Task 1.1: Sniffing Packets

Task 1.1A

Run as root:

```
[01/23/25]seed@VM:~/.../volumes$ sudo ./sniffer.py
###[ Ethernet ]###
  dst       = 52:55:0a:00:02:02
  src       = 08:00:27:33:45:40
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 17202
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x7d88
     src       = 10.0.2.15
     dst       = 142.251.222.228
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x3979
        id        = 0x3
        seq       = 0x1
###[ Raw ]###
           load      = '\xed\xfd\x91g\x00\x00\x00\x00{J\x05\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\
x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01234567'
```

Compare to run as user seed:

```
[01/23/25]seed@VM:~/.../volumes$ who
seed     :0           2025-01-23 01:57 (:0)
[01/23/25]seed@VM:~/.../volumes$ sniffer.py
Traceback (most recent call last):
  File "./sniffer.py", line 7, in <module>
    pkt = sniff(iface=['br-31c3436371e3','enp0s3'], filter='icmp', 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 894, in _run
    sniff_sockets.update(
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 895, in <genexpr>
    (L2socket(type=ETH_P_ALL, iface=ifname, *arg, **karg),
  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
[01/23/25]seed@VM:~/.../volumes$ ./sniffer.py
Traceback (most recent call last):
  File "./sniffer.py", line 7, in <module>
    pkt = sniff(iface=['br-31c3436371e3','enp0s3'], filter='icmp', 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 894, in _run
    sniff_sockets.update(
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 895, in <genexpr>
    (L2socket(type=ETH_P_ALL, iface=ifname, *arg, **karg),
  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
[01/23/25]seed@VM:~/.../volumes$ ls -lh
total 8.0K
-rw-rw-r-- 1 seed seed 140 Jan 23 02:29 mycode.py
-rwxrwxr-x 1 seed seed 159 Jan 23 03:29 sniffer.py
[01/23/25]seed@VM:~/.../volumes$
```

จาก Error: PermissionError: [Errno 1] Operation not permitted คือ แม้ว่าจะเปลี่ยน permission ไฟล์ให้ execute ได้ด้วย chmod แล้ว แต่ไม่สามารถ execute ไฟล์ได้ เนื่องจาก sniff ต้องการสิทธิ์ระดับ root เท่านั้น

Task 1.1B

• Capture only the ICMP packet

```
>>> pkt = sniff(iface="enp0s3", filter="icmp", count=2)
>>> pkt.show()
0000 Ether / IP / ICMP 10.0.2.15 > 8.8.8.8 echo-request 0 / Raw
0001 Ether / IP / ICMP 8.8.8.8 > 10.0.2.15 echo-reply 0 / Raw
>>> pkt[0].summary()
'Ether / IP / ICMP 10.0.2.15 > 8.8.8.8 echo-request 0 / Raw'
>>>
>>> pkt.summary()
Ether / IP / ICMP 10.0.2.15 > 8.8.8.8 echo-request 0 / Raw
Ether / IP / ICMP 8.8.8.8 > 10.0.2.15 echo-reply 0 / Raw
>>>
```

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

- ฝั่ง action

```
[01/23/25]seed@VM:~/.../volumes$ telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
aab885383d92 login: ^CConnection closed by foreign host.
[01/23/25]seed@VM:~/.../volumes$
```

- ฝั่ง sniffing

```
>>> pkt = sniff(iface=['br-31c3436371e3','enp0s3'], filter='host 10.9.0.1 and tcp port 23', count = 5)
>>> pkt.show()
0000 Ether / IP / TCP 10.9.0.1:44808 > 10.9.0.6:telnet S
0001 Ether / IP / TCP 10.9.0.6:telnet > 10.9.0.1:44808 SA
0002 Ether / IP / TCP 10.9.0.1:44808 > 10.9.0.6:telnet A
0003 Ether / IP / TCP 10.9.0.1:44808 > 10.9.0.6:telnet PA / Raw
0004 Ether / IP / TCP 10.9.0.6:telnet > 10.9.0.1:44808 A
>>>
```

• Capture packet(s) 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.

- ฝั่ง ping

```
[01/23/25]seed@VM:~/.../volumes$ ping 128.230.0.1
PING 128.230.0.1 (128.230.0.1) 56(84) bytes of data.
64 bytes from 128.230.0.1: icmp_seq=1 ttl=255 time=291 ms
64 bytes from 128.230.0.1: icmp_seq=2 ttl=255 time=291 ms
64 bytes from 128.230.0.1: icmp_seq=3 ttl=255 time=291 ms
64 bytes from 128.230.0.1: icmp_seq=4 ttl=255 time=290 ms
^C
--- 128.230.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 289.971/290.920/291.325/0.553 ms
[01/23/25]seed@VM:~/.../volumes$
```

- ฝั่ง sniffing

```
>>> pkt = sniff(iface=['br-31c3436371e3','enp0s3'], filter='net 128.230.0.0/16', count=5)
>>> pkt.show()
0000 Ether / IP / ICMP 10.0.2.15 > 128.230.0.1 echo-request 0 / Raw
0001 Ether / IP / ICMP 128.230.0.1 > 10.0.2.15 echo-reply 0 / Raw
0002 Ether / IP / ICMP 10.0.2.15 > 128.230.0.1 echo-request 0 / Raw
0003 Ether / IP / ICMP 128.230.0.1 > 10.0.2.15 echo-reply 0 / Raw
0004 Ether / IP / ICMP 10.0.2.15 > 128.230.0.1 echo-request 0 / Raw
>>>
```

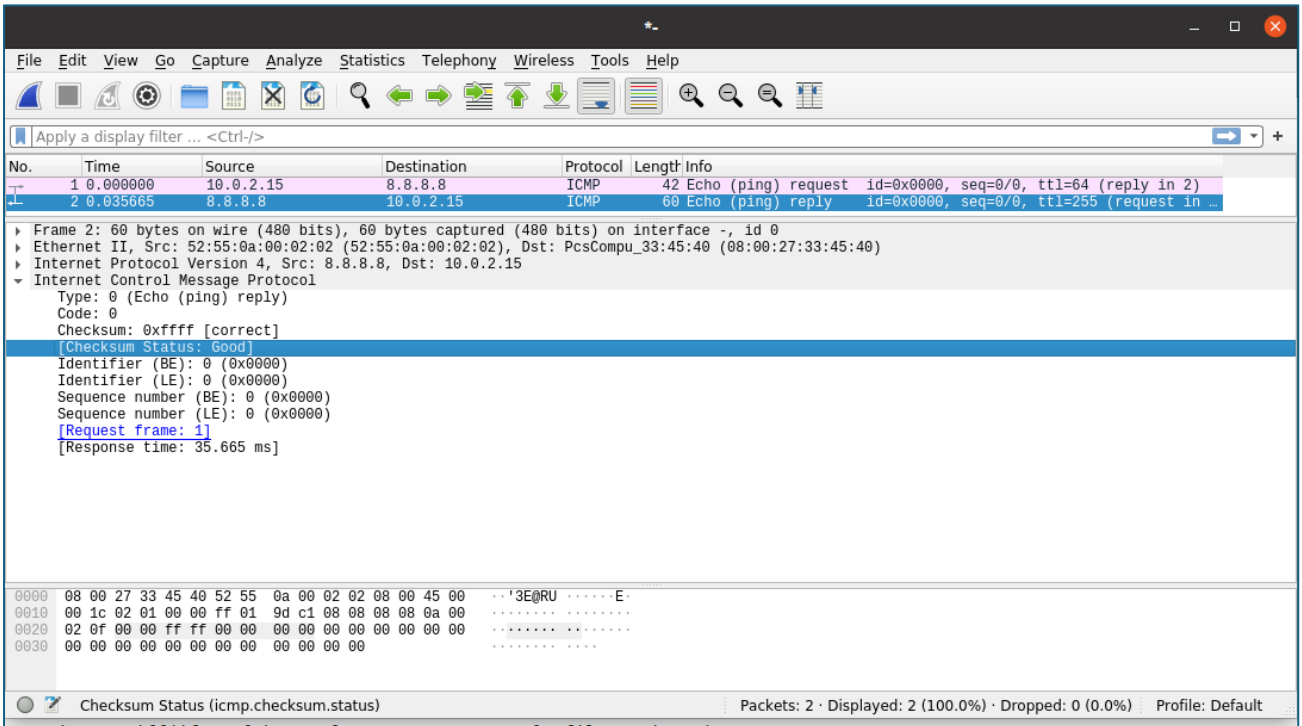
Task 1.2: Spoofing ICMP Packets

Demonstrate that you can spoof an ICMP echo request packet with an arbitrary source IP address.

- ฝั่ง spoofing

```
[01/24/25]seed@VM:~/.../Scapy$ cat icmp_spoof.py
#!/usr/bin/python3
from scapy.all import *

print("SENDING SPOOFED ICMP PACKET.........")
ip = IP(dst="8.8.8.8")
icmp = ICMP()
pkt = ip/icmp
pkt.show()
send(pkt,verbose=0)

[01/24/25]seed@VM:~/.../Scapy$ sudo ./icmp_spoof.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       = 10.0.2.15
  dst       = 8.8.8.8
  \options   \
###[ ICMP ]###
     type      = echo-request
     code      = 0
     chksum    = None
     id        = 0x0
     seq       = 0x0

[01/24/25]seed@VM:~/.../Scapy$
```

- ฝั่ง sniffing




```
>>> pkt = sniff(iface=['br-9c2f80916c49','enp0s3'], filter='icmp')

^C>>> pkt.show()
0000 Ether / IP / ICMP 10.0.2.15 > 8.8.8.8 echo-request 0
0001 Ether / IP / ICMP 8.8.8.8 > 10.0.2.15 echo-reply 0 / Padding
>>> wireshark(pkt)
/usr/lib/python3.8/subprocess.py:942: ResourceWarning: subprocess 7971 is still running
  _warn("subprocess %s is still running" % self.pid,
ResourceWarning: Enable tracemalloc to get the object allocation traceback
>>> QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

Task 1.3: Traceroute

Command for sending packet

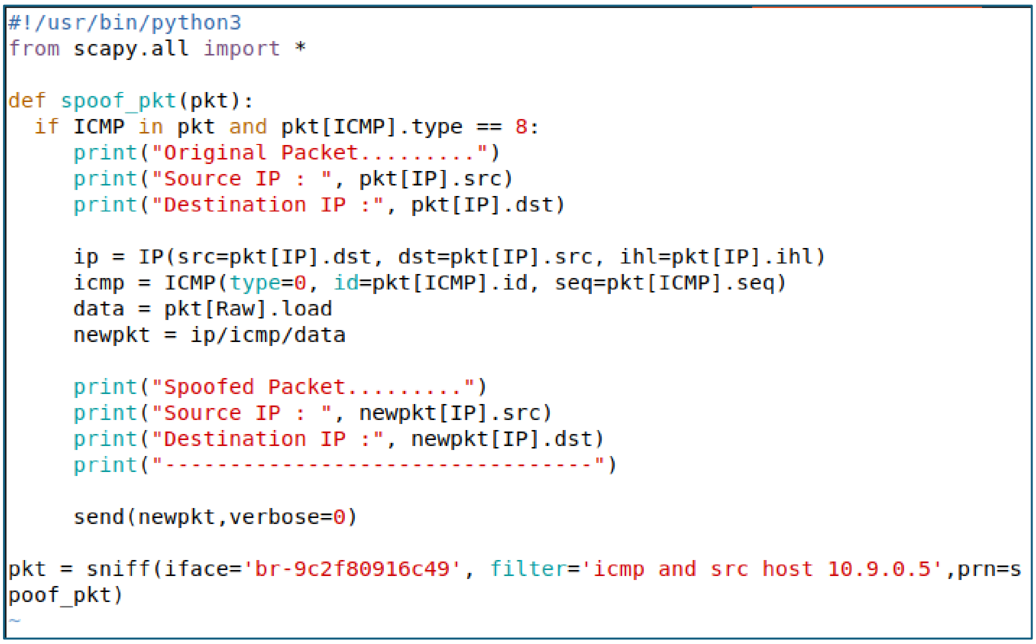
```
#!/usr/bin/env python3
from scapy.all import *

ip = IP()
ip.dst = sys.argv[1]
ttl = 1

while True:
        ip.ttl = ttl
        protocol = ICMP()
        packet = ip/protocol
        resp = sr1(packet, timeout=2, verbose=0)

        if resp is None:
                print("TTL =",ttl)
                print("No reply.")
        elif resp[ICMP].type == 0:
                print("TTL =",ttl)
                print("%d hops away: " % (ip.ttl), resp[IP].src)
                print("Arrived at destination ", resp[IP].src)
                break
        else:
                print("TTL =",ttl)
                print("%d hops away: " % (ip.ttl), resp[IP].src)
        ttl += 1

        if ttl > 30:
                break
~
```

```
[01/26/25]seed@VM:~/.../volumes$ sudo ./ttl2.py 8.8.8.8
TTL = 1
1 hops away:  8.8.8.8
Arrived at destination  8.8.8.8
[01/26/25]seed@VM:~/.../volumes$ sudo ./ttl2.py 10.9.0.6
TTL = 1
1 hops away:  10.9.0.6
Arrived at destination  10.9.0.6
[01/26/25]seed@VM:~/.../volumes$ sudo ./ttl2.py notion.com
TTL = 1
1 hops away:  208.103.161.2
Arrived at destination  208.103.161.2
[01/26/25]seed@VM:~/.../volumes$
```

จากการทดสอบ ping ผลมีค่า TTL =1 แล้วได้รับ echo-reply ทันที เนื่องจาก router รู้จัก IP เหล่านี้แล้ว

Task 1.4: Sniffing and-then Spoofing

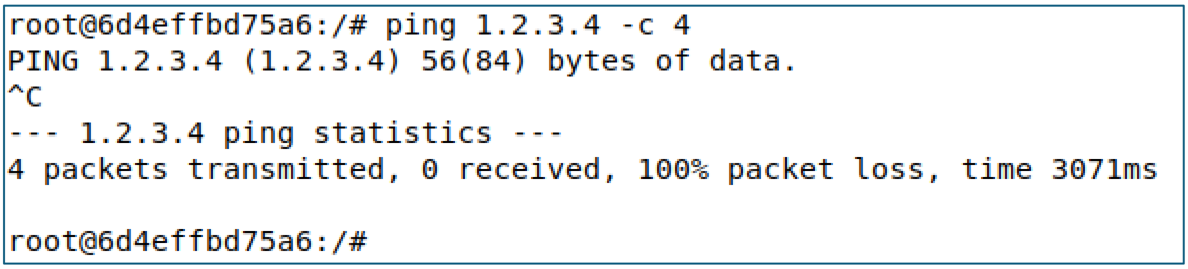
Code สำหรับ sniff แล้ว spoof กลับไป หลอกว่าเป็นเครื่องปลายทาง



```
#!/usr/bin/python3
from scapy.all import *

def spoof_pkt(pkt):
  if ICMP in pkt and pkt[ICMP].type == 8:
     print("Original Packet.........")
     print("Source IP : ", pkt[IP].src)
     print("Destination IP :", pkt[IP].dst)

     ip = IP(src=pkt[IP].dst, dst=pkt[IP].src, ihl=pkt[IP].ihl)
     icmp = ICMP(type=0, id=pkt[ICMP].id, seq=pkt[ICMP].seq)
     data = pkt[Raw].load
     newpkt = ip/icmp/data

     print("Spoofed Packet.........")
     print("Source IP : ", newpkt[IP].src)
     print("Destination IP :", newpkt[IP].dst)
     print("---------------------------------")

     send(newpkt,verbose=0)

pkt = sniff(iface='br-9c2f80916c49', filter='icmp and src host 10.9.0.5',prn=s
poof_pkt)
~
```

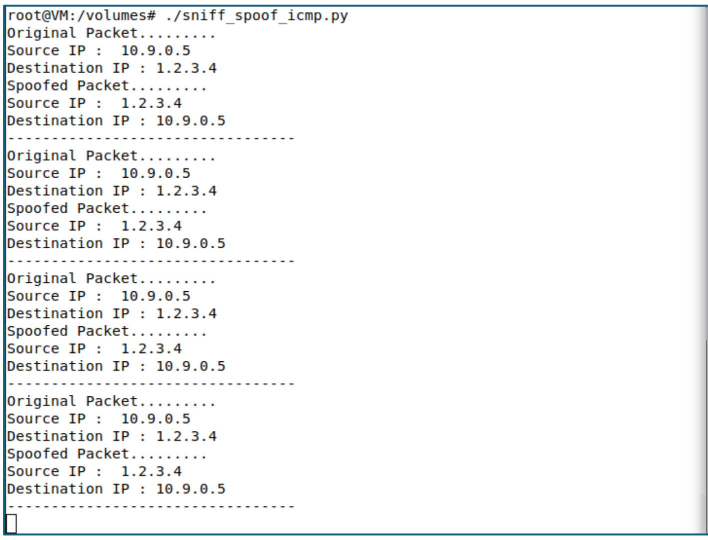
- ping 1.2.3.4 # a non-existing host on the Internet

  - ฝั่งส่ง ping โดยไม่โดน spoofing



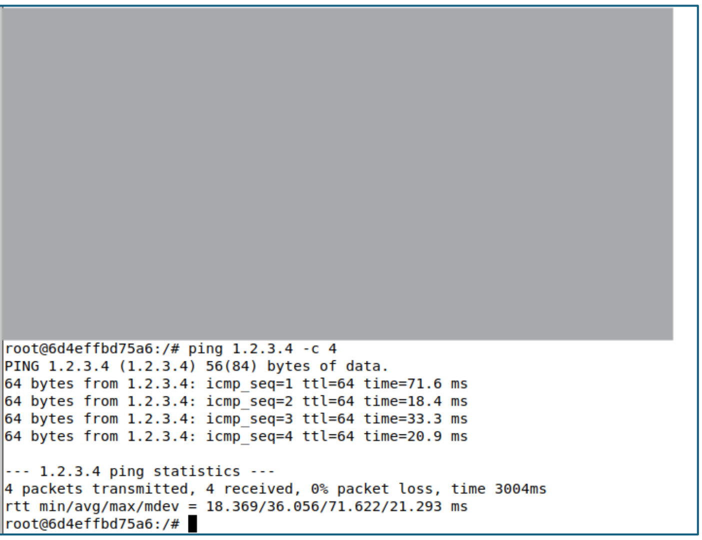
```
root@6d4effbd75a6:/# ping 1.2.3.4 -c 4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
^C
--- 1.2.3.4 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3071ms

root@6d4effbd75a6:/#
```

  - ฝั่ง spoofing



```
root@VM:/volumes# ./sniff_spoof_icmp.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
---------------------------------
```

  - ฝั่ง ping



```
root@6d4effbd75a6:/# ping 1.2.3.4 -c 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=71.6 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=18.4 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=33.3 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=20.9 ms

--- 1.2.3.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 18.369/36.056/71.622/21.293 ms
root@6d4effbd75a6:/#
```

ฝั่ง spoofing ประพฤติเป็น 1.2.3.4 ที่ไม่มีจริง ทางฝั่ง ping จึงได้รับ packet echo-reply สำเร็จ

- ping 10.9.0.99 # a non-existing host on the LAN

- ฝั่งส่ง ping โดยไม่โดน spoofing

```
root@6d4effbd75a6:/# ping 10.9.0.99 -c 4
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable

--- 10.9.0.99 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3050ms
pipe 4
root@6d4effbd75a6:/#
```

- ฝั่ง spoofing

```
^Croot@VM:/volumes#
root@VM:/volumes#
root@VM:/volumes# ./sniff_spoof_icmp.py
```

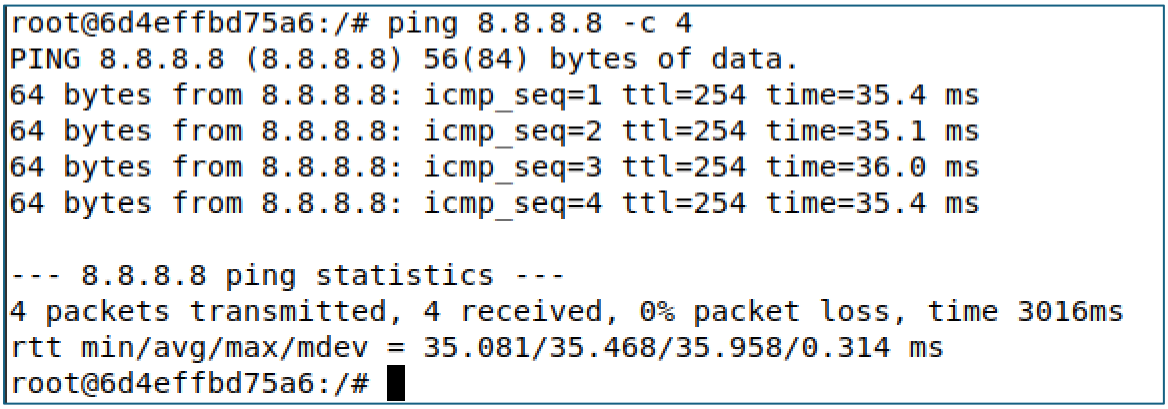
- ฝั่ง ping

```
root@6d4effbd75a6:/# ping 10.9.0.99 -c 4
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable

--- 10.9.0.99 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3067ms
pipe 4
root@6d4effbd75a6:/#
```

เนื่องจากในวง local subnet จะมี ARP ช่วย map MAC เครื่องกับ IP ในวง แต่ ARP ไม่รู้จัก 10.9.0.99 จึงตอบกลับมาเป็น Unreachable Host ดังนั้นจึงไม่โดนผลกระทบจาก spoofing

- ping 8.8.8.8 # an existing host on the Internet

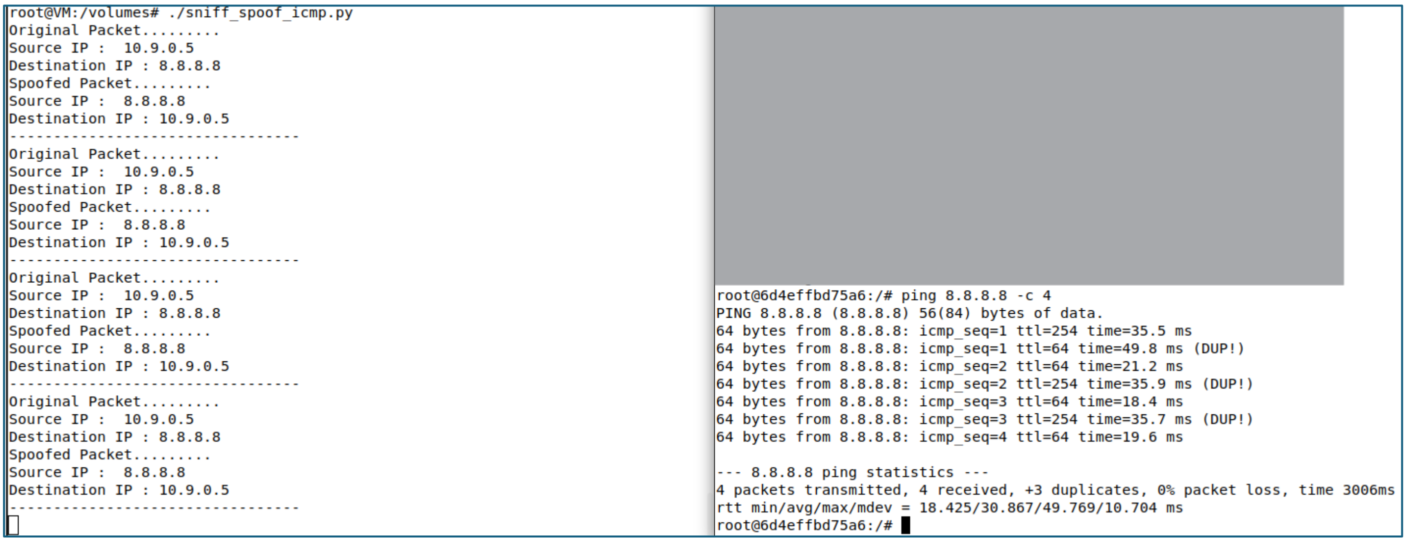
- ฝั่งส่ง ping โดยไม่โดน spoofing



```
root@6d4effbd75a6:/# ping 8.8.8.8 -c 4
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=254 time=35.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=254 time=35.1 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=254 time=36.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=254 time=35.4 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3016ms
rtt min/avg/max/mdev = 35.081/35.468/35.958/0.314 ms
root@6d4effbd75a6:/# 
```

- ฝั่ง spoofing

- ฝั่ง ping



```
root@VM:/volumes# ./sniff_spoof_icmp.py
Original Packet.........
Source IP :  10.9.0.5
Destination IP : 8.8.8.8
Spoofed Packet.........
Source IP :  8.8.8.8
Destination IP : 10.9.0.5
---------------------------------
Original Packet.........
Source IP :  10.9.0.5
Destination IP : 8.8.8.8
Spoofed Packet.........
Source IP :  8.8.8.8
Destination IP : 10.9.0.5
---------------------------------
Original Packet.........
Source IP :  10.9.0.5
Destination IP : 8.8.8.8
Spoofed Packet.........
Source IP :  8.8.8.8
Destination IP : 10.9.0.5
---------------------------------
Original Packet.........
Source IP :  10.9.0.5
Destination IP : 8.8.8.8
Spoofed Packet.........
Source IP :  8.8.8.8
Destination IP : 10.9.0.5
---------------------------------
```

[redacted]

```
root@6d4effbd75a6:/# ping 8.8.8.8 -c 4
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=254 time=35.5 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=64 time=49.8 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=21.2 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=254 time=35.9 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=18.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=254 time=35.7 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=19.6 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, +3 duplicates, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 18.425/30.867/49.769/10.704 ms
root@6d4effbd75a6:/# 
```

ทางฝั่ง ping ได้รับ echo-reply packet ที่เขียนว่า DUP! เพราะว่าเกิด packet ซ้ำ ซึ่งมาจากทั้งเครื่อง spoofing และต้นทางจริง (8.8.8.8) echo-reply ทั้งคู่