Nguyen Viet Hoang Nam

ITITIU19162

System and Network Security

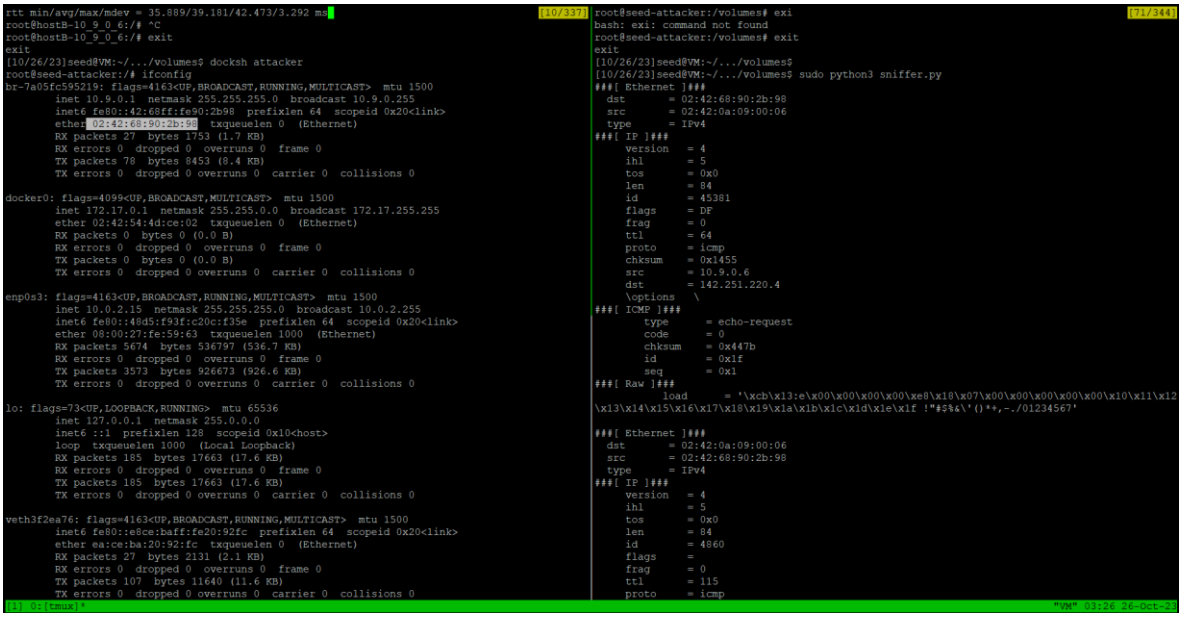
Lab 2

```
[10/26/23]seed@VM:~/.../Labsetup$ dockps
1ceb2fdba91f  attacker
d14314f6ad7e  hostA
6d756689287a  hostB
[10/26/23]seed@VM:~/.../Labsetup$ docksh attacker
root@seed-attacker:/# ip -br addr
lo               UNKNOWN        127.0.0.1/8 ::1/128
enp0s3           UP             10.0.2.15/24 fe80::48d5:f93f:c20c:f35e/64
docker0          DOWN           172.17.0.1/16
br-7a05fc595219  UP             10.9.0.1/24 fe80::42:68ff:fe90:2b98/64
veth3f2ea76@if5  UP             fe80::e8ce:baff:fe20:92fc/64
veth5ba7b2b@if7  UP             fe80::2435:1bff:feda:de68/64
root@seed-attacker:/#
```

My interface name of 10.9.0.1 is br-7a05fc595219

Task 1.1:

a) With the root privilege - sudo



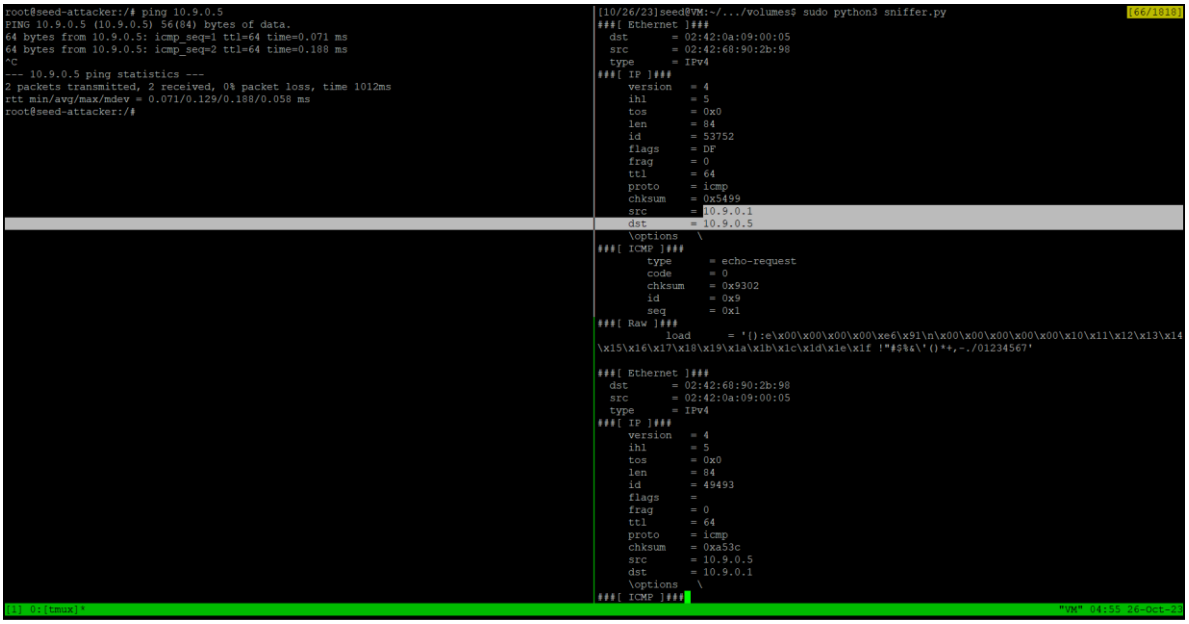
The sudo command not only gave the icmp request – reply but also show the source MAC address (mine was 02:42:68:90:2b:98) and the destination MAC address.

Without the root privilege:

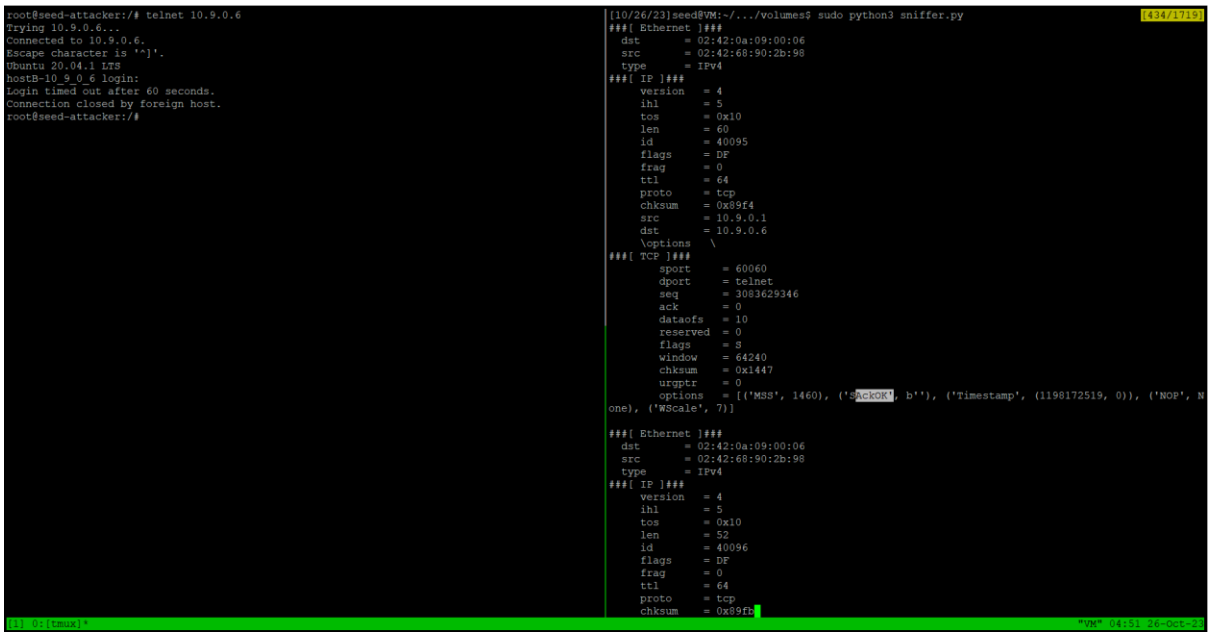
```
[10/26/23]seed@VM:~/.../volumes$ python3 sniffer.py
Traceback (most recent call last):
  File "sniffer.py", line 6, in <module>
    pkt = sniff(iface="br-7a05fc595219", filter="icmp", prn=print_pkt, count=5)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    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: E
501
  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
```

It seem that the file could not run without the root permission – sudo.

b) Only ICMP packets (ping 10.9.0.5 - hostA): with the source IP address is 10.9.0.1 – attacker to destination IP address 10.9.0.5.



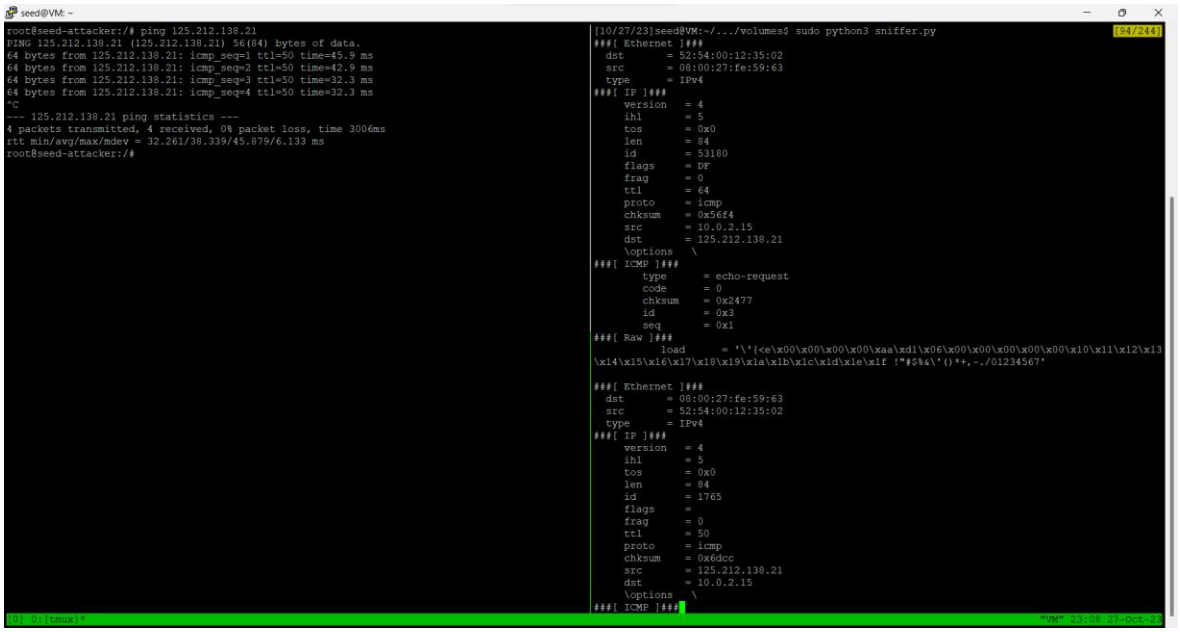
TCP packets:



A TCP packet showed an AckOK flag. And the sniffer.py was adjusted as follow:

```
# sniffer.py
#!/usr/bin/python3
from scapy.all import *
def print_pkt(pkt):
 pkt.show()
f = "tcp and dst port 23 and src net 10.9.0.1"
pkt = sniff(iface="br-7a05fc595219",filter=f, prn=print_pkt, count=100)
```

Any packets:

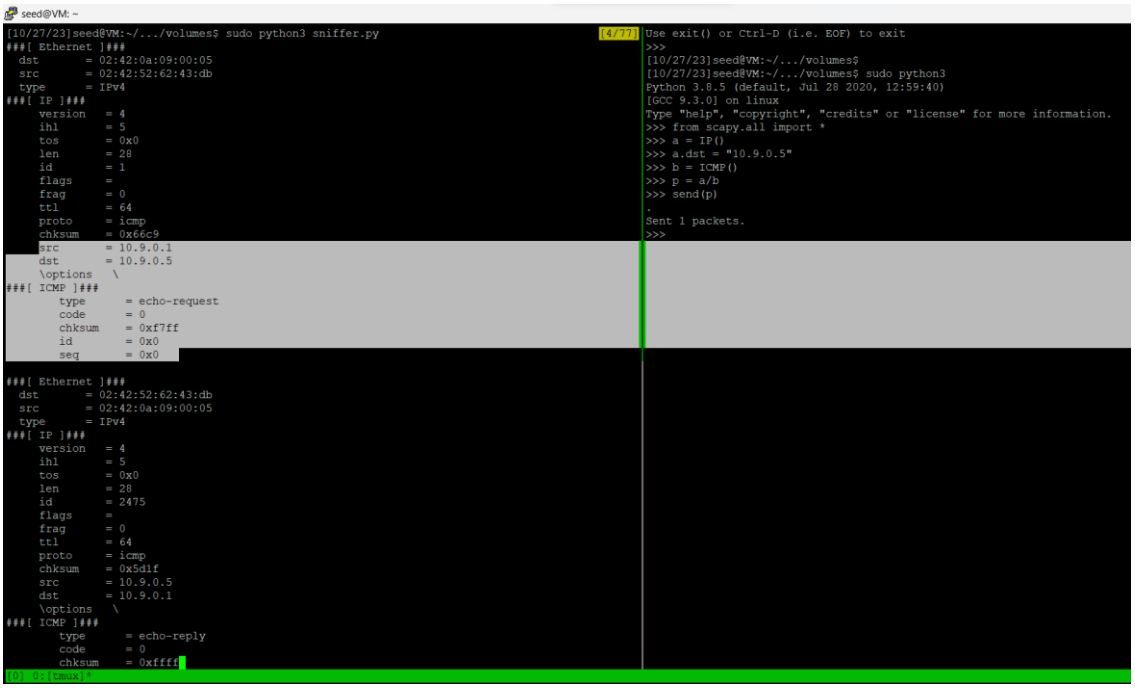


I have captured ICMP, Ethernet, IP packets.

Here is my sniffer.py:

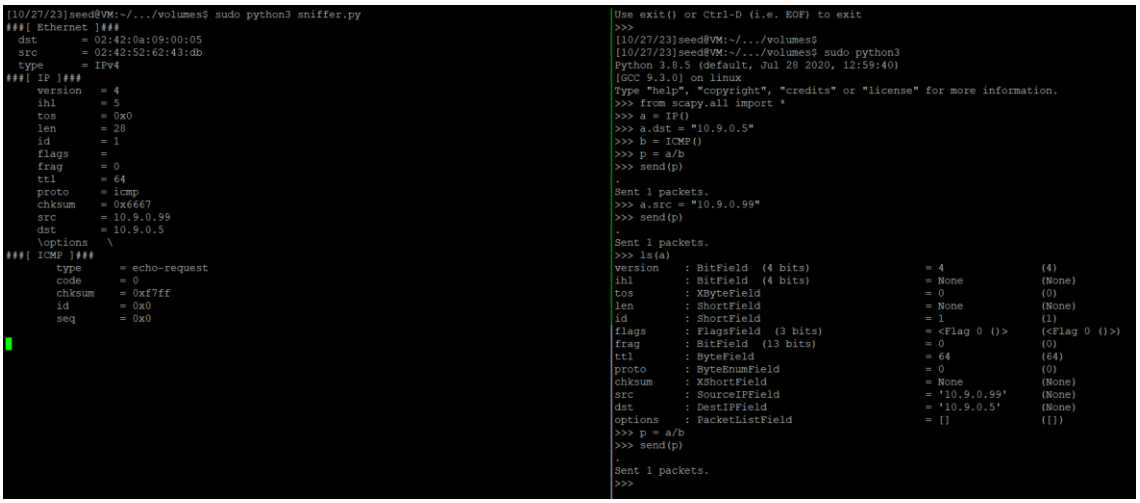
```
# sniffer.py
#!/usr/bin/python3
from scapy.all import *
def print_pkt(pkt):
 pkt.show()
target_subnet = "125.212.128.0/17"
filter_str = f"net {target_subnet}"
pkt = sniff(filter=filter_str, prn=print_pkt, count=5)
```

Task 1.2



The echo-request packet was captured which showed the source IP address is 10.9.0.1 and destination IP address is 10.9.0.5.

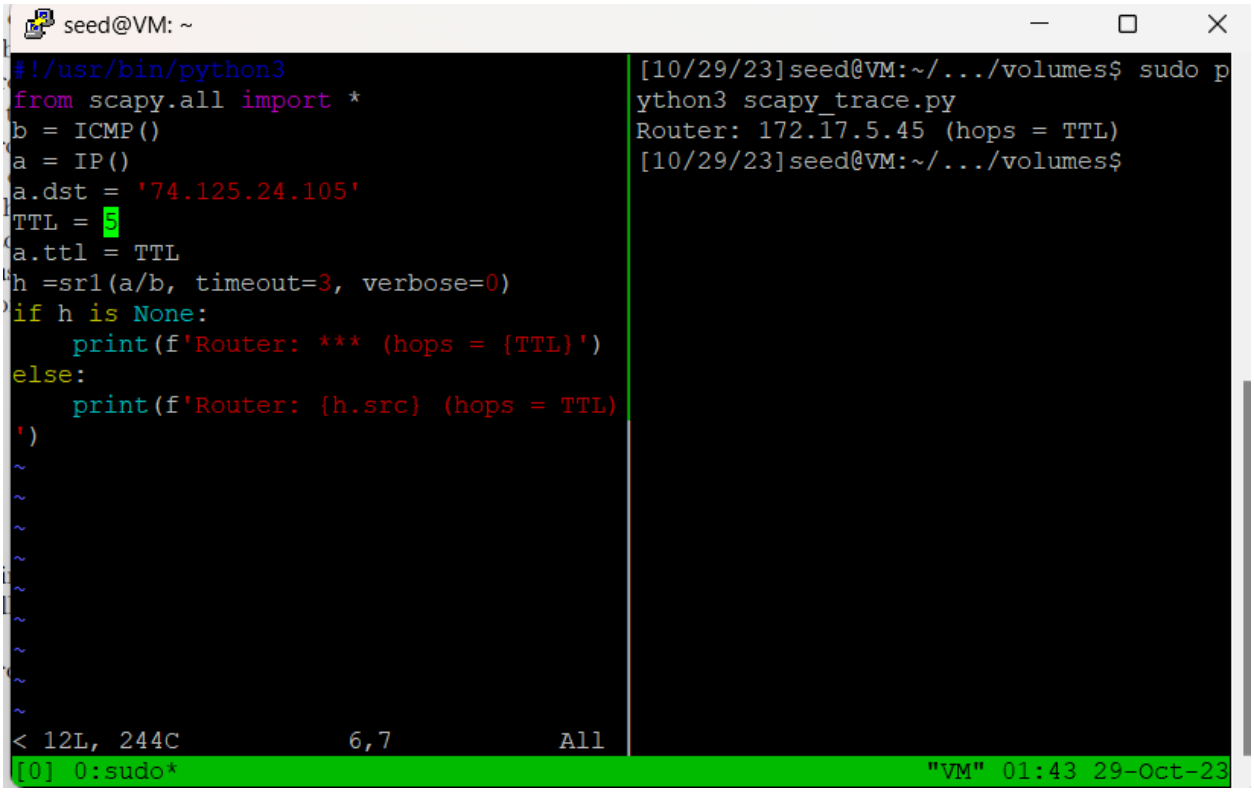
Now, if I changed the source IP address to 10.9.0.99 (a.dst = “10.9.0.99”) then the echo-request packet would also display the source IP address is it.



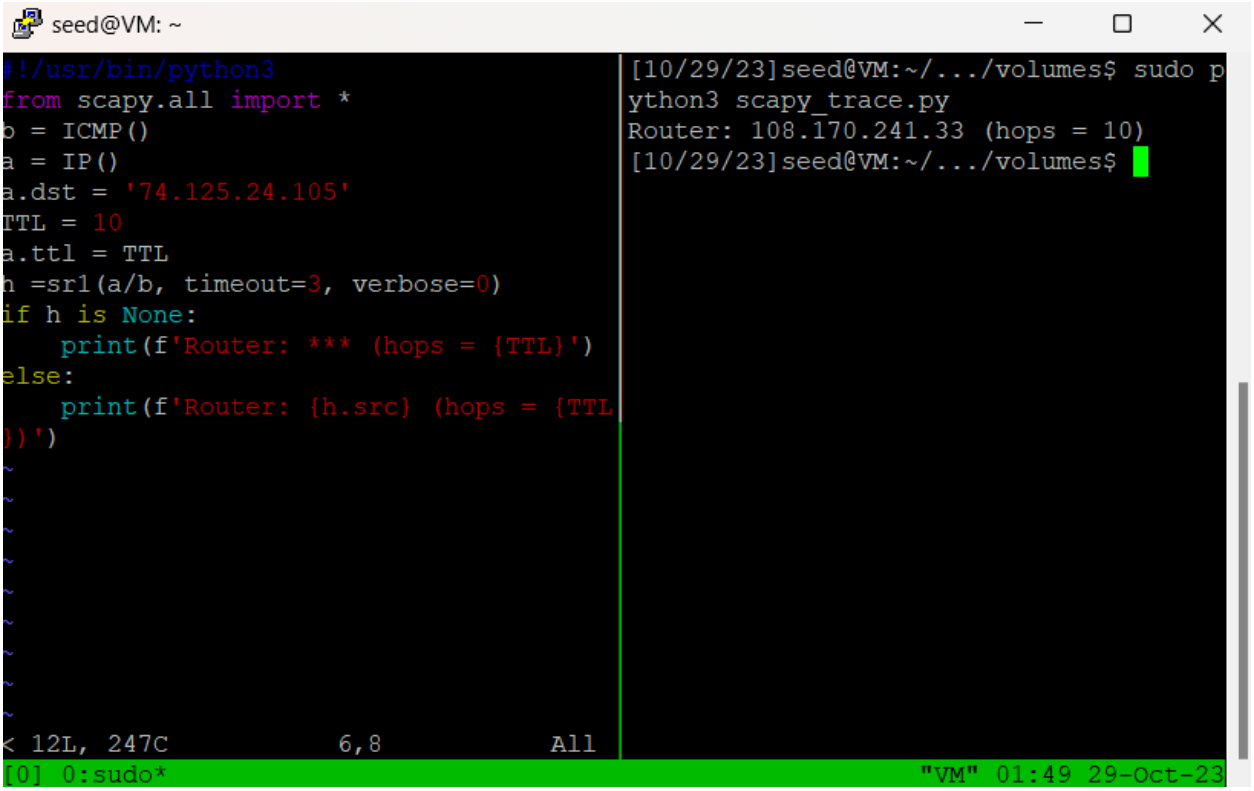
Task 1.3

When TTL = 1, the router IP address is 10.0.2.2

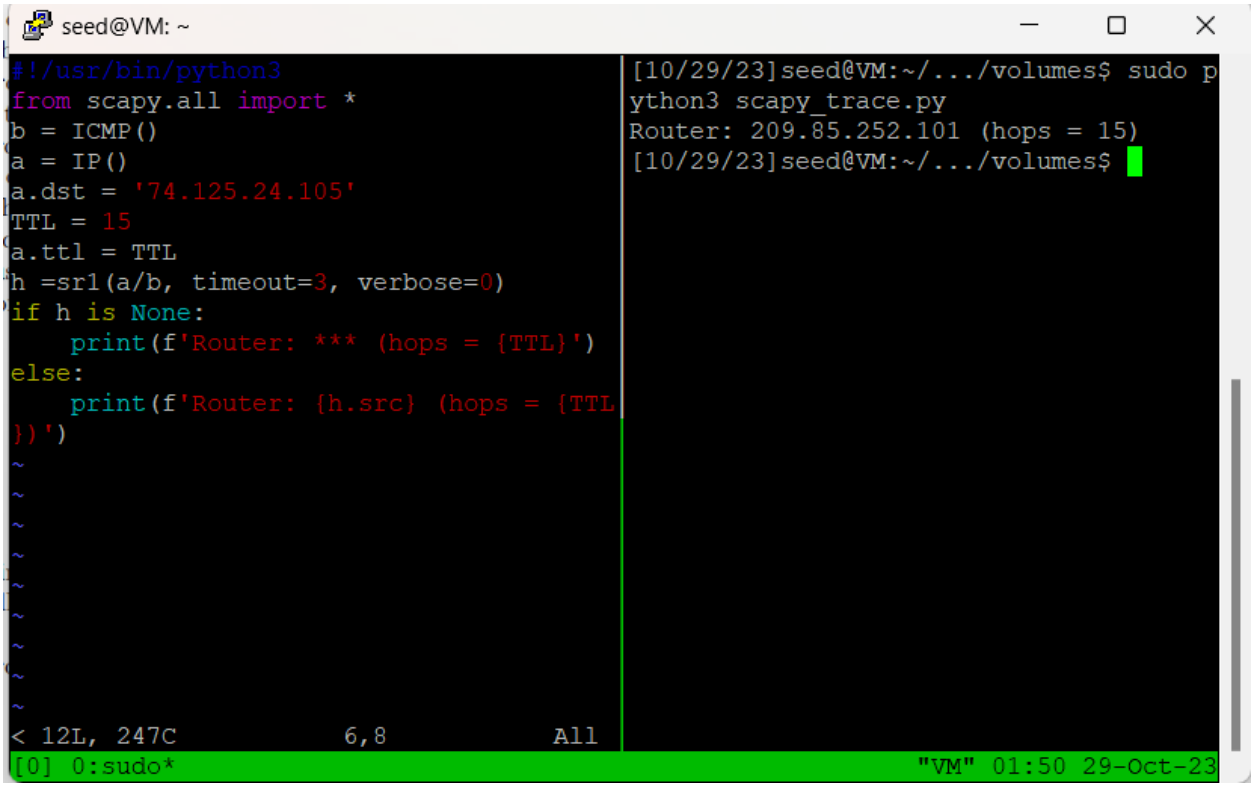
When TTL = 5, the router IP address is 172.17.5.45



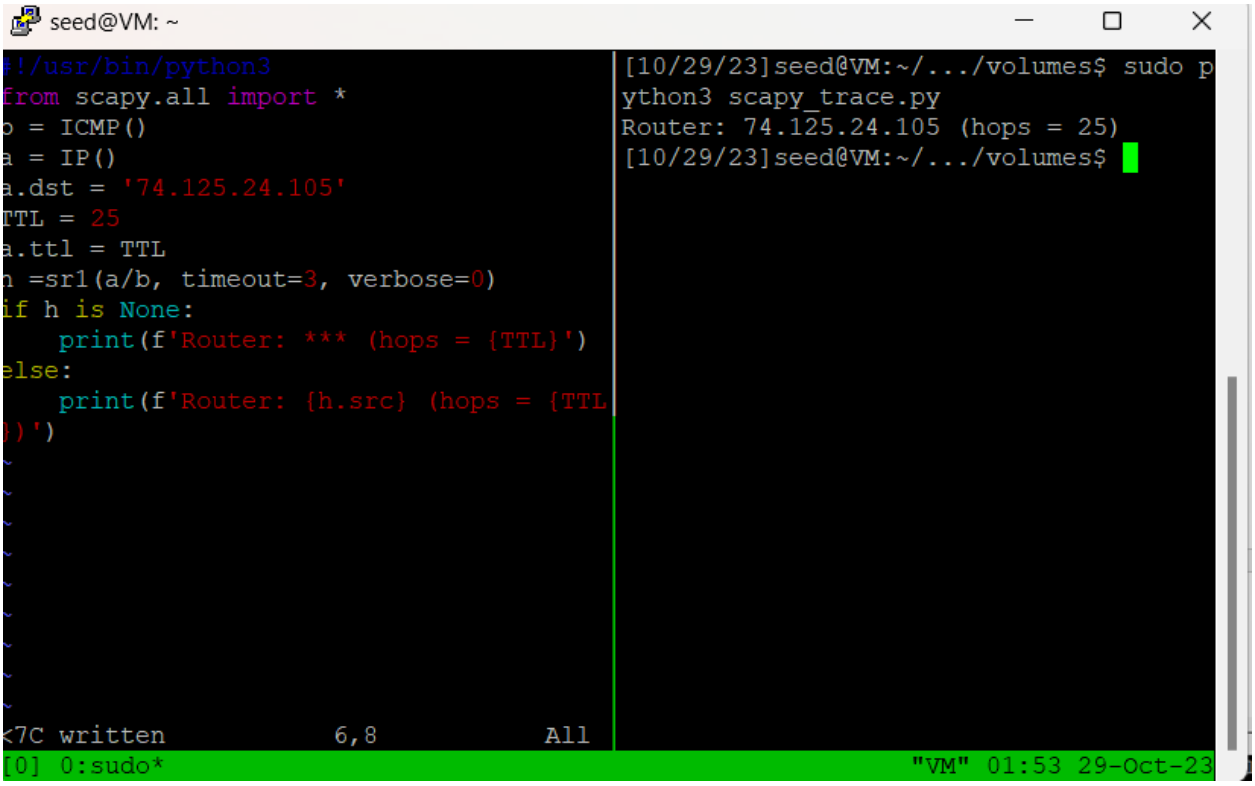
When TTL = 10, the router IP address is 108.170.241.33



When TTL = 15, the router IP address is 209.85.252.101

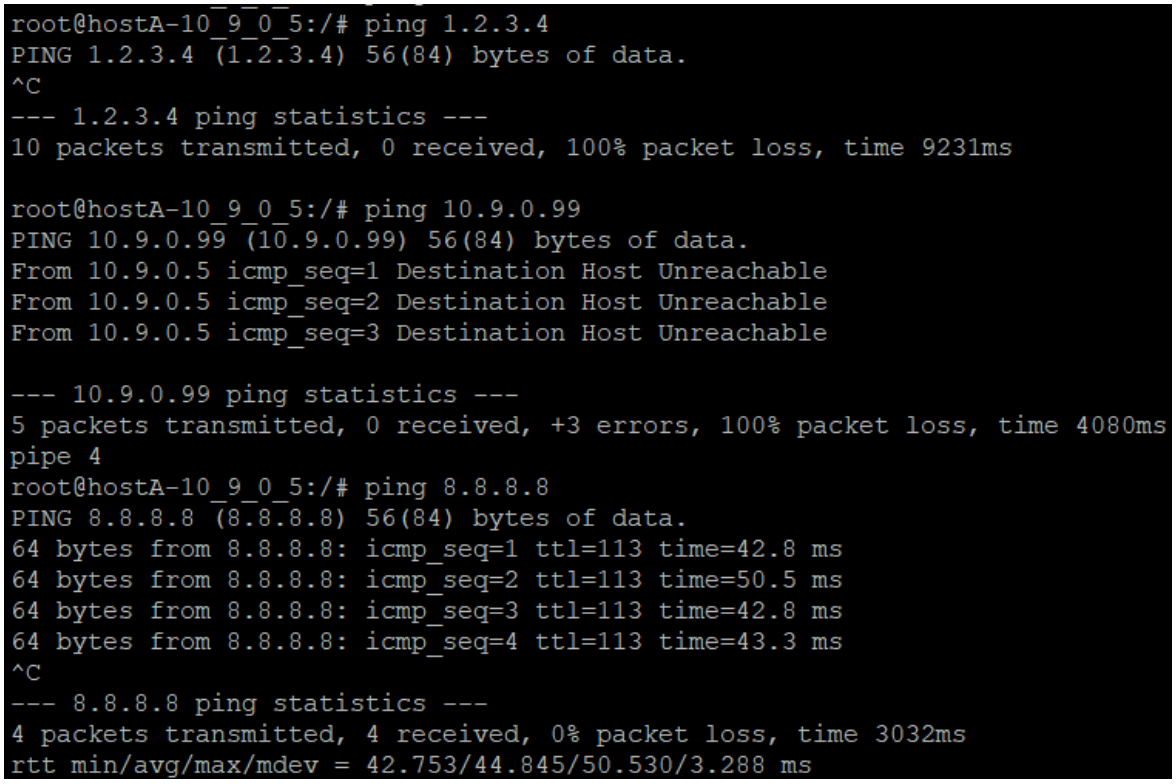


When TTL = 25, the router IP address is 74.125.24.105



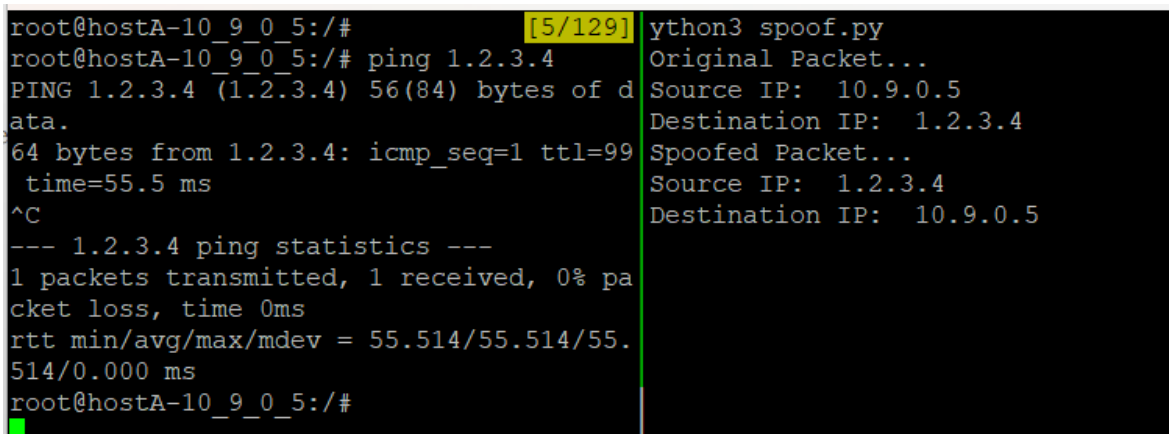
So after about 25 hops, the ICMP packet made it way to the destination hop - 74.125.24.105.

Task 1.4

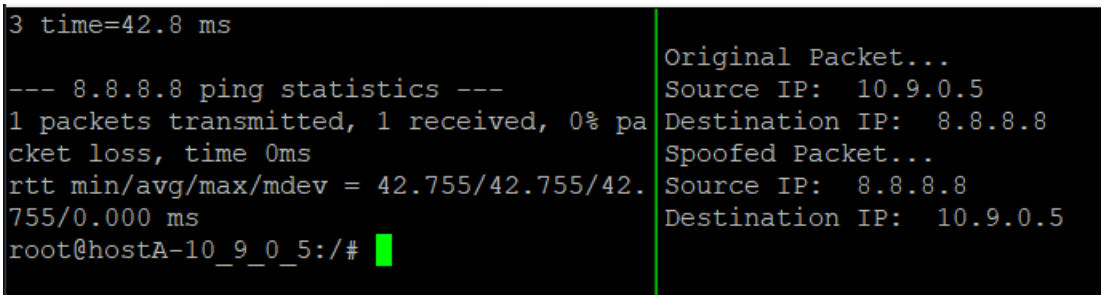


Before turning on the spoof.py, host A can not ping to the 1.2.3.4 and 10.9.0.99 because the IP address did not exist in this LAN or on the Internet.

Now, the spoof.py would be turned on to capture those ping packets from host A and send them a reply packet from the attacker but with the destination IP address.



Host A now would the receive the request back from the attacker pretending to be the destination IP address. And so as for the 8.8.8.8



However, the 10.9.0.99 can not be spoofed it was because the packet did not go to the Internet which the attacker can sniff it to spoof.