|  |  |
| --- | --- |
| Name | Naman Choudhary |
| SRN | PES2UG20CS209 |
| Section | D |

**Sniffing and Spoofing Lab**

Assignment 1

# **Lab Task Set-1: Using Tools to Sniff and Spoof Packets using Scapy**

# **Task 1.1: Sniffing Packets**

The objective of this task is to learn how to use Scapy to do packet sniffing in Python programs.

## **Task 1.1 A: Sniff IP packets using Scapy**

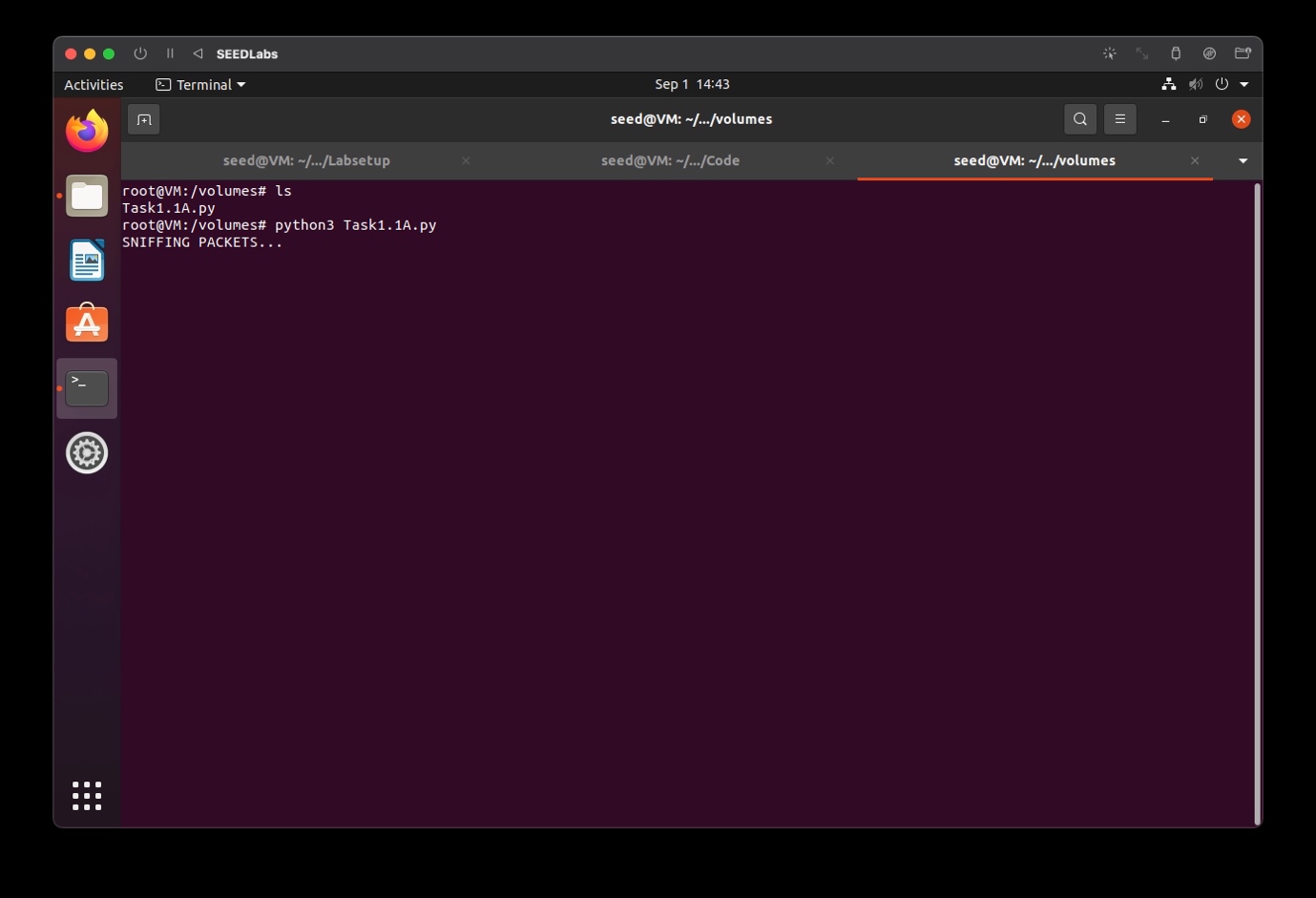
**On the Attacker terminal run the command:**

**# python3 Task1.1A.py**

**Explain on which VM you ran this command and why?**

* I ran the command `**python3 Task1.1A.py` on the attacker’s VM** terminal, as the attacker is the one to sniff the packers from the victim’s VM

**Provide a screenshot of your observations.**



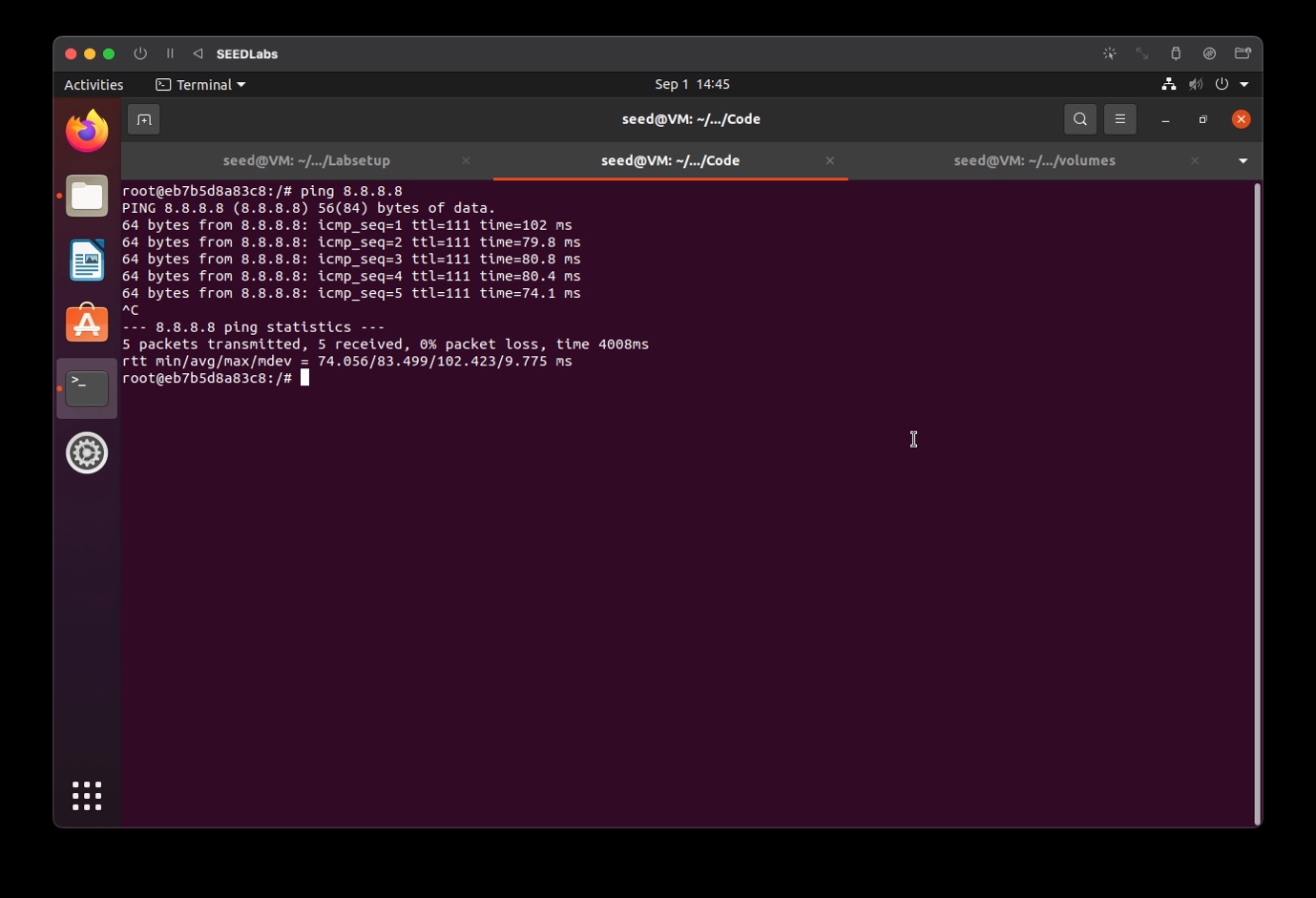
Sniffing on Attacker’s VM shown above

**From the host A machine’s terminal ping a random IP address(8.8.8.8)**

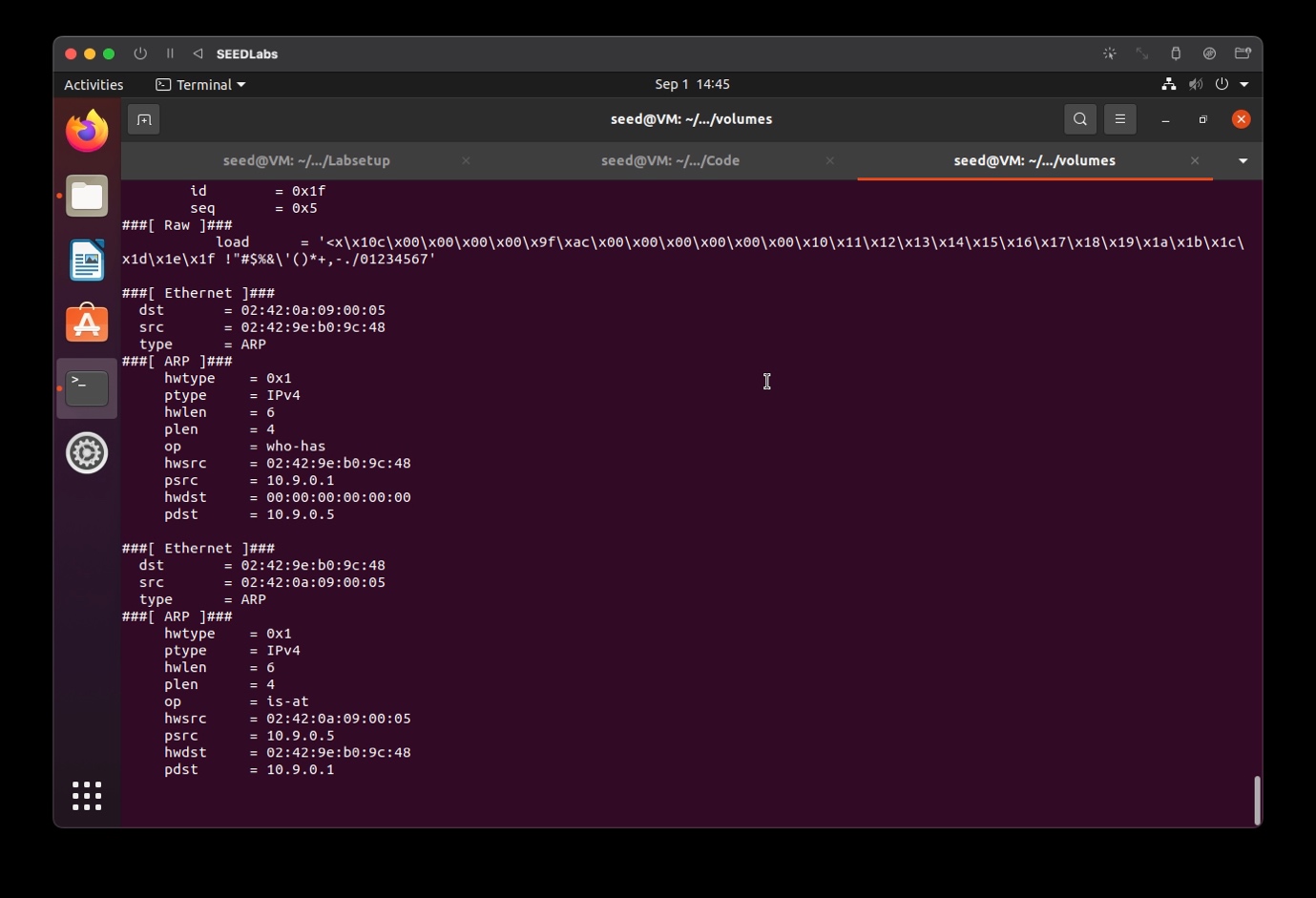
**On the Host A terminal run the command:**

**# ping 8.8.8.8**

**Host A terminal:**

****

**Attacker’s terminal:**

****

Sniffing is successful and its data is being shown on terminal continuously.

**Now, we run the same program without root privileges.**

**On the Attacker terminal run the command:**

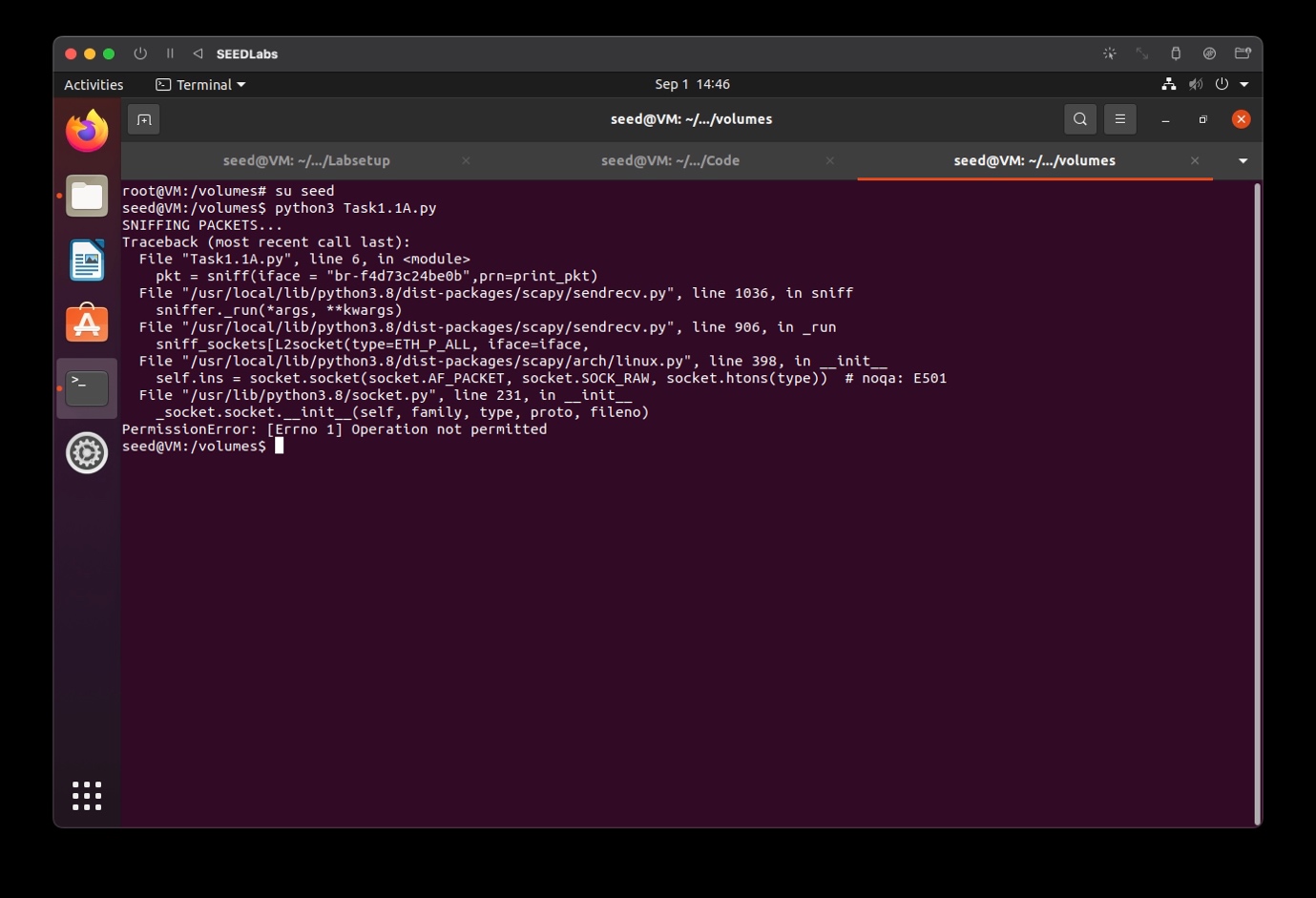
**# su seed**

**$ python3 Task1.1A.py**

**Do you find any issues? If so, why?**

* Yes, there are issues when the command is run, as the Operation is not permitted without root privilages. As seed user is not the root user, it does not have the required administrative access

**Provide a screenshot of your observations.**

****

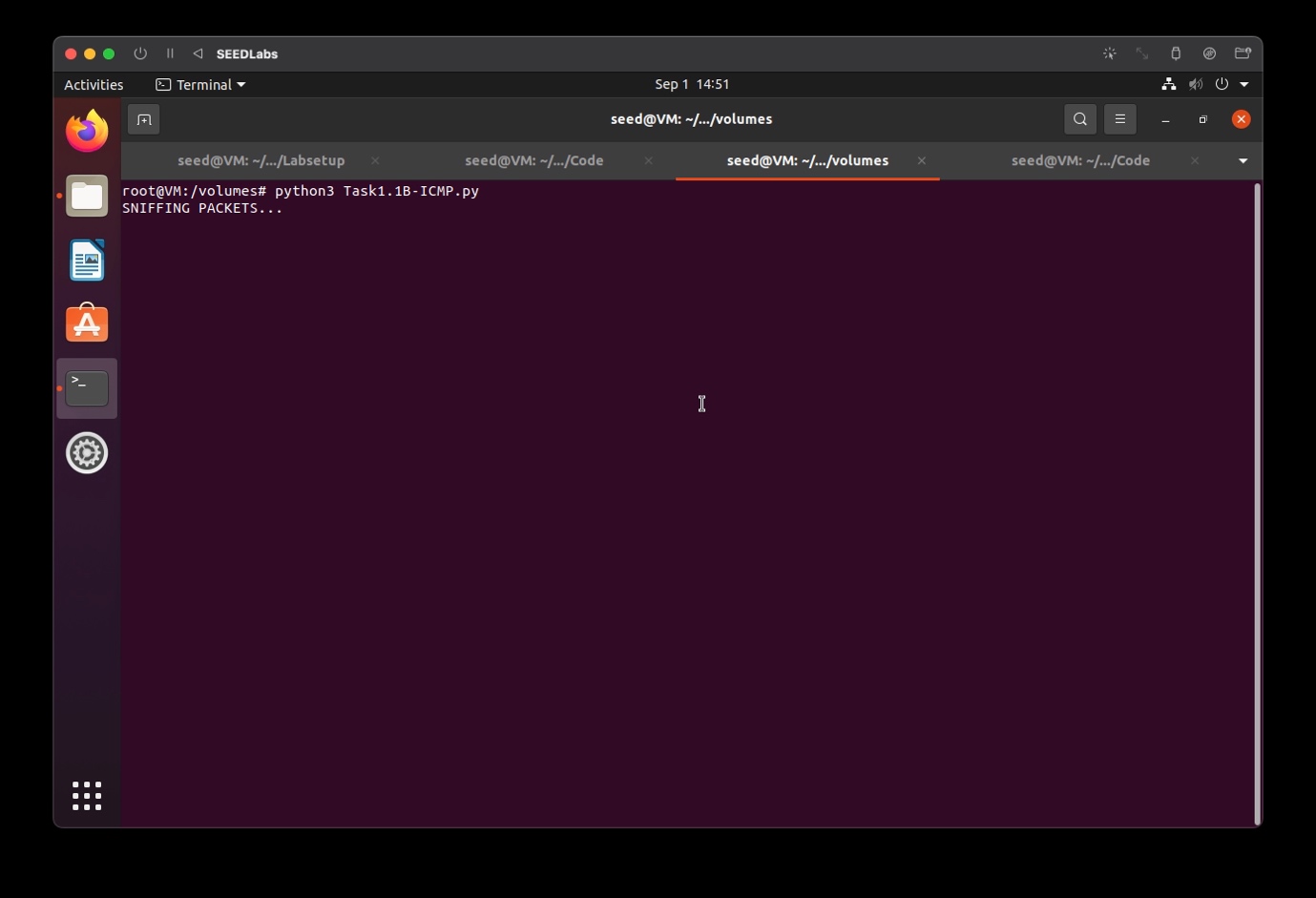
## **Task 1.1 B : Capturing ICMP, TCP packet and Subnet**

**Capture only the ICMP packet**

**On the Attacker terminal run the command:**

**# python3 Task1.1B-ICMP.py**

**Provide a screenshot of your observations**

****

The attacker starts sniffing.

**From the host A machine’s terminal ping a random IP address(8.8.8.8)**

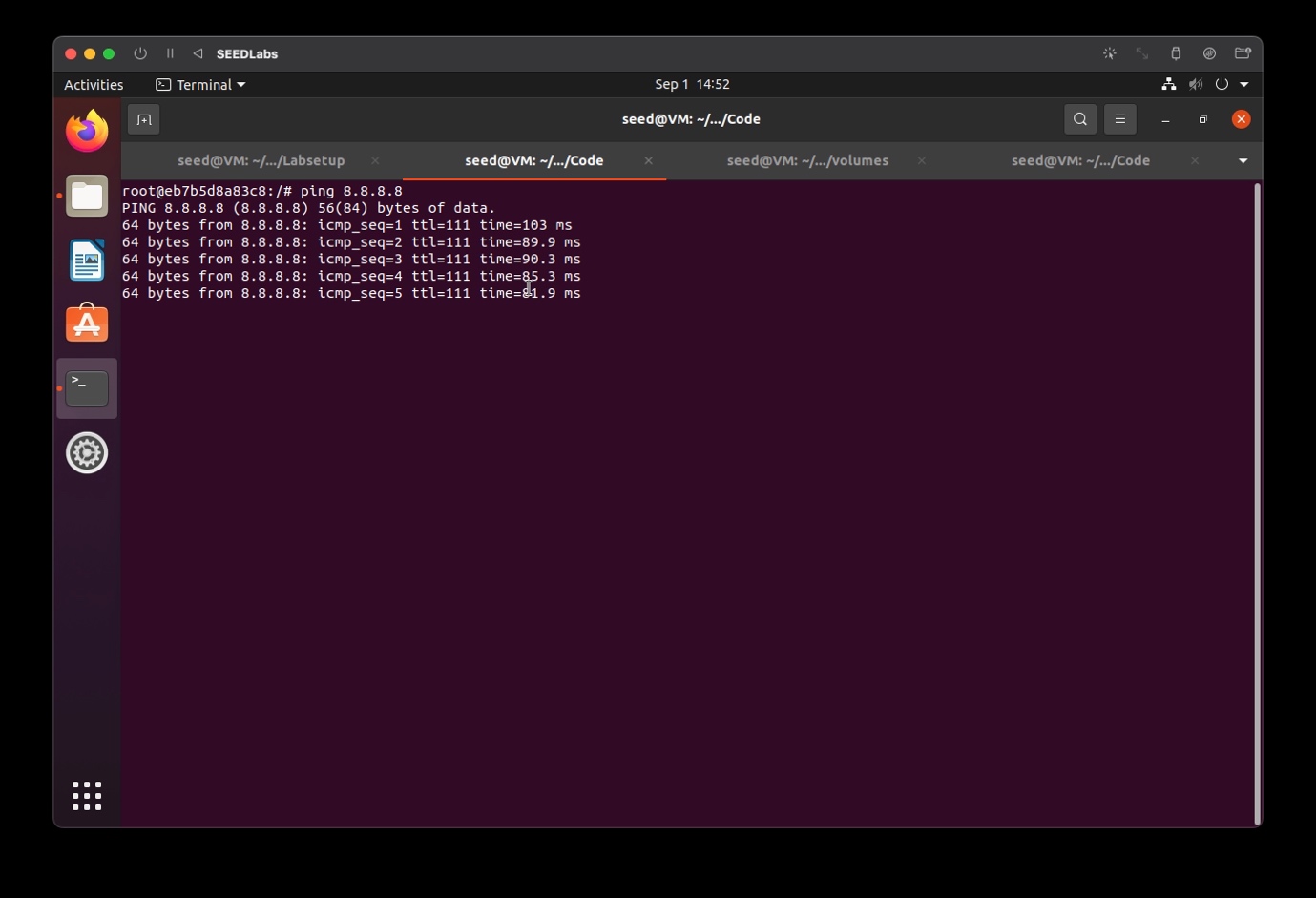
**On the Host A terminal run the command:**

**# ping 8.8.8.8**

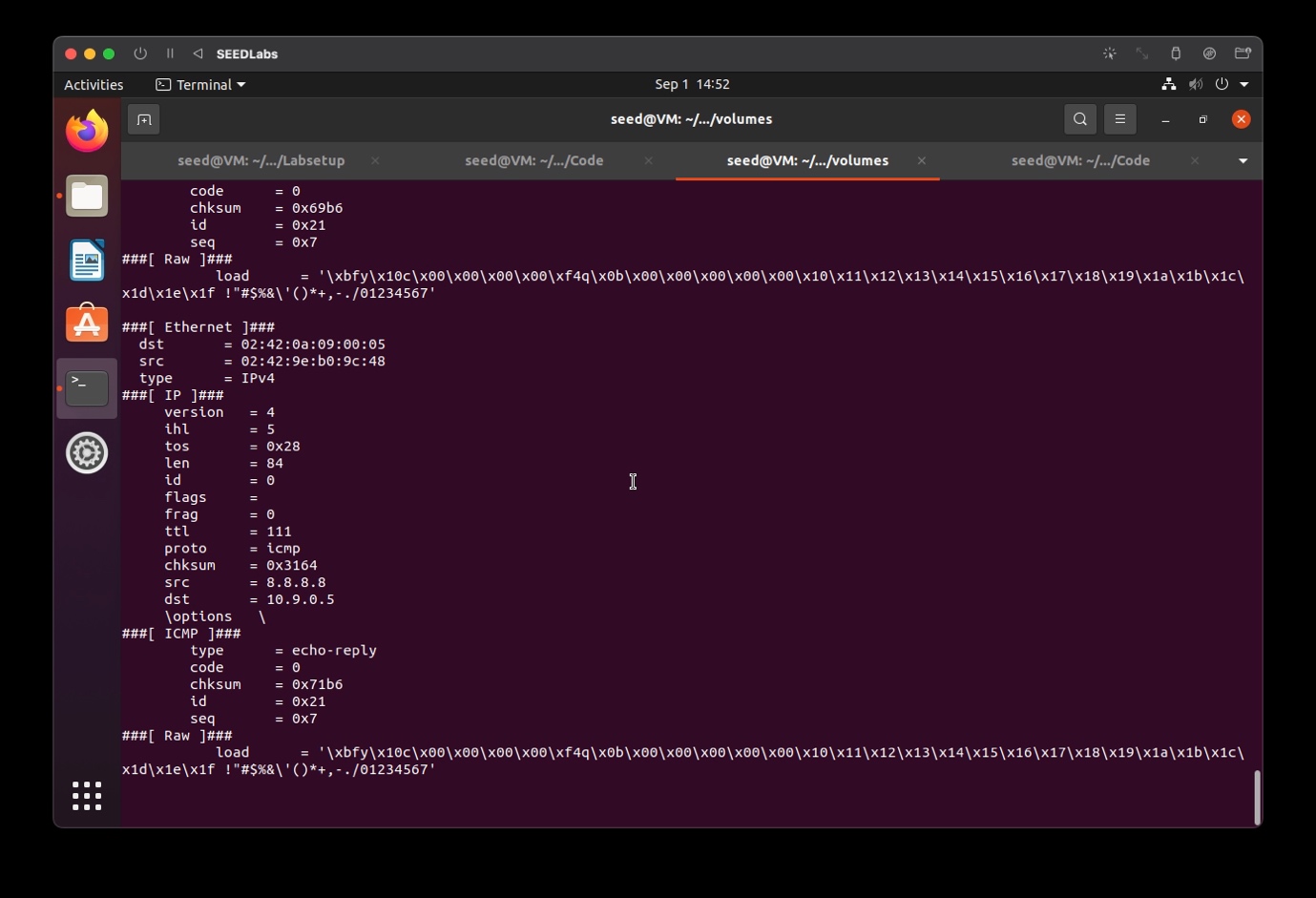
The ICMP packets are captured by the sniffer program.

**Provide a screenshot of your observations.**

**Host A terminal:**

****

**Attacker’s terminal:**

****

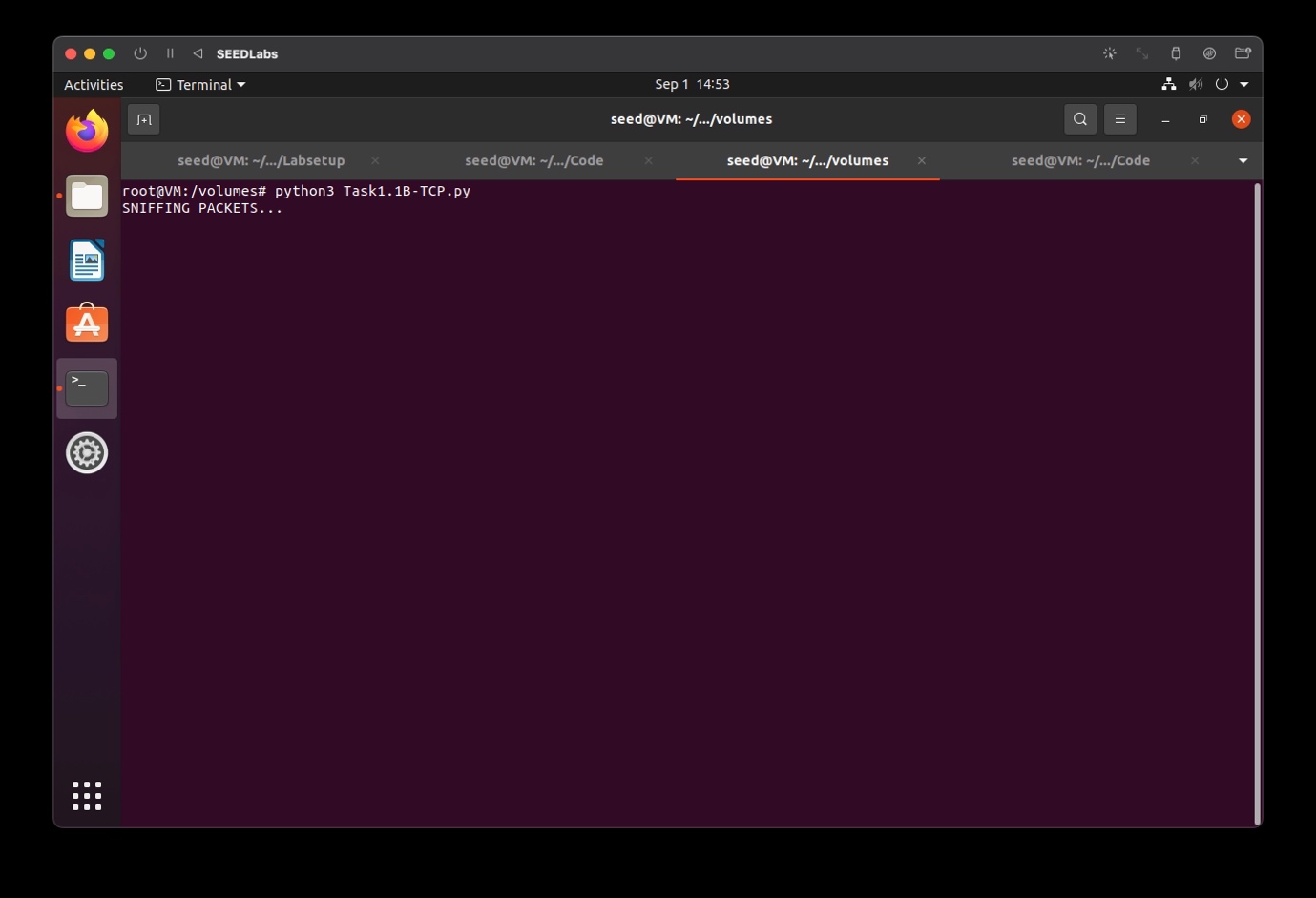
Sniffing attack is successful.

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

**On the Attacker terminal run the command:**

**# python3 Task1.1B-TCP.py**

**Provide a screenshot of your observations**

****

The attacker starts sniffing.

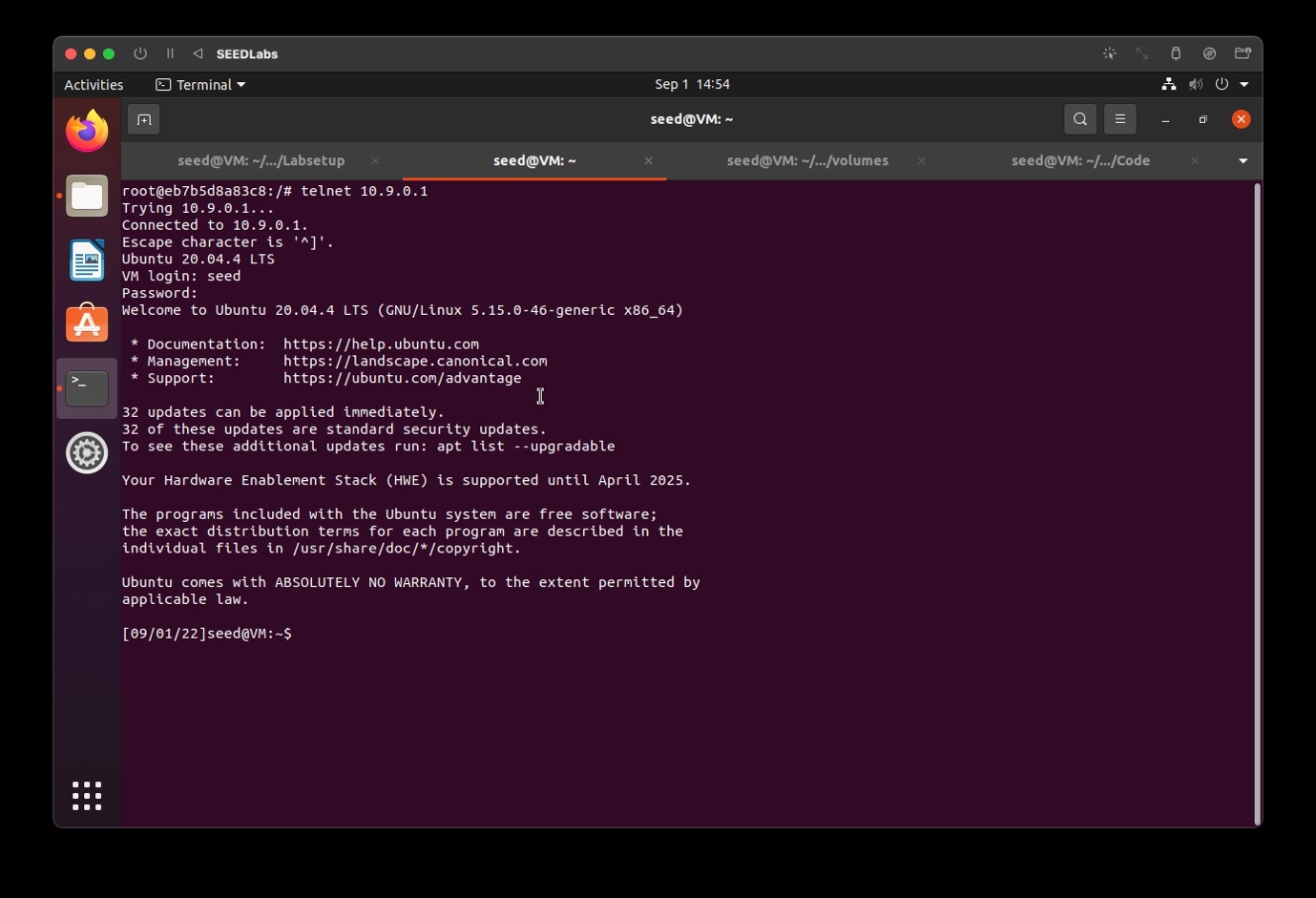
**From the host A machine’s terminal telnet to a random IP address.**

**On the Host A terminal run the command:**

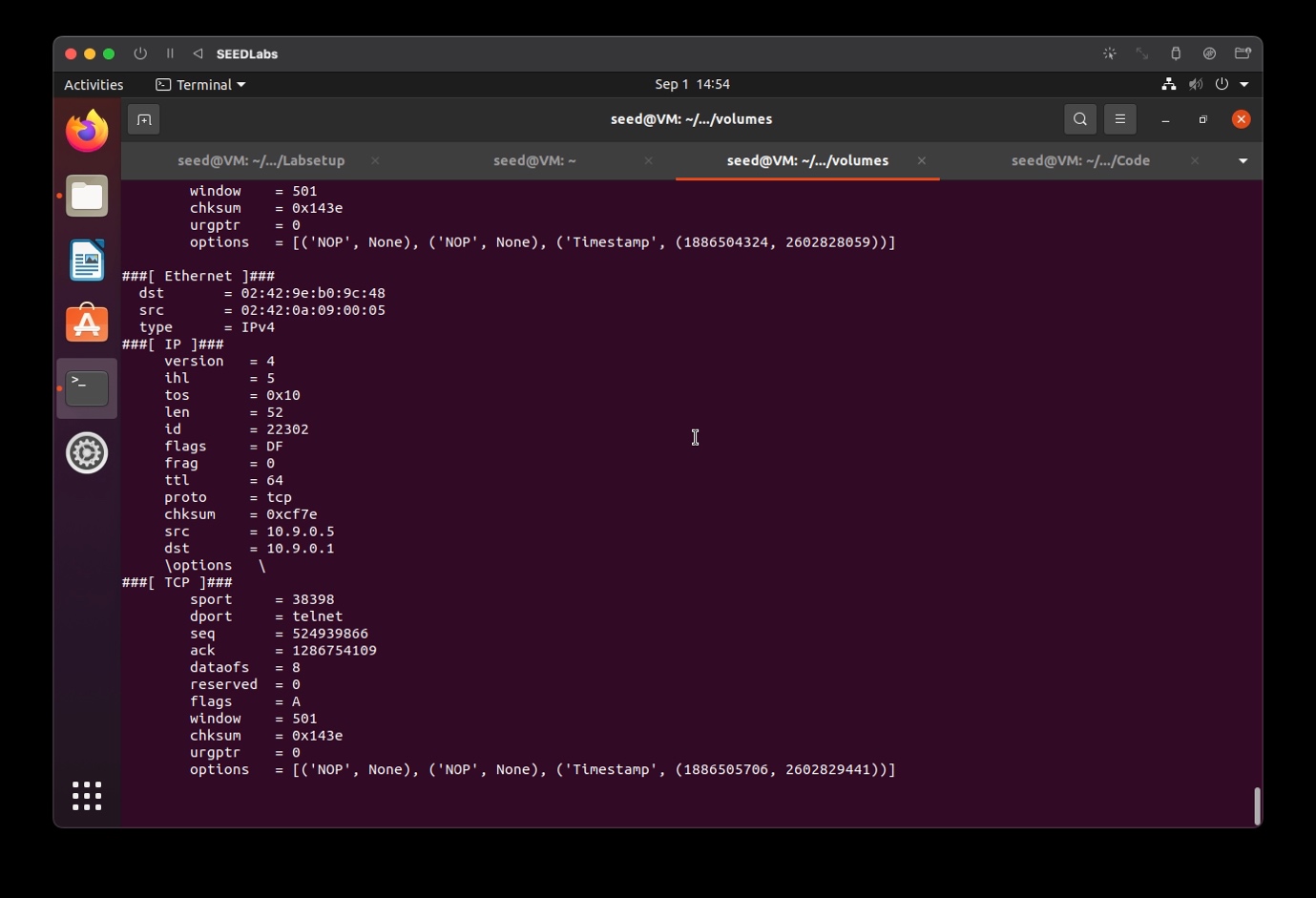
**# telnet 10.9.0.1**

**Provide screenshots of your observations.**

**Host A Terminal:**

****

**Attacker’s terminal:**

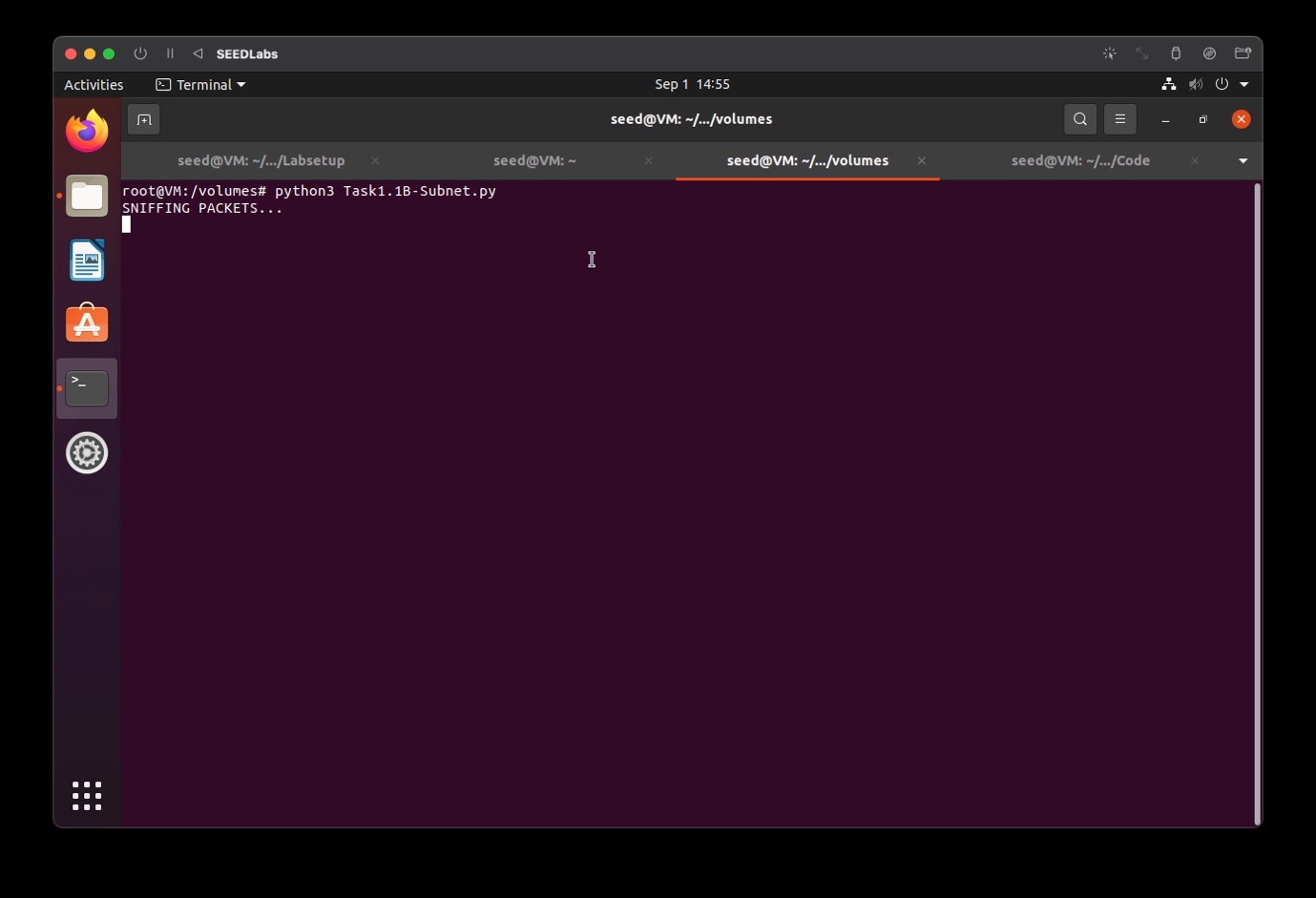
****

**Capture packets that come from or go to a particular subnet**

**On the Attacker terminal run the command:**

**# python3 Task1.1B-Subnet.py**

**Provide a screenshot of your observations**

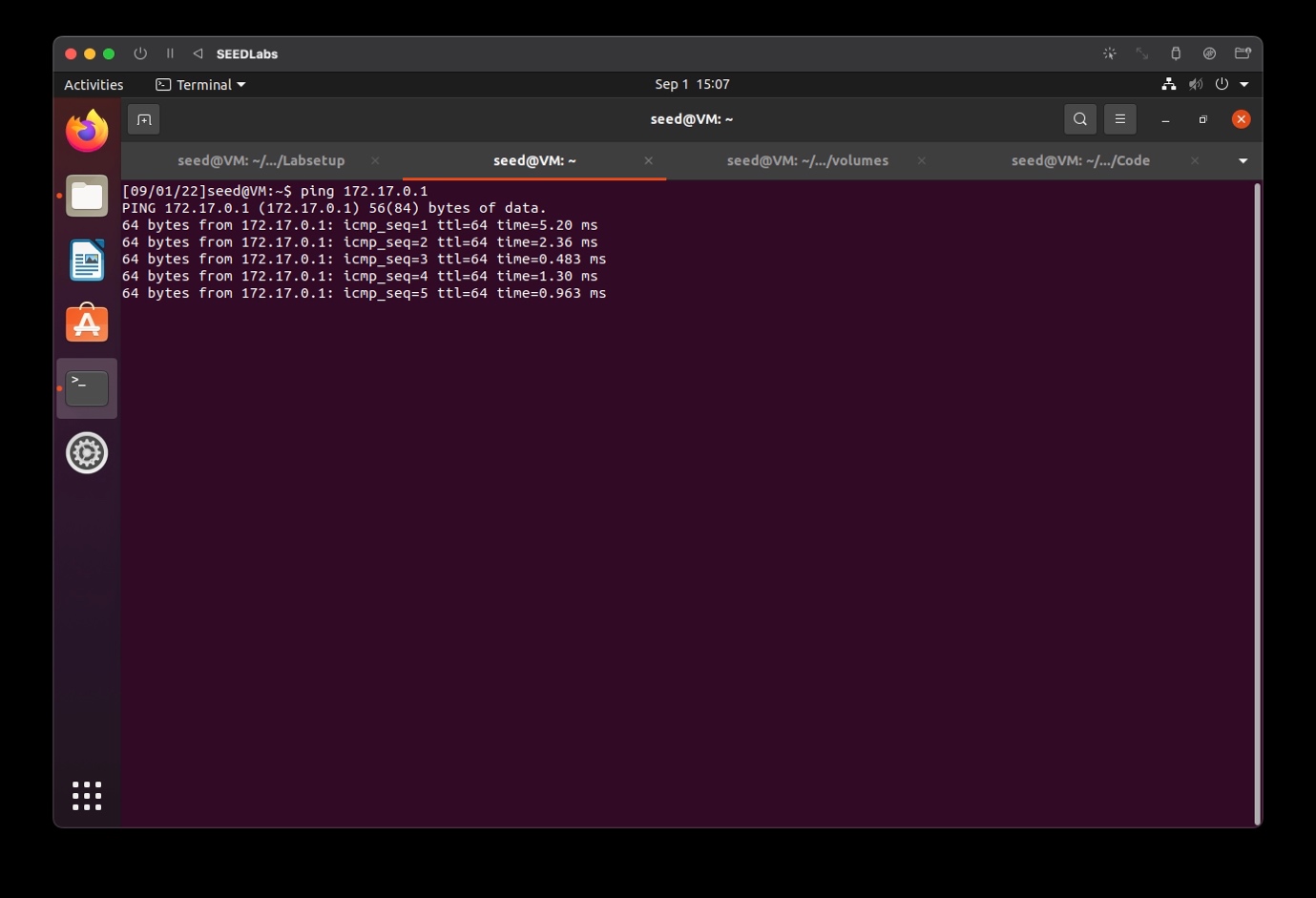
****

**On the Host A terminal run the command**

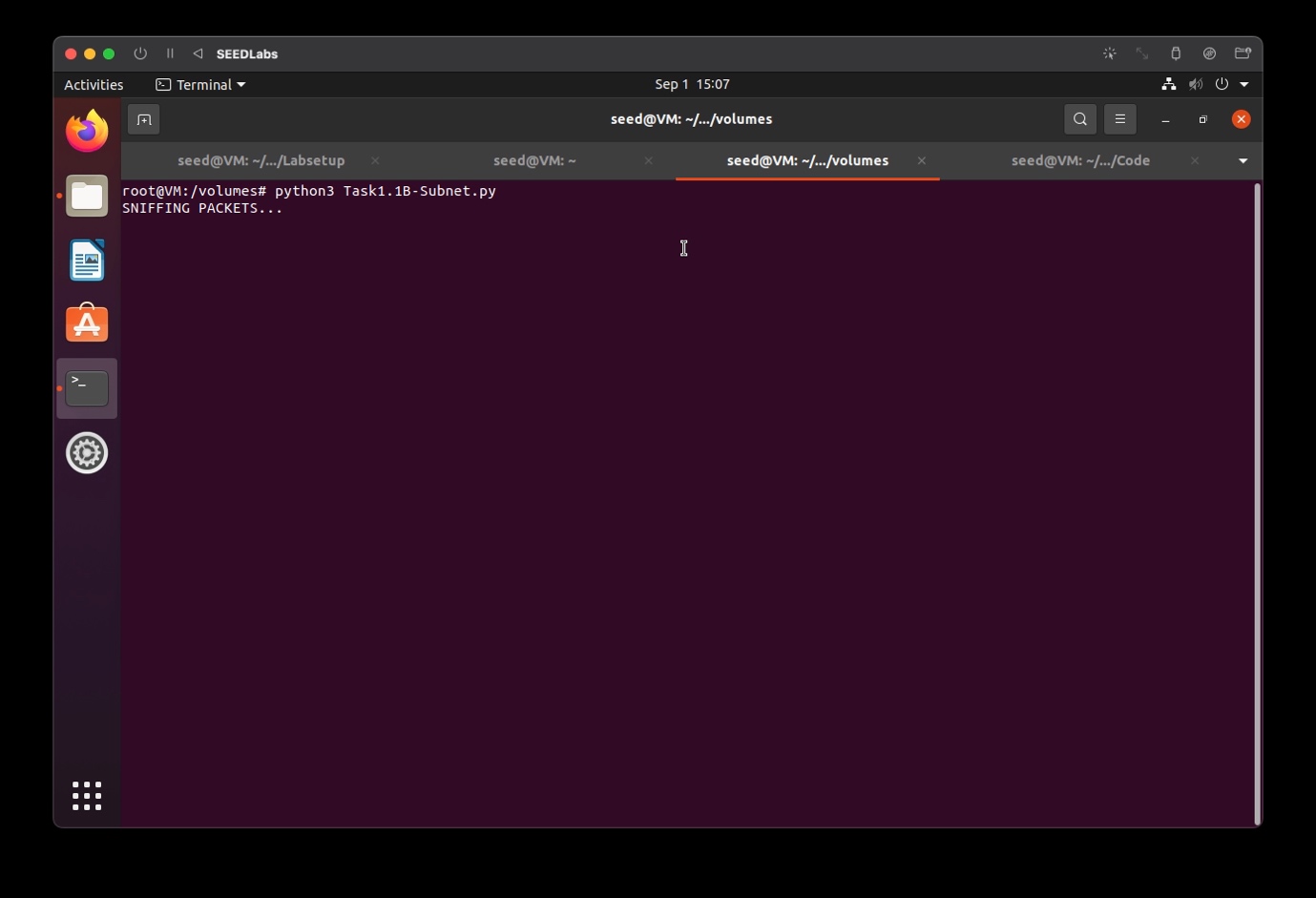
**# ping 172.17.0.1**

**Provide screenshots of your observations.**

**Host A terminal:**

****

**Attacker’s terminal:**

****

# **Task 1.2: Spoofing**

The objective of this task is to spoof IP packets with an arbitrary source IP address.

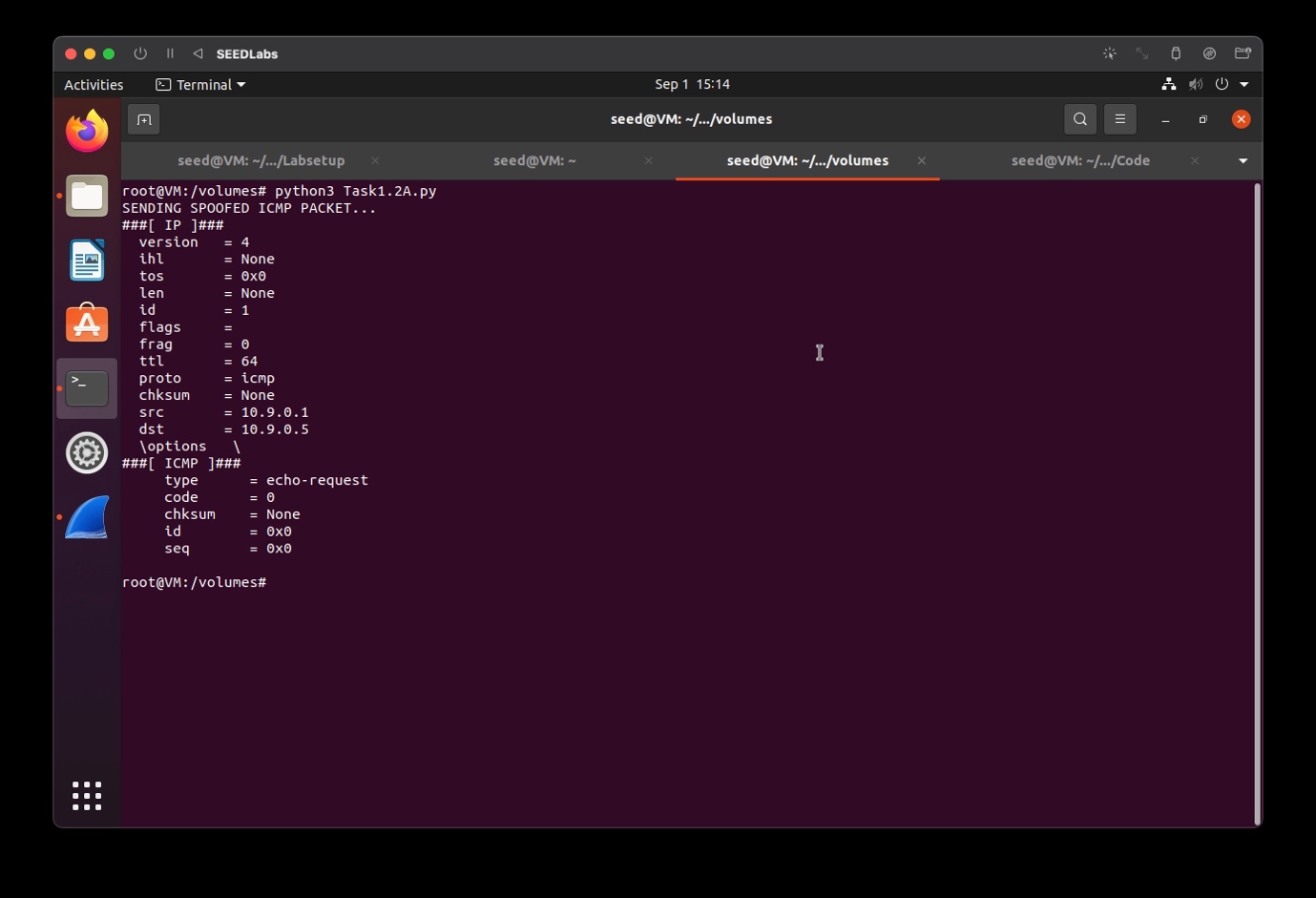
Wireshark is kept open before executing the program.

**On the Attacker terminal run the command:**

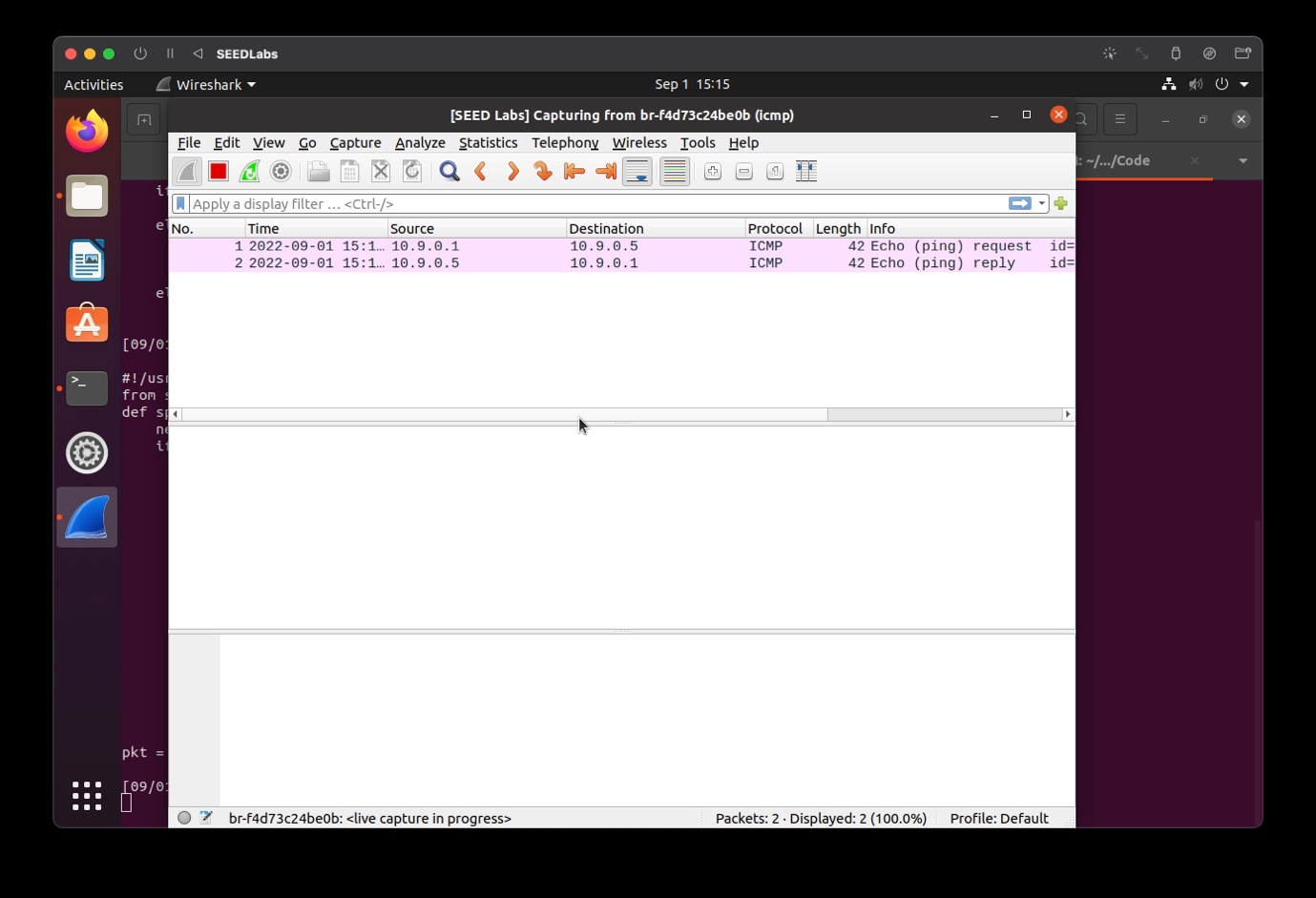
**# python3 Task1.2A.py**

**Provide a screenshot of your observations.**

**Attacker’s terminal:**

****

**Wireshark:**

****

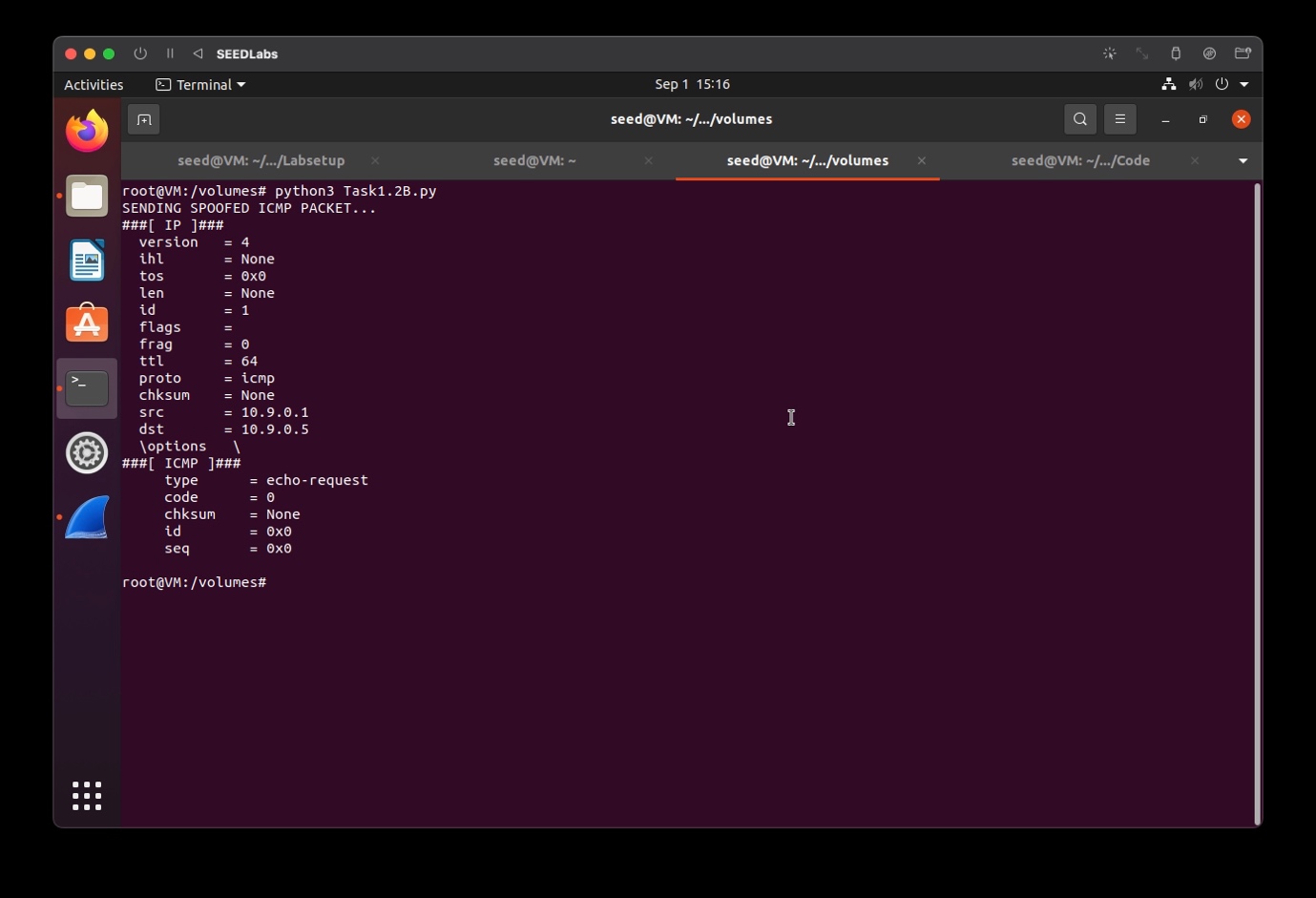
**With an arbitrary source IP address:**

**On the Attacker terminal run the command:**

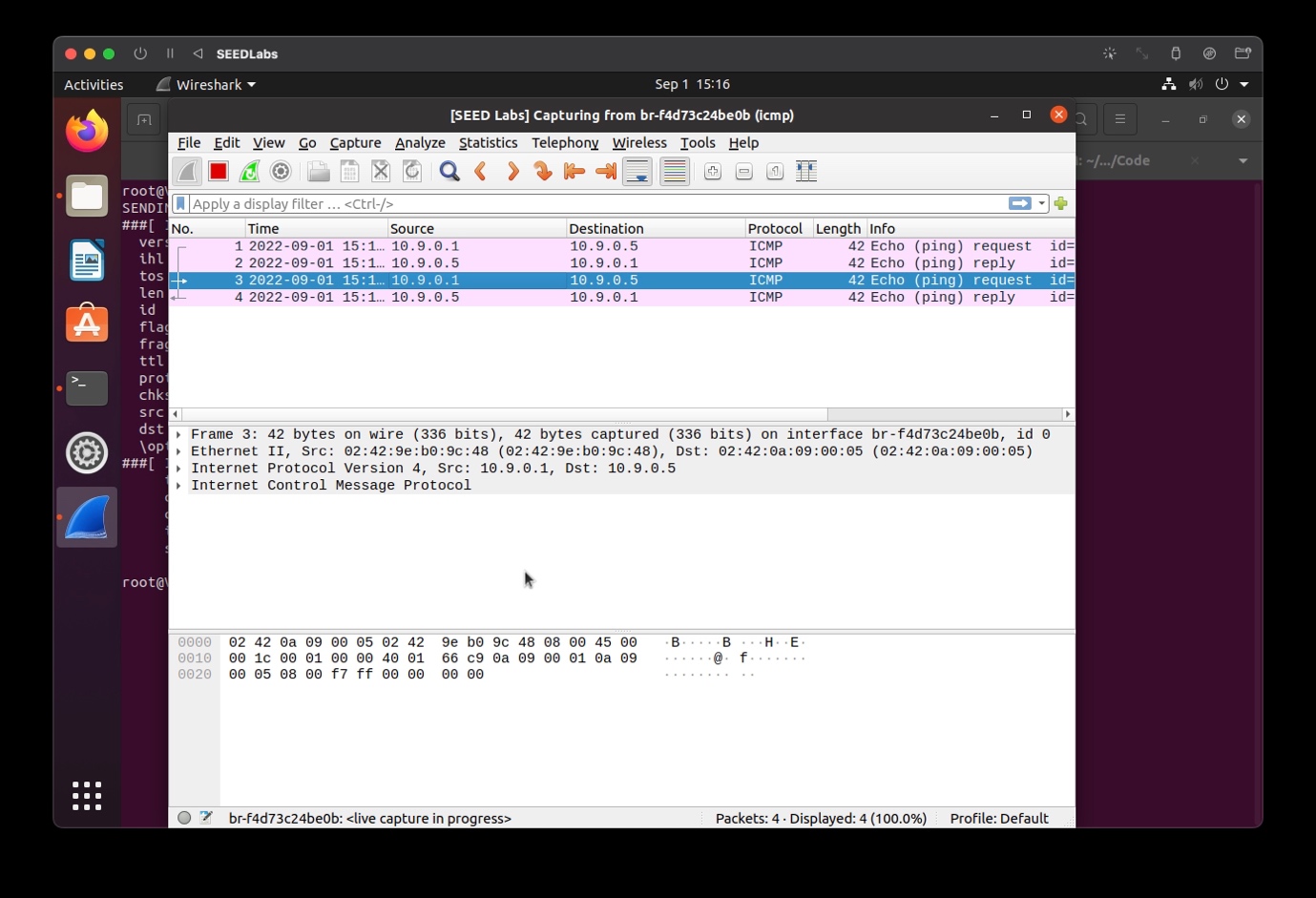
**# python3 Task1.2B.py**

**Provide a screenshot of your observations.**

**Attacker’s terminal:**

****

**Wireshark:**

****

When we use an arbitrary IP address, we notice in wireshark that the echo ping requests are not returned with a reply.

# **Task 1.3: Traceroute**

The objective of this task is to implement a simple traceroute tool using Scapy to estimate the distance, in terms of number of routers, between the VM and a selected destination.

Using the scapy library, we periodically increase the ttl value to check for different readings on Wireshark.

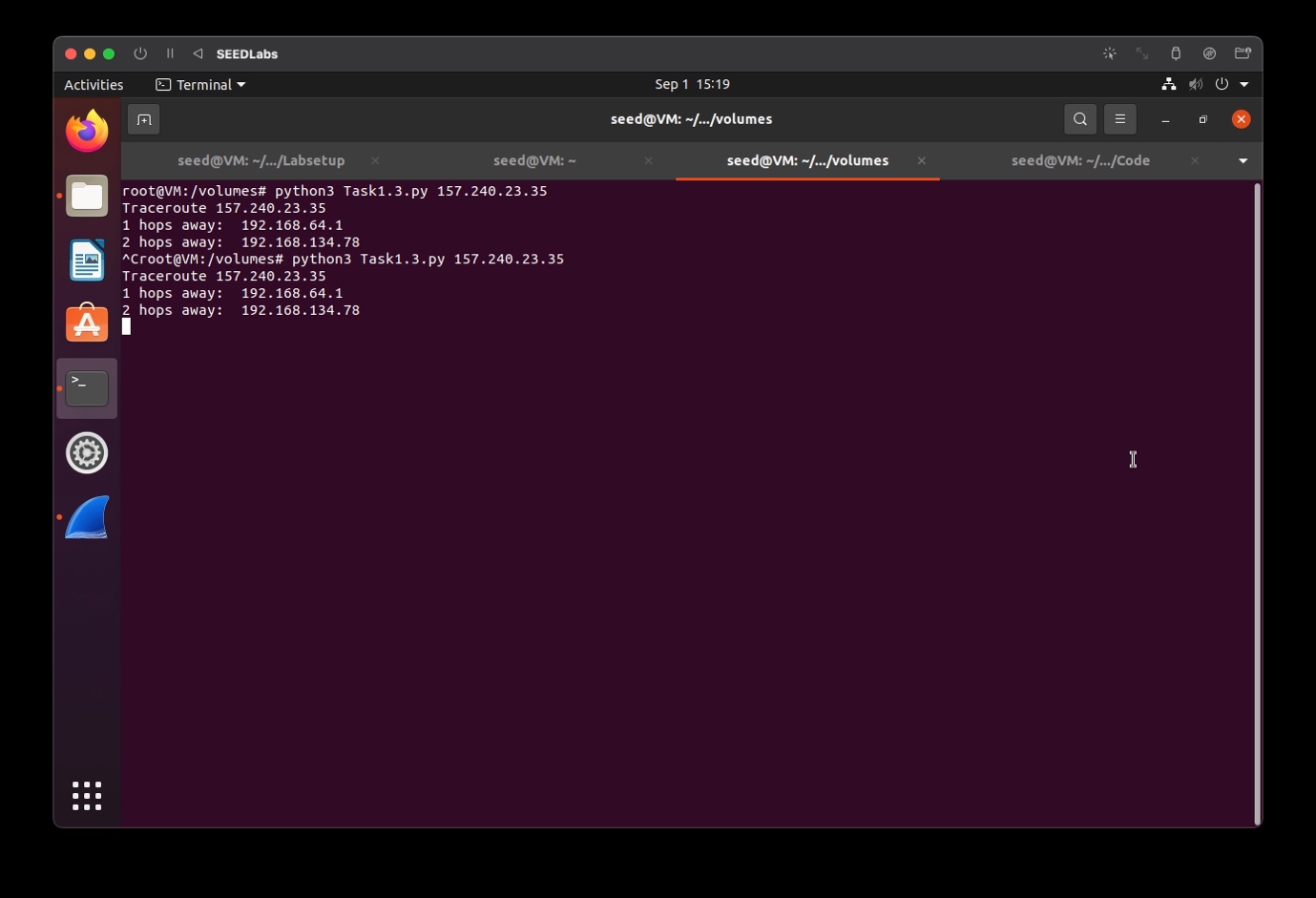
**On the Attacker terminal run the command:**

**# python3 Task1.3.py 157.240.23.35**

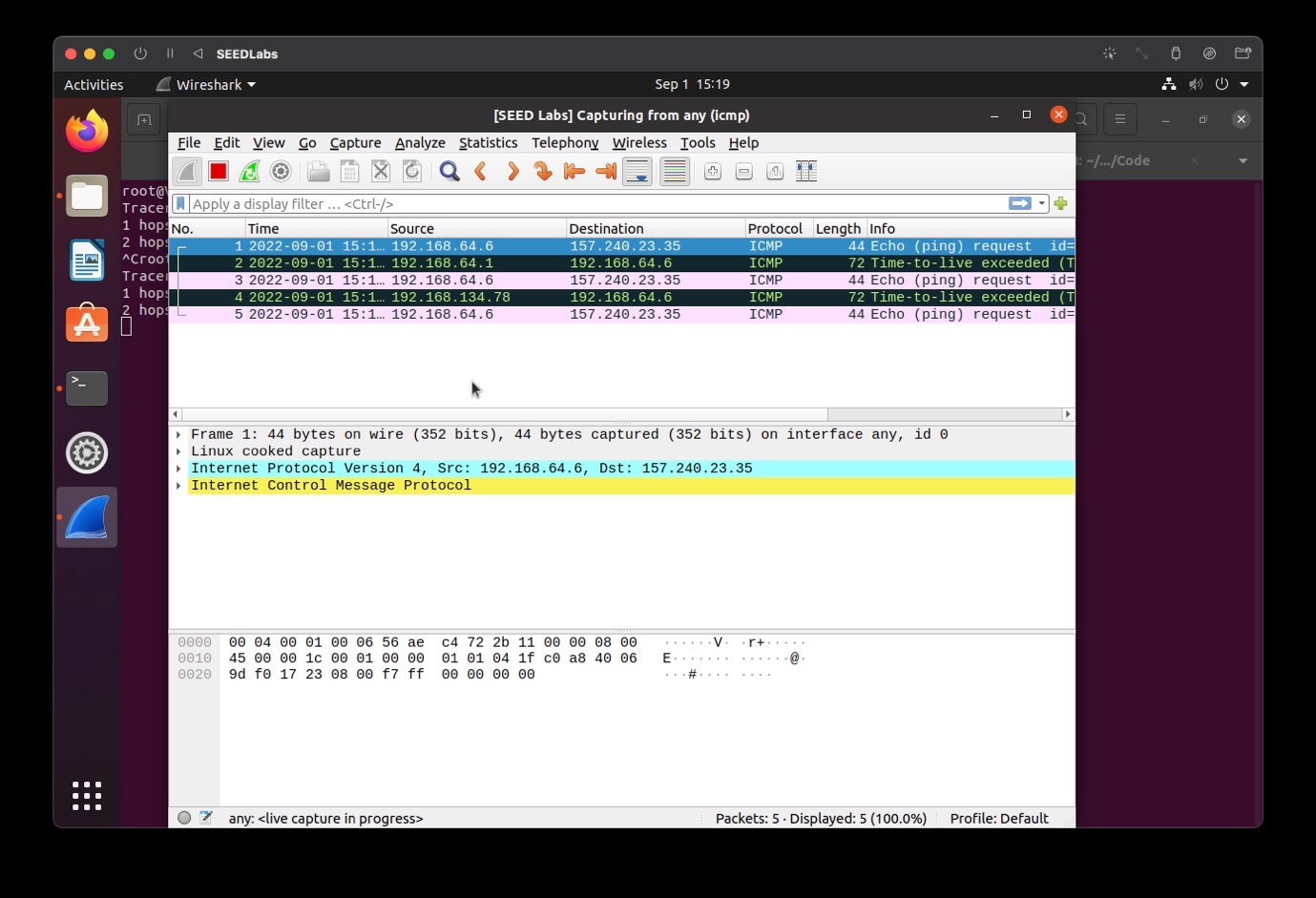
**157.240.23.35 is the IP address for facebook.com**

**For ttl=1,**

**Attacker’s terminal:**

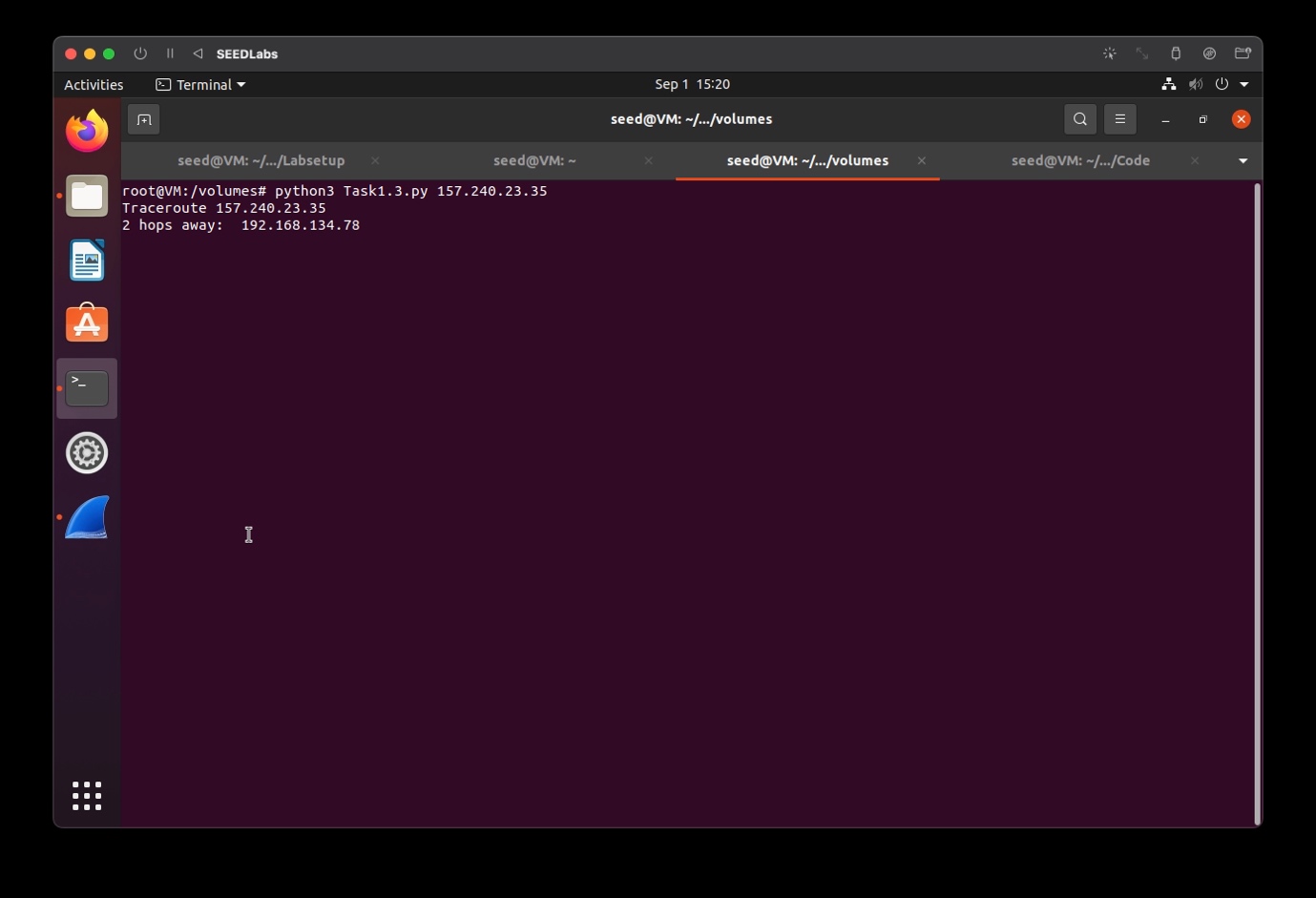
****

**Wireshark:**

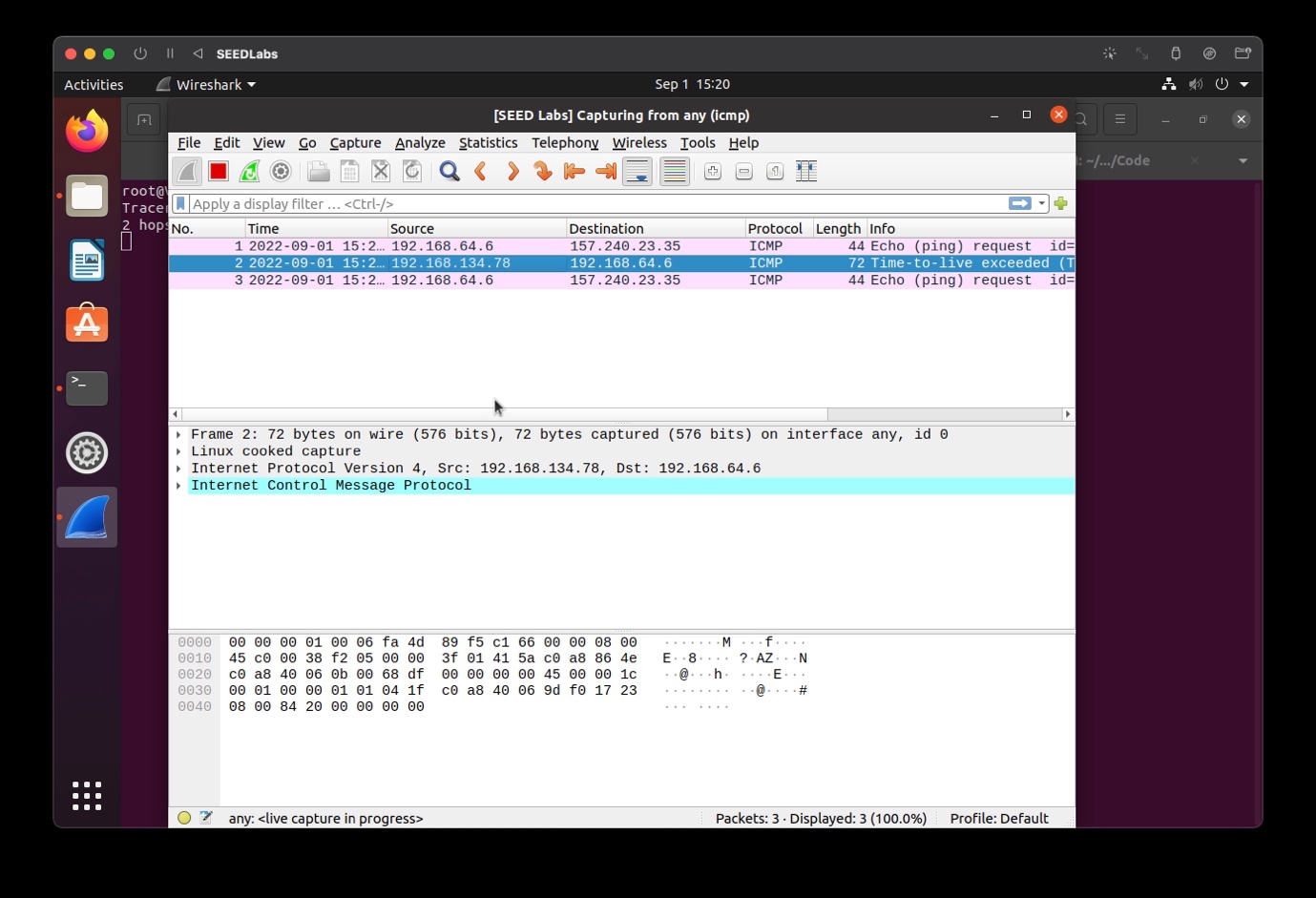


**For ttl=2,**

**Attacker’s terminal:**

****

**Wireshark:**



# **Task 1.4: Sniffing and-then Spoofing**

In this task, the victim machine pings a non-existing IP address “1.2.3.4”. As the attacker machine is on the same network, it sniffs the request packet, creates a new echo reply packet with IP and ICMP header and sends it to the victim machine. Hence, the user will always receive an echo reply from a non-existing IP address indicating that the machine is alive.

**On the Attacker terminal run the command:**

**# python3 Task1.4.py**

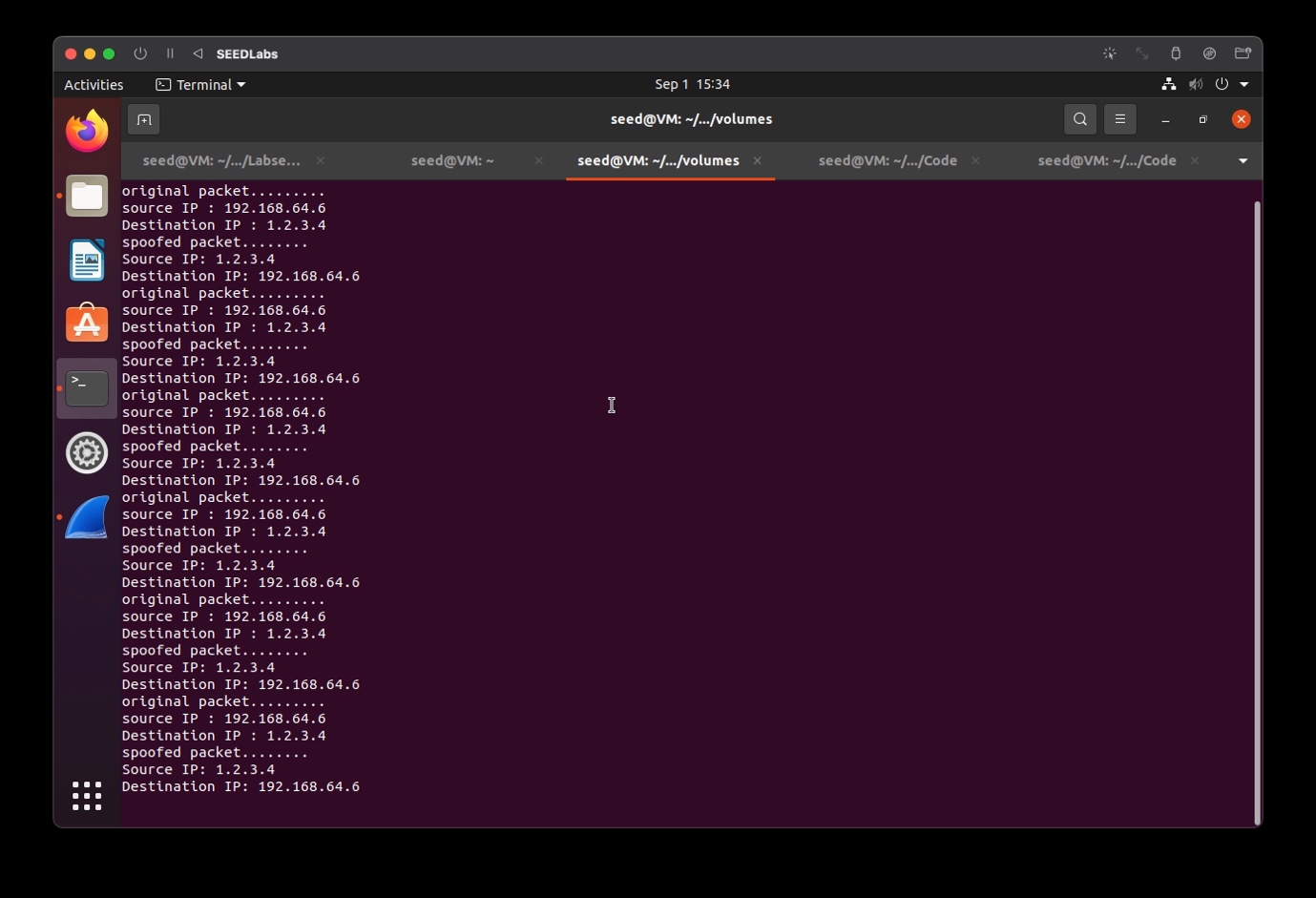
**From the host A machine’s terminal ping 1.2.3.4**

**On the Host A terminal run the command:**

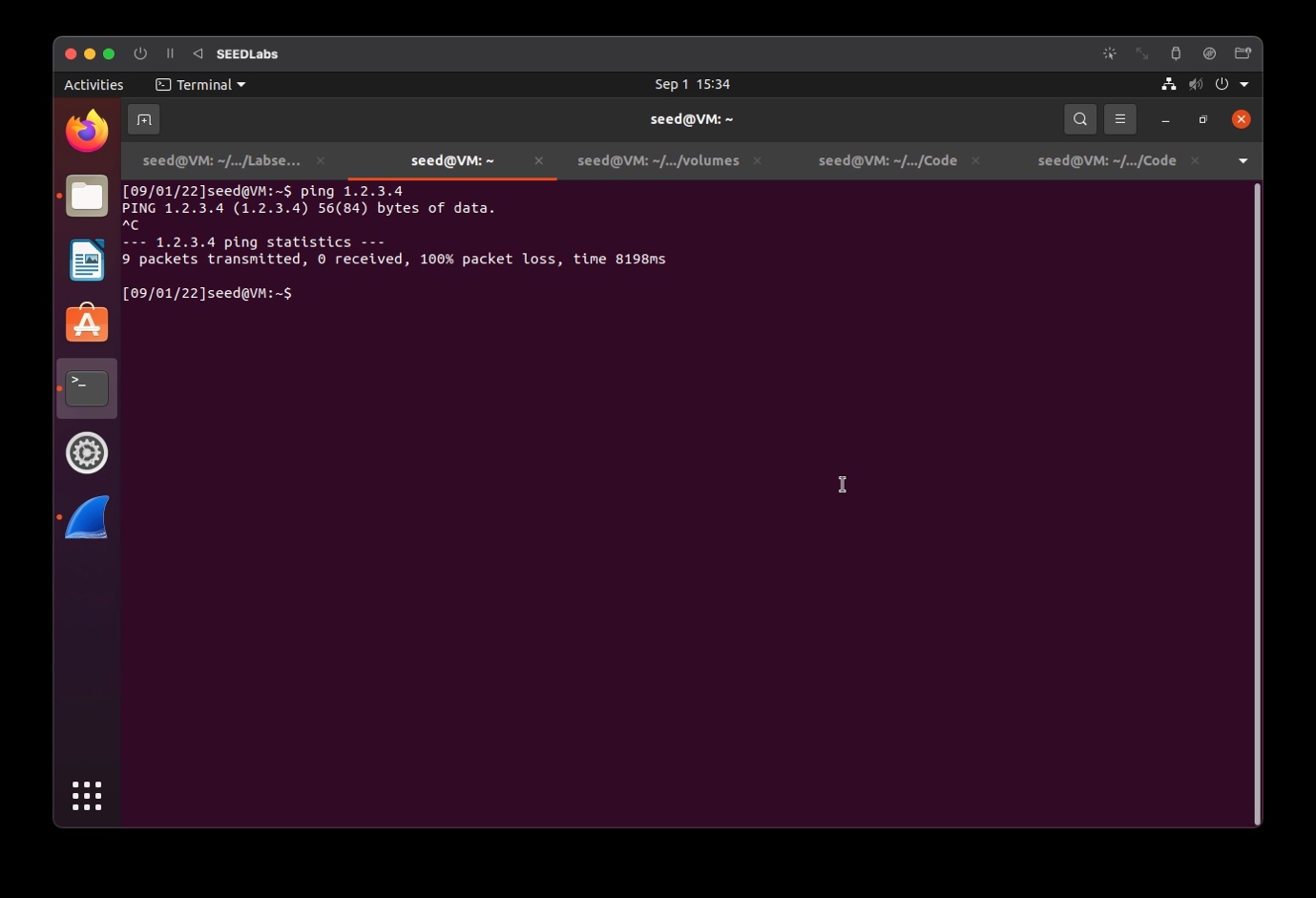
**# ping 1.2.3.4**

**Provide a screenshot of your observations**.

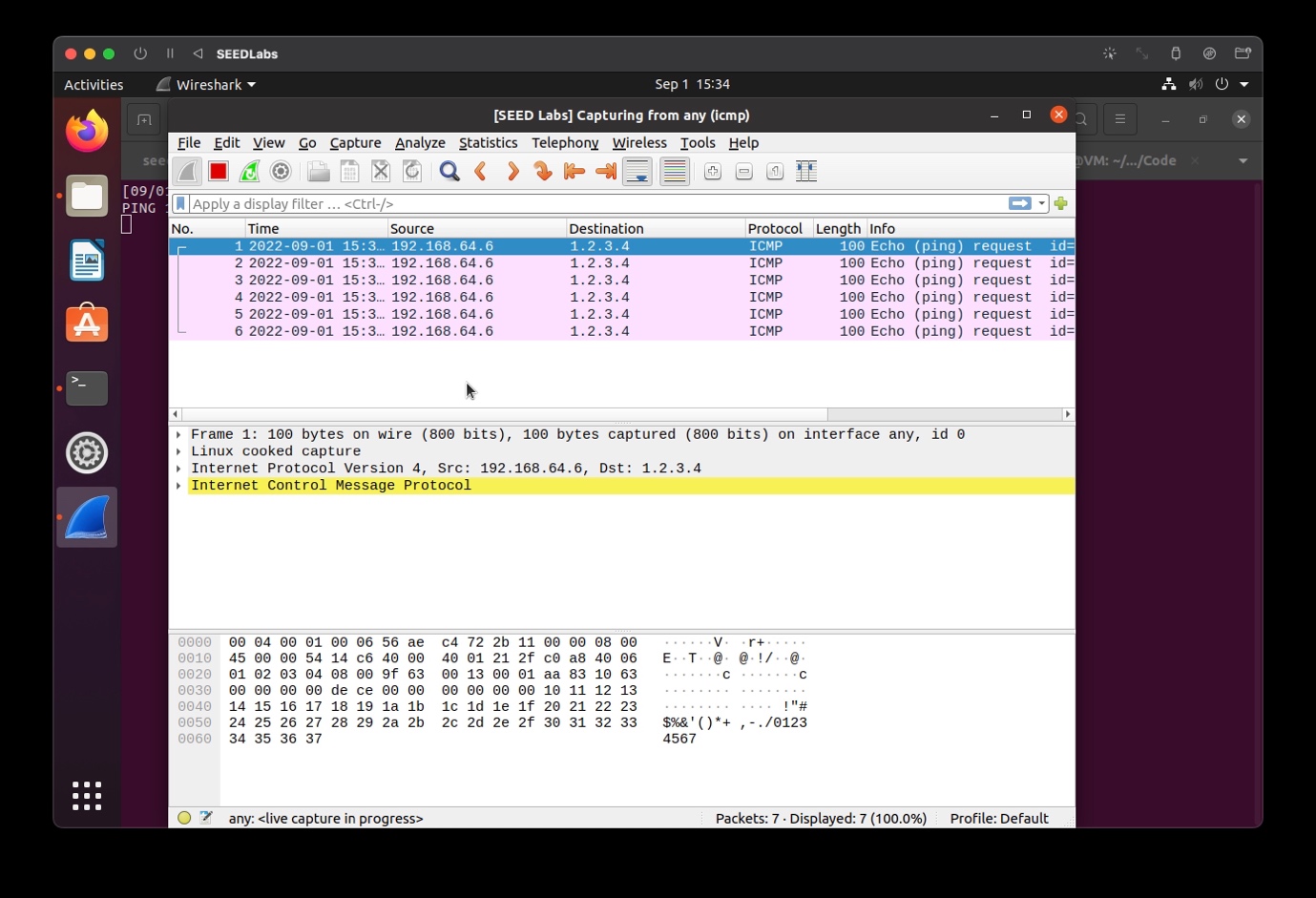
**Attacker’s terminal:**

****

**Host A terminal:**

****

**Wireshark:**

****