# Lab 1: Sniffing and Spoofing Lab

**Name: Maryam Khan**

**SRN: PES2UG21CS283**

**Date: 31/8/23**

## Table of Contents:

# Lab Environment Setup

Please download the **Labsetup.zip** file from the link given below :
https://seedsecuritylabs.org/Labs_20.04/Networking/Sniffing_Spoofing/

Follow the instructions in the **lab setup document** to set up the lab environment.

In this lab, we will use three machines that are connected to the same LAN. We can either use three VMs or three containers. Figure 1 depicts the lab environment setup using containers. We will do all the attacks on the attacker container, while using the other containers as the user machines.

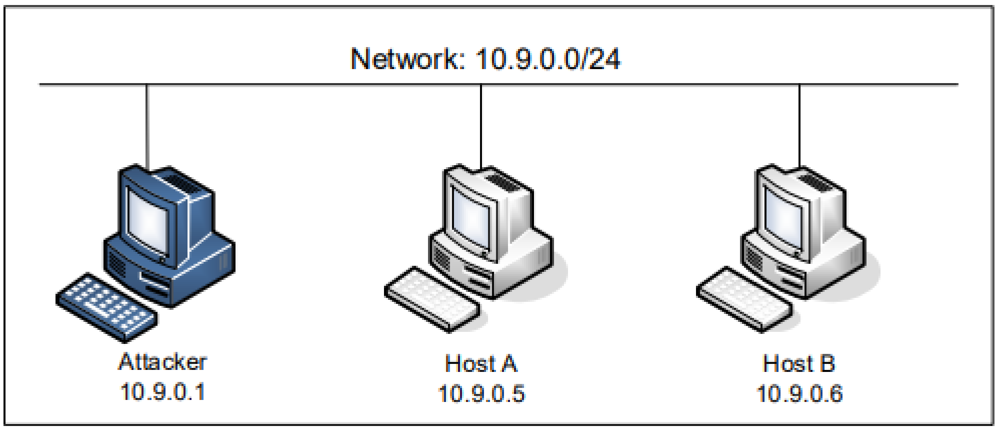
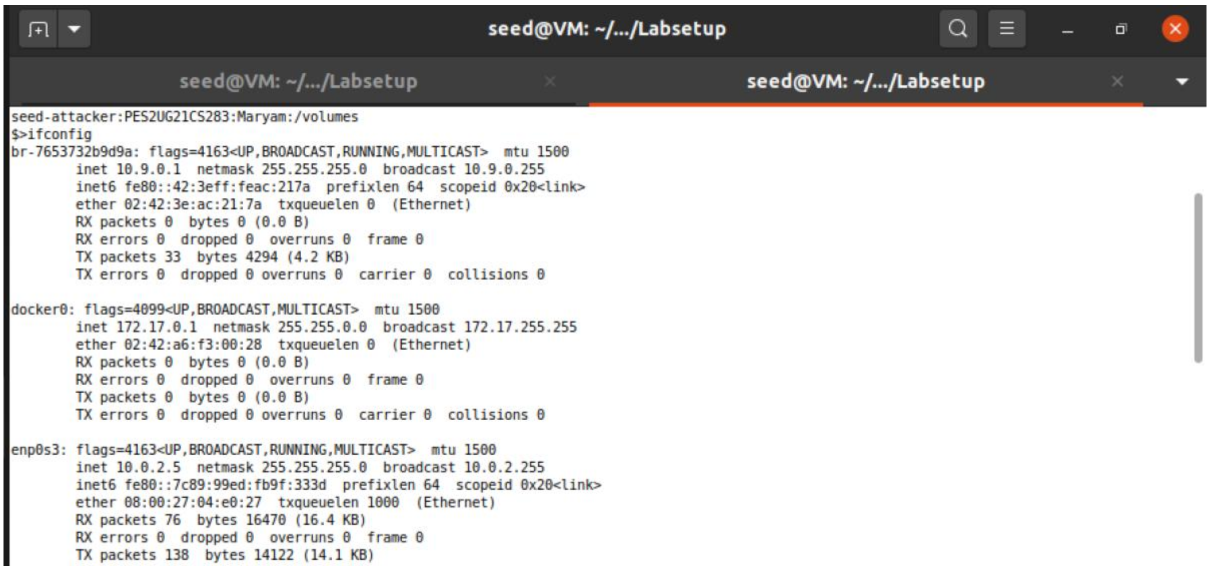
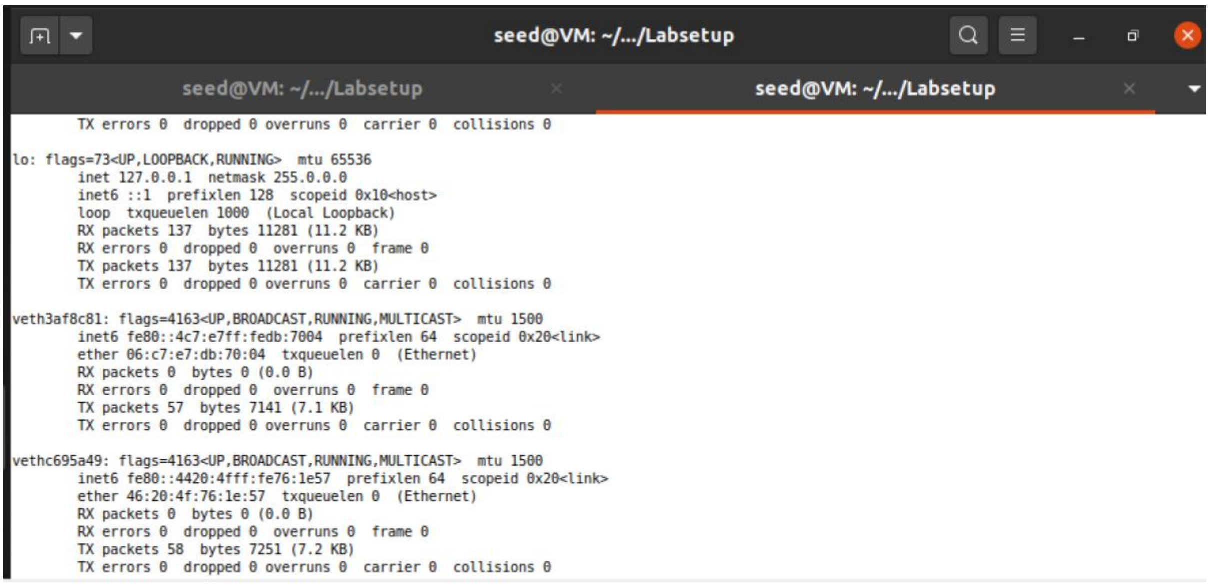


Figure 1 : Lab environment setup

Configuring seed-attacker

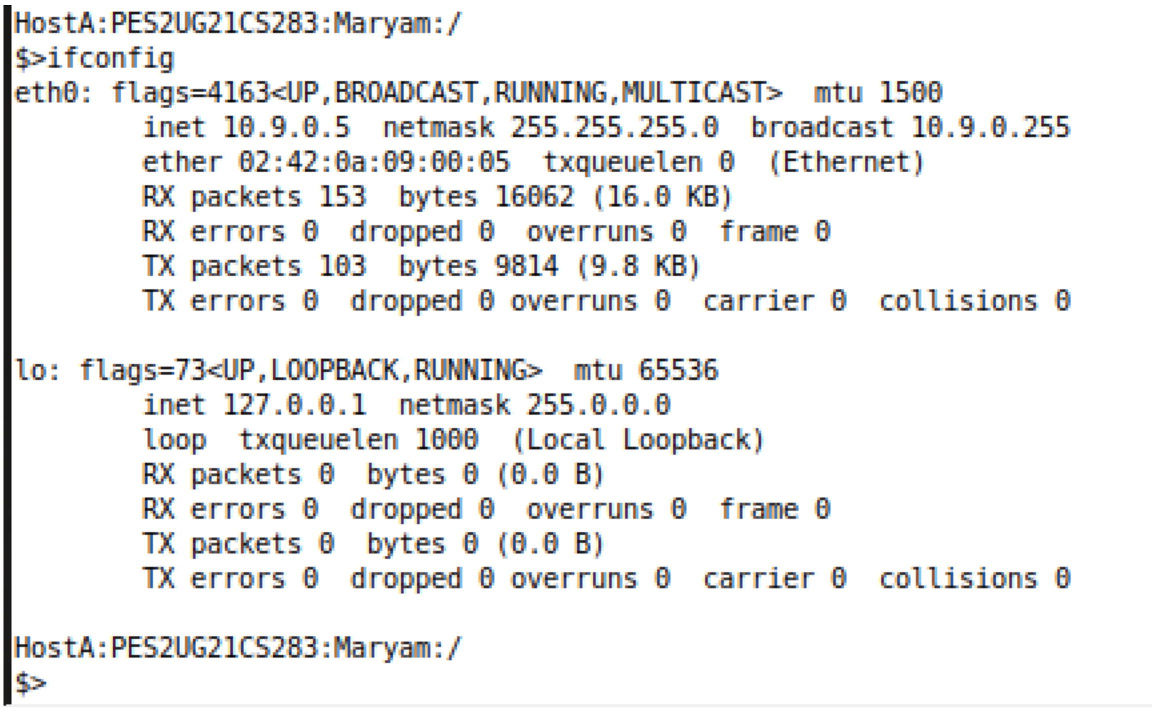


```
seed-attacker:PES2UG21CS283:Maryam:/volumes
$>ifconfig
br-7653732b9d9a: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:3eff:feac:217a  prefixlen 64  scopeid 0x20<link>
        ether 02:42:3e:ac:21:7a  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 33  bytes 4294 (4.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:a6:f3:00:28  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.5  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::7c89:99ed:fb9f:333d  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:04:e0:27  txqueuelen 1000  (Ethernet)
        RX packets 76  bytes 16470 (16.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 138  bytes 14122 (14.1 KB)
```



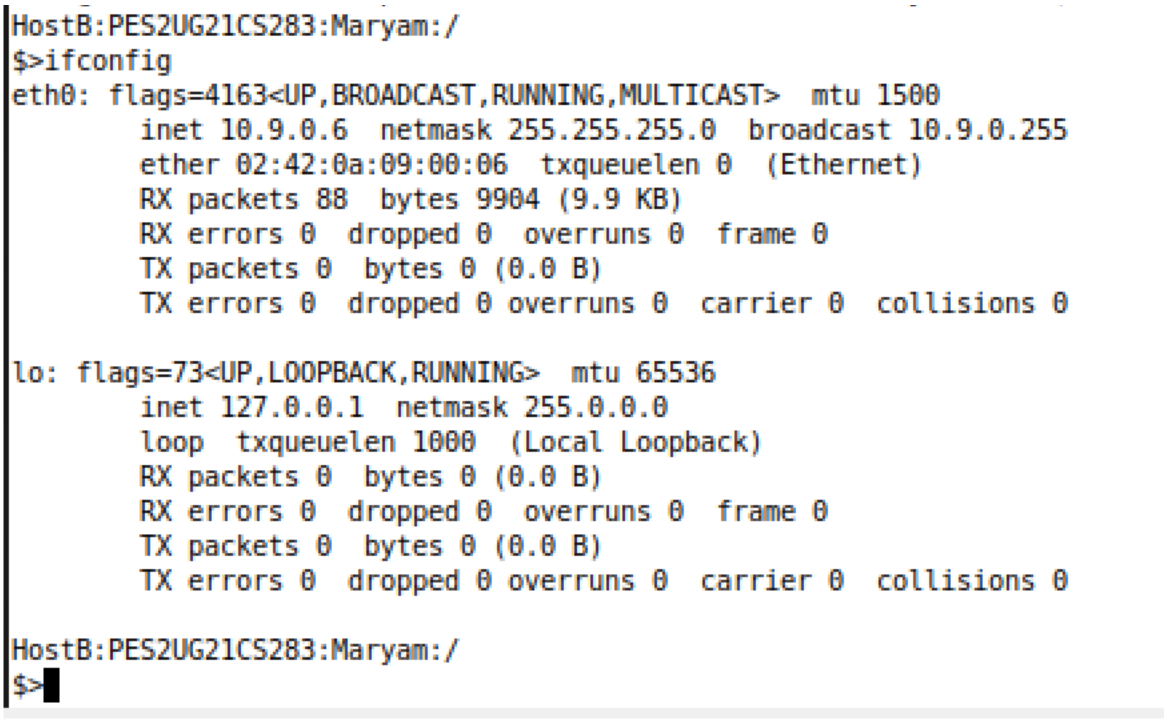
```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 137  bytes 11281 (11.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 137  bytes 11281 (11.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

veth3af8c81: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::4c7:e7ff:fedb:7004  prefixlen 64  scopeid 0x20<link>
        ether 06:c7:e7:db:70:04  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 57  bytes 7141 (7.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

vethc695a49: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::4420:4fff:fe76:1e57  prefixlen 64  scopeid 0x20<link>
        ether 46:20:4f:76:1e:57  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 58  bytes 7251 (7.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

## Configuring Host A 10.9.0.5



```
HostA:PES2UG21CS283:Maryam:/
$>ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:05  txqueuelen 0  (Ethernet)
        RX packets 153  bytes 16062 (16.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 103  bytes 9814 (9.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

HostA:PES2UG21CS283:Maryam:/
$>
```

Configuring Host B 10.9.0.6



```
HostB:PES2UG21CS283:Maryam:/
$>ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.6  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:06  txqueuelen 0  (Ethernet)
        RX packets 88  bytes 9904 (9.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

HostB:PES2UG21CS283:Maryam:/
$>
```

# 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.

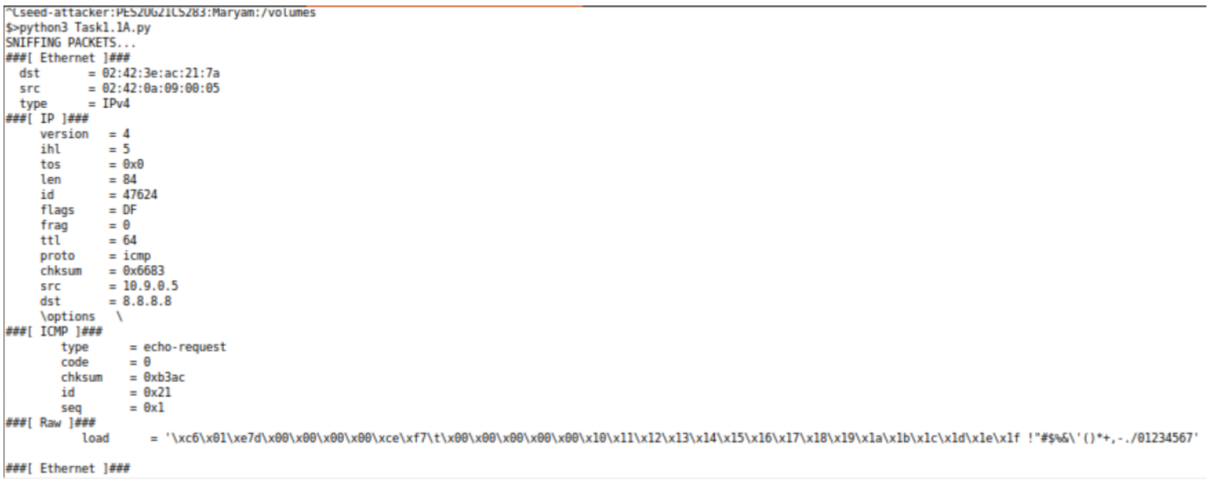
## Task 1.1 A : Sniff IP packets using Scapy

For each captured packet, the callback function print pkt() will be invoked; this function will print out some of the information about the packet. Run the program with the root privilege and demonstrate that you can indeed capture packets.

Check the **Lab setup instructions document** for detailed instructions on how to find out the interface of your attacker machine. **Replace the interface in the code wherever required**.

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

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

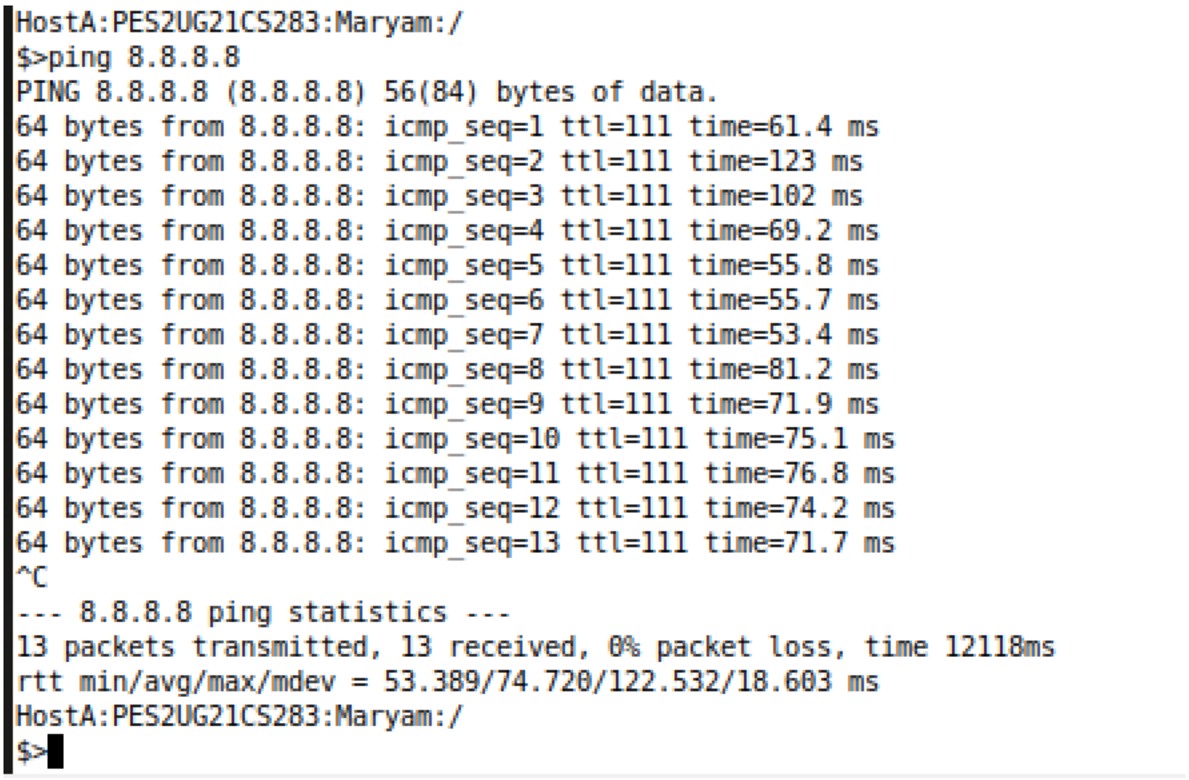


```
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.1A.py
SNIFFING PACKETS...
###[ Ethernet ]###
  dst       = 02:42:3e:ac:21:7a
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 47624
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x6683
     src       = 10.9.0.5
     dst       = 8.8.8.8
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0xb3ac
        id        = 0x21
        seq       = 0x1
###[ Raw ]###
           load      = '\xc6\x01\xe7d\x00\x00\x00\x00\xce\xf7\t\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01234567'

###[ Ethernet ]###
```

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**



```
HostA:PES2UG21CS283:Maryam:/
$>ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=111 time=61.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=111 time=123 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=111 time=102 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=111 time=69.2 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=111 time=55.8 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=111 time=55.7 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=111 time=53.4 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=111 time=81.2 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=111 time=71.9 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=111 time=75.1 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=111 time=76.8 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=111 time=74.2 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=111 time=71.7 ms
^C
--- 8.8.8.8 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12118ms
rtt min/avg/max/mdev = 53.389/74.720/122.532/18.603 ms
HostA:PES2UG21CS283:Maryam:/
$>
```

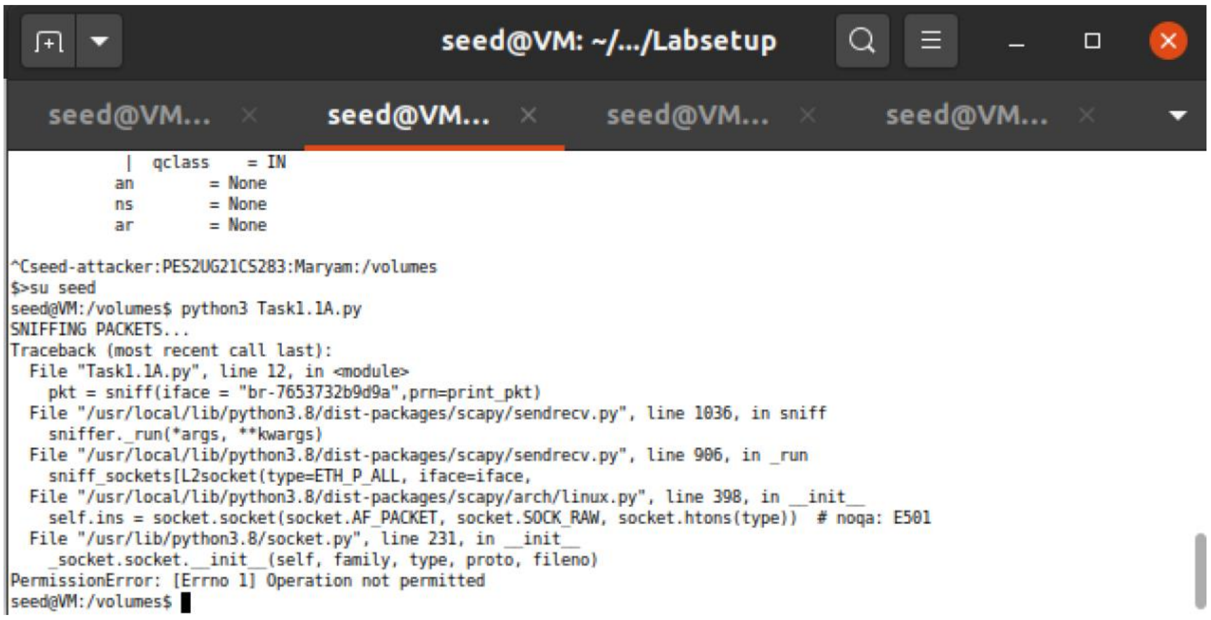
Now, we run the same program without root privileges. Do you find any issues? If so, why?

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

**# su seed**

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

Provide a screenshot of your observation



```
           |  qclass    = IN
           an        = None
           ns        = None
           ar        = None

^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>su seed
seed@VM:/volumes$ python3 Task1.1A.py
SNIFFING PACKETS...
Traceback (most recent call last):
  File "Task1.1A.py", line 12, in <module>
    pkt = sniff(iface = "br-7653732b9d9a",prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    sniffer._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in _run
    sniff_sockets[L2socket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type))  # noqa: E501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    _socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] Operation not permitted
seed@VM:/volumes$ 
```

Yes there are issues, since we do not have root privileges so we get a Permission error: Operation not permitted

**After running the sniff program run the command to return to root user on the attacker container:**

**# su root**

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

Usually, when we sniff packets, we are only interested in certain types of packets. We can do that by setting filters in sniffing. Scapy's filter uses the BPF (Berkeley Packet Filter) syntax; you can find the BPF manual from the Internet. Please set the following filters and demonstrate your sniffer program again (each filter should be set separately):

- Capture only ICMP packets.
- Capture any TCP packet that comes from a particular IP and with a destination port number 23.

- 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.

## Capture only the ICMP packet

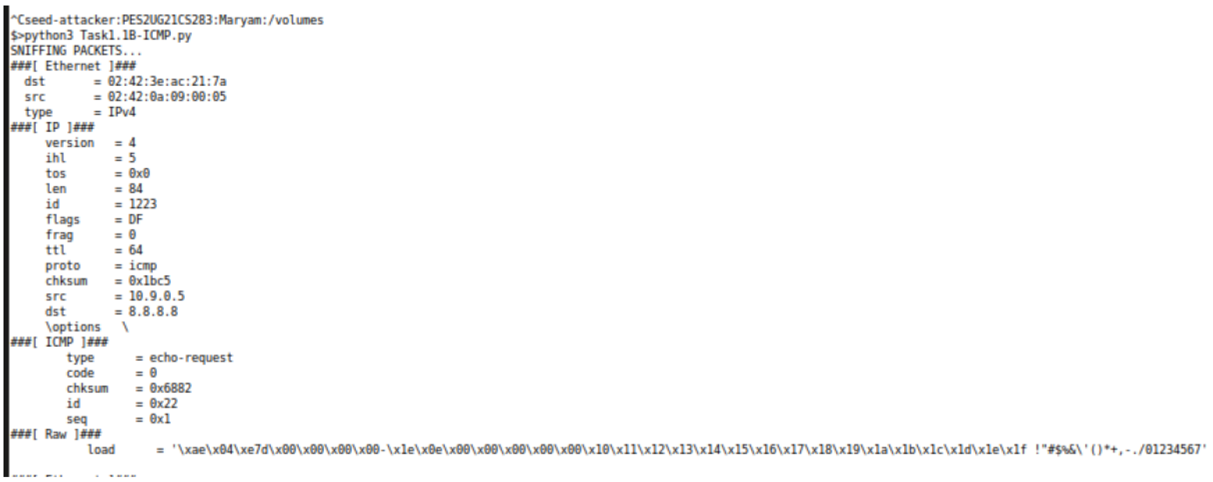
The ICMP packets are captured by the Scapy sniffer program. Hence, when some machine on the same network sends ping requests, the packets get captured by the sniffer.

**Fill in the interface of the attacker machine in the given program and run the code.**

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

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

Provide a screenshot of your observations



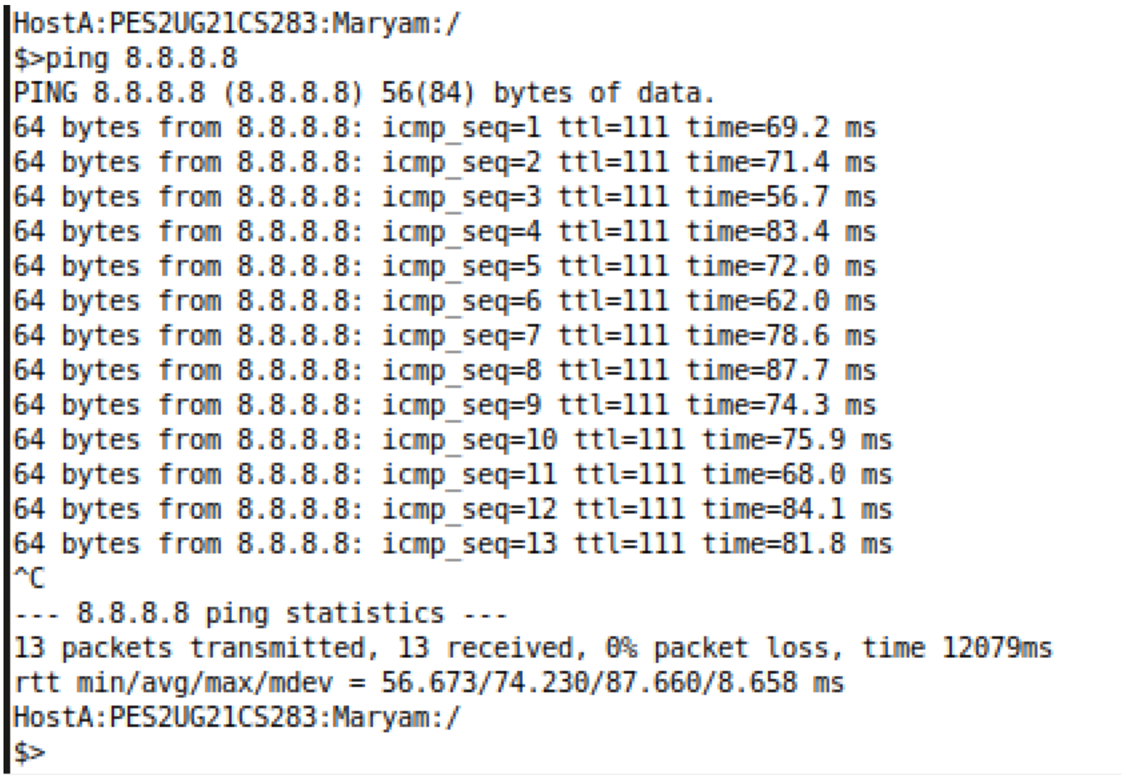
```
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.1B-ICMP.py
SNIFFING PACKETS...
###[ Ethernet ]###
  dst       = 02:42:3e:ac:21:7a
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 1223
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x1bc5
     src       = 10.9.0.5
     dst       = 8.8.8.8
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x6882
        id        = 0x22
        seq       = 0x1
###[ Raw ]###
           load      = '\xae\x04\xe7d\x00\x00\x00\x00-\x1e\x0e\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01234567'
```

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
```

Provide a screenshot of your observations.



```
HostA:PES2UG21CS283:Maryam:/
$>ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=111 time=69.2 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=111 time=71.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=111 time=56.7 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=111 time=83.4 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=111 time=72.0 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=111 time=62.0 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=111 time=78.6 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=111 time=87.7 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=111 time=74.3 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=111 time=75.9 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=111 time=68.0 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=111 time=84.1 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=111 time=81.8 ms
^C
--- 8.8.8.8 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12079ms
rtt min/avg/max/mdev = 56.673/74.230/87.660/8.658 ms
HostA:PES2UG21CS283:Maryam:/
$>
```

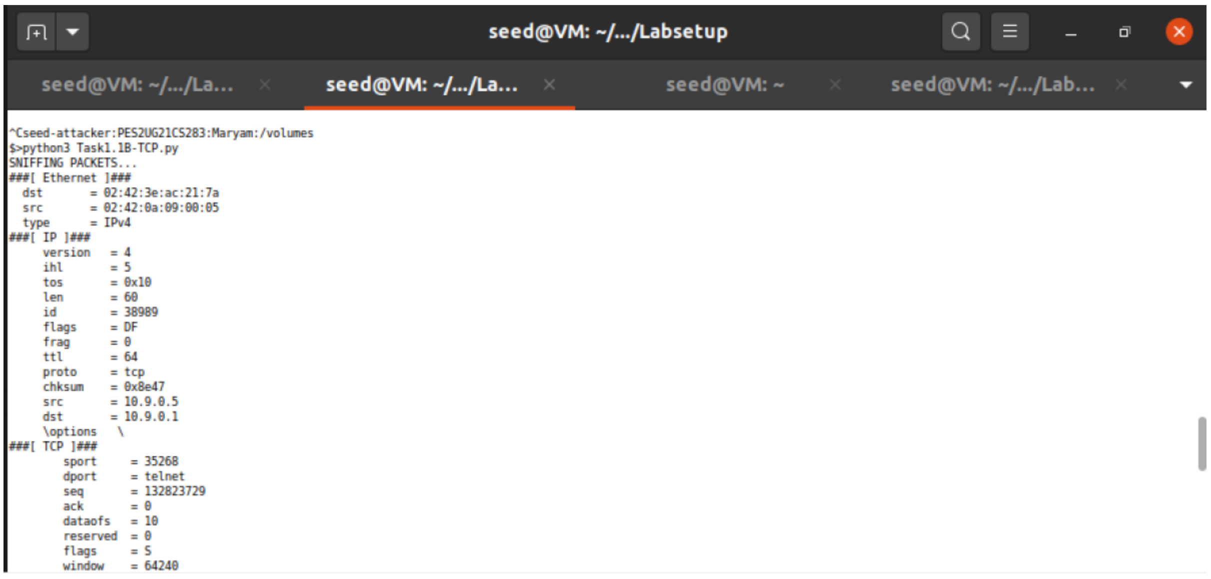
## Capture any TCP packet that comes from a particular IP and with a destination port number 23

The program must capture the TCP packets being sent from the specified IP address on the port 23.
**Fill in the interface of the attacker machine in the given program and run the code.**

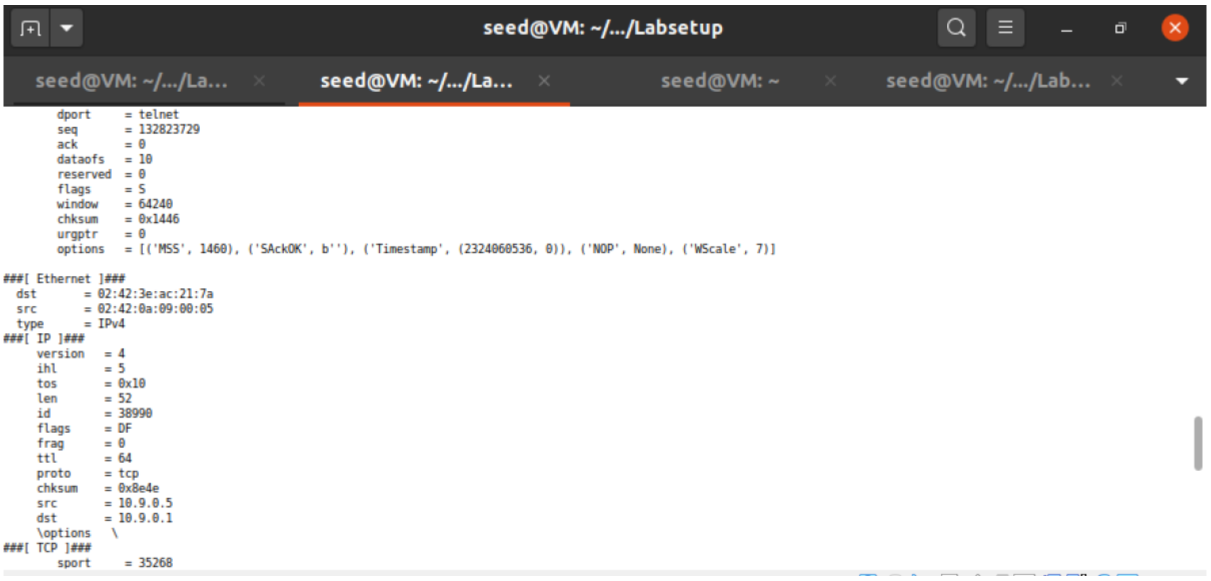
**On the Attacker terminal run the command:**

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

Provide a screenshot of your observations



```
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.1B-TCP.py
SNIFFING PACKETS...
###[ Ethernet ]###
  dst       = 02:42:3e:ac:21:7a
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 60
     id        = 38989
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x8e47
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 35268
        dport     = telnet
        seq       = 132823729
        ack       = 0
        dataofs   = 10
        reserved  = 0
        flags     = S
        window    = 64240
```



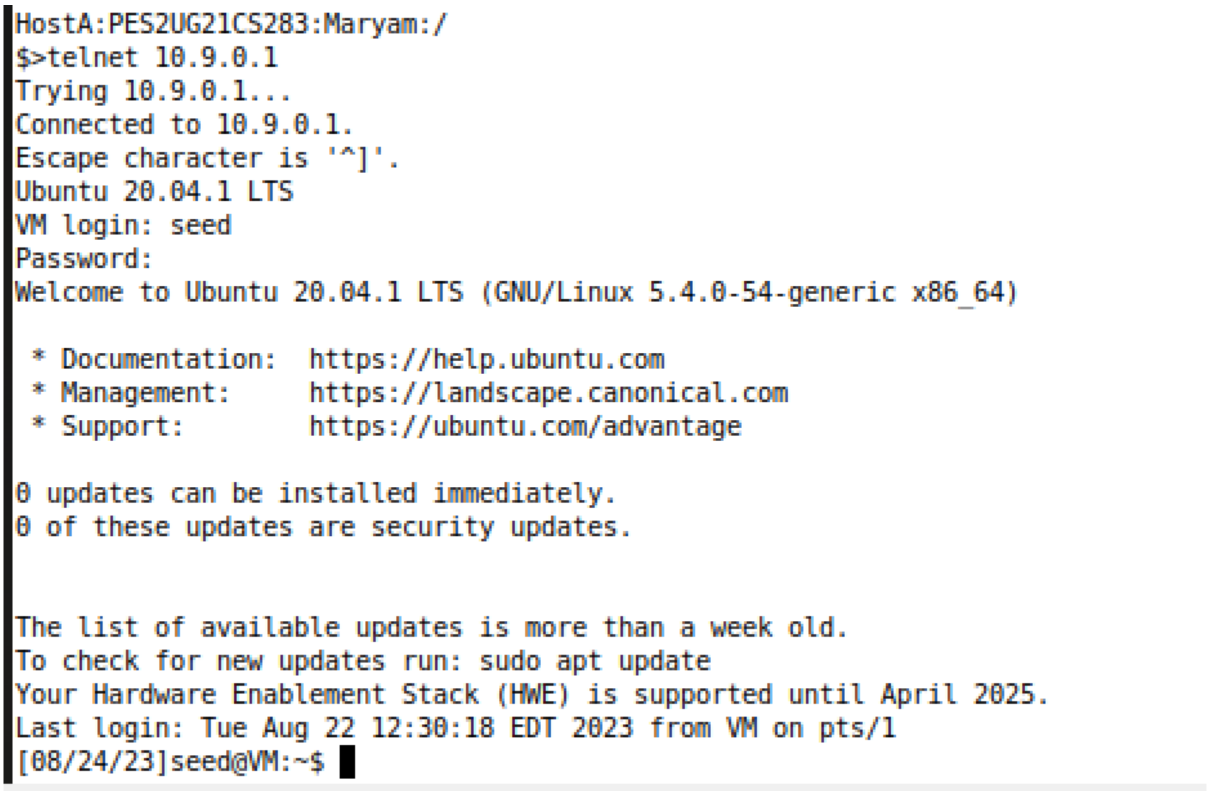
```
        dport     = telnet
        seq       = 132823729
        ack       = 0
        dataofs   = 10
        reserved  = 0
        flags     = S
        window    = 64240
        chksum    = 0x1446
        urgptr    = 0
        options   = [('MSS', 1460), ('SAckOK', b''), ('Timestamp', (2324060536, 0)), ('NOP', None), ('WScale', 7)]

###[ Ethernet ]###
  dst       = 02:42:3e:ac:21:7a
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 52
     id        = 38990
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x8e4e
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 35268
```

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.



```
HostA:PES2UG21CS283:Maryam:/
$>telnet 10.9.0.1
Trying 10.9.0.1...
Connected to 10.9.0.1.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 updates can be installed immediately.
0 of these updates are security updates.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Tue Aug 22 12:30:18 EDT 2023 from VM on pts/1
[08/24/23]seed@VM:~$ 
```

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

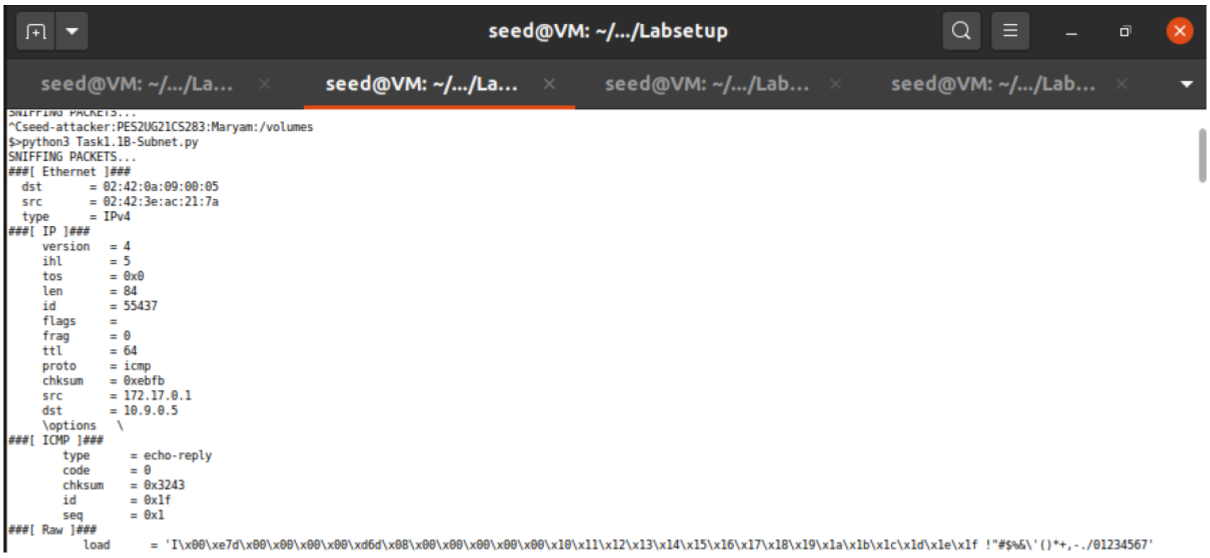
You can pick any subnet, such as 192.168.254.0/24; you should not pick the subnet that your VM is attached to.

Show that on sending ICMP packets to 192.168.254.1, the sniffer program captures the packets sent out from 192.168.254.1 .

**Fill in the interface of the attacker machine in the given program and run the code.**

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

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



seed@VM: ~/.../Labsetup

```
SNIFFING PACKETS...
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.1B-Subnet.py
SNIFFING PACKETS...
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:3e:ac:21:7a
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 55437
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0xebfb
     src       = 172.17.0.1
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-reply
        code      = 0
        chksum    = 0x3243
        id        = 0x1f
        seq       = 0x1
###[ Raw ]###
           load      = 'I\x00\xe7d\x00\x00\x00\x00\xd6d\x08\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01234567'
```

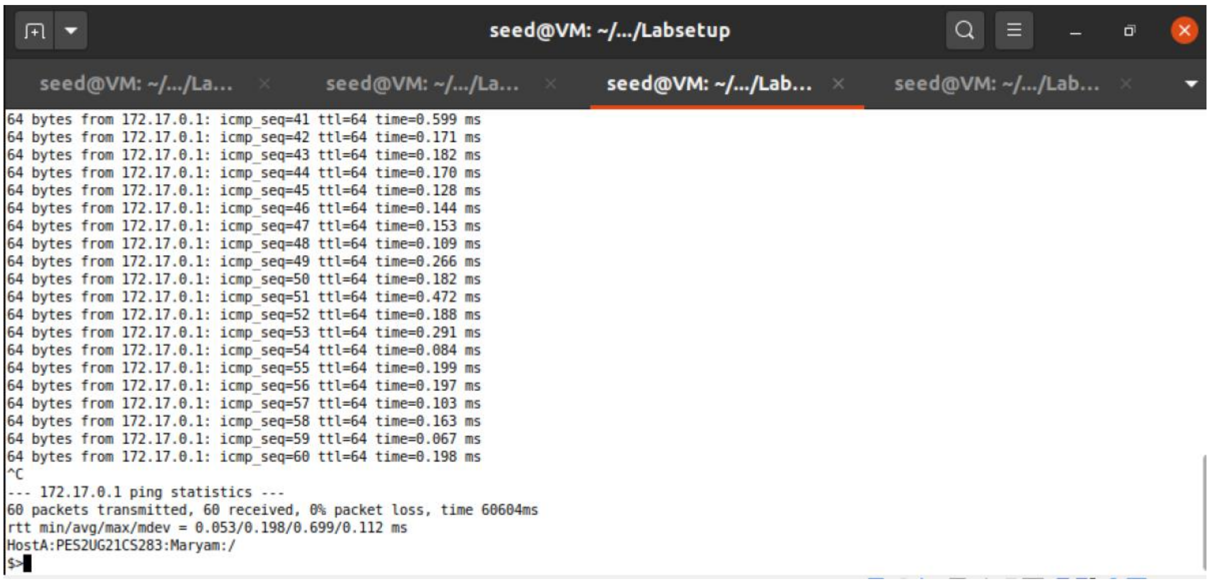
Provide a screenshot of your observations

From the host A machine's terminal, ping a random IP address on the chosen subnet.

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

**# ping 192.168.254.1**

Provide screenshots of your observations.



seed@VM: ~/.../Labsetup

```
64 bytes from 172.17.0.1: icmp_seq=41 ttl=64 time=0.599 ms
64 bytes from 172.17.0.1: icmp_seq=42 ttl=64 time=0.171 ms
64 bytes from 172.17.0.1: icmp_seq=43 ttl=64 time=0.182 ms
64 bytes from 172.17.0.1: icmp_seq=44 ttl=64 time=0.170 ms
64 bytes from 172.17.0.1: icmp_seq=45 ttl=64 time=0.128 ms
64 bytes from 172.17.0.1: icmp_seq=46 ttl=64 time=0.144 ms
64 bytes from 172.17.0.1: icmp_seq=47 ttl=64 time=0.153 ms
64 bytes from 172.17.0.1: icmp_seq=48 ttl=64 time=0.109 ms
64 bytes from 172.17.0.1: icmp_seq=49 ttl=64 time=0.266 ms
64 bytes from 172.17.0.1: icmp_seq=50 ttl=64 time=0.182 ms
64 bytes from 172.17.0.1: icmp_seq=51 ttl=64 time=0.472 ms
64 bytes from 172.17.0.1: icmp_seq=52 ttl=64 time=0.188 ms
64 bytes from 172.17.0.1: icmp_seq=53 ttl=64 time=0.291 ms
64 bytes from 172.17.0.1: icmp_seq=54 ttl=64 time=0.084 ms
64 bytes from 172.17.0.1: icmp_seq=55 ttl=64 time=0.199 ms
64 bytes from 172.17.0.1: icmp_seq=56 ttl=64 time=0.197 ms
64 bytes from 172.17.0.1: icmp_seq=57 ttl=64 time=0.103 ms
64 bytes from 172.17.0.1: icmp_seq=58 ttl=64 time=0.163 ms
64 bytes from 172.17.0.1: icmp_seq=59 ttl=64 time=0.067 ms
64 bytes from 172.17.0.1: icmp_seq=60 ttl=64 time=0.198 ms
^C
--- 172.17.0.1 ping statistics ---
60 packets transmitted, 60 received, 0% packet loss, time 60604ms
rtt min/avg/max/mdev = 0.053/0.198/0.699/0.112 ms
HostA:PES2UG21CS283:Maryam:/
$>
```

## Task 1.2 : Spoofing

The objective of this task is to spoof IP packets with an arbitrary source IP address. We will spoof **ICMP echo request packets** and send them to another VM on the same network. We will use Wireshark to observe whether our request will be accepted by the receiver. If it is accepted, an echo reply packet will be sent to the spoofed IP address. Below shows the code to create the ICMP packet. The spoofed request is formed by creating our own packet with the header specifications.

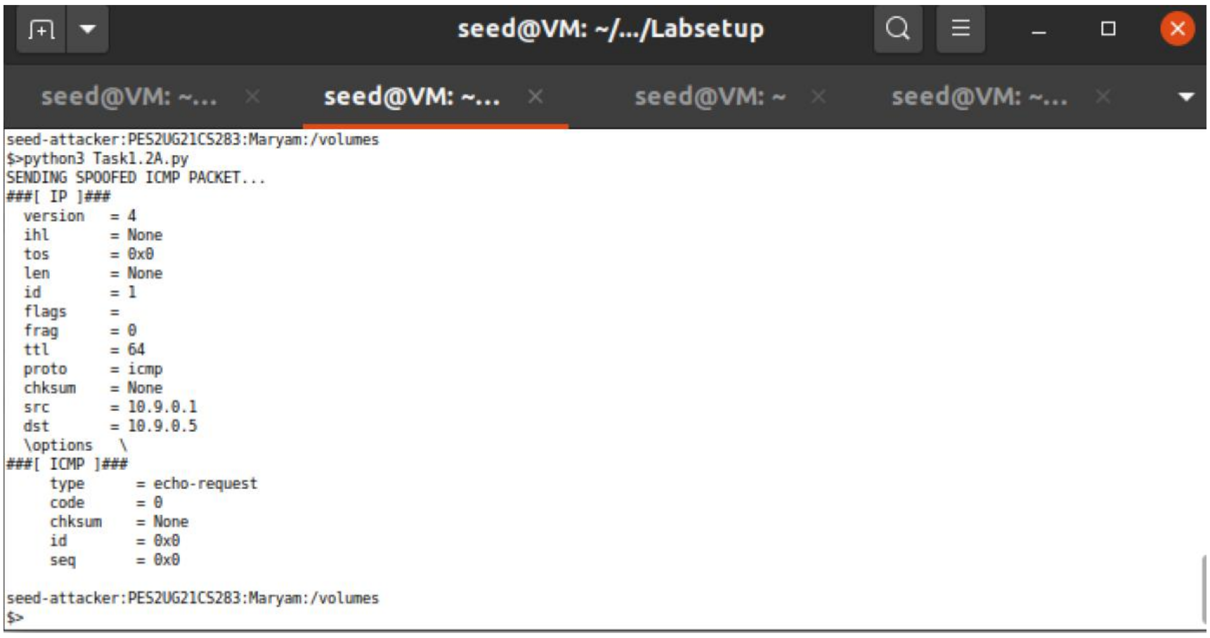
Similarly, we fill the IP header with source IP address of any machine within the local network and destination IP address of any remote machine on the internet which is alive.

Open wireshark before executing the program and select the same interface in wireshark, as used in the code for each task ie. the attacker machines interface. Show that Wireshark captures the live machine sending back an ICMP response.

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

**# python Task1.2A.py**

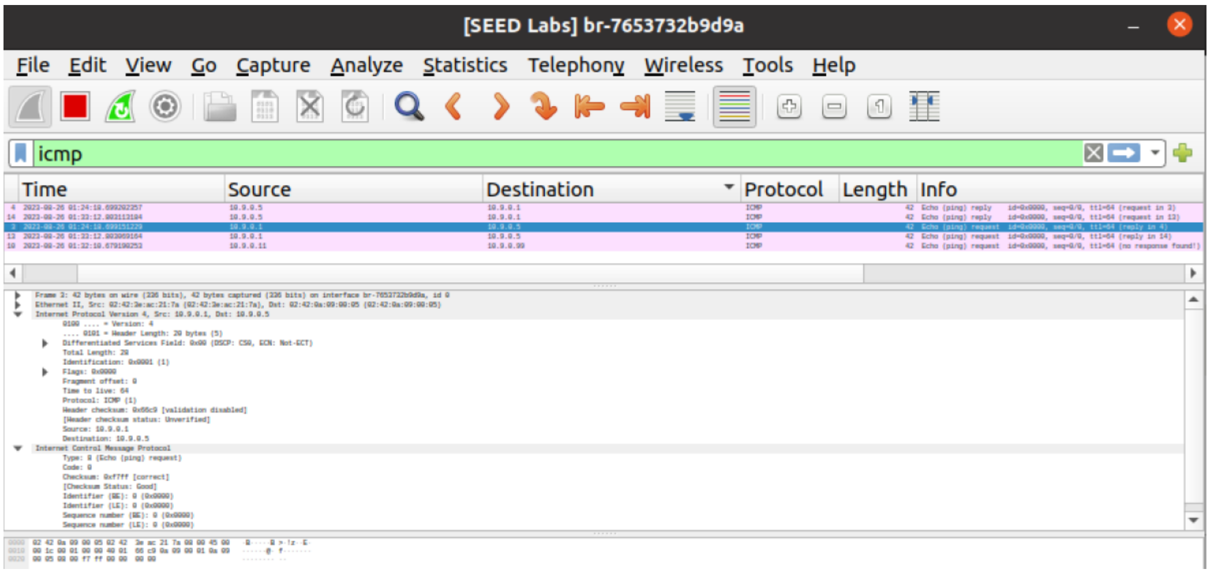
Provide a screenshot of your observations.



```
seed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.2A.py
SENDING SPOOFED ICMP PACKET...
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = icmp
  chksum    = None
  src       = 10.9.0.1
  dst       = 10.9.0.5
  \options   \
###[ ICMP ]###
     type      = echo-request
     code      = 0
     chksum    = None
     id        = 0x0
     seq       = 0x0

seed-attacker:PES2UG21CS283:Maryam:/volumes
$>
```

Demonstrate that you can spoof an ICMP echo request packet with an **arbitrary source IP address**. Open Wireshark and observe the ICMP packets as they are being captured. Select the same interface in wireshark, as used in the code for each task ie. the attacker containers interface.

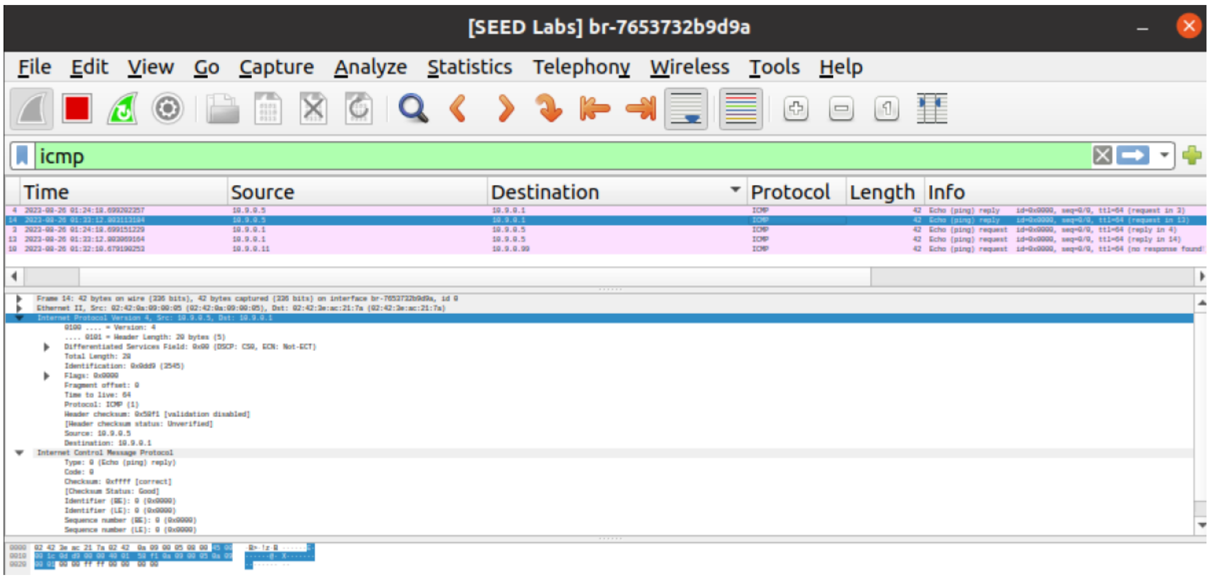
Observation: From frame number 2 we can see both request and reply when we give the destination address of the host



[SEED Labs] br-7653732b9d9a

```
icmp

     Time                            Source      Destination  Protocol Length Info
 4   2023-08-26 01:24:18.699202357   10.9.0.5    10.9.0.1     ICMP     42 Echo (ping) reply    id=0x0000, seq=0/0, ttl=64 (request in 3)
14   2023-08-26 01:33:12.803113184   10.9.0.5    10.9.0.1     ICMP     42 Echo (ping) reply    id=0x0000, seq=0/0, ttl=64 (request in 13)
 3   2023-08-26 01:24:18.699151229   10.9.0.1    10.9.0.5     ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (reply in 4)
13   2023-08-26 01:33:12.803069164   10.9.0.1    10.9.0.5     ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (reply in 14)
10   2023-08-26 01:32:10.679190252   10.9.0.11   10.9.0.99    ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (no response found!)

Frame 3: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-7653732b9d9a, id 0
Ethernet II, Src: 02:42:3e:ac:21:7a (02:42:3e:ac:21:7a), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
Internet Protocol Version 4, Src: 10.9.0.1, Dst: 10.9.0.5
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 28
    Identification: 0x0001 (1)
    Flags: 0x0000
    Fragment offset: 0
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0x66c9 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.9.0.1
    Destination: 10.9.0.5
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xf7ff [correct]
    [Checksum Status: Good]
    Identifier (BE): 0 (0x0000)
    Identifier (LE): 0 (0x0000)
    Sequence number (BE): 0 (0x0000)
    Sequence number (LE): 0 (0x0000)

0000  02 42 0a 09 00 05 02 42  3e ac 21 7a 08 00 45 00   ·B·····B >·!z··E·
0010  00 1c 00 01 00 00 40 01  66 c9 0a 09 00 01 0a 09   ······@· f·······
0020  00 05 08 00 f7 ff 00 00  00 00                     ········ ··
```

Frame number 14



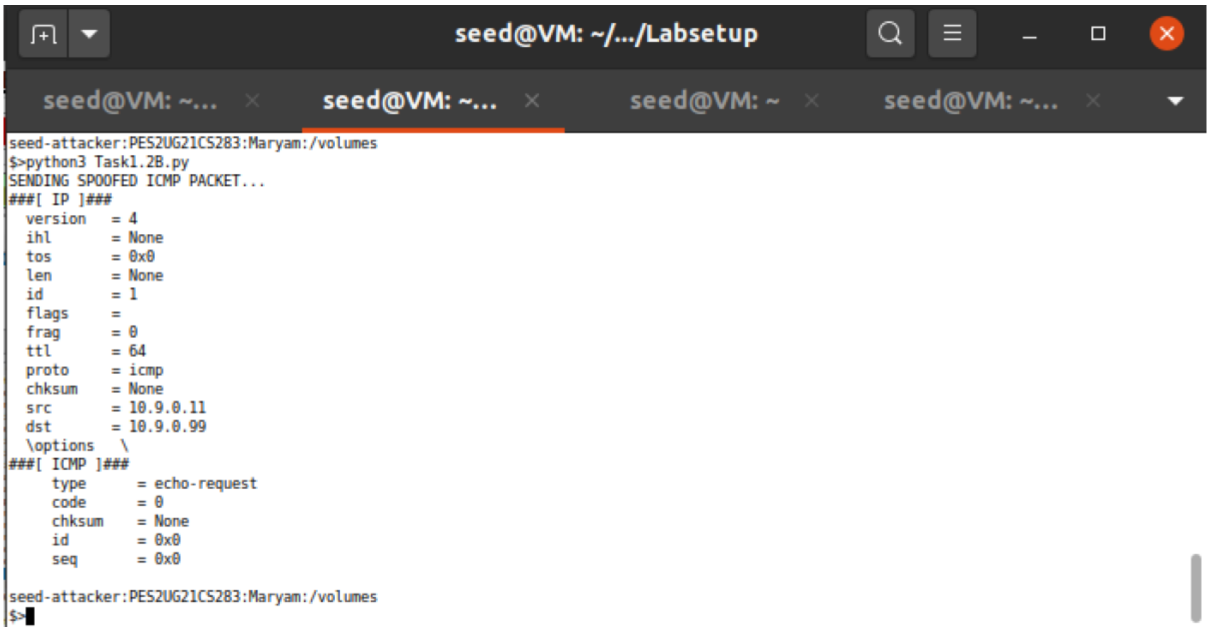
[SEED Labs] br-7653732b9d9a

```
icmp

     Time                            Source      Destination  Protocol Length Info
 4   2023-08-26 01:24:18.699202357   10.9.0.5    10.9.0.1     ICMP     42 Echo (ping) reply    id=0x0000, seq=0/0, ttl=64 (request in 3)
14   2023-08-26 01:33:12.803113184   10.9.0.5    10.9.0.1     ICMP     42 Echo (ping) reply    id=0x0000, seq=0/0, ttl=64 (request in 13)
 3   2023-08-26 01:24:18.699151229   10.9.0.1    10.9.0.5     ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (reply in 4)
13   2023-08-26 01:33:12.803069164   10.9.0.1    10.9.0.5     ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (reply in 14)
10   2023-08-26 01:32:10.679190252   10.9.0.11   10.9.0.99    ICMP     42 Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (no response found!)

Frame 14: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-7653732b9d9a, id 0
Ethernet II, Src: 02:42:0a:09:00:05 (02:42:0a:09:00:05), Dst: 02:42:3e:ac:21:7a (02:42:3e:ac:21:7a)
Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 28
    Identification: 0x09d9 (2521)
    Flags: 0x0000
    Fragment offset: 0
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0x58f1 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.9.0.5
    Destination: 10.9.0.1
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0xffff [correct]
    [Checksum Status: Good]
    Identifier (BE): 0 (0x0000)
    Identifier (LE): 0 (0x0000)
    Sequence number (BE): 0 (0x0000)
    Sequence number (LE): 0 (0x0000)

0000  02 42 3e ac 21 7a 02 42  0a 09 00 05 08 00 45 00   ·B>·!z·B ······E·
0010  00 1c 0d d9 00 00 40 01  58 f1 0a 09 00 05 0a 09   ······@· X·······
0020  00 01 00 00 ff ff 00 00  00 00                     ········ ··
```

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

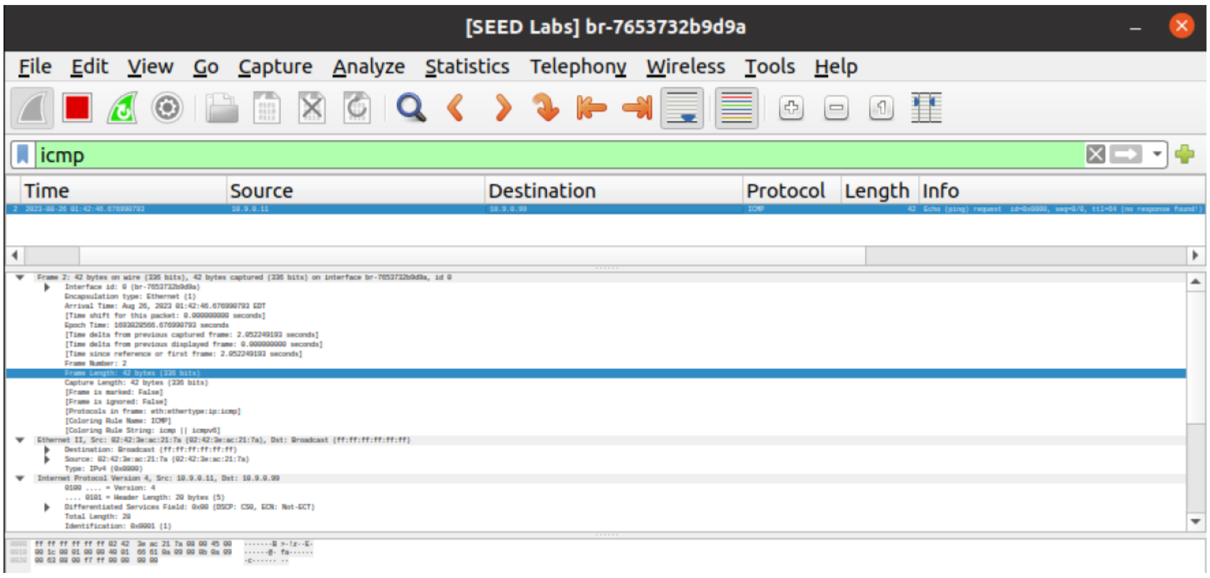
**# python Task1.2B.py**

Provide a screenshot of your observations.



```
seed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.2B.py
SENDING SPOOFED ICMP PACKET...
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = icmp
  chksum    = None
  src       = 10.9.0.11
  dst       = 10.9.0.99
  \options   \
###[ ICMP ]###
     type      = echo-request
     code      = 0
     chksum    = None
     id        = 0x0
     seq       = 0x0

seed-attacker:PES2UG21CS283:Maryam:/volumes
$>
```

Frame number 2: no response seen



```
[SEED Labs] br-7653732b9d9a
icmp

Time                          Source      Destination  Protocol  Length  Info
2 2023-08-26 01:42:46.676990793  10.9.0.11   10.9.0.99    ICMP      42  Echo (ping) request  id=0x0000, seq=0/0, ttl=64 (no response found!)

Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-7653732b9d9a, id 0
    Interface id: 0 (br-7653732b9d9a)
    Encapsulation type: Ethernet (1)
    Arrival Time: Aug 26, 2023 01:42:46.676990793 EDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1693028566.676990793 seconds
    [Time delta from previous captured frame: 2.052249193 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 2.052249193 seconds]
    Frame Number: 2
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:icmp]
    [Coloring Rule Name: ICMP]
    [Coloring Rule String: icmp || icmpv6]
Ethernet II, Src: 02:42:3e:ac:21:7a (02:42:3e:ac:21:7a), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
    Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    Source: 02:42:3e:ac:21:7a (02:42:3e:ac:21:7a)
    Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 10.9.0.11, Dst: 10.9.0.99
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 28
    Identification: 0x0001 (1)

0000  ff ff ff ff ff ff 02 42  3e ac 21 7a 08 00 45 00   .......B >.!z..E.
0010  00 1c 00 01 00 00 40 01  66 61 0a 09 00 0b 0a 09   ......@. fa......
0020  00 63 08 00 f7 ff 00 00  00 00                     .c...... ..
```

## 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 your VM and a selected destination.

The below code is a simple traceroute implementation using Scapy. It takes hostname or IP address as the input. We create an IP packet with destination address and TTL value and ICMP packet. We send the packet using function sr1(). This function waits for the reply from the destination. If the ICMP reply type is 0, we receive an echo response from the destination, else we increase the TTL value and resend the packet.

Provide a screenshot of the Wireshark capture that shows the ICMP requests sent with increasing TTL and the error response from the routers with a message as “Time to live exceeded”.

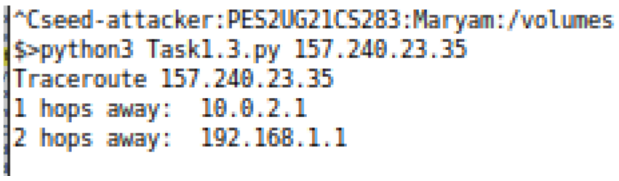
Open wireshark before executing the program and select the same interface in wireshark, as used in the code for each task ie. the attacker machine’s interface.

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

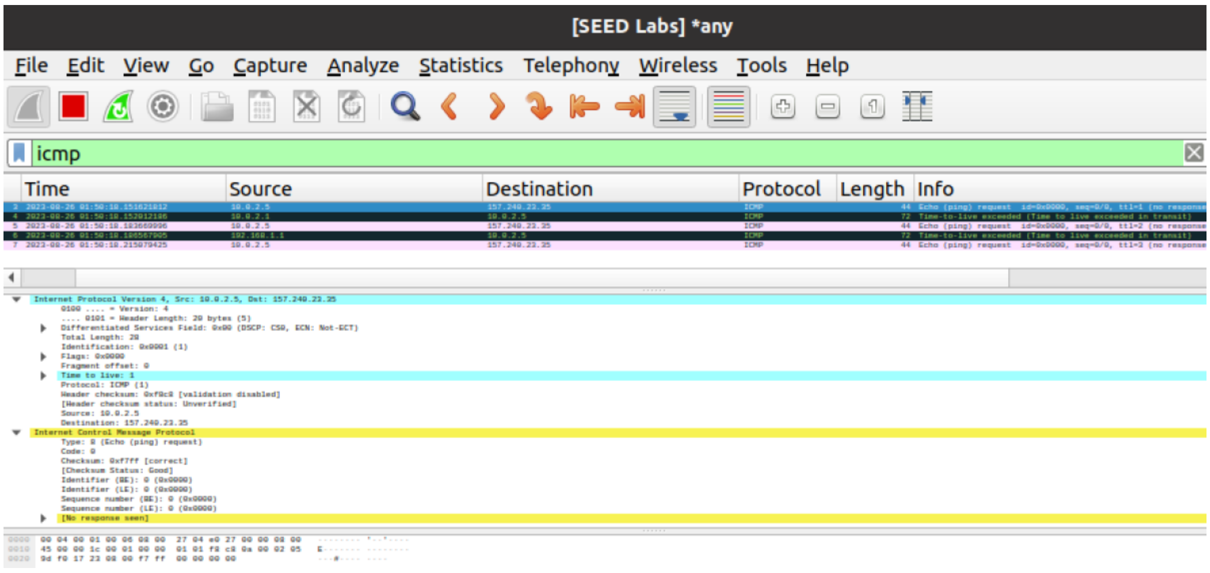
**# python3 Task1.3.py 157.240.23.35**

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

Provide a screenshot of your observations.



```
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.3.py 157.240.23.35
Traceroute 157.240.23.35
1 hops away:  10.0.2.1
2 hops away:  192.168.1.1
```



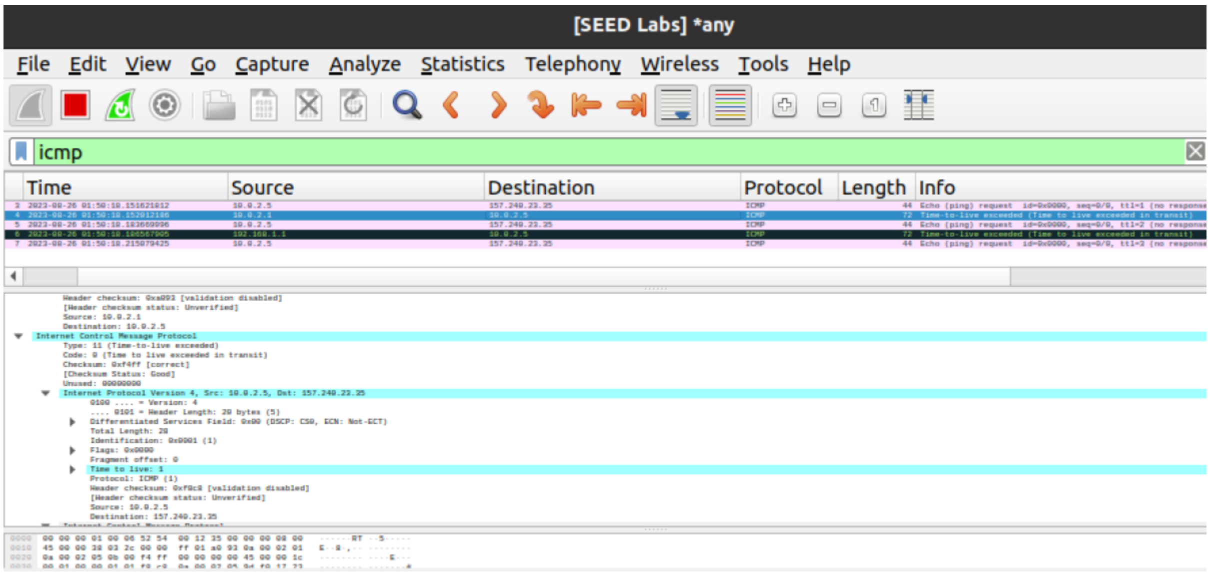
[SEED Labs] *any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Filter: icmp

```
  Time                             Source          Destination      Protocol  Length  Info
3 2023-08-26 01:50:18.151621812    10.0.2.5        157.240.23.35    ICMP      44      Echo (ping) request  id=0x0000, seq=0/0, ttl=1 (no response
4 2023-08-26 01:50:18.152012186    10.0.2.1        10.0.2.5         ICMP      72      Time-to-live exceeded (Time to live exceeded in transit)
5 2023-08-26 01:50:18.183669996    10.0.2.5        157.240.23.35    ICMP      44      Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (no response
6 2023-08-26 01:50:18.186567905    192.168.1.1     10.0.2.5         ICMP      72      Time-to-live exceeded (Time to live exceeded in transit)
7 2023-08-26 01:50:18.215079425    10.0.2.5        157.240.23.35    ICMP      44      Echo (ping) request  id=0x0000, seq=0/0, ttl=3 (no response
```

```
▼ Internet Protocol Version 4, Src: 10.0.2.5, Dst: 157.240.23.35
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 28
      Identification: 0x0001 (1)
    ▶ Flags: 0x0000
      Fragment offset: 0
    ▶ Time to live: 1
      Protocol: ICMP (1)
      Header checksum: 0xf0c8 [validation disabled]
      [Header checksum status: Unverified]
      Source: 10.0.2.5
      Destination: 157.240.23.35
▼ Internet Control Message Protocol
      Type: 8 (Echo (ping) request)
      Code: 0
      Checksum: 0xf7ff [correct]
      [Checksum Status: Good]
      Identifier (BE): 0 (0x0000)
      Identifier (LE): 0 (0x0000)
      Sequence number (BE): 0 (0x0000)
      Sequence number (LE): 0 (0x0000)
    ▶ [No response seen]
```

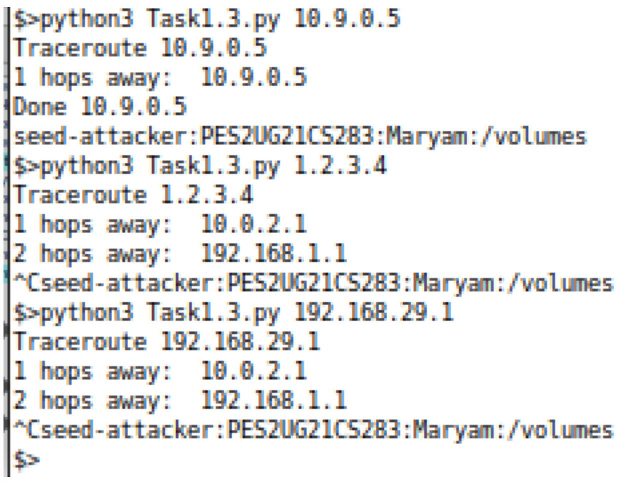
```
0000  00 04 00 01 00 06 08 00  27 04 e0 27 00 00 08 00   ........ '..'....
0010  45 00 00 1c 00 01 00 00  01 01 f0 c8 0a 00 02 05   E....... ........
0020  9d f0 17 23 08 00 f7 ff  00 00 00 00               ...#.... ....
```


[SEED Labs] *any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp

```
Time                             Source        Destination     Protocol Length Info
3 2023-08-26 01:50:18.151621812  10.0.2.5      157.240.23.35   ICMP     44 Echo (ping) request  id=0x0000, seq=0/0, ttl=1 (no response
4 2023-08-26 01:50:18.152012186  10.0.2.1      10.0.2.5        ICMP     72 Time-to-live exceeded (Time to live exceeded in transit)
5 2023-08-26 01:50:18.183669996  10.0.2.5      157.240.23.35   ICMP     44 Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (no response
6 2023-08-26 01:50:18.186567905  192.168.1.1   10.0.2.5        ICMP     72 Time-to-live exceeded (Time to live exceeded in transit)
7 2023-08-26 01:50:18.215079425  10.0.2.5      157.240.23.35   ICMP     44 Echo (ping) request  id=0x0000, seq=0/0, ttl=3 (no response
```

```
        Header checksum: 0xa093 [validation disabled]
        [Header checksum status: Unverified]
        Source: 10.0.2.1
        Destination: 10.0.2.5
    Internet Control Message Protocol
        Type: 11 (Time-to-live exceeded)
        Code: 0 (Time to live exceeded in transit)
        Checksum: 0xf4ff [correct]
        [Checksum Status: Good]
        Unused: 00000000
        Internet Protocol Version 4, Src: 10.0.2.5, Dst: 157.240.23.35
            0100 .... = Version: 4
            .... 0101 = Header Length: 20 bytes (5)
            Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
            Total Length: 28
            Identification: 0x0001 (1)
            Flags: 0x0000
            Fragment offset: 0
            Time to live: 1
            Protocol: ICMP (1)
            Header checksum: 0xf9c8 [validation disabled]
            [Header checksum status: Unverified]
            Source: 10.0.2.5
            Destination: 157.240.23.35
```

```
0000  00 00 00 01 00 06 52 54  00 12 35 00 00 00 08 00   ......RT ..5.....
0010  45 00 00 38 02 2c 00 00  ff 01 a0 93 0a 00 02 01   E..8.,.. ........
0020  0a 00 02 05 0b 00 f4 ff  00 00 00 00 45 00 00 1c   ........ ....E...
```



```
$>python3 Task1.3.py 10.9.0.5
Traceroute 10.9.0.5
1 hops away:  10.9.0.5
Done 10.9.0.5
seed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.3.py 1.2.3.4
Traceroute 1.2.3.4
1 hops away:  10.0.2.1
2 hops away:  192.168.1.1
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.3.py 192.168.29.1
Traceroute 192.168.29.1
1 hops away:  10.0.2.1
2 hops away:  192.168.1.1
^Cseed-attacker:PES2UG21CS283:Maryam:/volumes
$>
```

## 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.

The below code sniffs ICMP packets sent out by the victim machine. Using the callback function, we can use the packets to send the spoofed packets. We retrieve source IP and destination IP from the sniffed packet and create a new IP packet. The new source IP of the spoofed packet is the sniffed packet’s destination IP address and vice versa. We also generate ICMP packets with id and sequence number. In the new packet, ICMP type should be 0 (ICMP reply). To avoid truncated packets, we also add the data to the new packet.

Open wireshark before executing the program and select the same interface in wireshark, as used in the code for each task ie. the attacker machine’s interface.

**Fill in the interface of the attacker machine in the given program and run the code.**

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

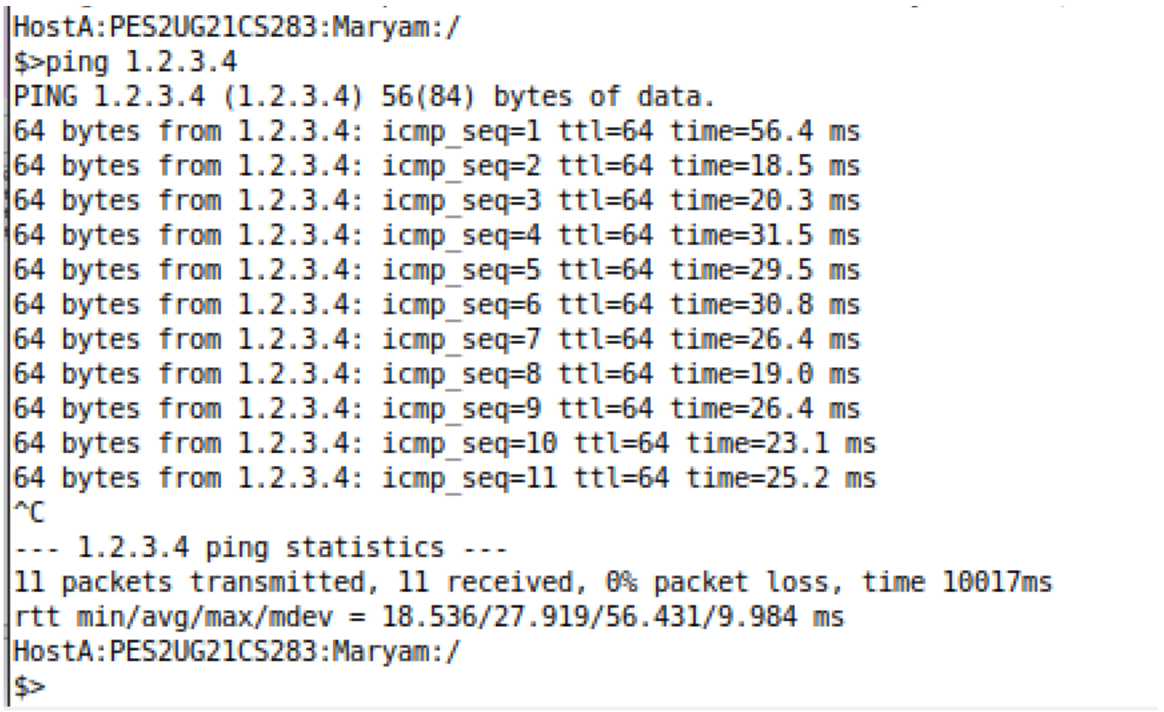
```
# python3 Task1.4.py
```

From the host A machine’s terminal ping 1.2.3.4

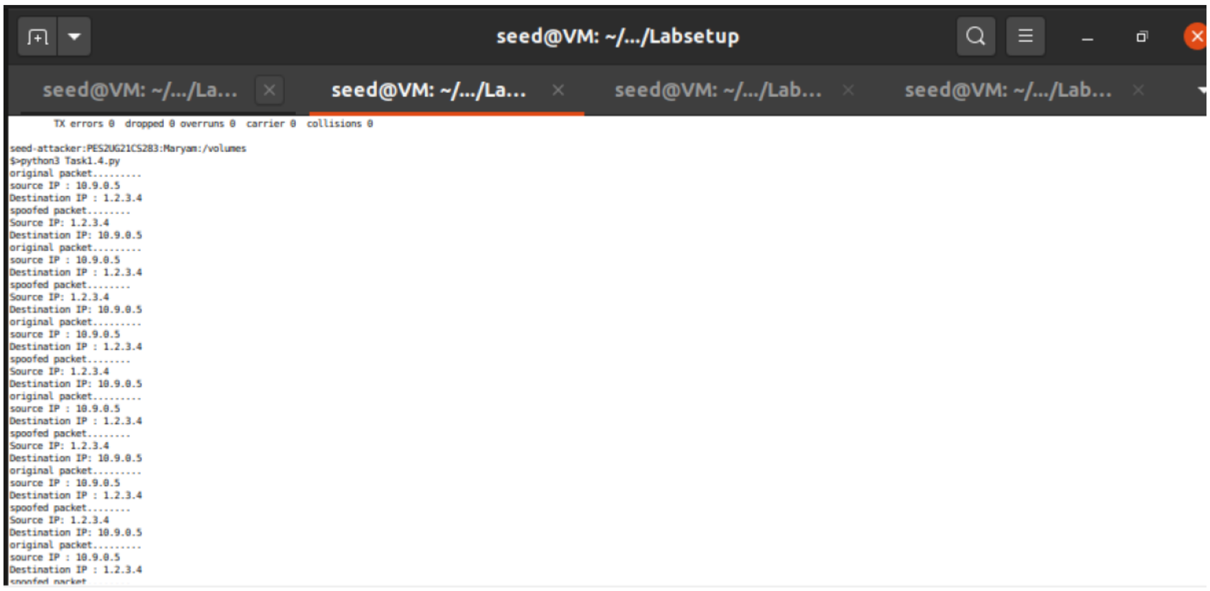
**On the Host A terminal run the command:**

```
# ping 1.2.3.4
```

Provide a screenshot of your observations.

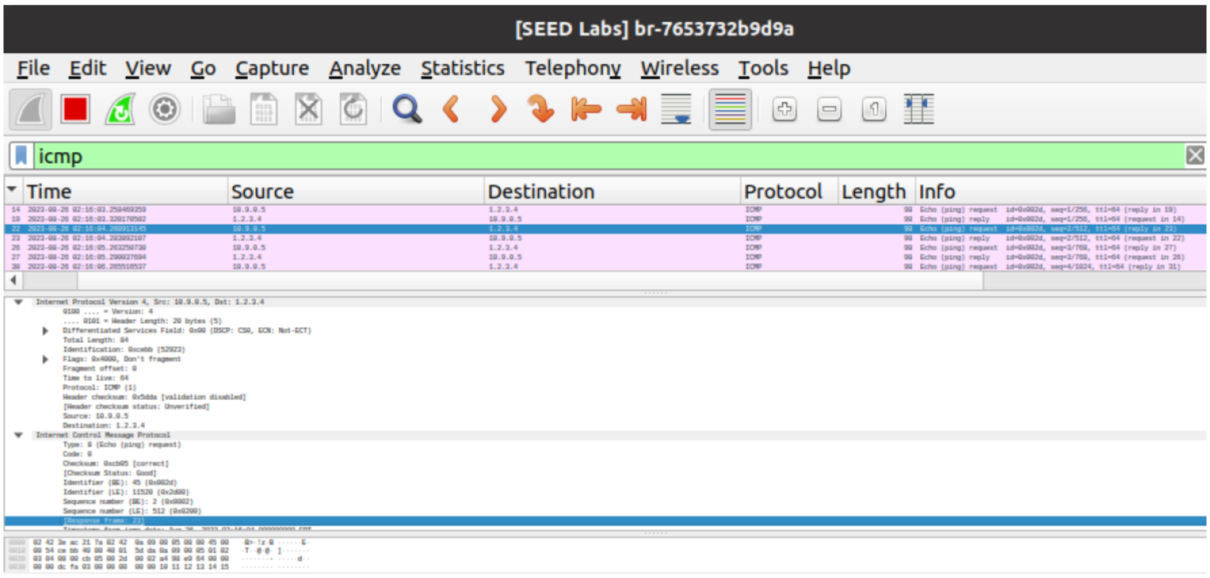


```
HostA:PES2UG21CS283:Maryam:/
$>ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=56.4 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=18.5 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=20.3 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=31.5 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=29.5 ms
64 bytes from 1.2.3.4: icmp_seq=6 ttl=64 time=30.8 ms
64 bytes from 1.2.3.4: icmp_seq=7 ttl=64 time=26.4 ms
64 bytes from 1.2.3.4: icmp_seq=8 ttl=64 time=19.0 ms
64 bytes from 1.2.3.4: icmp_seq=9 ttl=64 time=26.4 ms
64 bytes from 1.2.3.4: icmp_seq=10 ttl=64 time=23.1 ms
64 bytes from 1.2.3.4: icmp_seq=11 ttl=64 time=25.2 ms
^C
--- 1.2.3.4 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10017ms
rtt min/avg/max/mdev = 18.536/27.919/56.431/9.984 ms
HostA:PES2UG21CS283:Maryam:/
$>
```



```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

seed-attacker:PES2UG21CS283:Maryam:/volumes
$>python3 Task1.4.py
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet........
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet........
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet........
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet........
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet........
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
original packet.........
source IP : 10.9.0.5
Destination IP : 1.2.3.4
spoofed packet
```

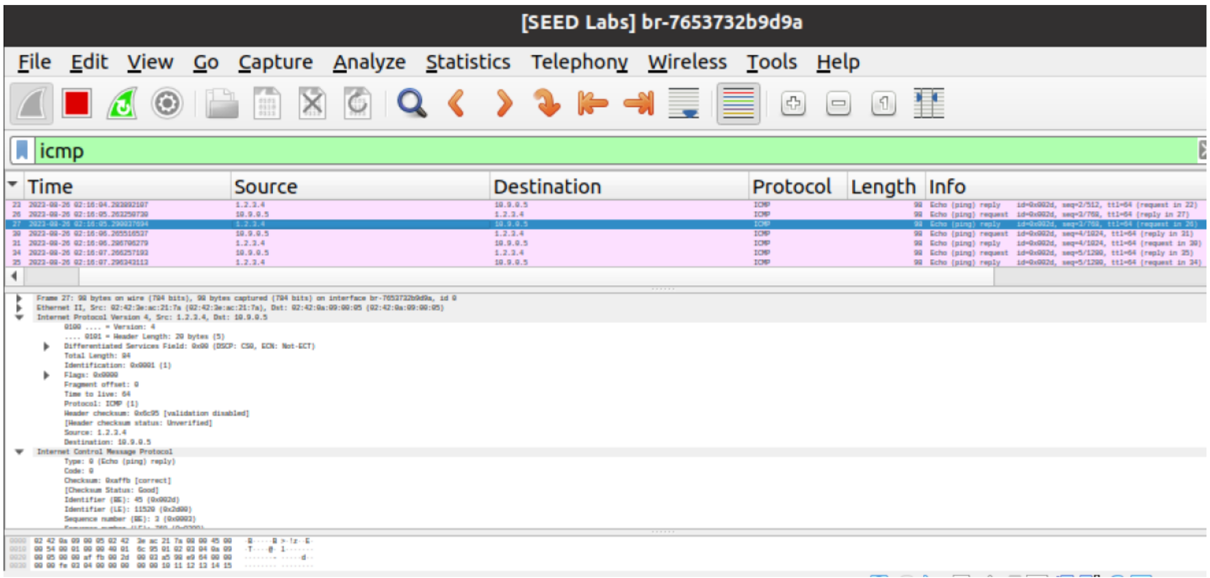
Frame 22


[SEED Labs] br-7653732b9d9a
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp

```
   Time                           Source     Destination  Protocol  Length  Info
14 2023-08-26 02:16:03.258469259  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=1/256, ttl=64 (reply in 19)
19 2023-08-26 02:16:03.320170502  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=1/256, ttl=64 (request in 14)
22 2023-08-26 02:16:04.269913145  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=2/512, ttl=64 (reply in 23)
23 2023-08-26 02:16:04.293892107  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=2/512, ttl=64 (request in 22)
26 2023-08-26 02:16:05.263250730  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=3/768, ttl=64 (reply in 27)
27 2023-08-26 02:16:05.290037694  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=3/768, ttl=64 (request in 26)
30 2023-08-26 02:16:06.265516537  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=4/1024, ttl=64 (reply in 31)
```

```
Internet Protocol Version 4, Src: 10.9.0.5, Dst: 1.2.3.4
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0xcebb (52923)
    Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0x5dda [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.9.0.5
    Destination: 1.2.3.4
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xcb05 [correct]
    [Checksum Status: Good]
    Identifier (BE): 45 (0x002d)
    Identifier (LE): 11520 (0x2d00)
    Sequence number (BE): 2 (0x0002)
    Sequence number (LE): 512 (0x0200)
    [Response frame: 23]

0000  02 42 3e ac 21 7a 02 42  0a 09 00 05 08 00 45 00   ·B>·!z·B ······E·
0010  00 54 ce bb 40 00 40 01  5d da 0a 09 00 05 01 02   ·T··@·@· ]·······
0020  03 04 08 00 cb 05 00 2d  00 02 a4 98 e9 64 00 00   ·······- ·····d··
0030  00 00 dc fa 03 00 00 00  00 00 10 11 12 13 14 15   ········ ········
```

Frame 27


[SEED Labs] br-7653732b9d9a
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp

```
   Time                           Source     Destination  Protocol  Length  Info
23 2023-08-26 02:16:04.293892107  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=2/512, ttl=64 (request in 22)
26 2023-08-26 02:16:05.263250730  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=3/768, ttl=64 (reply in 27)
27 2023-08-26 02:16:05.290037694  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=3/768, ttl=64 (request in 26)
30 2023-08-26 02:16:06.265516537  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=4/1024, ttl=64 (reply in 31)
31 2023-08-26 02:16:06.286706279  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=4/1024, ttl=64 (request in 30)
34 2023-08-26 02:16:07.266257193  10.9.0.5   1.2.3.4      ICMP      98      Echo (ping) request  id=0x002d, seq=5/1280, ttl=64 (reply in 35)
35 2023-08-26 02:16:07.296343113  1.2.3.4    10.9.0.5     ICMP      98      Echo (ping) reply    id=0x002d, seq=5/1280, ttl=64 (request in 34)
```

```
Frame 27: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface br-7653732b9d9a, id 0
Ethernet II, Src: 02:42:3e:ac:21:7a (02:42:3e:ac:21:7a), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
Internet Protocol Version 4, Src: 1.2.3.4, Dst: 10.9.0.5
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x0001 (1)
    Flags: 0x0000
    Fragment offset: 0
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0x6c95 [validation disabled]
    [Header checksum status: Unverified]
    Source: 1.2.3.4
    Destination: 10.9.0.5
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0xaffb [correct]
    [Checksum Status: Good]
    Identifier (BE): 45 (0x002d)
    Identifier (LE): 11520 (0x2d00)
    Sequence number (BE): 3 (0x0003)

0000  02 42 0a 09 00 05 02 42  3e ac 21 7a 08 00 45 00   ·B·····B >·!z··E·
0010  00 54 00 01 00 00 40 01  6c 95 01 02 03 04 0a 09   ·T····@· l·······
0020  00 05 00 00 af fb 00 2d  00 03 a5 98 e9 64 00 00   ·······- ·····d··
0030  00 00 fe 03 04 00 00 00  00 00 10 11 12 13 14 15   ········ ········
```