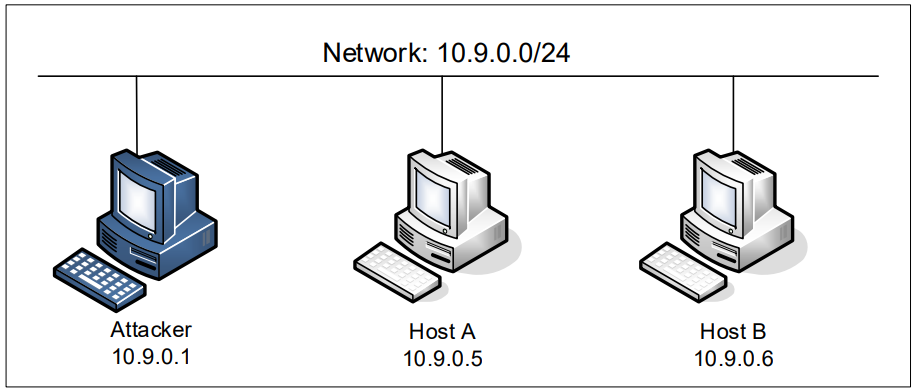
מטלה 6

סטודנט ראשון : אילן מאיר סופיר, ת"ז : **342615648**

סטודנט שני : בן כהן , ת"ז : **207029786**



Task 1.1: Sniffing Packets

Task1.1A :

To know the network’s name, we use « ifconfig » on the attacker and then we have the network’s name :

**br-9ce32ebbb4e2**.

Une image contenant texte

Description générée automatiquement

Python code to sniff icmp packets :

Une image contenant texte

Description générée automatiquement

With the root (sudo) :

Une image contenant texte, capture d’écran, équipement électronique, ordinateur

Description générée automatiquement

All the ICMP packets sent with the command ping 8.8.8.8 (Google DNS) are sniffed successfully.

Without the root :

Une image contenant texte, capture d’écran, équipement électronique, ordinateur

Description générée automatiquement

We can see that the python code task1.1A.py works, but in the ubuntu terminal when I run the code I have :

Une image contenant texte

Description générée automatiquement

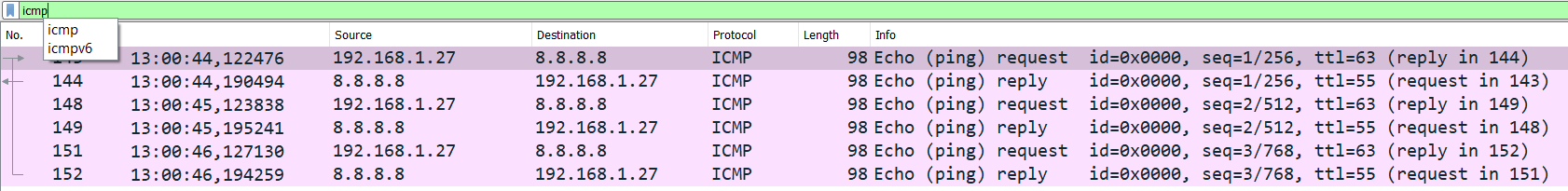
We have this message error : PermissionError: [Errno 1] Operation not permitted.

Why in the docker it’s work for the two mode : with and without the root ? I think it’s beacause when i wrote the command « sudo » for the first time, and then we don’t need to write « sudo » each time.

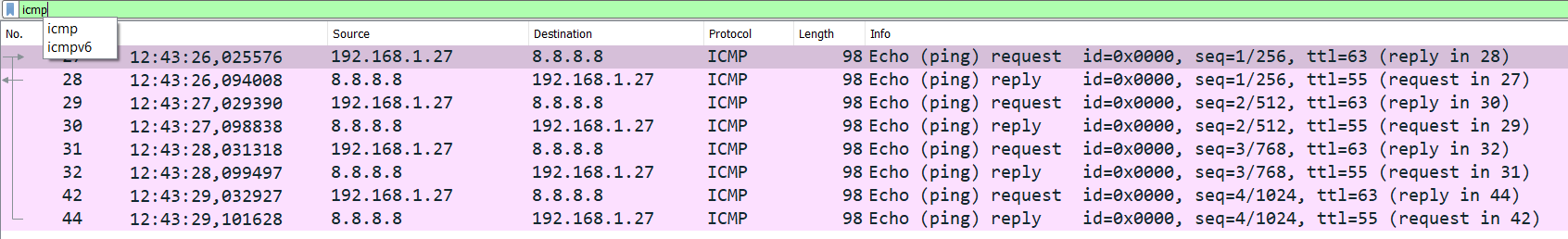
Captures Wireshark :

I work with the docker in Windows. So in Wireshark I don’t know why but I have the IP adress 192.168.1.27 and not the IP adress 10.9.0.5 which is the adress of the HostA.

With sudo : (3 pings)



Without sudo : (4 pings)



Task1.1B :

• Capture only the ICMP packet :

-> see the task 1.1A.

• Capture any TCP packet that comes from a particular IP and with a destination port number 23 :

Code python :

Une image contenant texte

Description générée automatiquement

telnet : works on port 23, telnet is a tool for working exclusively through the terminal, it is a tool for connecting with a far system. It allows you to edit documents, run features, check whether a particular port in the remote system is available or not.

Wireshark capture 1.1B\_2 :

Une image contenant texte, capture d’écran, ordinateur, afficher

Description générée automatiquement

• Capture packets comes from or to go to a particular subnet. You can pick any subnet, such as 128.230.0.0/16; you should not pick the subnet that your VM is attached to :

With ping from 10.9.0.5 to one of the subnet addresses I mentioned:

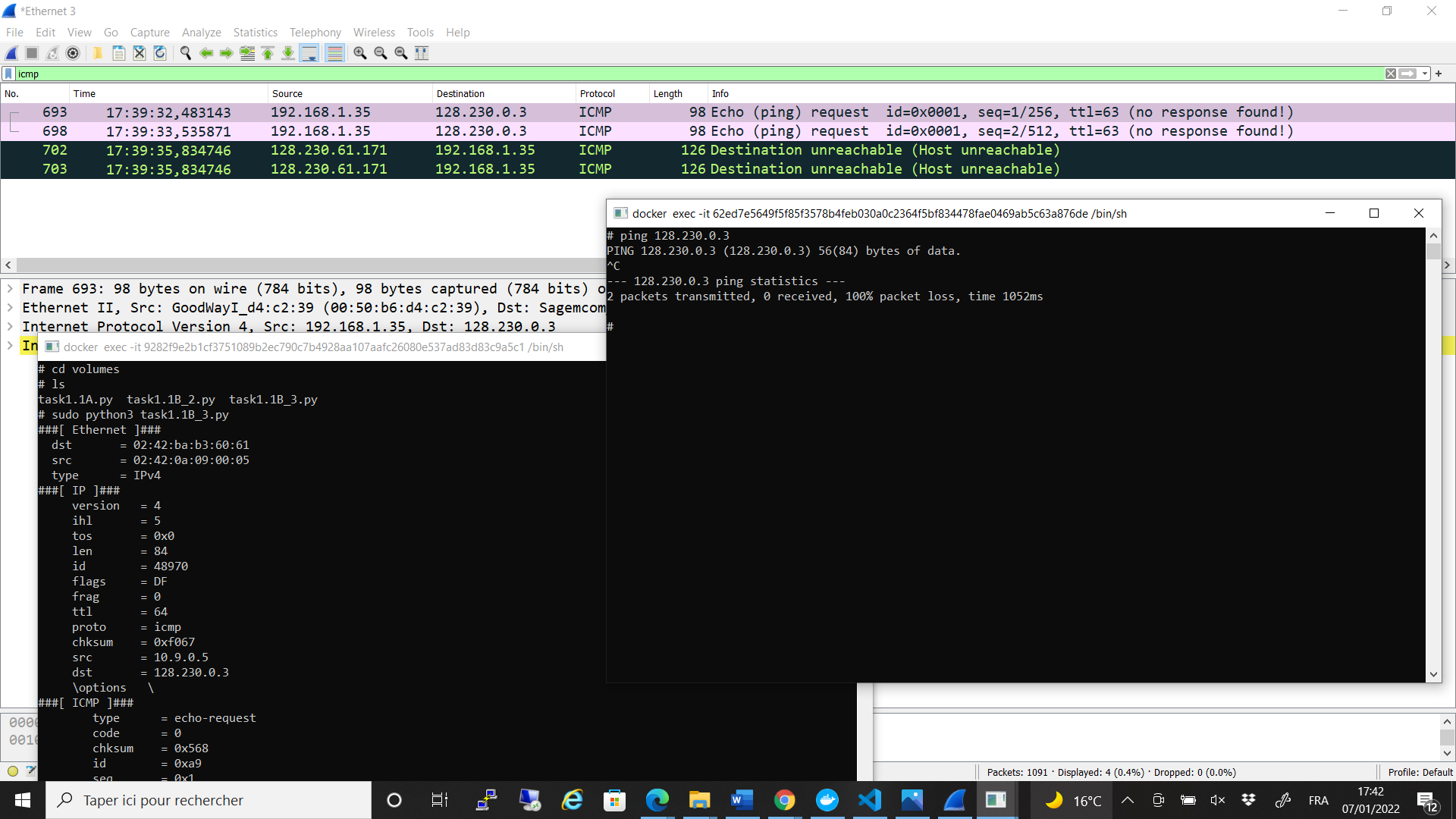
I chose to pingto 128.230.0.3, there is no answer from this address, these are just request-echo (look wireshark) icmp requests.

Python code :

Une image contenant texte

Description générée automatiquement

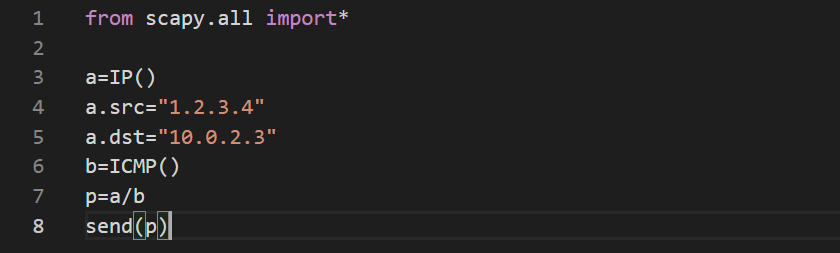
Wireshark capture 1.1B\_3 :



Task 1.2: Spoofing ICMP Packets

In this task we were asked to create our own fact and send it to the address we decide and from the address we decide, I chose the address given in the example of the exercise: 10.0.2.3, the packet does come from an IP (1.2.3.4) that does not exist in the subnet of the containers. We have only ICMP request.

Python code :



Wireshark capture 1.2 :

Une image contenant texte

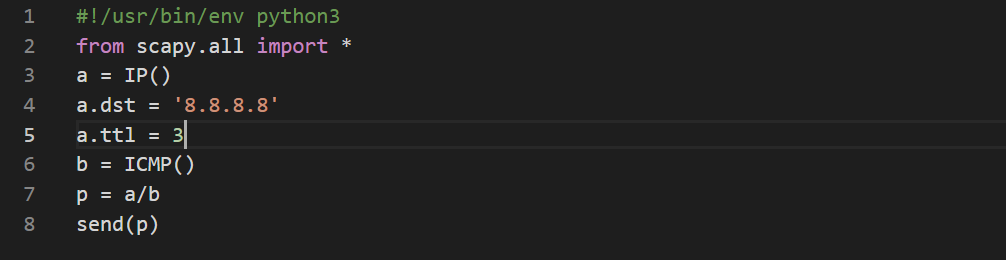
Description générée automatiquement

Task 1.3: Traceroute

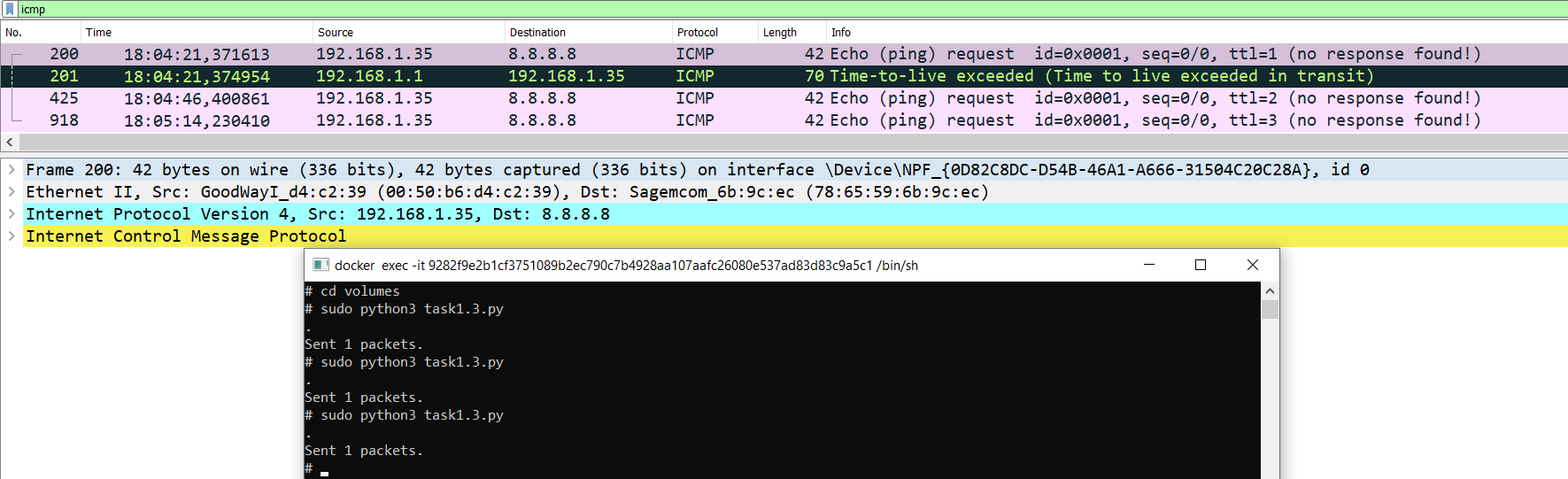
In this question we were asked to know how many routers move from our personal network to a specific IP address when sending Ping to this address. We checked this by sending Ping to google's address (8.8.8.8) and found that as much as TTL (To Time Live) Larger. This way you can get to the router further away until you reach the desired address.

From the picture you can see that there were 2 transitions between routers until we reached the google’s address.

Python code :



Wireshark capture 1.3 :



Task 1.4: Sniffing and-then Spoofing

To answer this question we have developed two computers that simulate attack and user validity (the victim). The attacker runs the program to steal information from the victim.

At first the attacker pings to two non-existent addresses (1.2.3.4 and 10.9.0.99), the attacker branches the passing ICMP facts on the network, and sends the attacker a reply message from the same ping as if the address exists.

In addition, the attacker also sends a Ping to an existing address ("8.8.8.8)" which sends him a reply message back. The victim i think received a reply from "8.8.8.8" but the truth is that the attacker was the one who sent the reply message. An address that already exists on the Internet, a reply will be sent twice - both from the attacker and from the address itself (the original), can be see it in the image below - a message will be printed for us (DUP !) at the end of the double message line that alerts the user that this fact is sent from two different places.

Python code :

Une image contenant texte

Description générée automatiquement

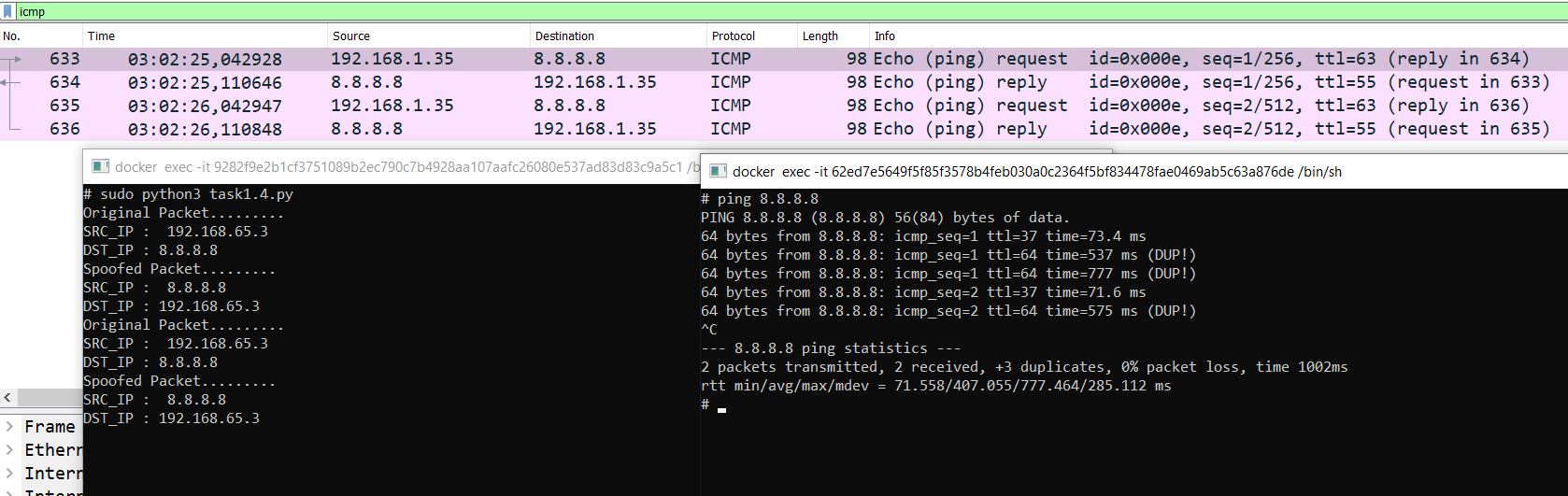
1.2.3.4 :

We can see that in wireshark (1.4\_1) i don’t have a reply but in my terminal yes.

Une image contenant texte

Description générée automatiquement

8.8.8.8 :



Task 2.1A: Understanding How a Sniffer Works

Code C :

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquement

Question 1 : Please use your own words to describe the sequence of the library calls that are essential for sniffer programs. This is meant to be a summary, not detailed explanation like the one in the tutorial or book.

A :

The following calls are essential for this sniffer program :

- pcap\_lookupdev : Find the default device on which to capture.

- pcap\_lookupnet : Is used to determine the IPv4 network number and mask associated with the network device.

- pcap\_open\_live : Open a device to start sniffing.

- pcap\_datalink : Get the link-layer header type to know what type.

- pcap\_compile : Compile a filter expression.

- pcap\_setfilter : Sets the compiled filter.

- pcap\_loop : Process packets from the capture.

- pcap\_freecode : Frees up allocated memory generated.

- pcap\_close : Closes the sniffing session.

Question 2 : Why do you need the root privilege to run a sniffer program ? Where does the program fail if it is executed without the root privilege ?

A : We need root privilege to access to the Network Interface Card which needs admin authorization.

Question 3 : Please turn on and turn off the promiscuous mode in your sniffer program. The value 1 of the third parameter in pcap open live() turns on the promiscuous mode (use 0 to turn it off). Can you demonstrate the difference when this mode is on and off ? Please describe how you can demonstrate this. You can use the following command to check whether an interface’s promiscuous mode is on or off (look at the promiscuity’s value).

A : It can be deduced from the pictures that when Promiscuous mode is off in "0," mode our branch software does not manages to detect any facts that pass through Internet traffic from the second (attacked) VM.

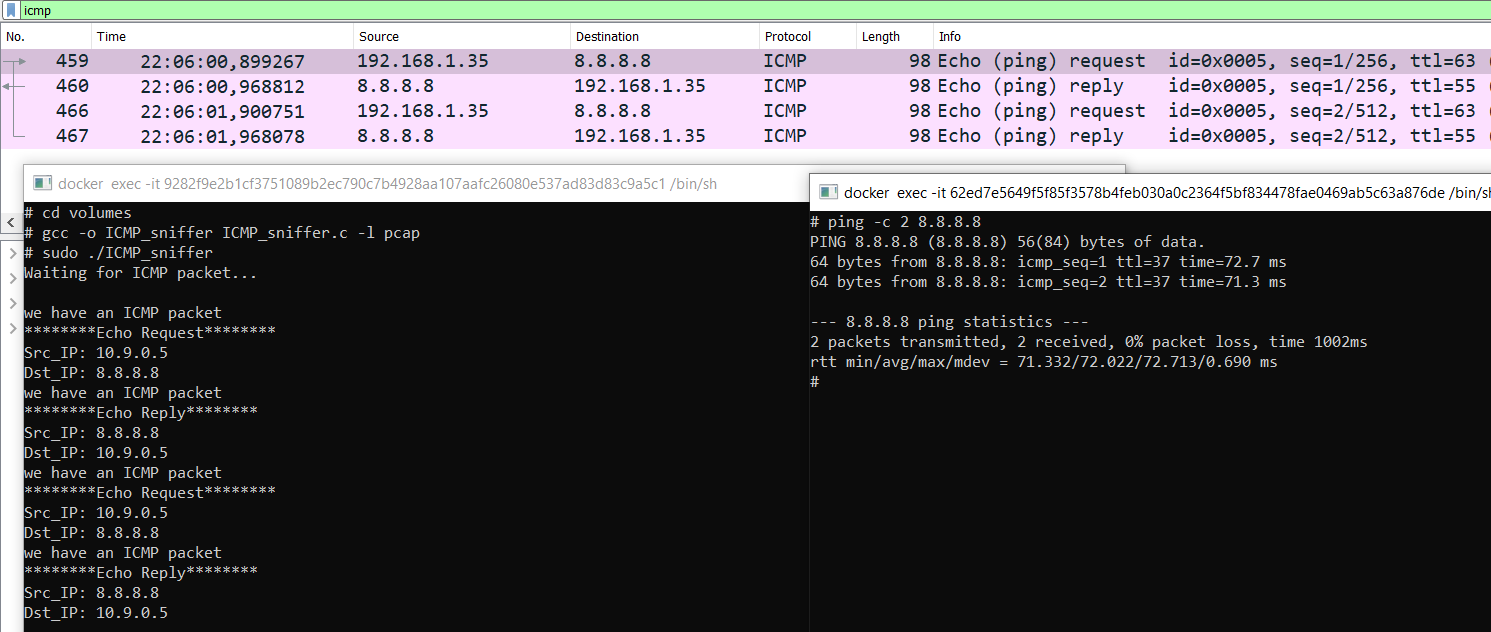
On the other hand, when Promiscuous mode is on in "1" mode, our branch software manages to branch information from users on the network.

Promiscuous mode OFF :

Une image contenant texte

Description générée automatiquement

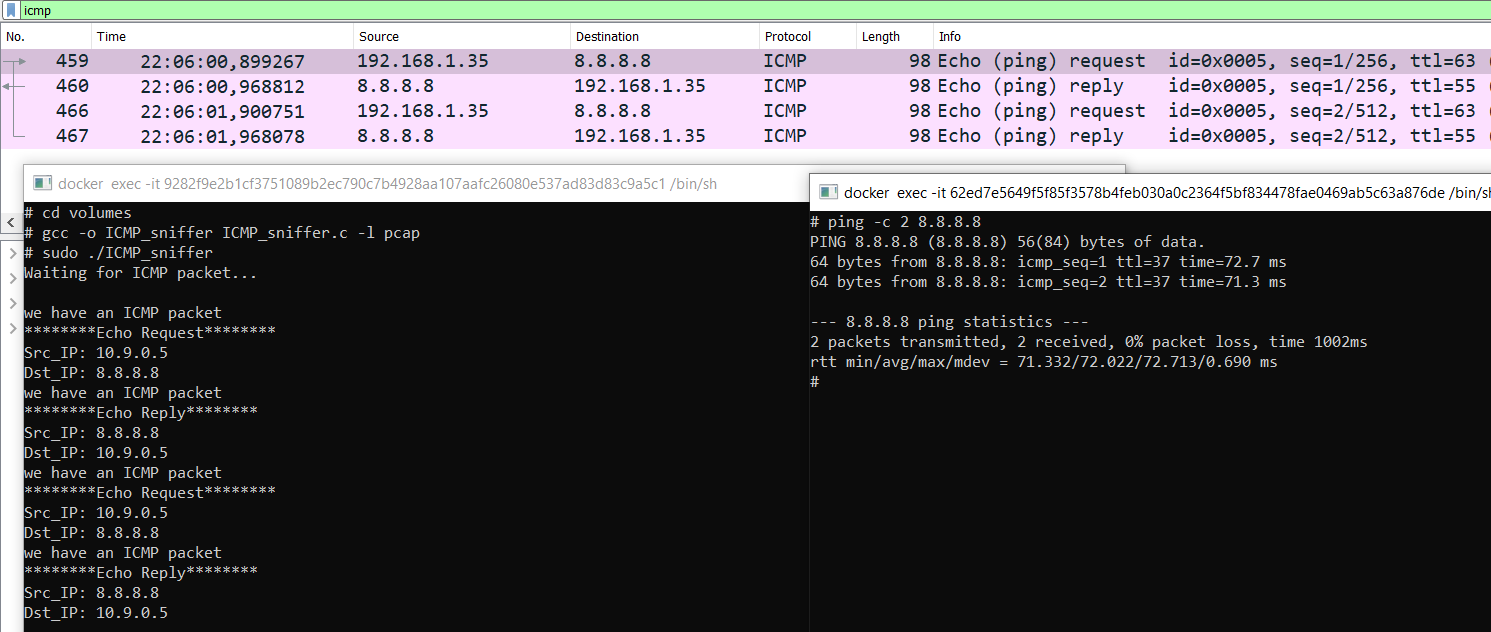
Promiscuous mode ON :



Task 2.1B: Writing Filters :

• Capture the ICMP packets between two specific hosts.

-> look task 2.1A :



• Capture the TCP packets with a destination port number in the range from 10 to 100.

We can capture only the TCP packets with a destination port number in the given range from 10 to 100 by modifying the filter\_exp[] string in the code ICMP\_sniffer:

Une image contenant texte, capture d’écran, écran

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

telnet = port 23 and 10<23<100



Task 2.1C: Sniffing Passwords :

You need to run the command : sudo ./out2

And ping : telnet 10.9.0.6

Then you enter the login : seed

And the password : dees

Une image contenant texte

Description générée automatiquement

seed

In this question the attacker wants to branch out the password of the user (victim), the attacker tries to access the VM by Operation : 10.9.0.6. After that, the user is asked to enter his password for login.

The attacker branches the TCP packets that pass through the Internet traffic and discovers the user's password by the information provided from the data of the facts.

The attacker sees the password at the end of te data. When we read them in a sequence from bottom to the top, the password will be revealed.

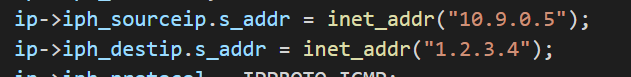
In the image shown above, the user's password is « dees ».

Task 2.2A: Write a spoofing program :

I wrote the code but I can’t see in wireshark like all my matala, I can’t see the reply

The important part of the code :

Une image contenant texte

Description générée automatiquement

Task 2.2B: Spoof an ICMP Echo Request :

A big part is take from the tirgoul.

In this section we were asked to return an ICMP reply to an ICMP request from any machine.

I issue an unreal request spoofedon 10.9.0.5 to the google DNS (8.8.8.8).

And he gives me a reply.

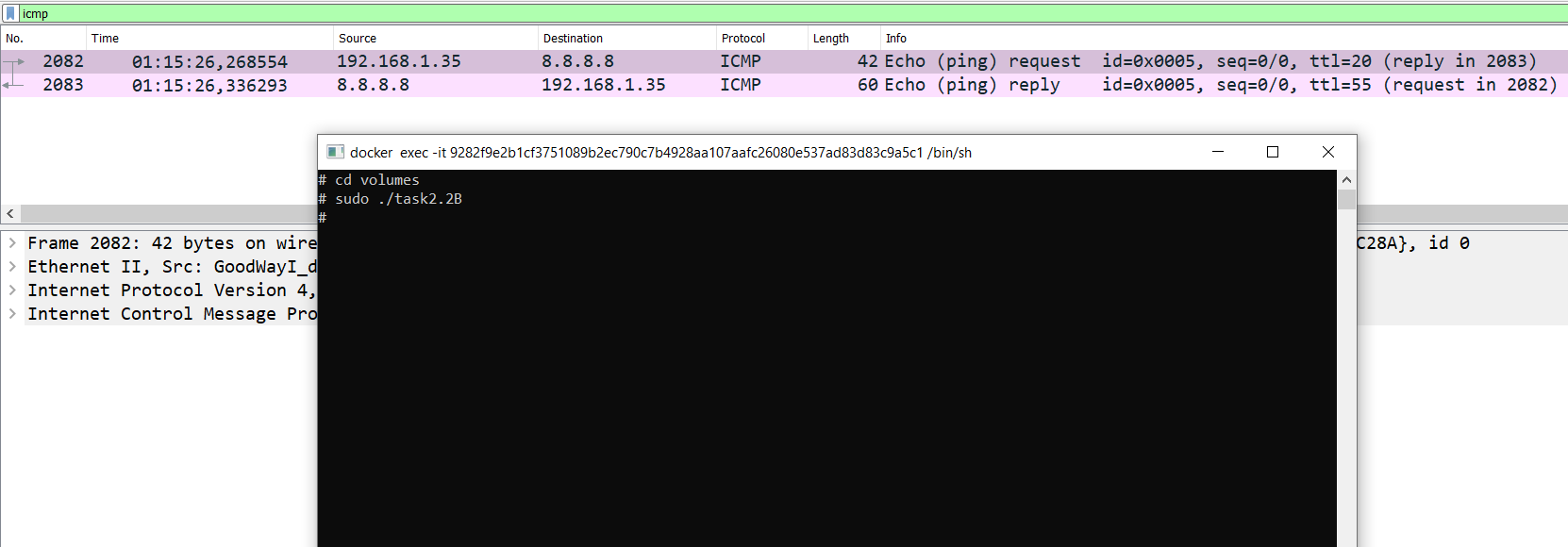
It can be seen that a request was issued to 8.8.8.8 google DNS server, it can be seen that the request came from the IP of me and this is because when creating the containers then the attacker also gets the address 10.9.0.1 and he also gets the address of my so that he can branch because otherwise he will only be able to branch the traffic on his LAN so it came out here that even when he issued a fake fact it came out on the IP of the VM but the answer ICMP reply as requested by the code back to 10.9.0.5 .

The important part of the code :

Une image contenant texte

Description générée automatiquement

Capture Wireshark 2.2B :



• Question 4 : Can you set the IP packet length field to an arbitrary value, regardless of how big the actual packet is ?

The length field in this case should be the length of the IP packet, if not the sendto method (or function) will show an error ‘Invalid Argument’. So yes, it is important that it is the length of the IP packet.

• Question 5 : Using the raw socket programming, do you have to calculate the checksum for the IP header ?

No you don’t need to calculate the checksum, it will be filled by the system.

• Question 6 : Why do you need the root privilege to run the programs that use raw sockets ? Where does the program fail if executed without the root privilege ?

It's restricted to root because it would break some rules for networking that are in place, without root you can't bind a port lower than 1024. With raw sockets you can simulate a server on any port and spoof custom packets which can interfere with inbound traffic.

Task 2.3: Sniff and then Spoof :

We can see that Wireshark displays a successful ping request even though no packet were really sent (100% packet loss), and in the attacker terminal we can also see that the source IP and the destination IP have been successfully swapped, meaning that the spoofing was successful. I don’t know why in wireshark I can’t see this swap.

Une image contenant texte, portable, ordinateur, capture d’écran

Description générée automatiquement