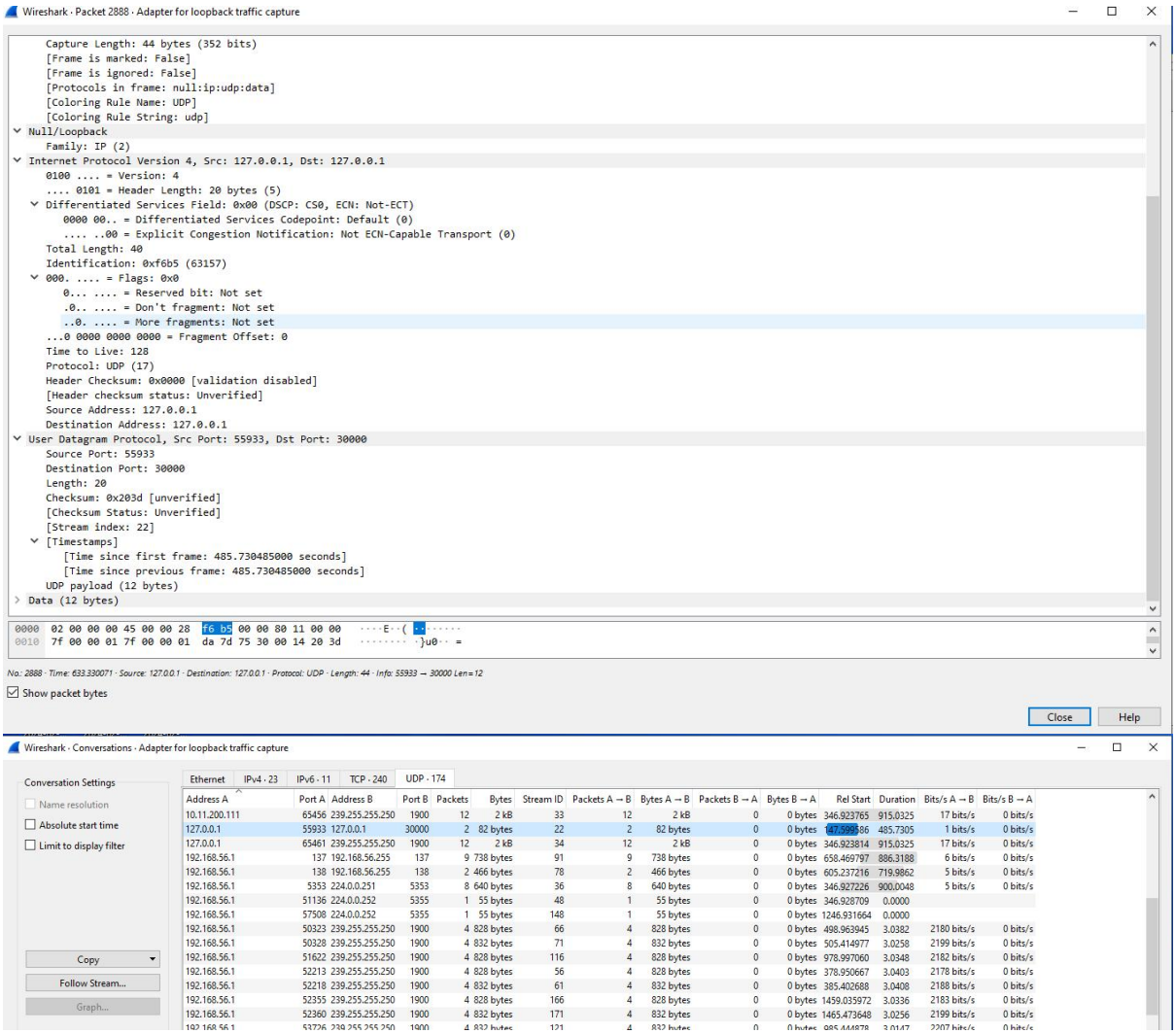
Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
127.0.0.1	33996	127.0.0.1	1277	8	444 bytes	1	4	226 bytes	4	218 bytes	14.867144	60.0029	30 bits/s	29 bits/s
127.0.0.1	33997	127.0.0.1	1277	8	444 bytes	0	4	226 bytes	4	218 bytes	14.866999	60.0031	30 bits/s	29 bits/s
127.0.0.1	56414	127.0.0.1	53280	23	2 kB	2	12	1 kB	11	924 bytes	18.715195	0.0040		
127.0.0.1	56415	127.0.0.1	53280	23	2 kB	3	12	1 kB	11	924 bytes	18.722763	0.0017		
127.0.0.1	56416	127.0.0.1	53280	23	2 kB	4	12	1 kB	11	933 bytes	18.726399	0.0007		
127.0.0.1	56417	127.0.0.1	53280	23	2 kB	5	12	1 kB	11	924 bytes	38.741496	0.0017		
127.0.0.1	56418	127.0.0.1	53280	23	2 kB	6	12	1 kB	11	924 bytes	38.745046	0.0007		
127.0.0.1	56419	127.0.0.1	53280	23	2 kB	7	12	1 kB	11	933 bytes	38.746975	0.0006		
127.0.0.1	56426	127.0.0.1	53280	23	2 kB	8	12	1 kB	11	924 bytes	58.747184	0.0037		
127.0.0.1	56427	127.0.0.1	53280	23	2 kB	9	12	1 kB	11	924 bytes	58.755033	0.0021		
127.0.0.1	56428	127.0.0.1	53280	23	2 kB	10	12	1 kB	11	933 bytes	58.759264	0.0008		
127.0.0.1	56429	127.0.0.1	53280	23	2 kB	11	12	1 kB	11	924 bytes	78.764571	0.0040		
127.0.0.1	56430	127.0.0.1	53280	23	2 kB	12	12	1 kB	11	924 bytes	78.771454	0.0014		
127.0.0.1	56431	127.0.0.1	53280	23	2 kB	13	12	1 kB	11	933 bytes	78.774689	0.0006		
127.0.0.1	56433	127.0.0.1	53280	23	2 kB	14	12	1 kB	11	924 bytes	98.786590	0.0017		
127.0.0.1	56434	127.0.0.1	53280	23	2 kB	15	12	1 kB	11	924 bytes	98.791768	0.0010		
127.0.0.1	56435	127.0.0.1	53280	23	2 kB	16	12	1 kB	11	933 bytes	98.794118	0.0011		

This response explains that in a TCP connection, the client port is dynamically chosen from an available port range by the operating system. Therefore, it is not fixed and can vary with each connection attempt. In contrast, the server port is fixed because the server needs to listen on a specific port to accept incoming connections.



Communicate successfully!

8. **What are the differences from the TCP experiment you did above?**



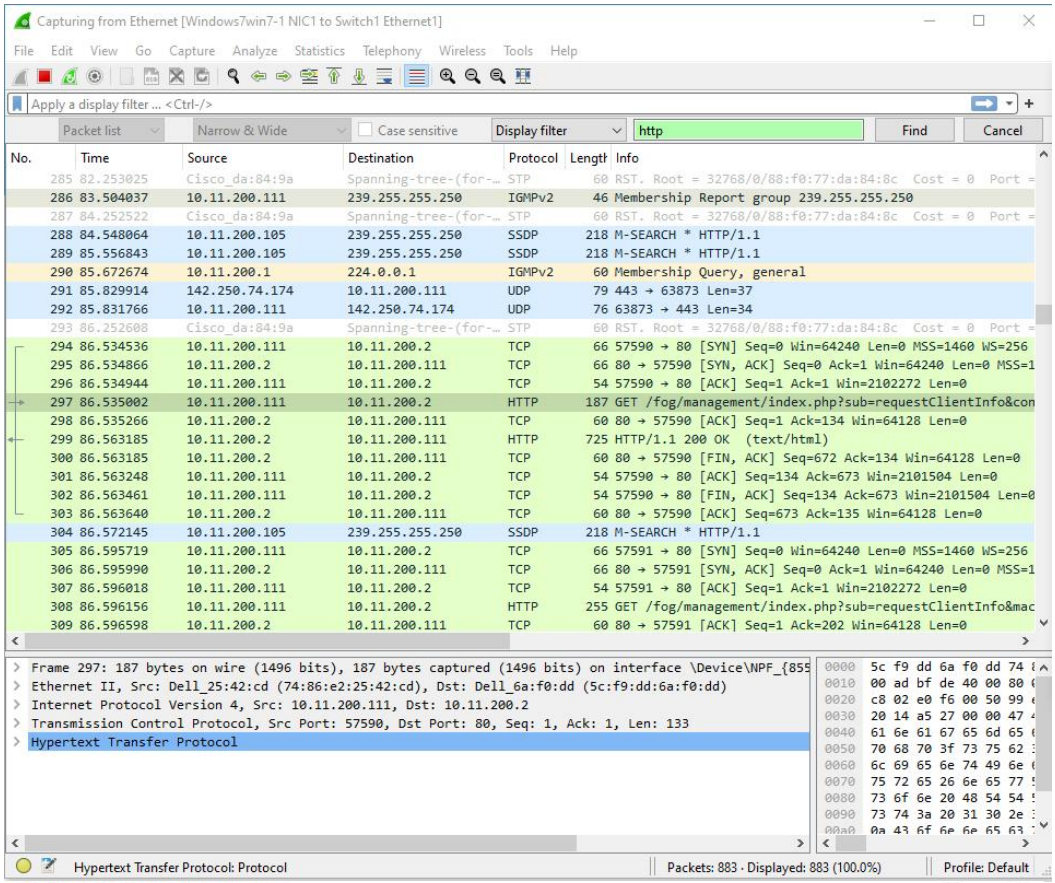
TCP Experiment Differences:

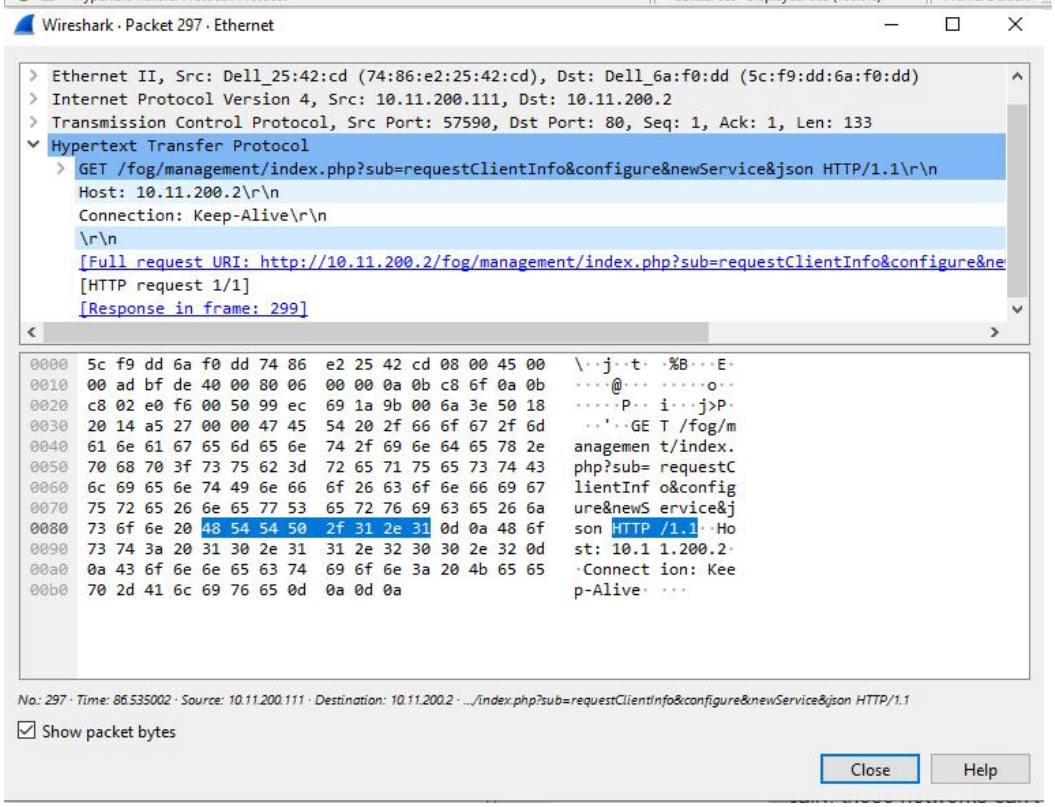
In the TCP experiment, both the client port and server port can transmit data bidirectionally. Wireshark captures traffic from both directions, allowing us to see packets transmitted from the client to the server (A to B) as well as from the server to the client (B to A).
UDP Experiment Differences:

In the UDP experiment, Wireshark only captures traffic in one direction.
UDP is connectionless, so there is no inherent concept of a client or server; data transmission occurs between two endpoints.
Wireshark may capture packets transmitted from one endpoint to another (e.g., from A to B), but it may not capture packets transmitted in the opposite direction (e.g., from B to A).
Therefore, in this experiment, only the records of packets from A to B are visible, and packets from B to A are not captured by Wireshark.

Exercise 4: Application Layer

Activity 1





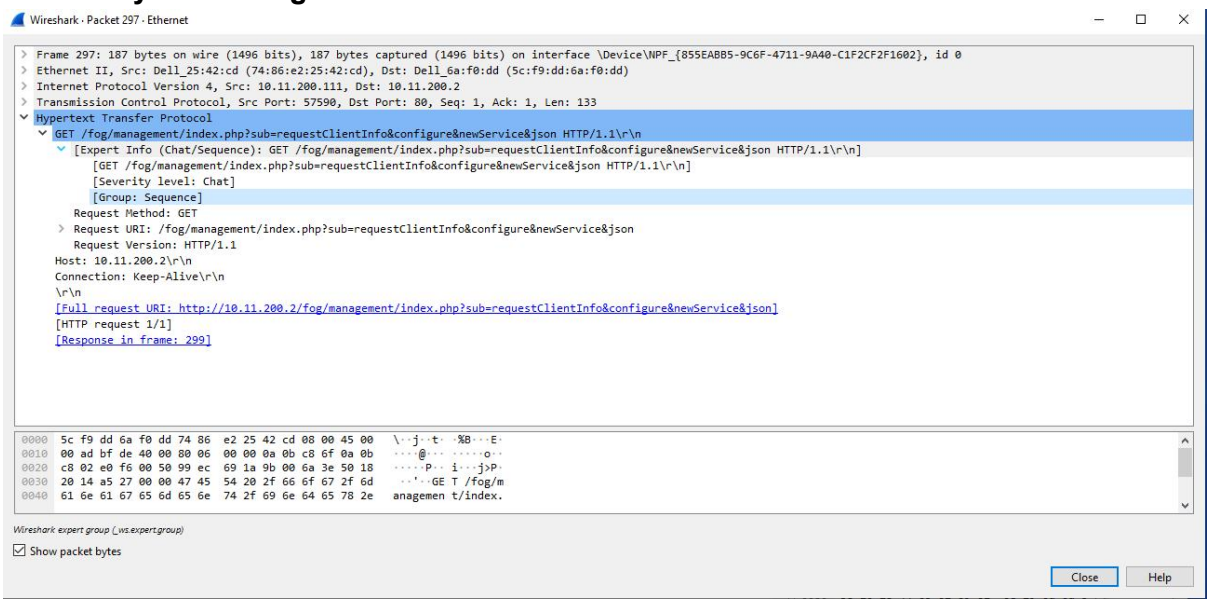
1. **Start Wireshark and a web browser. Surf to http://neverssl.com Examine the captured packets that were generated by this request.**

The first packet with the HTTP protocol observed in Wireshark is packet 297. This packet represents the initial HTTP protocol data packet generated when accessing http://neverssl.com through the web browser.

2. **Which application layer protocol is used? Can you identify a packet that carries the initial web request?**

The application layer protocol used in this case is HTTP (Hypertext Transfer Protocol), as it is the standard protocol for transferring hypertext requests and responses on the World Wide Web. You can identify the packet that carries the initial web request by looking for packets with the HTTP GET method sent from your computer to the neverssl.com server.

3. **Write down which metadata fields appear in the request and what their value means. Can you identify the packet that carries the response? Can you identify any interesting metadata fields?**

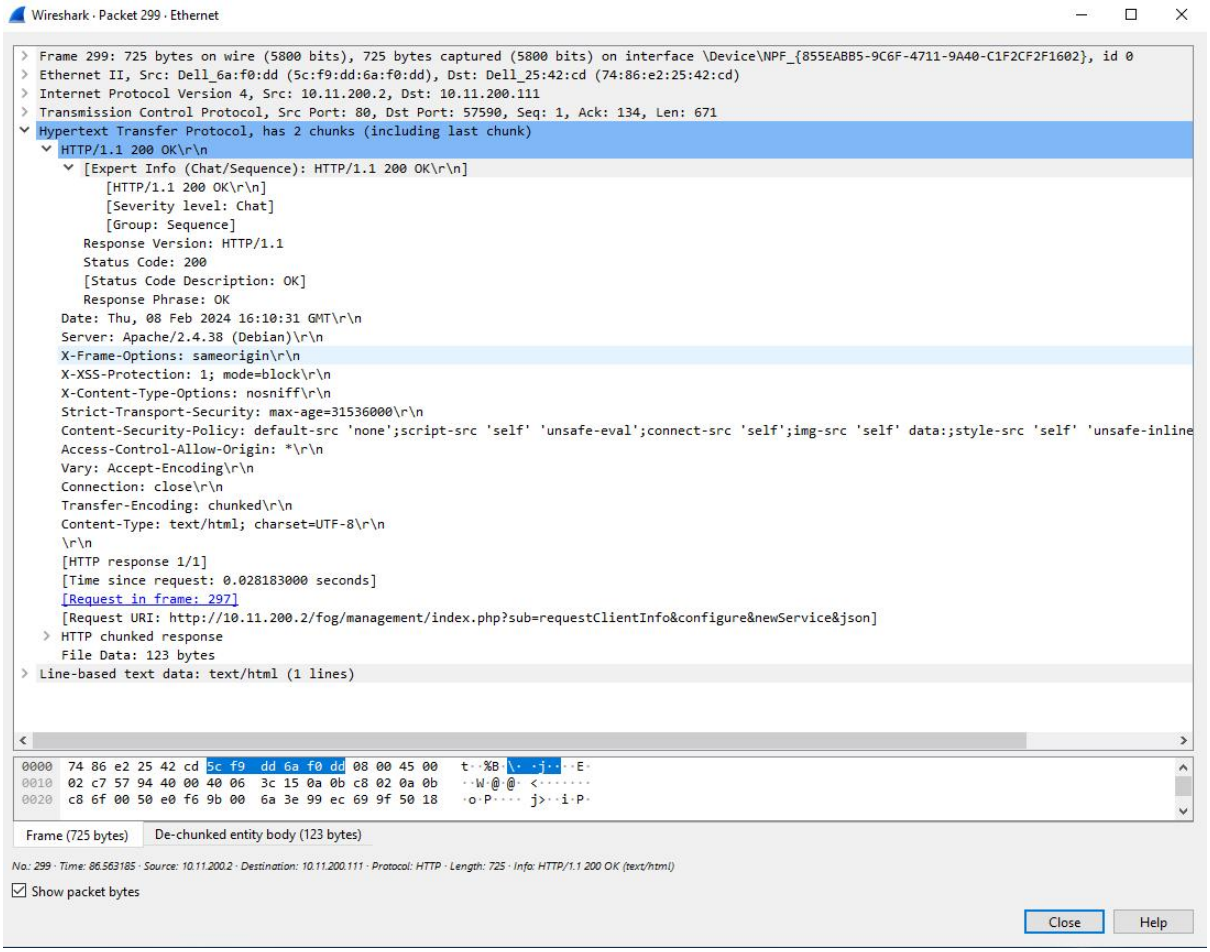


Metadata fields that appear in the request include:

- Request Method(GET): GET requests are used to get data from the server. Indicates the type of request made to the web server.
- Request URI Path: Represents the path of the requested resource.
- Request Query: Represents the query parameters.
- Request Version(HTTP): The version of the HTTP protocol being used.
- Host(10.11.200.2): Specifies the hostname of the target server.
- Host Request: Indicates the host information included in the request.

The packet that carries the response

- Yes, I can identify the packet that carries the response. a status code of 200 and a Content-Type of text/HTML typically indicate that the packet carries the response.
- The status code 200 in HTTP indicates that the request has succeeded.
- The Content-Type of text/html specifies that the content being sent back is HTML.
- These characteristics are indicative of a successful response containing HTML content, which is typically the case for web pages. Therefore, packets with these attributes can be identified as carrying the response in Wireshark.



Interesting metadata fields:

X-Frame-Options: This field is used to control whether a webpage is allowed to be embedded in <frame>, <iframe>, <embed>, or <object> tags on other websites. SAMEORIGIN indicates that the page can be displayed in an iframe within the same domain.

X-XSS-Protection: This field is used to enable or disable the browser's built-in Cross-Site Scripting (XSS) filter. It is typically set to 1; mode=block, indicating that the XSS filter is enabled, and if an XSS attack is detected, the browser will stop loading the page and display an error message.

X-Content-Type-Options: This field is used to control whether the browser should attempt to sniff the content type of the response and modify the response's MIME type accordingly. The value nosniff indicates that the browser should not attempt to sniff the response's content type and should respect the Content-Type header provided by the server.

4: Can you extract the transmitted HTML code of the visited web page? If not, why? (Note that there is no encryption: neverssl is called that for a reason).

Yes. The transmitted HTML code can be extracted of the visited web page.

```
<html>
    <head>
        <title>NeverSSL - helping you get online</title>

        <style>
        body {
            font-family: Montserrat, helvetica, arial, sans-serif;
            font-size: 16x;
            color: #444444;
            margin: 0;
        }
        h2 {
            font-weight: 700;
            font-size: 1.6em;
            margin-top: 30px;
        }
        p {
            line-height: 1.6em;
        }
        .container {
            max-width: 650px;
            margin: 20px auto 20px auto;
            padding-left: 15px;
            padding-right: 15px
        }
        .header {
            background-color: #42C0FD;
            color: #FFFFFF;
            padding: 10px 0 10px 0;
            font-size: 2.2em;
        }
        .header {
            background-color: #42C0FD;
            color: #FFFFFF;
            padding: 10px 0 10px 0;
            font-size: 2.2em;
        }
        <!-- CSS from Mark Webster https://gist.github.com/markcwebster/9bdf30655cdd5279bad13993ac87c85d -->
        </style>
    </head>
    <body>

    <div class="header">
        <div class="container">
        <h1>NeverSSL</h1>
        </div>
    </div>

    <div class="content">
    <div class="container">

    <h2>What?</h2>
    <p>This website is for when you try to open Facebook, Google, Amazon, etc
    on a wifi network, and nothing happens. Type "http://neverssl.com"
    into your browser's url bar, and you'll be able to log on.</p>

    <h2>How?</h2>
    <p>neverssl.com will never use SSL (also known as TLS). No
    encryption, no strong authentication, no <a
    href="https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security">HSTS</a>,
    no HTTP/2.0, just plain old unencrypted HTTP and forever stuck in the dark
    ages of internet security.</p>
```

```
    ages of internet security.</p>

    <h2>Why?</h2>
    <p>Normally, that's a bad idea. You should always use SSL and secure
    encryption when possible. In fact, it's such a bad idea that most websites
    are now using https by default.</p>

    <p>And that's great, but it also means that if you're relying on
    poorly-behaved wifi networks, it can be hard to get online.  Secure
    browsers and websites using https make it impossible for those wifi
    networks to send you to a login or payment page. Basically, those networks
    can't tap into your connection just like attackers can't. Modern browsers
    are so good that they can remember when a website supports encryption and
    even if you type in the website name, they'll use https.</p>

    <p>And if the network never redirects you to this page, well as you can
    see, you're not missing much.</p>

    <a href="https://twitter.com/neverssl">Follow @neverssl</a>

    </div>
    </div>
    </body>
</html>
```

Exercise 5: IP Addressing and Subnets

1. **What is the lowest and the highest IP address that belongs in this IP address range? What is the broadcast address of this IP address range? How many hosts can your network support?**

The network address is 172.29.8.0, the lowest IP address is 172.29.8.1, the highest is 127.29.8.254, the broadcast address is 172.29.8.255, and 254 hosts can be used in this network range.

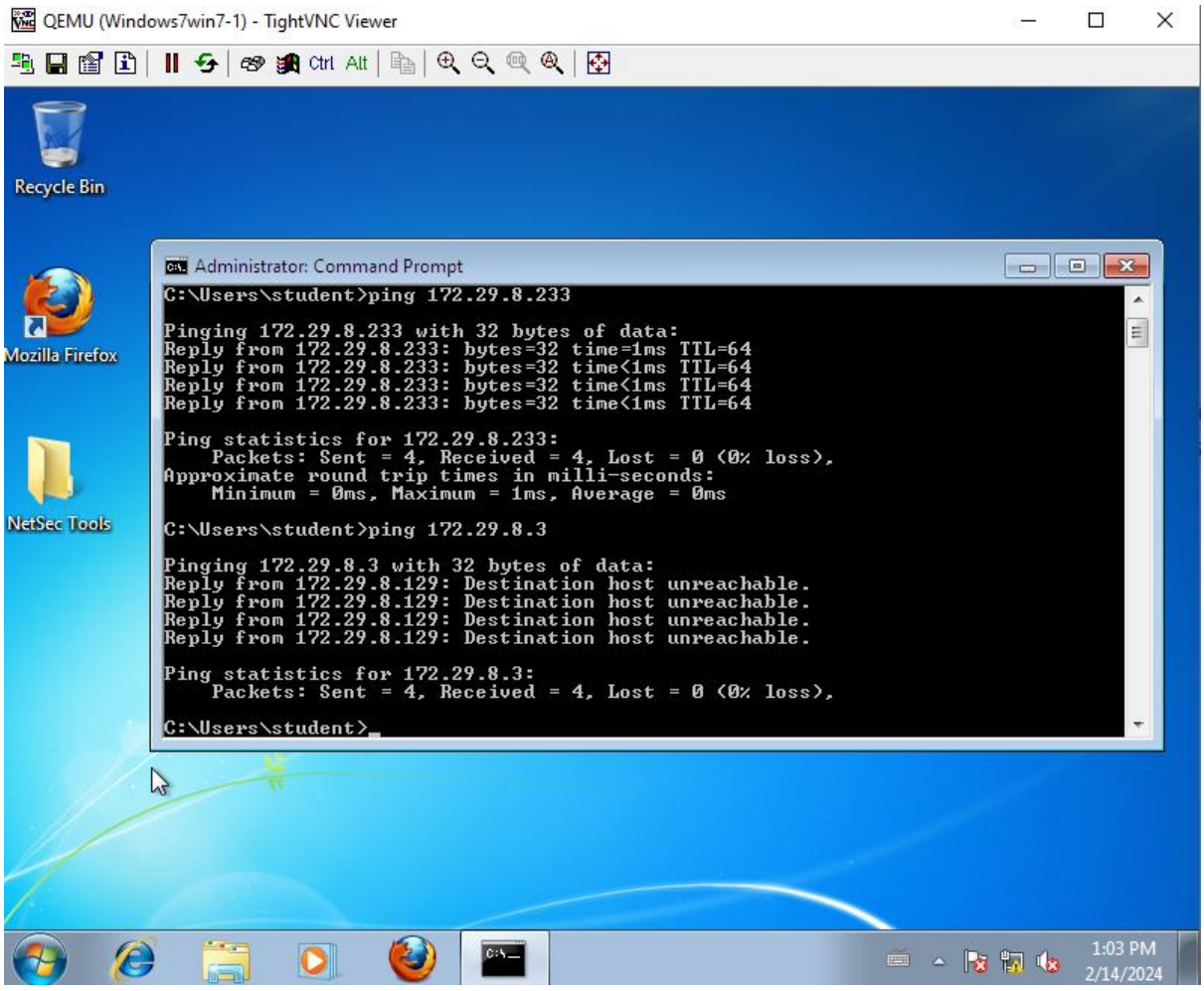
2. **Write down the IP address that you have manually assigned to the 'Windows 7 VM. Identify the network ID of this IP address as well as its host ID of it. A technique called subnetting allows breaking a given IP address range into smaller, better manageable blocks. A subnet mask groups the network prefix along with some high-order bits from the host part into forming an 'extended' network prefix. The subnet mask is used to determine the Network ID of an IP address via the logical AND operation in the process known as "Binary AND–ing".**

The IP address is 172.29.8.2 and the subnet mask is 255.255.255.0. It's obtained by performing a bitwise AND operation between the IP address and the subnet mask, so Network ID: 172.29.8.0. The remaining bits after determining the network ID constitute the host ID. So the host ID is 2.

3. **Splitting the above IP address range in half, gives you two smaller subnets, each with a subnet mask of /25 (i.e. 255.255.255.128). What is the IP address range and the broadcast address of each subnet? How many hosts can they support each? Pick one IP address belonging to each subnet and write them down along with their network ID and host ID.**

- Subnet 1:
    - Network Address: 172.29.8.0
    - Broadcast Address: 172.29.8.127
    - Usable Host Range: 172.29.8.1 - 172.29.8.126
    - Number of Usable Hosts: 126
- Subnet 2:
    - Network Address: 172.29.8.128
    - Broadcast Address: 172.29.8.255
    - Usable Host Range: 172.29.8.129 - 172.29.8.254
    - Number of Usable Hosts: 126

4. **Is the ping successful? Why or why not? Can you capture ping packets on any of the links?**



The ping from Win 7 to Linux Server is successful because they are in the same subnet, but the ping from Win 7 to Linux Client failed because they are from two different subnets. In general, if the two hosts belong to different subnets, they may not be able to communicate directly without proper routing configured. If there is a router connecting the two subnets and routing is configured correctly, the ping should be successful.

10	7.354207	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
11	8.354178	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
12	9.544950	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
13	10.542295	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
14	11.542299	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
15	12.757567	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
16	13.754713	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
17	14.754397	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
18	15.945423	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
19	16.942584	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
20	17.942341	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
21	19.159357	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
22	20.154499	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
23	21.154515	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
24	22.346761	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
25	23.347189	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
26	24.346469	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
27	25.558717	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
28	26.558536	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
29	27.558772	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
30	28.747021	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
31	29.747001	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
32	30.746917	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
33	36.357504	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
34	37.354909	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
35	38.354997	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
36	42.758336	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
37	43.755857	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
38	44.755072	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
39	49.158899	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
40	50.155104	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
41	51.155157	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233
42	55.559052	0c:54:53:5e:00:00	Broadcast	ARP	42 Who has 172.29.8.133? Tell 172.29.8.233

There is only an ARP request captured when using ping from Win 7 to LinuxServer.

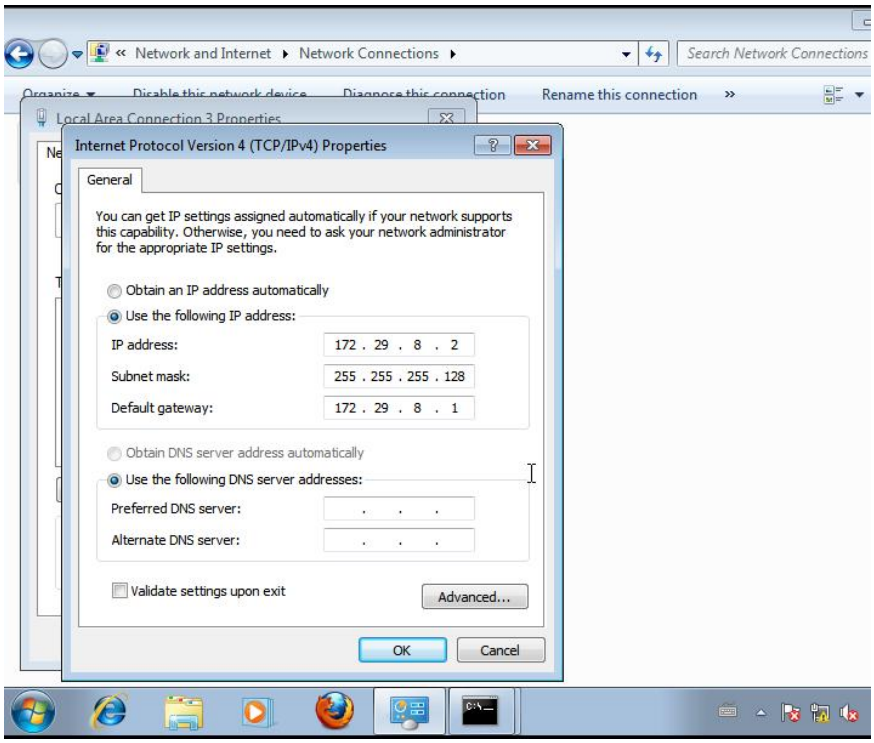
Exercise 6: Basic Internetworking

Activity 1

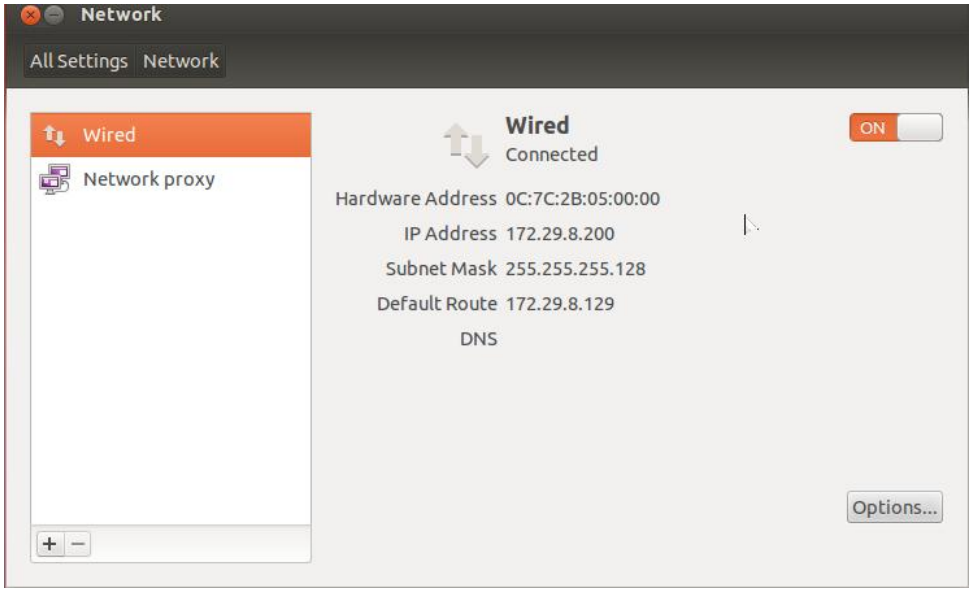
Issue a ping command from one VM to another. Is the ping completed successfully? Why or why not? If not how can the problem be solved? Which entry has been added to the VMs' host routing tables?

Successfully.

Windows VM and Linux VM are located in two different subnets and connected through a router. The router's two interfaces (FastEthernet0/0 and FastEthernet0/1) are connected to these subnets and configured with two different gateways.
The IP address of the Windows device is 172.29.8.2 and the subnet mask is 255.255.255.128, which means that the subnet ranges from 172.29.8.0 to 172.29.8.127. The IP address of the Windows device is 172.29.8.2, which belongs to the subnet 172.29. 8.0 is the subnet of the network address, the subnet mask is /25, and the default gateway is 172.29.8.1



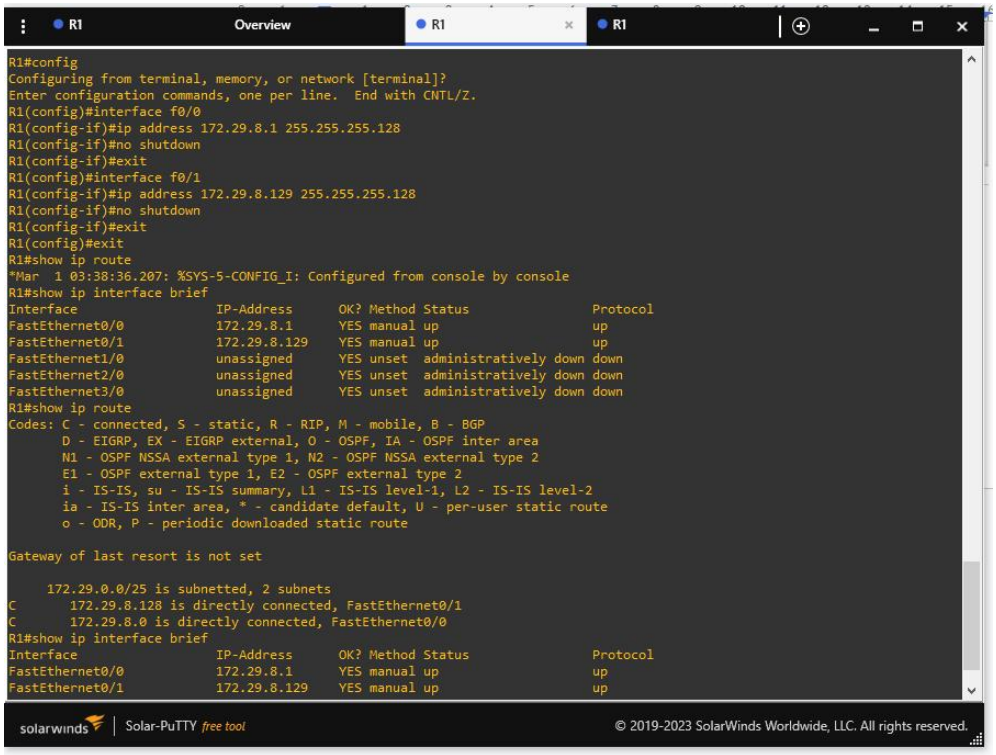
The Linux IP address is 172.29.8.200 and the subnet mask is 255.255.255.128, which means the subnet ranges from 172.29.8.128 to 172.29.8.255 and the default gateway is 172.29.8.129.



The interface on the router is configured correctly and can forward data between the two different subnets.

The gateway address for Windows VM is 172.29.8.1. We connected the F0/0 interface of the router to the Windows VM, the F0/0 interface was configured to be on the same subnet as the Windows VM, and we set its IP address to the gateway address of the Windows VM, 172.29.8.1.

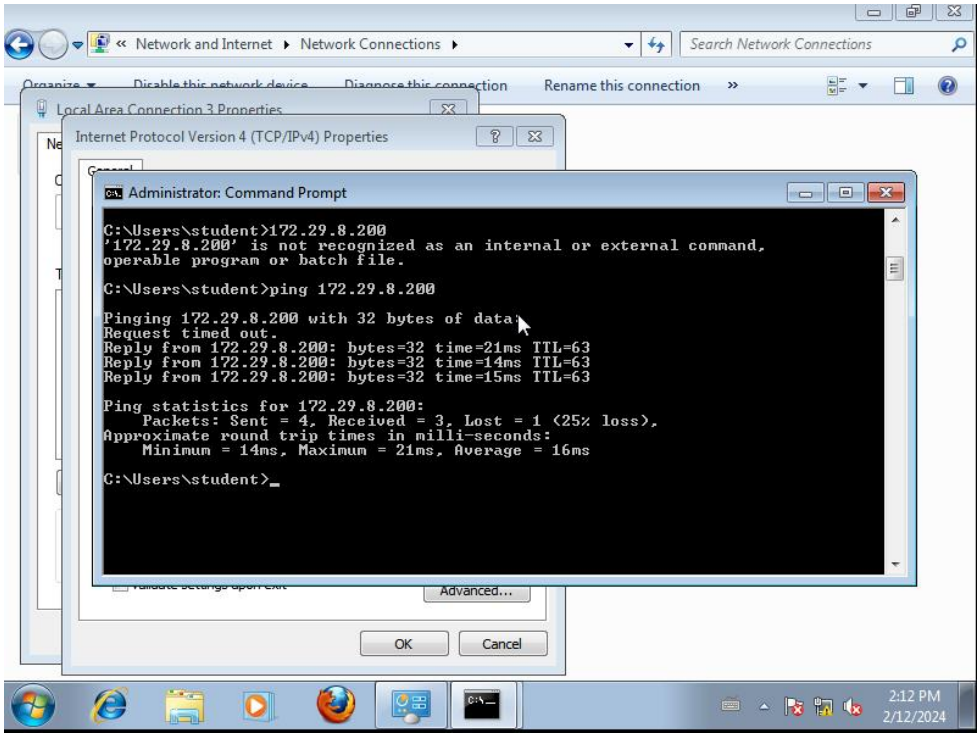
The gateway address of the Linux VM is 172.29.8.129. Therefore, we connected the router's F0/1 interface to the Linux VM and configured the F0/1 interface to be on the same subnet as the Linux VM. We set its address to the Linux VM's gateway address 172.29.8.129.

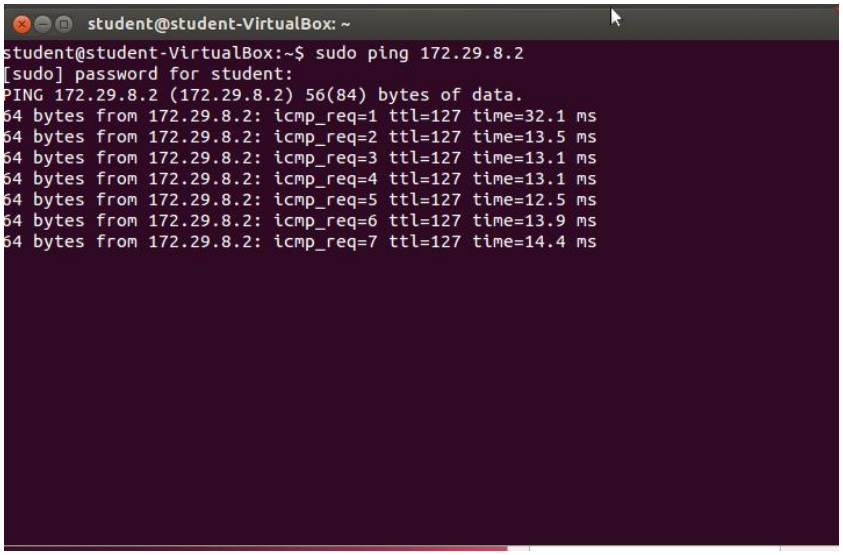


After setting up the F0/0 and F0/1 interfaces on the router, it indicates that the router has established effective connections in two different subnets.

- The F0/0 interface is connected to the subnet where the Windows VM resides and is configured with the same IP address as the gateway of the Windows VM, making the router the default gateway for the subnet where the Windows VM resides.
- The F0/1 interface is connected to the subnet where the Linux VM resides and is configured with the same IP address as the gateway of the Linux VM, making the router the default gateway for the subnet where the Linux VM resides.

With these settings, the router can now forward data between these two different subnets, enabling communication across networks. The Windows VM and Linux VM can now communicate with each other through the router and can access other networks or the internet.





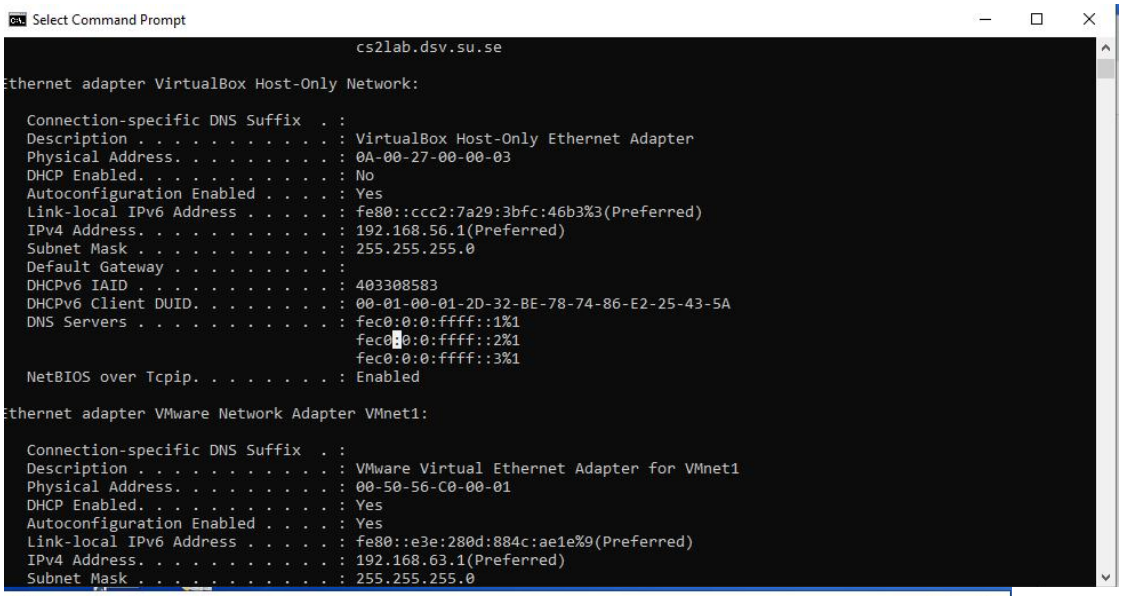
Windows VM and Linux VM are located in two different subnets and connected through a router. The router's two interfaces (FastEthernet0/0 and FastEthernet0/1) are connected to these subnets and configured with two different gateways.

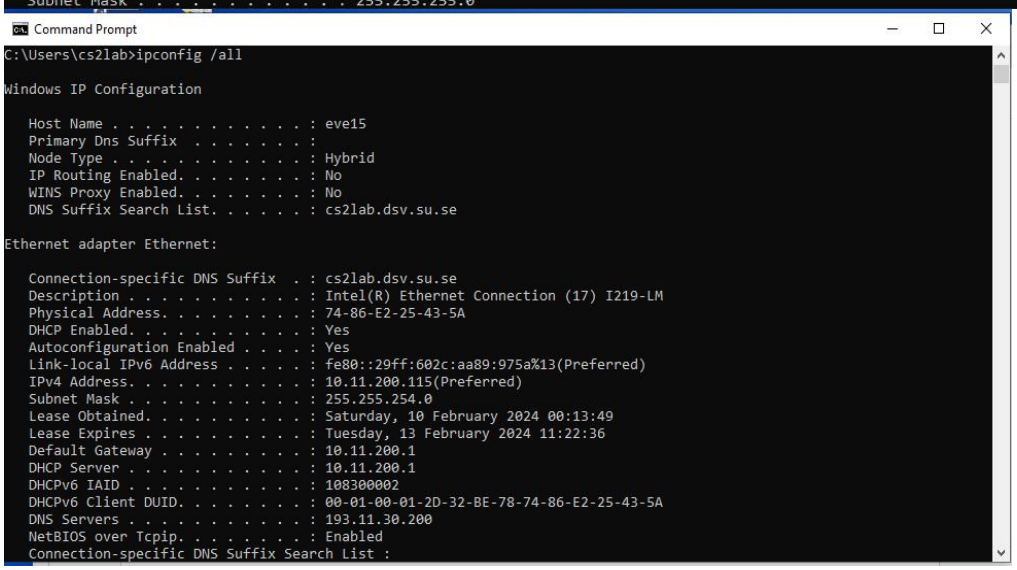
Exercise 7: Application Layer Protocols (DHCP and DNS)

1. **Where is that server (i.e. who runs that server)? Do you have one or two (or several) DNS servers configured? Why may there be more than one?**

Where is that server:

DNS server: 193.11.30.200 (We have only one DNS server configured). We found the owner's information of 193.11.30.200 in the whois online query service tool as follows. (OrgName: RIPE Network Coordination Centre; OrgId: RIPE; Address: P.O. Box 10096; City: Amsterdam)





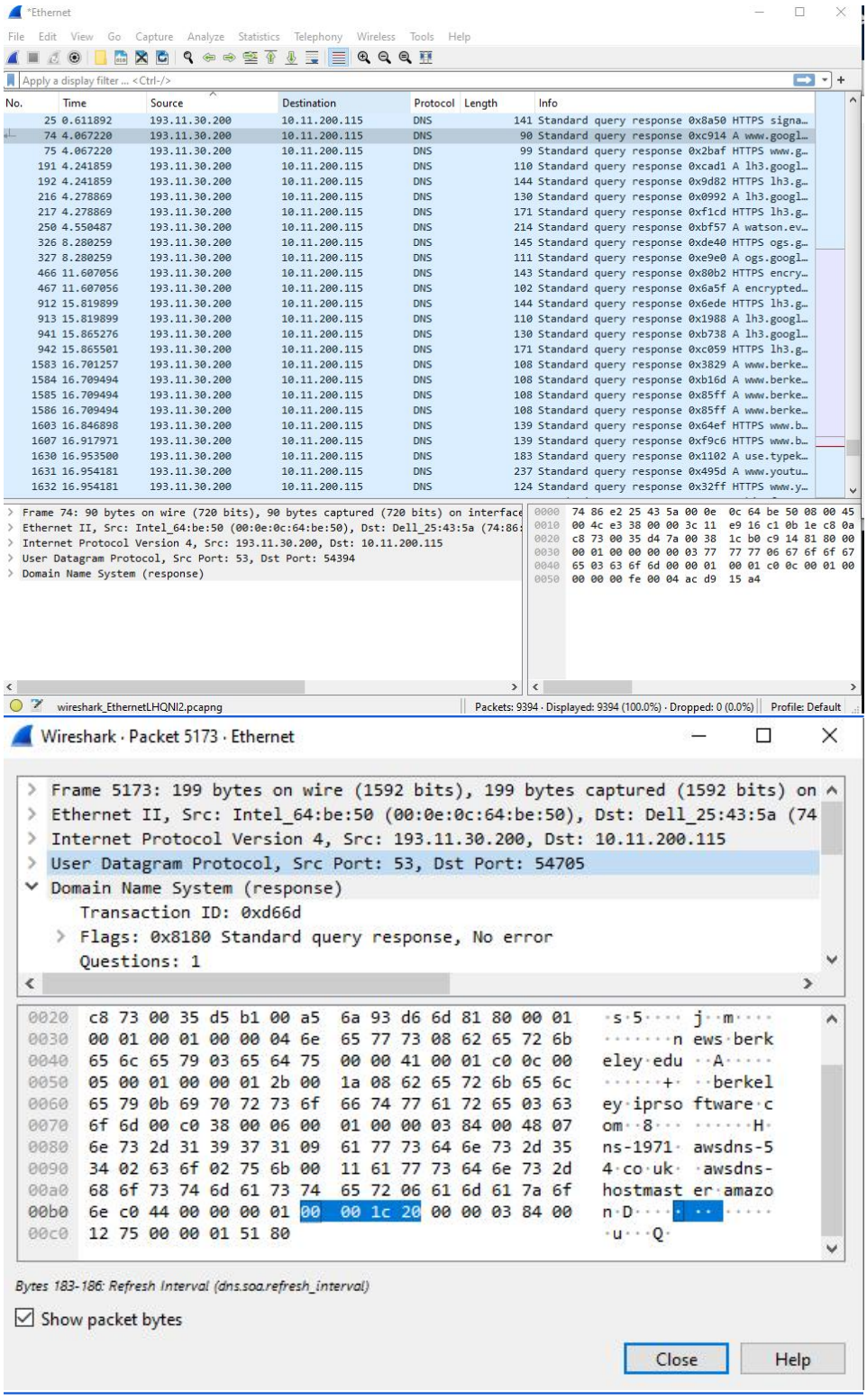
```
NetRange:       193.0.0.0 - 193.255.255.255
CIDR:           193.0.0.0/8
NetName:        RIPE-CBLK
NetHandle:      NET-193-0-0-0-1
Parent:          ()
NetType:        Allocated to RIPE NCC
OriginAS:
Organization:   RIPE Network Coordination Centre (RIPE)
RegDate:        1992-08-12
Updated:        2009-03-25
Comment:        These addresses have been further assigned to users in
Comment:        the RIPE NCC region. Contact information can be found in
Comment:        the RIPE database at http://www.ripe.net/whois
Ref:            https://rdap.arin.net/registry/ip/193.0.0.0

ResourceLink:  https://apps.db.ripe.net/search/query.html
ResourceLink:  whois.ripe.net


OrgName:        RIPE Network Coordination Centre
OrgId:          RIPE
Address:        P.O. Box 10096
City:           Amsterdam
StateProv:
PostalCode:     1001EB
Country:        NL
RegDate:
Updated:        2013-07-29
Ref:            https://rdap.arin.net/registry/entity/RIPE

ReferralServer:  whois://whois.ripe.net
ResourceLink:  https://apps.db.ripe.net/search/query.html

OrgAbuseHandle: ABUSE3850-ARIN
OrgAbuseName:   Abuse Contact
OrgAbusePhone:  +31205354444
OrgAbuseEmail:  abuse@ripe.net
OrgAbuseRef:    https://rdap.arin.net/registry/entity/ABUSE3850-ARIN

OrgTechHandle: RNO29-ARIN
OrgTechName:   RIPE NCC Operations
OrgTechPhone:  +31 20 535 4444
OrgTechEmail:  hostmaster@ripe.net
OrgTechRef:    https://rdap.arin.net/registry/entity/RNO29-ARIN
```

Why may there be more than one:

Having more than one DNS server can prevent the system from resolving domain names when the server encounters problems. It can also prevent the server from being overloaded.

2. **Check the captured DNS-traffic: What port does the DNS server use for the communication? Which transport layer protocol is used? Which flag(s) did the DNS query header have enabled and what is the Transaction ID? Of which type is the query?" The received DNS reply is cached locally by the OS for as much time as the TTL field indicates. You can clear the local DNS cache on Windows by issuing the command 'ipconfig /flushdns' in an administrative command line."**

We visit 1 website: https://www.berkeley.edu/ Then we open wireshark to capture DNS traffic. We chose packet 5173 for analysis.



Port, transport layer protocol, flag, transaction ID, and query type are displayed on the interface:

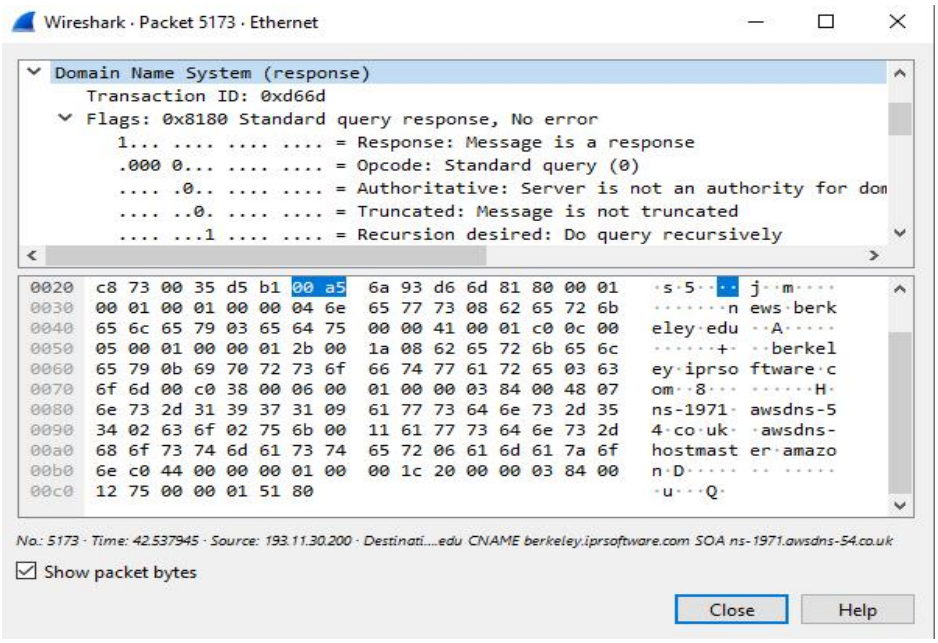
PORT: 53

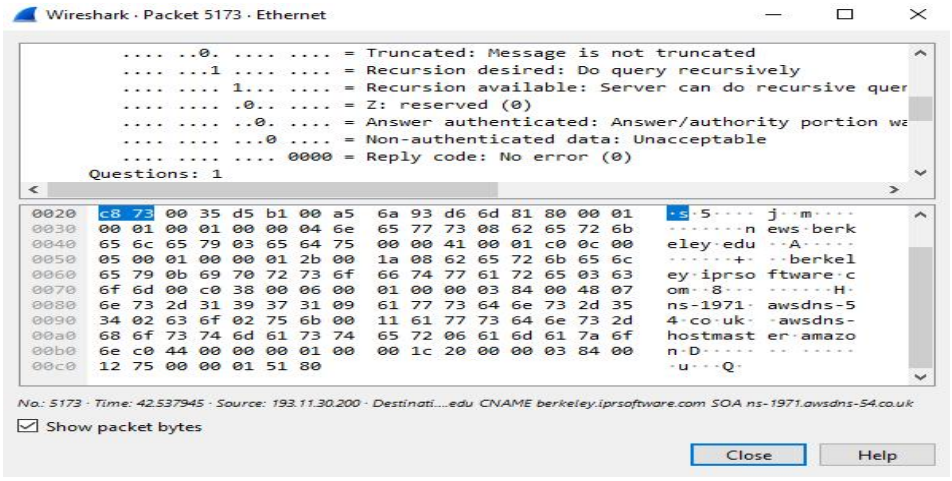
Transport layer protocol: The user datagram protocol(UDP) is used.

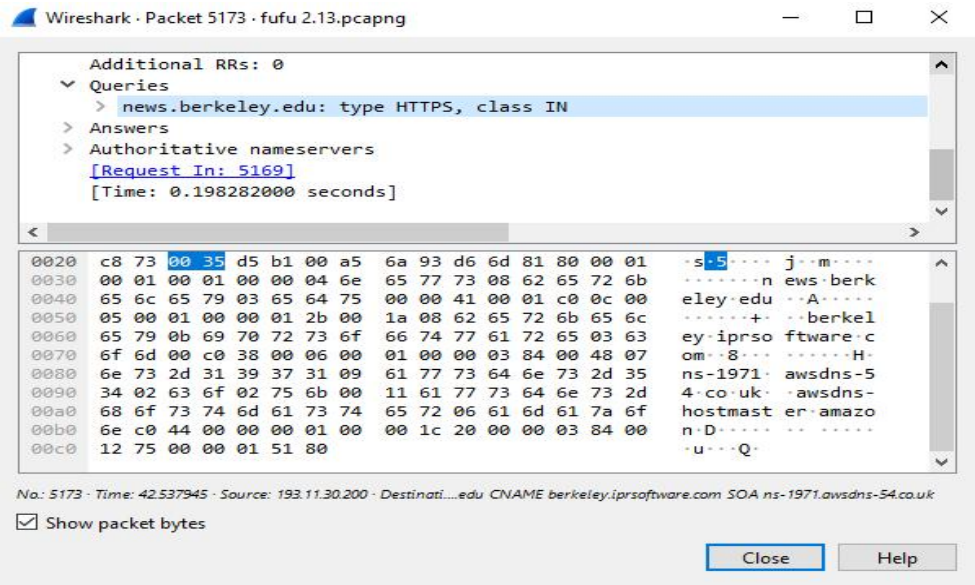
Flag : 0x8180

Transaction ID: 0xd66d

Query type: HTTPS

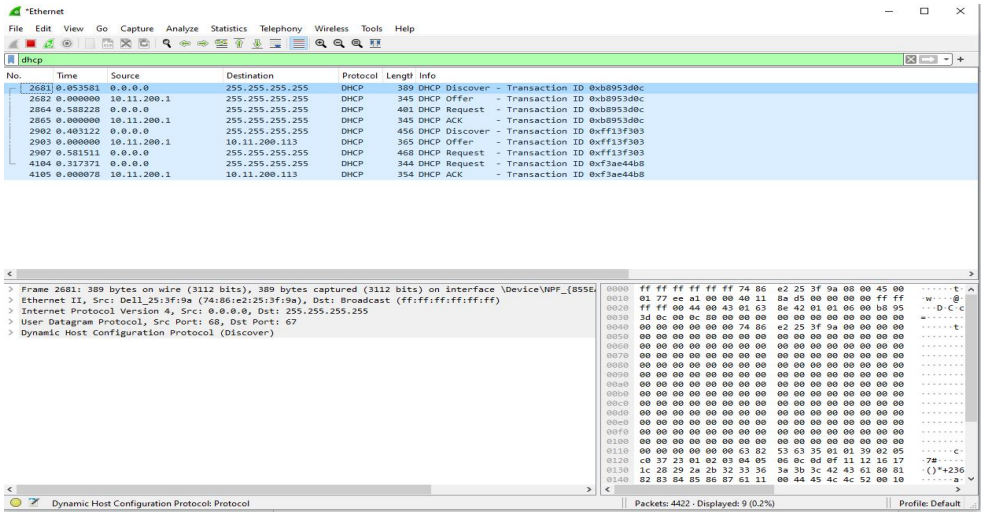




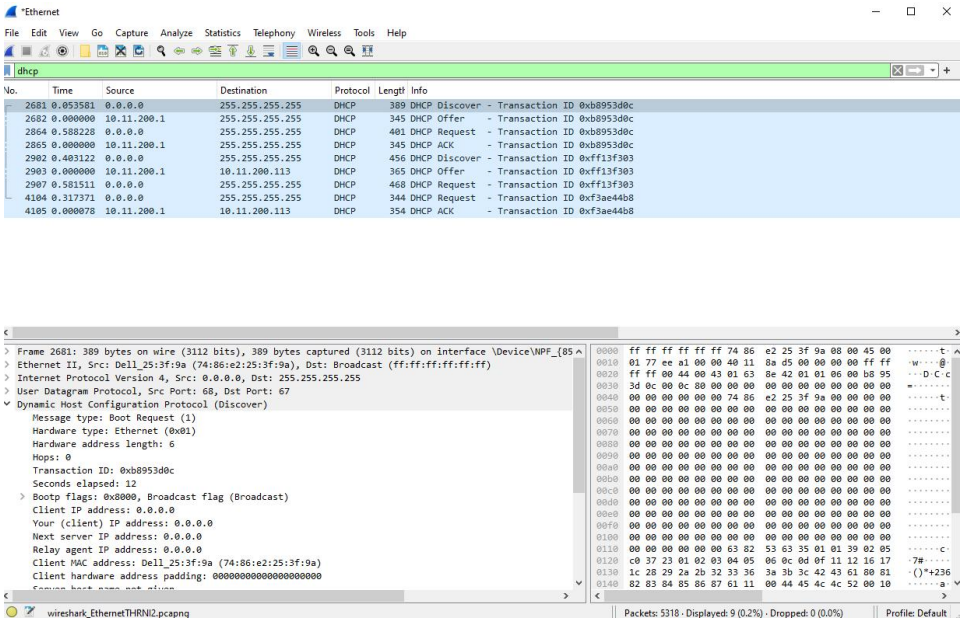


3. **Do that and see if you can capture the DHCP request/response (if not just say that it wasn't possible). Study the DHCP request; how does the computer know which host can give it an IP address before it can communicate on the IP network (as it doesn't yet have an address)?**

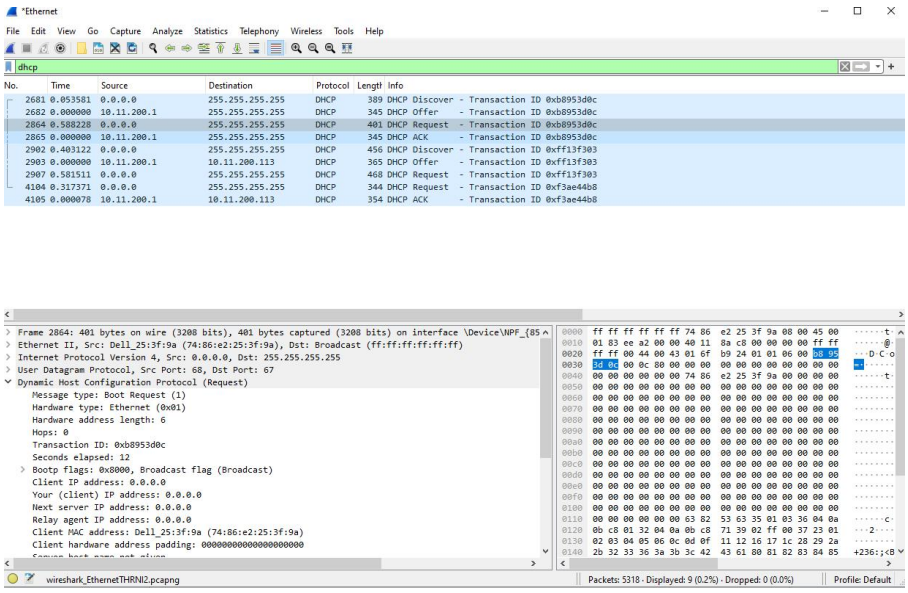
The DHCP request/ response is shown under:



DHCP first discover:



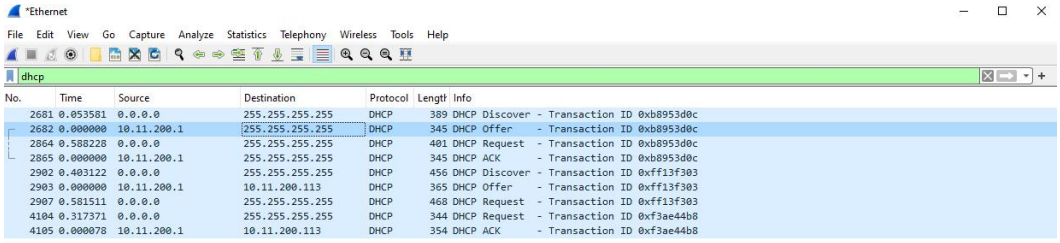
DHCP first request:

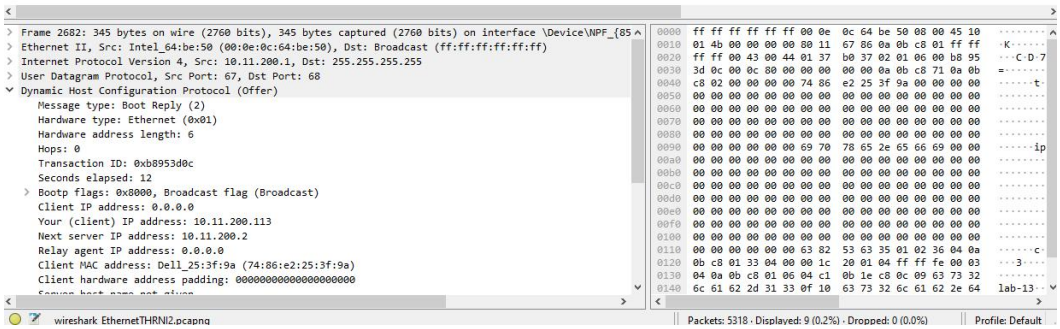


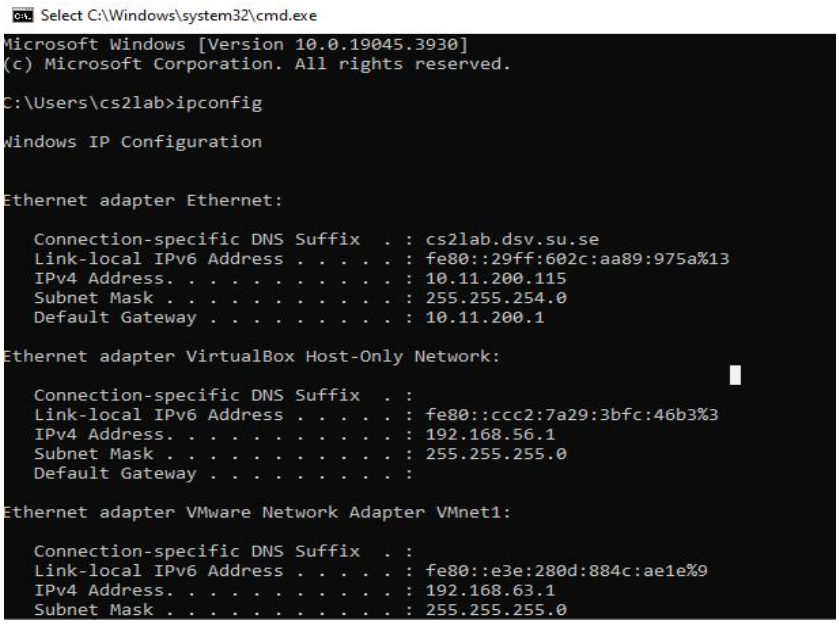
Analysis of a DHCP request packet:

We can see from the DHCP request‘s destination that the DHCP server IP address is 255.255.255.255 The DHCP server listens to this address and will respond to the request.
Client IP address: 0.0.0.0

DHCP first offer:







We use the command ipconfig to check the computer's IP address, Subnet Mask, and Default Gateway.
Computer's IPv4 Address: 10.11.200.115
Subnet Mask: 255.255.254.0
Default Gateway: 10.11.200.1
The source of the DHCP offer is 10.11.200.1, this is the same as the computer's Default Gateway.

How does the computer know which host can give it an IP address before it can communicate on the IP network (as it doesn't yet have an address)?

Because DHCP allows a computer to request an IP address from a DHCP server. And this is how it works:
The computer sends out DHCP discovery to find the DHCP server. DHCP discover asks an available DHCP server for an IP address. The DHCP server responds to the DHCP offer, which includes the IP address, subnet mask, and default gateway that the DHCP server can assign to the computer. The computer selects a provided IP address and requests it through a DHCP request.

Reference

A full guide to the 7 layers of the OSI model. (n.d.). Retrieved 1 March 2024, from https://nordlayer.com/blog/guide-to-the-osi-model/

An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware (Request for Comments RFC 826). (1982). Internet Engineering Task Force. https://doi.org/10.17487/RFC0826

Balakrishnan, H., Seshan, S., Amir, E., & Katz, R. H. (1995). Improving TCP/IP performance over wireless networks. *Proceedings of the 1st Annual International Conference on Mobile Computing and Networking*, 2–11. https://doi.org/10.1145/215530.215544

Beale, J., Orebaugh, A., & Ramirez, G. (2006). *Wireshark & Ethereal Network Protocol Analyzer Toolkit*. Elsevier.

Bishop, M., Sullivan, E., & Ruppel, M. (2019). *Computer security: Art and science* (Second edition). Addison-Wesley.

Dynamic Host Configuration Protocol. (2024). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Dynamic_Host_Configuration_Protocol&oldid=1209149891

Martin, J., Mayberry, T., Donahue, C., Foppe, L., Brown, L., Riggins, C., Rye, E. C., & Brown, D. (2017). *A Study of MAC Address Randomization in Mobile Devices and When it Fails* (arXiv:1703.02874). arXiv. https://doi.org/10.48550/arXiv.1703.02874

Nath, A. (2015). *Packet Analysis with Wireshark*. Packt Publishing Ltd.