



**slington college**  
(इरिलिङ्टन कलेज)

**Module Code & Module Title**

**CC5009NI Cyber Security in Computing**

**Assessment Weightage & Type**

**60% Group Coursework 02**

**Year and Semester**

**2024 -25 Autumn Semester**

**Student Name: Bibhuti Sigdel London Met ID: 23047458**

**Student Name: Shraddha Budhathoki London Met ID: 23047397**

**Student Name: Gaurav Pratap Malla London Met ID: 23047411**

**Student Name: Piyush Shrestha London Met ID: 23047449**

**Assignment Due Date: Monday, May 12, 2025**

**Assignment Submission Date: Monday, May 12, 2025**

**Word Count (Where Required):3449**

*I confirm that I understand my coursework needs to be submitted online via MST under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

# Similarity Report

23047458 Bibhuti Sigdel.docx

Introduction

Cyberattack

A cyberattack refers to an attempt by hackers or malicious actors to exploit vulnerabilities in a system to steal data, disrupt services, or cause damage. Attacks can take various forms, such as malware infections, denial-of-service (DoS) attacks, phishing scams, and hacking attempts (fortinet, 2025).

DOS (Denial of service) attack

A Denial-of-Service (DoS) attack is a cyberattack in which a malicious actor attempts to make a computer or device inaccessible to its intended users by disrupting its normal operations. These attacks typically work by overwhelming the targeted system with excessive requests, preventing it from processing legitimate traffic and causing service denial for other users. A key characteristic of a DoS attack is that it originates

Page 7 of 27

4060 words

176%

geeks, 2024)

Filters

[Back to Similarity Report](#)

12% Overall Similarity

35 Matching Text Blocks

Compare submissions against ?

Select at least one source type to check for similarity.

☒ Submitted Works

☒ Internet content

☒ Publications

Exclusion filters ?

☒ Exclude bibliography

☒ Exclude quoted text

☒ Exclude cited text

☒ Exclude small matches

Cancel

Apply Filters

## Table of Contents

<b>1. Introduction.....</b>	<b>1</b>
<b>1.1. Cyberattack.....</b>	<b>1</b>
<b>1.2. DOS (Denial of service) attack .....</b>	<b>1</b>
<b>1.3. Motivation For DOS attack .....</b>	<b>2</b>
<b>1.4. Key Terminologies .....</b>	<b>3</b>
<b>1.5. Aims and Objectives .....</b>	<b>3</b>
<b>1.6. Report structure .....</b>	<b>4</b>
<b>2. Background .....</b>	<b>5</b>
<b>2.1. The Origin of Denial of service (DOS) attacks .....</b>	<b>5</b>
<b>2.2. Types of DOS attacks .....</b>	<b>5</b>
<b>2.3. DOS attack in the past year .....</b>	<b>7</b>
<b>3. Demonstration.....</b>	<b>9</b>
<b>3.1. Operating System Used .....</b>	<b>9</b>
<b>3.2. Tools Used .....</b>	<b>9</b>
<b>3.3. Lab environment configuration .....</b>	<b>9</b>
<b>3.4. Port reconnaissance .....</b>	<b>11</b>
<b>3.5. Port Scanning with Hping3 .....</b>	<b>12</b>
<b>3.6. SYN Flood attack demonstration .....</b>	<b>13</b>
<b>3.7. Monitoring wireshark to observe spoofed source Ips during SYN Flood attack...17</b>	
<b>4. Mitigation Strategy and Evaluation .....</b>	<b>18</b>
<b>4.1. Disabling ICMP Echo Requests Temporarily (Blocking Ping) .....</b>	<b>18</b>
<b>4.2. Rate Limiting ICMP Requests Using iptables .....</b>	<b>20</b>
<b>4.3. Enabling SYN Cookies .....</b>	<b>22</b>
<b>4.4. Blocking a Malicious IP .....</b>	<b>22</b>
<b>4.5. Limiting Number of Connections Per IP .....</b>	<b>23</b>

4.6.	Using Fail2Ban to Ban Repeat Offenders .....	23
4.7.	Dropping Fragmented Packets .....	24
4.8.	Using NGINX as a Reverse Proxy with Rate Limiting .....	24
5.	Evaluation and Analysis .....	25
5.1.	Effectiveness of Implemented Techniques .....	25
5.2.	Comparison and Practicality.....	25
5.3.	Advantages and Disadvantages.....	25
5.4.	Real-World Applicability.....	26
6.	Conclusion .....	27
7.	Bibliography .....	28

## Table of Figures

Figure 1: Different types of cyberattacks .....	1
Figure 2:DOS attack .....	2
Figure 3: Working mechanism of botnet.....	7
Figure 4: Checking ip .....	9
Figure 5: Checking connection by doing ping .....	10
Figure 6: Nmap port scan to identify open ports and active devices.....	11
Figure 7: Scanning port using hping3 .....	12
Figure 8: Scan ports using random spoofed source ip.....	13
Figure 9: SYN flood attack with spoofed IP using hping3 .....	14
Figure 10: Simulating a DoS attack by flooding port 135 with large packets from random IPs...	15
Figure 11: Sending a large data payload (1MB) to 192.168.56.103 using hping3 .....	16
Figure 12: Monitoring Wireshark to observe spoofed source IPs during a SYN flood attack .....	17
Figure 13:Disabling ICMP Echo response.....	19
Figure 14: Ping was unsuccessful .....	20
Figure 15: Limiting ICMP request .....	21
Figure 16: Ping from two machine and packets were dropped.....	21
Figure 17: Enabling SYN cookies .....	22
Figure 18: Blocking malicious IP .....	23

## **Abstract**

This report presents a comprehensive analysis and practical demonstration of Denial-of-Service (DoS) attacks, with a particular focus on SYN flood and ICMP flood techniques. The research explores the background and motivations behind DoS attacks, common types, and real-world incidents that highlight their disruptive potential. A practical lab-based simulation was conducted using Kali Linux and Metasploitable to perform a SYN flood attack using the hping3 tool. The impact of the attack was observed through system resource monitoring and network traffic analysis. Following the attack, various mitigation strategies were implemented and evaluated, including ICMP blocking, rate limiting with iptables, enabling SYN cookies, using Fail2Ban, limiting concurrent connections, and configuring NGINX as a reverse proxy. Each method was tested for its effectiveness and practicality in preventing or reducing the attack's impact. The report concludes that while basic mitigation techniques are effective against single-source DoS attacks, a layered security approach combining multiple tools and methods is essential for robust defense against advanced and distributed threats.

## 1. Introduction

### 1.1. Cyberattack

A cyberattack refers to an attempt by hackers or malicious actors to exploit vulnerabilities in a system to steal data, disrupt services, or cause damage. Attacks can take various forms, such as malware infections, denial-of-service (DoS) attacks, phishing scams, and hacking attempts (fortinet, 2025).

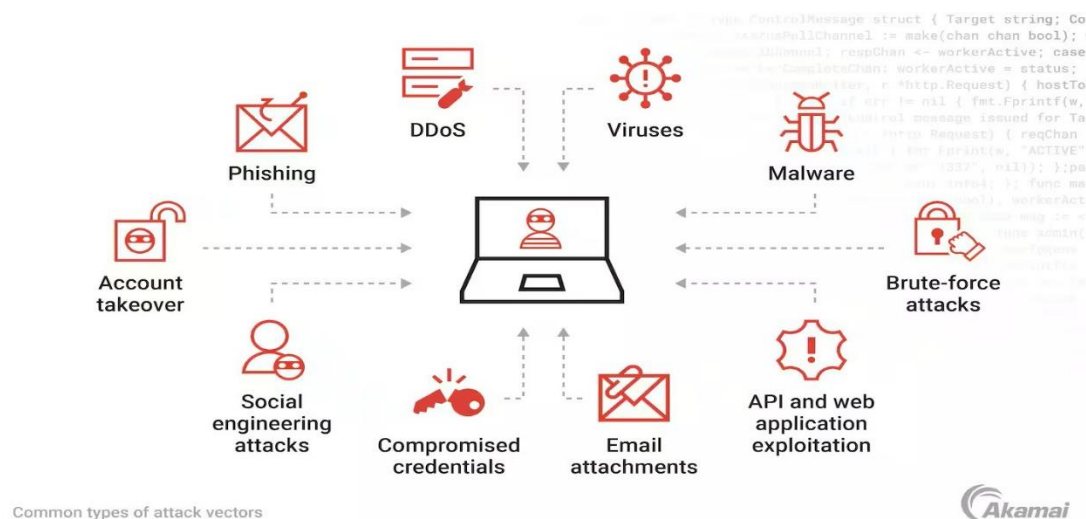


Figure 1: Different types of cyberattacks

### 1.2. DOS (Denial of service) attack

A Denial-of-Service (DoS) attack is a cyberattack in which a malicious actor attempts to make a computer or device inaccessible to its intended users by disrupting its normal operations. These attacks typically work by overwhelming the targeted system with excessive requests, preventing it from processing legitimate traffic and causing service denial for other users. A key characteristic of a DoS attack is that it originates from a single computer (geeksforgeeks, 2024)

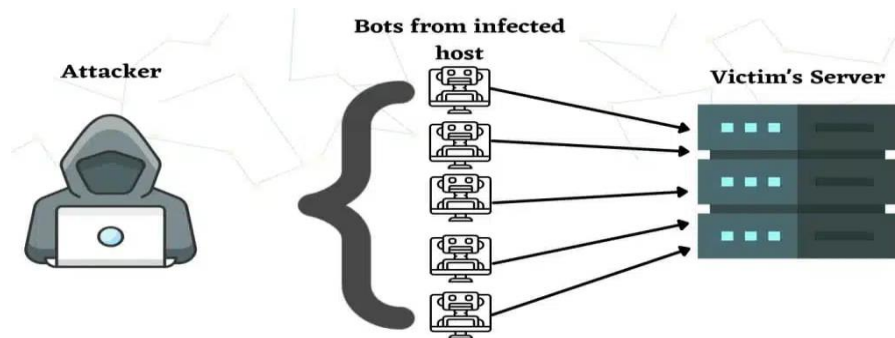


Figure 2: DOS attack

### 1.3. Motivation For DOS attack

Denial of Service (DoS) attacks are carried out for various reasons, ranging from financial motives to personal vendettas. The key motivations behind such attacks include:

- **Financial Gain:** Attackers often execute Ransom DoS attacks, where they demand payment in exchange for stopping the attack. Businesses, especially those heavily reliant on online services, may be forced to pay to restore functionality.
- **Hactivism:** Cyber activists, use DoS attacks as a tool for protest against governments, corporations, or organizations. These attacks aim to disrupt services to raise awareness about political, social, or ideological causes.
- **Business Competition:** In highly competitive industries, unethical businesses may hire attackers to launch DoS attacks against rivals. By rendering a competitor's website or service unavailable, attackers can give an unfair advantage to another entity.
- **Revenge or Personal Reasons:** Some attackers target individuals, organizations, or institutions due to personal grudges, past disputes, or dissatisfaction. Former employees or dissatisfied customers may attempt to disrupt a company's operations as an act of retaliation (GarySockerider, 2023).



## 1.4. Key Terminologies

- **DoS (Denial of Service):** A cyberattack aimed at disrupting the availability of a service or system.
- **DDoS (Distributed Denial of Service):** A DoS attack conducted using multiple compromised devices.
- **Botnet:** A botnet is a network of internet-connected devices, including personal computers, servers, mobile phones, and IoT devices, that have been infected with malware and are controlled by an attacker, often without the owners' knowledge (geeksforgeeks, 2024).
- **Amplification Attack:** A method where attackers send small requests that result in large responses from publicly available services.
- **SYN Flood:** A SYN flood, also known as a half-open attack, that overwhelms a server by consuming its available resources, making it inaccessible to legitimate users (Cloudflare, 2025)

## 1.5. Aims and Objectives

### Aim

This coursework aims to research, describe, analyze, and evaluate DoS attacks while demonstrating practical cases and exploring effective mitigation techniques related to information and network security

### Objectives

- Researching different types of DoS attacks and their mechanisms.
- Demonstrating a practical DoS attack scenario.
- Exploring various mitigation techniques.
- Evaluating the strengths and weaknesses of prevention methods.

## 1.6. Report structure

This report is systematically structured to provide a comprehensive analysis of Denial-of-Service (DoS) attacks, covering their mechanisms, impact, and countermeasures. The key sections of the report are as follows:

- **Background:**

Here, the report explores the different types of DoS attacks, the techniques used to launch them, and the damage they can cause to IT systems. It also includes real-world examples to show just how disruptive these attacks can be.

- **Demonstration:**

This section presents a simulated or controlled example of a DoS attack. It explains how the experiment was set up, how the attack was performed, and what results were observed, all while ensuring that ethical standards were maintained.

- **Mitigation Techniques:**

This part looks at different methods for defending against DoS attacks. It covers tools and strategies like firewalls, traffic filtering, rate limiting, and intrusion detection systems. Each method is discussed in terms of how it works and how effective it can be.

- **Evaluation and Analysis:**

The effectiveness of the various mitigation techniques is critically assessed in this section. It looks at the pros and cons of each approach and considers how practical they are in real-world environments.

- **Conclusion:**

The report wraps up by summarizing the main findings, stressing the importance of having strong security measures in place, and offering recommendations for better preparedness and response in the future.

## **2. Background**

### **2.1. The Origin of Denial of service (DOS) attacks**

The first recorded cases of denial-of-service (DoS) attacks date back to the early days of the internet, when a deluge of malicious traffic brought the Panix Internet Service Provider offline in 1996 (smarterMSP, 2025). By overloading a system, server, or network with requests or taking advantage of hardware and software flaws, these attacks aim to interfere with regular operations and prevent authorised users from accessing services. As technology developed, denial-of-service (DoS) attacks changed into Distributed Denial-of-Service (DDoS) attacks, which leverage a number of compromised systems, frequently as part of a botnet, to perform coordinated, large-scale attacks. Attackers, particularly those with little technical expertise, can now perform these tasks more easily because to the availability of automated tools. Modern DoS assaults are usually motivated by more serious goals like political activism, financial extortion, competitive sabotage, or cyber warfare, whereas early attempts were typically motivated by curiosity or the desire for attention. DoS and DDoS attacks have consequently emerged as one of the most pervasive and destructive dangers in the realm of cybersecurity, with the capacity to seriously impair websites, interfere with vital infrastructure, and result in substantial financial and reputational losses. (Raza, 2025)

### **2.2. Types of DOS attacks**

- **ICMP Flood (Ping Flood)**

An ICMP flood, commonly referred to as a ping flood, is a straightforward yet powerful type of DoS attack that targets a system by sending a massive number of ICMP Echo Request (ping) packets. The victim's system is forced to handle and respond to each incoming packet, which drains its CPU power and network bandwidth. When done on a large scale, this flood of traffic can slow down system performance significantly or even crash the system entirely. Despite being one of the oldest DoS attack methods, it still poses a serious threat to systems that lack proper safeguards, although many modern networks now limit or ignore ICMP traffic to reduce vulnerability. (Twingate Team, 2024)

- **SYN Flood**

A **SYN flood** attack takes advantage of the way computers establish a TCP connection. Normally, when a device wants to connect to a server, it goes through a three-step process called the **TCP handshake**. In a SYN flood, the attacker sends a lot of **SYN** packets with fake source addresses but never responds to the server's reply (called a **SYN-ACK**). As a result, the server waits for a response that never comes, leaving open connections in its memory. When too many of these open connections pile up, the server can't handle new, legitimate requests, preventing real users from accessing the server. This type of attack targets weaknesses in the way TCP/IP systems handle connections. (Lutkevich, 2025)

- **HTTP Flood**

An HTTP flood is a type of attack targeting the application layer, where the attacker sends what looks like normal **HTTP GET or POST requests** to a web server. Unlike attacks that rely on massive amounts of traffic, HTTP floods focus on using up the server's resources by making repeated requests that require heavy processing, like fetching data from a database, downloading files, or loading web pages. Since these requests seem like regular user activity, it's harder to spot and stop them with basic network filters. HTTP floods are usually carried out with bots or scripts and often need more advanced protection, like deep packet inspection or specialized application firewalls, to defend against them. (geeksforgeeks, 2025)

- **Teardrop Attack**

A Teardrop attack is a type of Denial-of-Service (DoS) attack that manipulates IP packet fragmentation to destabilize a target system. The attacker sends IP packets with intentionally altered or overlapping fragment offsets, which the receiving system cannot correctly reassemble. This manipulation causes instability, often leading to system crashes or freezes, particularly in older operating systems like Windows 95 and early versions of Linux. The inability to properly handle these malformed packets results in a denial-of-service condition, rendering the system unresponsive and disrupting normal operations. This can lead to significant downtime and potential damage, especially if the affected system is critical to business operations. (Markova, 2024)

### 2.3. DOS attack in the past year

#### Dyn Attack (2016)

In October 2016, a major cyberattacks hit a company called Dyn, which plays an important role in helping people connect to websites. Dyn provides DNS services basically, it helps turn website names (like twitter.com) into IP addresses that computers use. When Dyn was attacked, many popular websites like Twitter, Netflix, PayPal, and Reddit stopped working for millions of users, especially in the U.S. and Europe.

The attack used a dangerous malware called **Mirai**. It infected thousands of smart devices, like cameras and internet routers, that had weak or unchanged default passwords. These infected devices were then controlled by the attackers and turned into a **botnet** a large group of devices working together to send massive amounts of fake traffic to Dyn's servers. This traffic was so heavy that Dyn's systems crashed and couldn't handle normal requests from real users.

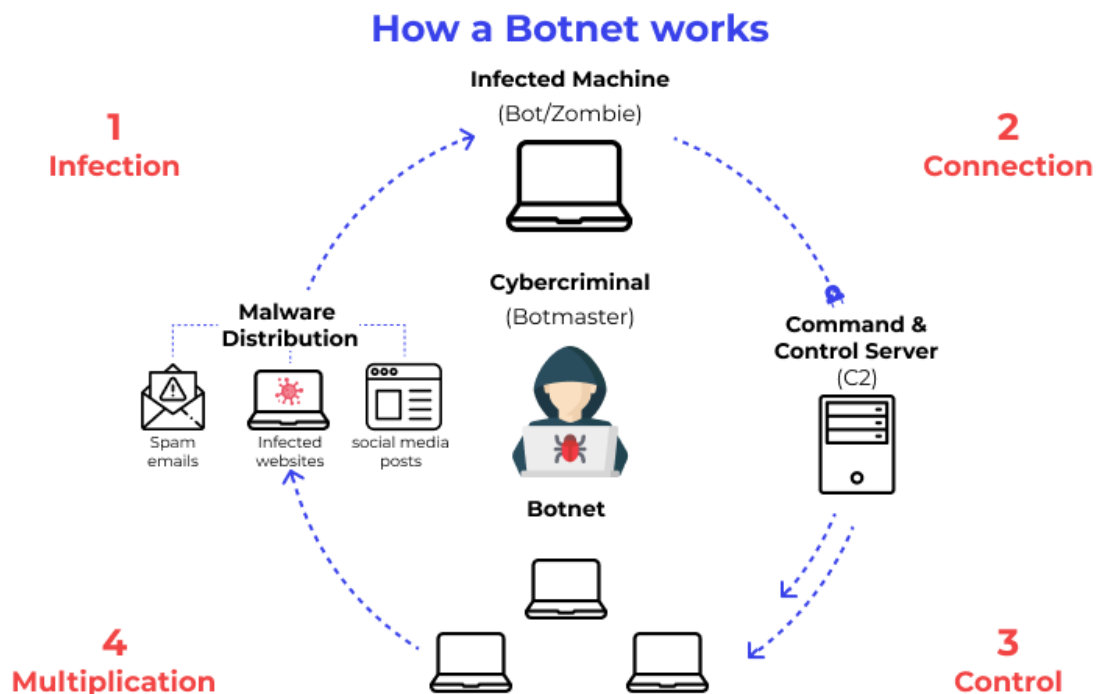


Figure 3: Working mechanism of botnet

Since Dyn helps connect users to many websites, when their servers stopped working, it affected the entire internet experience for many people. Even though some hacker groups

claimed responsibility at first, three individuals were eventually identified as the creators of the Mirai botnet. They were later caught and cooperated with the authorities.

This attack showed how easy it can be to abuse poorly secured smart devices. It was a wake-up call about the risks of not changing default passwords on connected gadgets. After the attack, experts and companies started focusing more on improving device security, setting stronger password rules, and using better tools to detect these types of attacks early. (Medium, 2023)

### 3. Demonstration

#### 3.1. Operating System Used

- Attacker Machine: Kali Linux
- Victim Machine: Metasploitable 2

#### 3.2. Tools Used

- Nmap: Utilized for network scanning to discover active hosts and open ports on the target system.
- Hping3: Used to carry out a SYN flood attack on the target system.

#### 3.3. Lab environment configuration

The lab environment was set up using virtualization software, with both the Kali Linux and Metasploitable 2 virtual machines configured on the same virtual network to ensure seamless communication. Network connectivity between the attacker and victim machines was verified using tools like ping to confirm successful communication.

Step 1: Use ifconfig command to view ip of both the system.

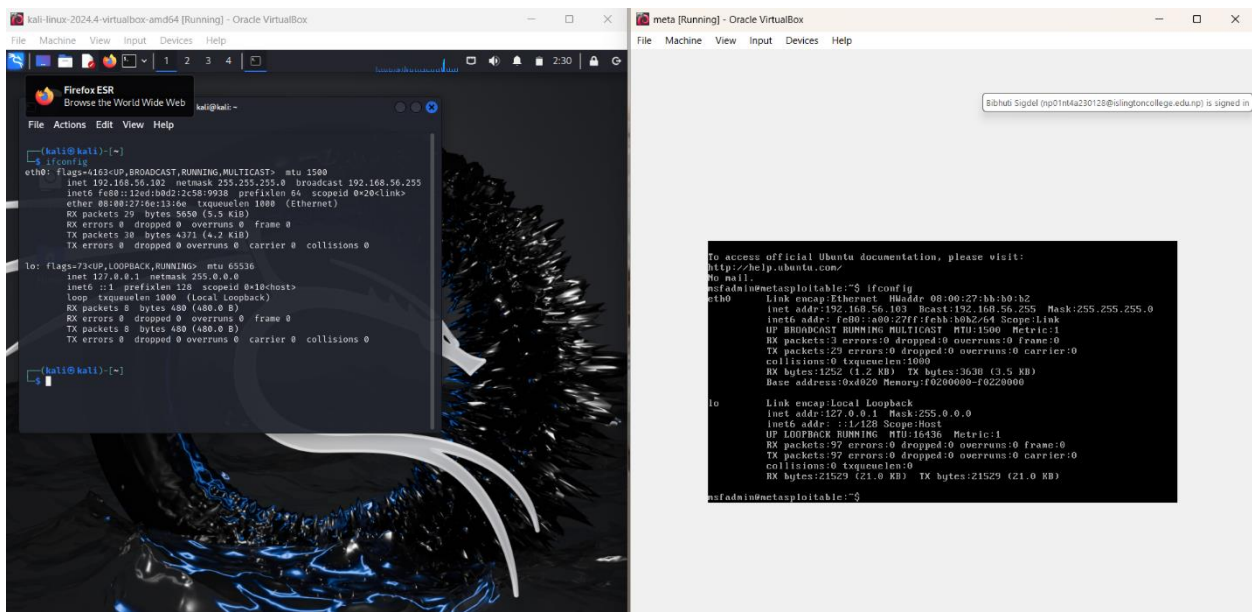
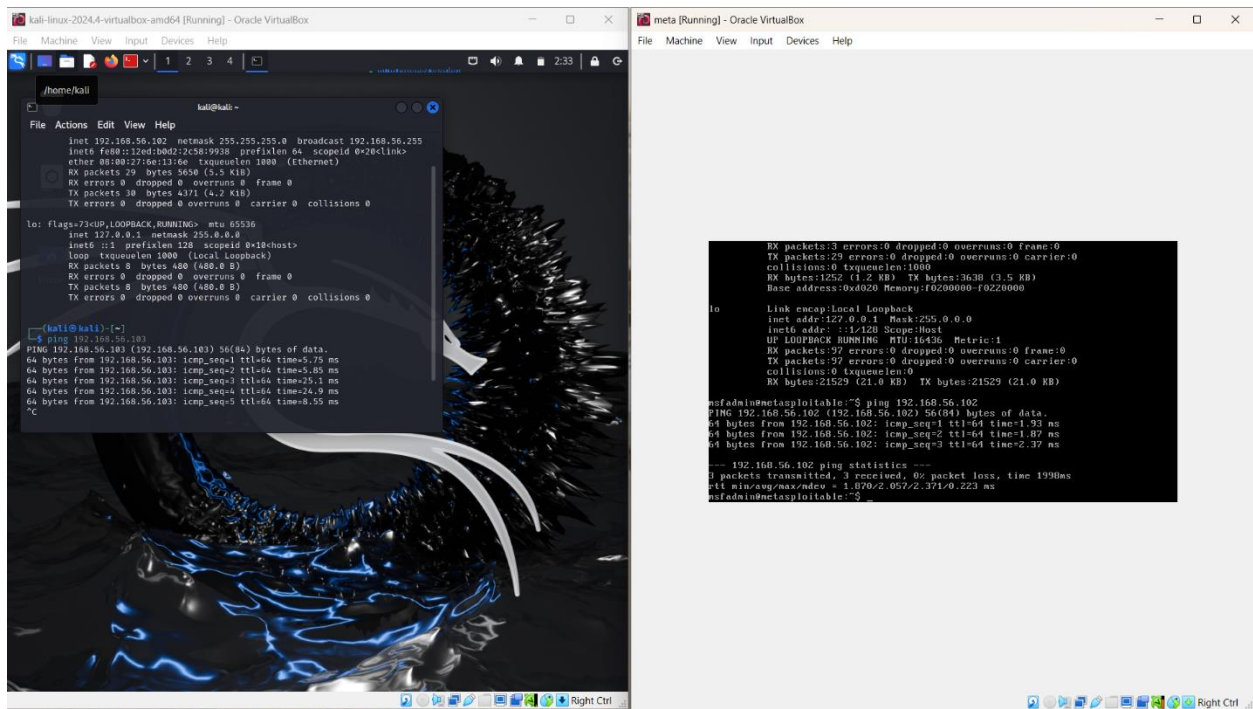


Figure 4: Checking ip

Step 2: Use ping command to check connectivity of both the system.



The image shows two terminal windows from Oracle VM VirtualBox. The left window is titled 'kali-linux-2024.4-virtualbox-amd64 [Running]' and shows the output of the 'ifconfig' command for the 'lo' interface. The right window is titled 'meta [Running]' and shows the output of the 'ifconfig' command for the 'lo' interface, followed by a 'ping' command to 192.168.56.102.

```
kali@kali:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

kali@kali:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data:
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=5.75 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=5.85 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=25.1 ms
64 bytes from 192.168.56.102: icmp_seq=4 ttl=64 time=26.9 ms
64 bytes from 192.168.56.102: icmp_seq=5 ttl=64 time=8.55 ms
^C

```

```
meta@meta:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

meta@meta:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data:
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=1.33 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=1.87 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=2.37 ms
^C

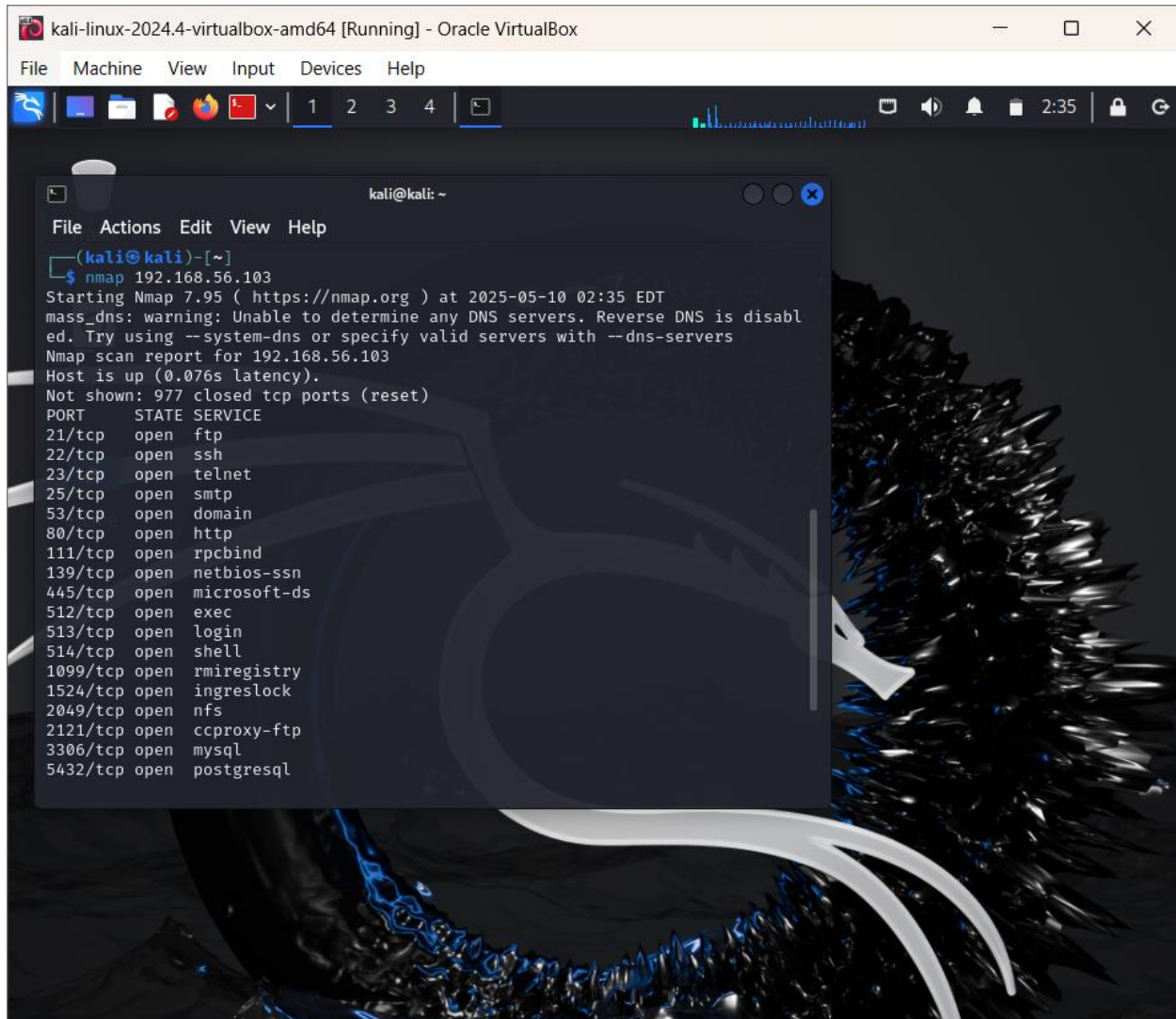
```

Figure 5: Checking connection by doing ping



### 3.4. Port reconnaissance

Step 1: An initial port scan was conducted using Nmap to discover open ports on the target machine. The scan revealed the following open ports:



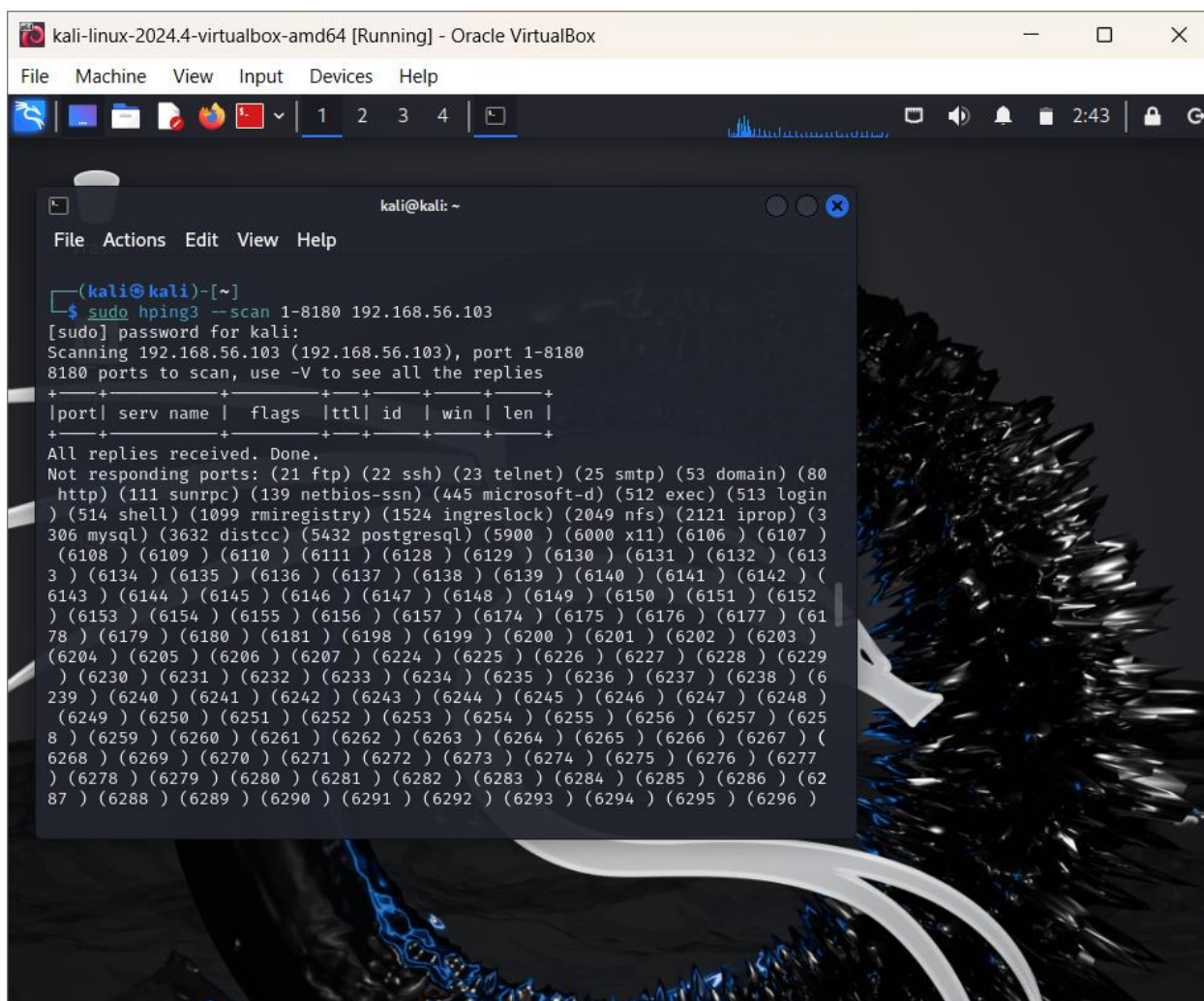
```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ nmap 192.168.56.103  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-10 02:35 EDT  
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers  
Nmap scan report for 192.168.56.103  
Host is up (0.076s latency).  
Not shown: 977 closed tcp ports (reset)  
PORT      STATE SERVICE  
21/tcp    open  ftp  
22/tcp    open  ssh  
23/tcp    open  telnet  
25/tcp    open  smtp  
53/tcp    open  domain  
80/tcp    open  http  
111/tcp   open  rpcbind  
139/tcp   open  netbios-ssn  
445/tcp   open  microsoft-ds  
512/tcp   open  exec  
513/tcp   open  login  
514/tcp   open  shell  
1099/tcp  open  rmiregistry  
1524/tcp  open  ingreslock  
2049/tcp  open  nfs  
2121/tcp  open  ccproxy-ftp  
3306/tcp  open  mysql  
5432/tcp  open  postgresql
```

Figure 6: Nmap port scan to identify open ports and active devices

### 3.5. Port Scanning with Hping3

To continue analyzing the target, Hping3 was used to manually scan the ports. At first, the scan failed because it was run without administrator rights, causing permission errors. After re-running the command using sudo, the scan worked and confirmed that the previously discovered ports were indeed open.

Step 1: Use hping3 to scan ports to identify open ports.



```
kali@kali: ~
File Actions Edit View Help

(kali@kali)-[~]
$ sudo hping3 --scan 1-8180 192.168.56.103
[sudo] password for kali:
Scanning 192.168.56.103 (192.168.56.103), port 1-8180
8180 ports to scan, use -V to see all the replies
+-----+-----+-----+-----+-----+
|port| serv name | flags | ttl | id | win | len |
+-----+-----+-----+-----+-----+
All replies received. Done.
Not responding ports: (21 ftp) (22 ssh) (23 telnet) (25 smtp) (53 domain) (80
http) (111 sunrpc) (139 netbios-ssn) (445 microsoft-d) (512 exec) (513 login
) (514 shell) (1099 rmiregistry) (1524 ingreslock) (2049 nfs) (2121 iprop) (3
306 mysql) (3632 distcc) (5432 postgresql) (5900 ) (6000 x11) (6106 ) (6107 )
(6108 ) (6109 ) (6110 ) (6111 ) (6128 ) (6129 ) (6130 ) (6131 ) (6132 ) (613
3 ) (6134 ) (6135 ) (6136 ) (6137 ) (6138 ) (6139 ) (6140 ) (6141 ) (6142 ) (
6143 ) (6144 ) (6145 ) (6146 ) (6147 ) (6148 ) (6149 ) (6150 ) (6151 ) (6152
) (6153 ) (6154 ) (6155 ) (6156 ) (6157 ) (6174 ) (6175 ) (6176 ) (6177 ) (61
78 ) (6179 ) (6180 ) (6181 ) (6198 ) (6199 ) (6200 ) (6201 ) (6202 ) (6203 )
(6204 ) (6205 ) (6206 ) (6207 ) (6224 ) (6225 ) (6226 ) (6227 ) (6228 ) (6229
) (6230 ) (6231 ) (6232 ) (6233 ) (6234 ) (6235 ) (6236 ) (6237 ) (6238 ) (6
239 ) (6240 ) (6241 ) (6242 ) (6243 ) (6244 ) (6245 ) (6246 ) (6247 ) (6248 )
(6249 ) (6250 ) (6251 ) (6252 ) (6253 ) (6254 ) (6255 ) (6256 ) (6257 ) (625
8 ) (6259 ) (6260 ) (6261 ) (6262 ) (6263 ) (6264 ) (6265 ) (6266 ) (6267 ) (
6268 ) (6269 ) (6270 ) (6271 ) (6272 ) (6273 ) (6274 ) (6275 ) (6276 ) (6277
) (6278 ) (6279 ) (6280 ) (6281 ) (6282 ) (6283 ) (6284 ) (6285 ) (6286 ) (62
87 ) (6288 ) (6289 ) (6290 ) (6291 ) (6292 ) (6293 ) (6294 ) (6295 ) (6296 )
```

Figure 7: Scanning port using hping3

Step 2: Use following command to scan ports of target using random spoofed source ip address to avoid detection.

The screenshot shows a Kali Linux terminal window titled 'kali@kali: ~'. The terminal displays the output of the command `sudo hping3 --scan 1-8180 192.168.56.103 -S --rand-source`. The output indicates that 8180 ports are being scanned. Below the command, a table lists the results of the scan for various ports, including ftp, ssh, telnet, smtp, domain, http, sunrpc, netbios-ssn, microsoft-d, exec, login, shell, rmiregistry, ingreslock, nfs, and iprop. Each entry shows the port number, service name, flags, ttl, id, win, and len.

```

kali@kali: ~
File Actions Edit View Help
8158 ) (8159 ) (8160 ) (8161 ) (8162 ) (8163 ) (8164 ) (8165 ) (8166 ) (8167
) (8168 ) (8169 ) (8170 ) (8171 ) (8172 ) (8173 ) (8174 ) (8175 ) (8176 ) (81
77 ) (8178 ) (8179 ) (8180 )

(kali@kali)-[~]
$ sudo hping3 --scan 1-8180 192.168.56.103 -S --rand-source
Scanning 192.168.56.103 (192.168.56.103), port 1-8180
8180 ports to scan, use -V to see all the replies
+-----+-----+-----+-----+-----+-----+
|port| serv name | flags | ttl | id | win | len |
+-----+-----+-----+-----+-----+-----+
21 ftp      : .S..A... 64    0 5840 46
22 ssh      : .S..A... 64    0 5840 46
23 telnet   : .S..A... 64    0 5840 46
25 smtp     : .S..A... 64    0 5840 46
53 domain   : .S..A... 64    0 5840 46
80 http     : .S..A... 64    0 5840 46
111 sunrpc  : .S..A... 64    0 5840 46
139 netbios-ssn: .S..A... 64    0 5840 46
445 microsoft-d: .S..A... 64    0 5840 46
512 exec    : .S..A... 64    0 5840 46
513 login   : .S..A... 64    0 5840 46
514 shell   : .S..A... 64    0 5840 46
1099 rmiregistry: .S..A... 64    0 5840 46
1524 ingreslock: .S..A... 64    0 5840 46
2049 nfs    : .S..A... 64    0 5840 46
2121 iprop   : .S..A... 64    0 5840 46

```

Figure 8: Scan ports using random spoofed source ip

### 3.6. SYN Flood attack demonstration

A SYN flood attack was simulated using Hping3 by sending a rapid succession of SYN packets to port 135 on the target machine. This type of attack aims to overwhelm the target's

system by initiating numerous half-open TCP connections, thereby consuming server resources and potentially leading to service disruption.

Step 1: Use Hping3 to send Syn packets and flood the victim with following command.

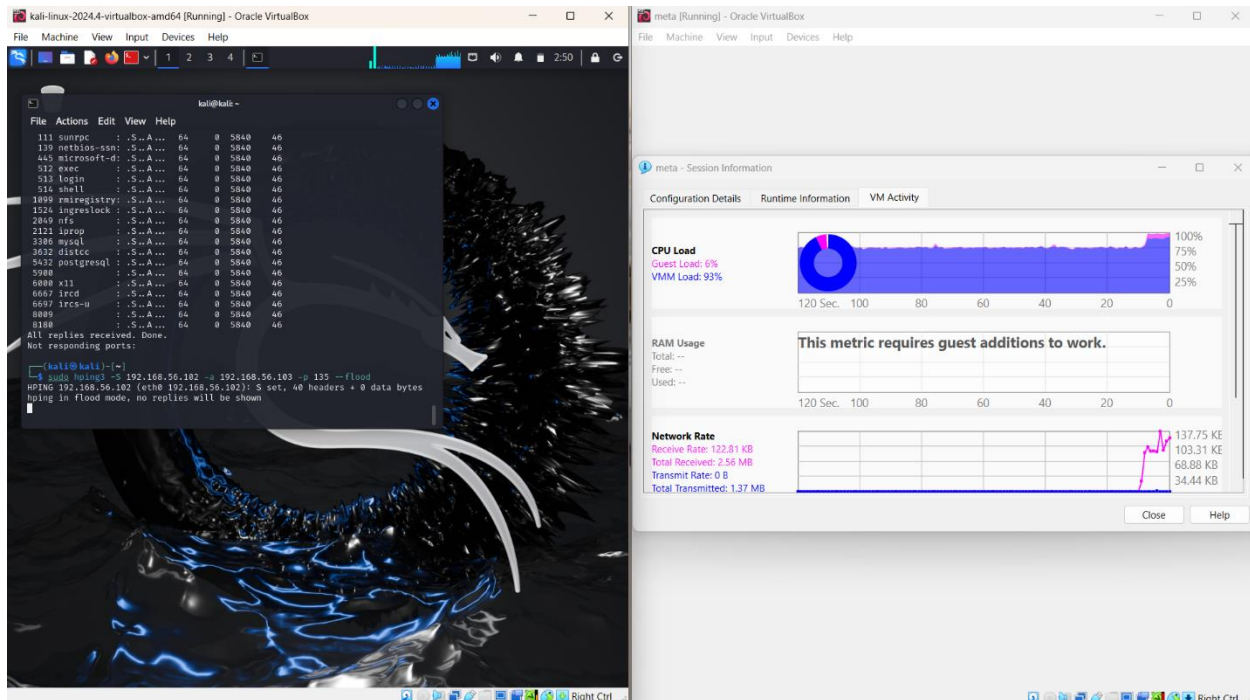


Figure 9: SYN flood attack with spoofed IP using hping3



Step 2: Use `--rand-source` to spoof IP and send SYN packets from different IPs.

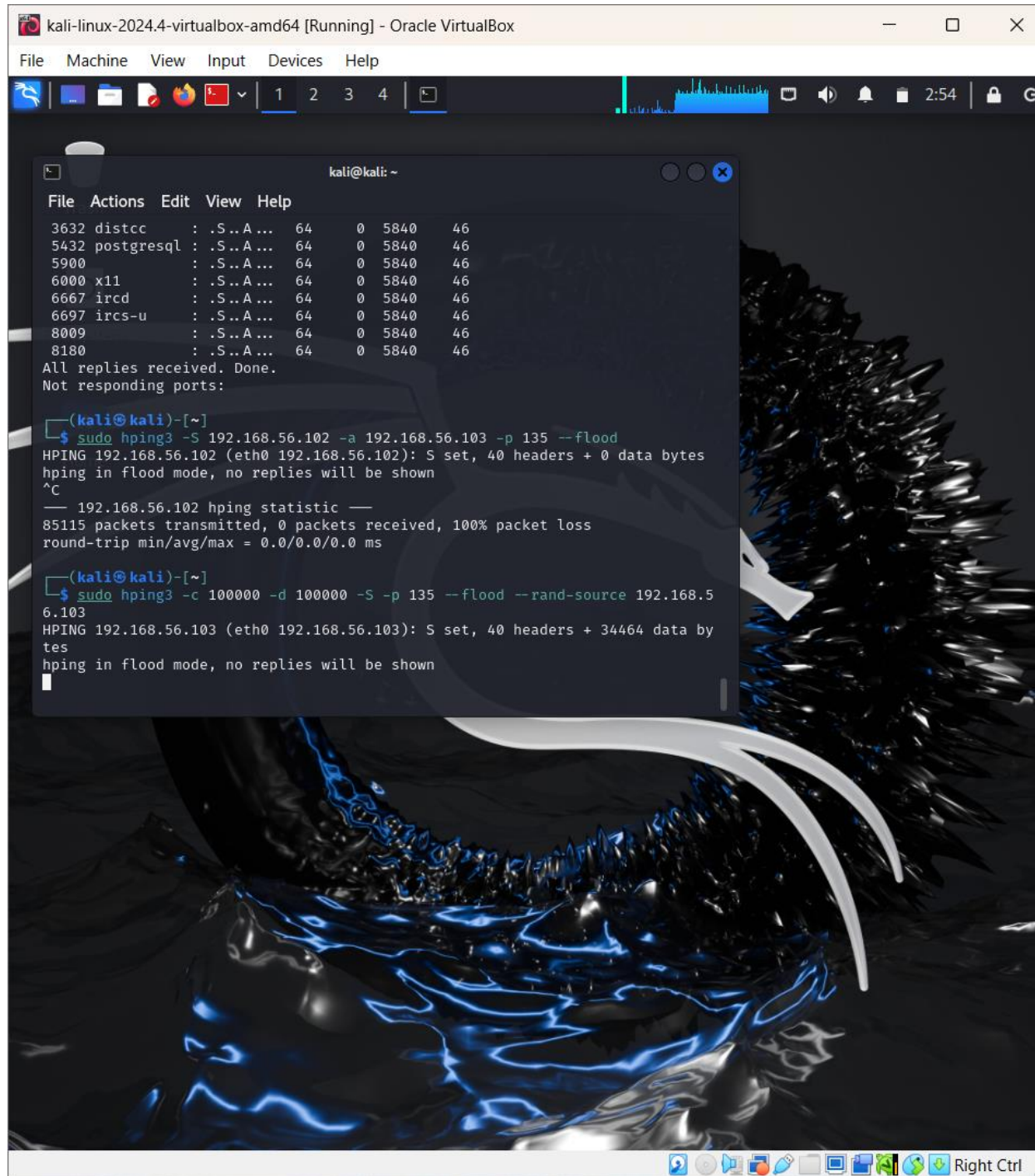
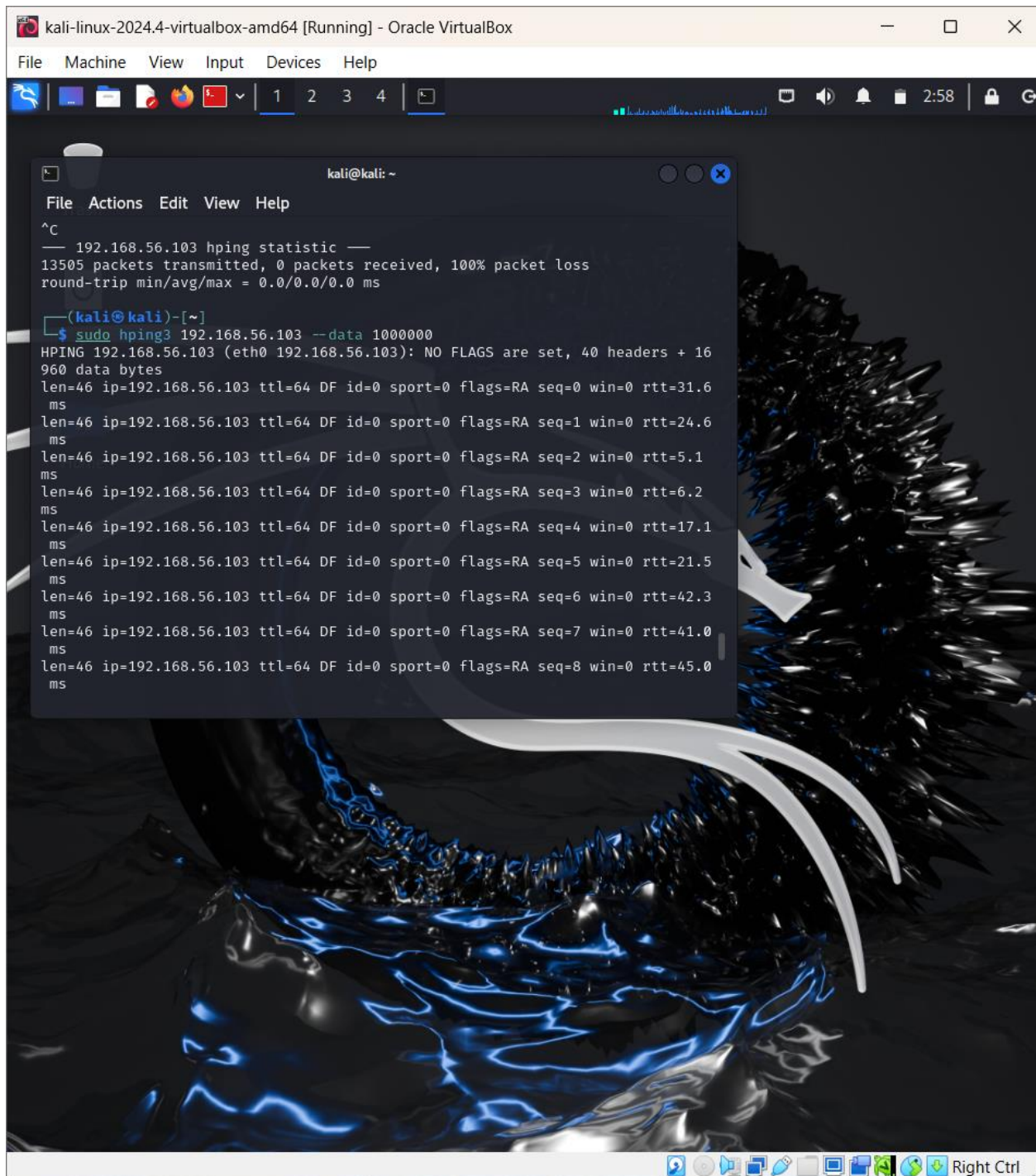


Figure 10: Simulating a DoS attack by flooding port 135 with large packets from random IPs

Step 3: Use following command to send large packets to target IP to create heavy network traffic.



```
kali-linux-2024.4-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4

kali@kali: ~
File Actions Edit View Help
^C
— 192.168.56.103 hping statistic —
13505 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

(kali@kali)-[~]
$ sudo hping3 192.168.56.103 --data 1000000
HPING 192.168.56.103 (eth0 192.168.56.103): NO FLAGS are set, 40 headers + 16
960 data bytes
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=31.6
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=1 win=0 rtt=24.6
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=2 win=0 rtt=5.1
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=3 win=0 rtt=6.2
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=4 win=0 rtt=17.1
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=5 win=0 rtt=21.5
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=6 win=0 rtt=42.3
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=7 win=0 rtt=41.0
ms
len=46 ip=192.168.56.103 ttl=64 DF id=0 sport=0 flags=RA seq=8 win=0 rtt=45.0
ms
```

Figure 11: Sending a large data payload (1MB) to 192.168.56.103 using hping3

### 3.7. Monitoring wireshark to observe spoofed source Ips during SYN Flood attack

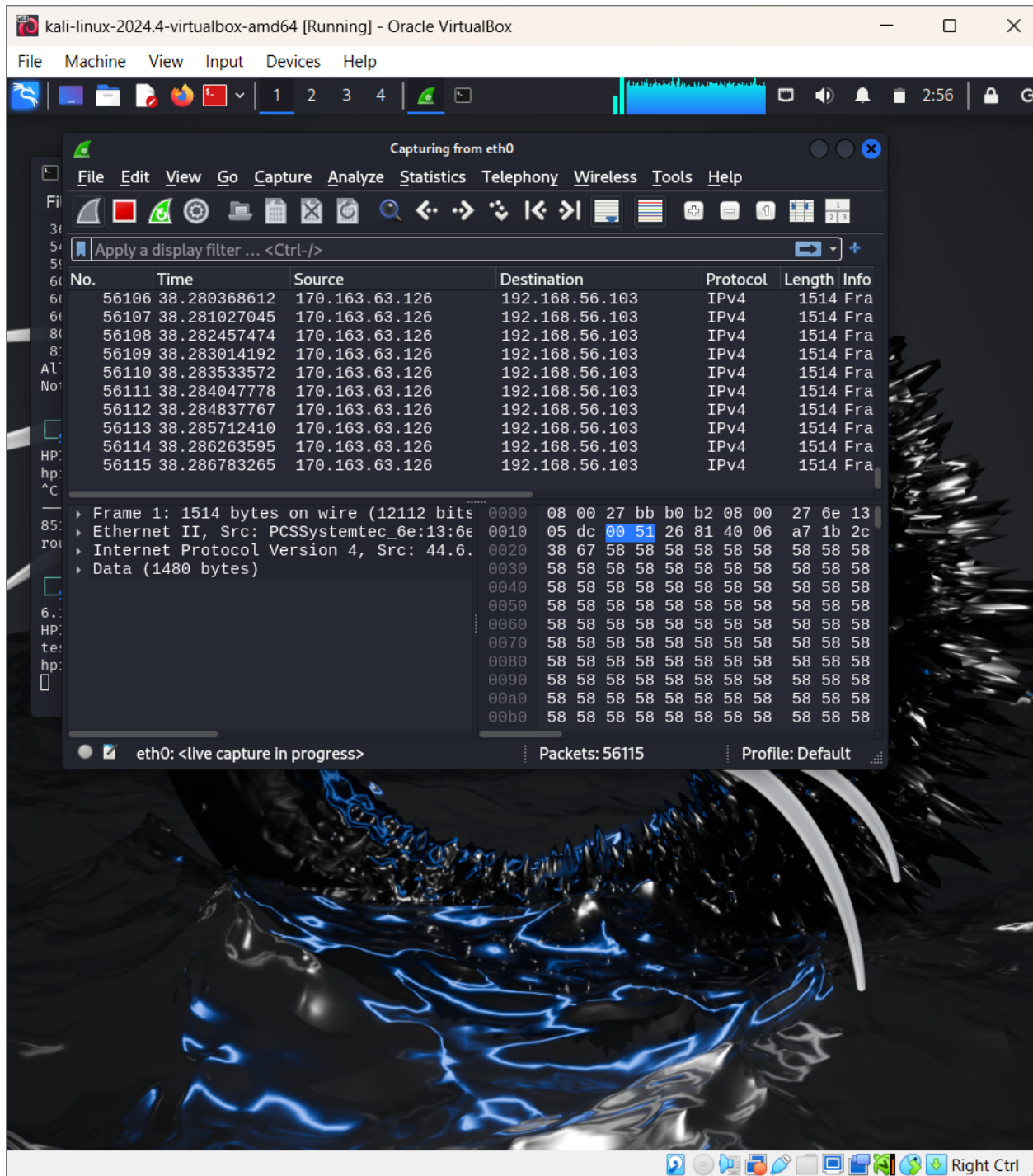


Figure 12: Monitoring Wireshark to observe spoofed source IPs during a SYN flood attack

## 4. Mitigation Strategy and Evaluation

We tested a **ping flood attack**, a type of DoS attack that floods a system with ICMP Echo Request packets, commonly known as pings. This simple yet disruptive attack attempts to exhaust the system's processing power or bandwidth by overwhelming it with continuous ping requests. While testing this practically, it was evident how even a basic command like ping can be misused to cause significant downtime.

To prevent such an attack, we explored several defensive measures, particularly those that involve limiting or disabling ICMP traffic. Below are the techniques I personally tested during the practical test, along with a brief explanation of their effectiveness.

### 4.1. Disabling ICMP Echo Requests Temporarily (Blocking Ping)

One of the simplest and quickest ways to protect a machine from ping floods is by **disabling ICMP Echo responses**, essentially making the system invisible to pings.

```
sudo sysctl -w net.ipv4.icmp_echo_ignore_all=1
```

When this command is active, the system stops responding to any ping requests. I tested this by initiating a ping flood attack from another machine, and once ICMP was disabled, the target machine didn't respond, which effectively neutralized the attack. This technique is especially helpful during an ongoing attack when immediate mitigation is needed.



Step 1: Disabling ICMP Echo response using following command.

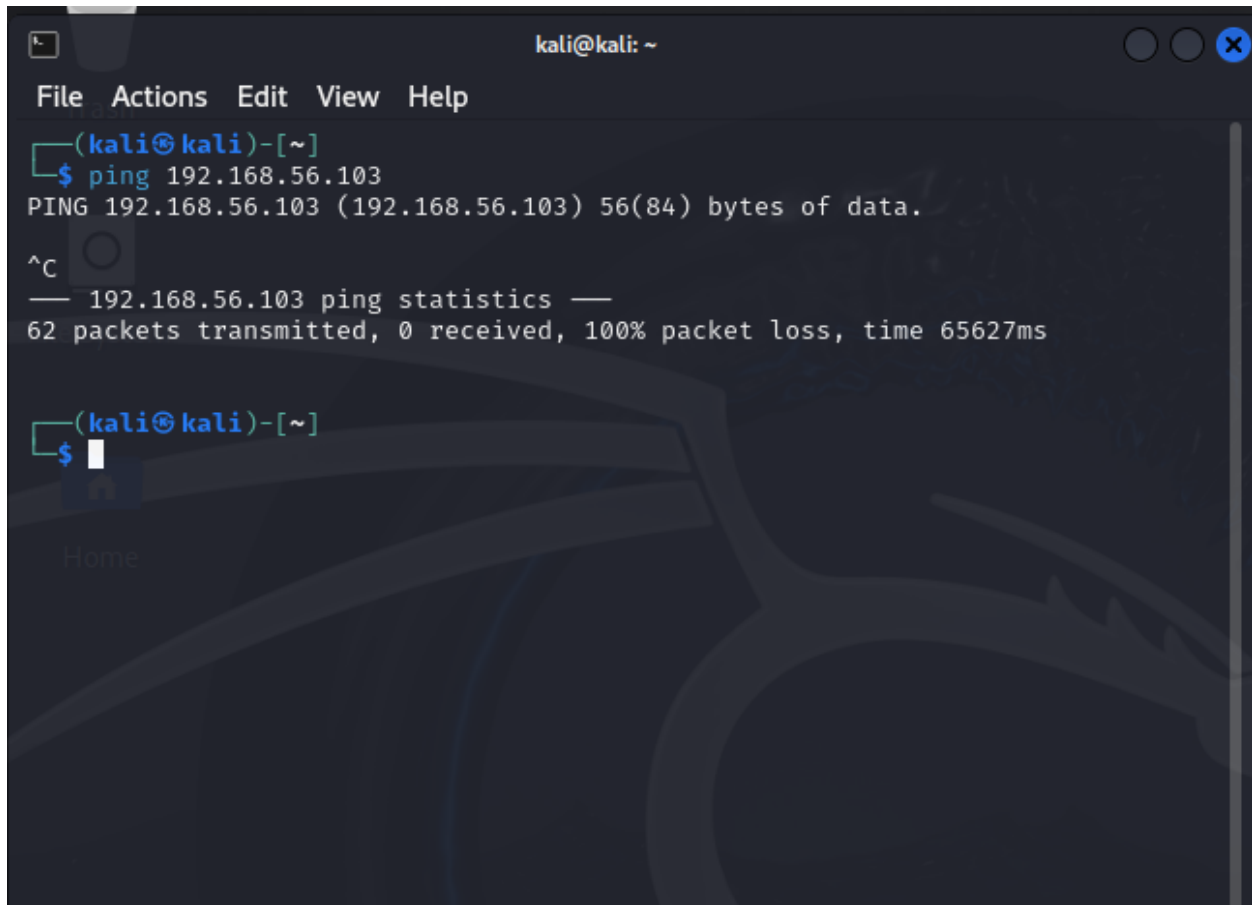
```
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:bb:b0:b2
          inet addr:192.168.56.103  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:febb:b0b2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11 errors:0 dropped:0 overruns:0 frame:0
          TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2564 (2.5 KB)  TX bytes:3924 (3.8 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:97 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:21529 (21.0 KB)  TX bytes:21529 (21.0 KB)

msfadmin@metasploitable:~$ sudo sysctl -w net.ipv4.icmp_echo_ignore_all=1
[sudo] password for msfadmin:
net.ipv4.icmp_echo_ignore_all = 1
msfadmin@metasploitable:~$
```

Figure 13:Disabling ICMP Echo response

Step 2: Ping was unsuccessful after disabling ICMP echo response

A terminal window titled 'kali@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~]'. The user enters '\$ ping 192.168.56.103'. The output shows 'PING 192.168.56.103 (192.168.56.103) 56(84) bytes of data.' followed by a carriage return '^C'. Then, it shows '— 192.168.56.103 ping statistics —' and '62 packets transmitted, 0 received, 100% packet loss, time 65627ms'. The prompt returns to '(kali@kali)-[~]' with a cursor on the next line.

```
kali@kali: ~
File Actions Edit View Help
(kali@kali)-[~]
$ ping 192.168.56.103
PING 192.168.56.103 (192.168.56.103) 56(84) bytes of data.
^C
— 192.168.56.103 ping statistics —
62 packets transmitted, 0 received, 100% packet loss, time 65627ms
(kali@kali)-[~]
$
```

Figure 14: Ping was unsuccessful

#### 4.2. Rate Limiting ICMP Requests Using iptables

To avoid completely disabling ICMP (which may be needed for diagnostics), another approach is **rate limiting**. This lets the system respond to ping requests, but only up to a safe limit. The following command restricted the system to respond to just one ping request per second:

### Step 1: Limiting ICMP Requests Using iptables

```
msfadmin@metasploitable:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit-burst 1 -j ACCEPT
msfadmin@metasploitable:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ _
```

Figure 15: Limiting ICMP request

Step 2: Tried to ping from two machine i.e kali and windows server at a same time and some packets were dropped in windows server to control the traffic.

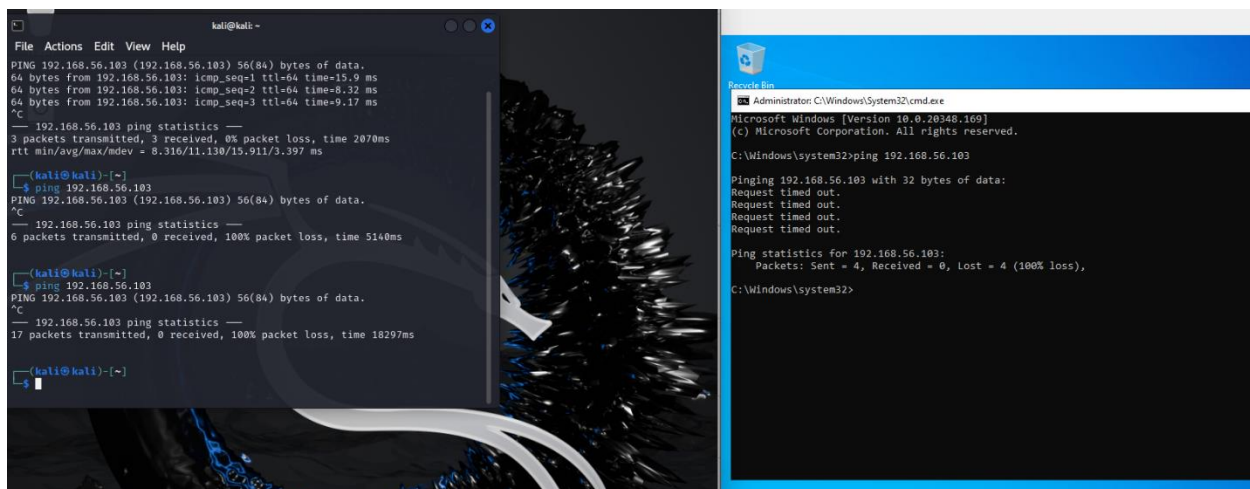
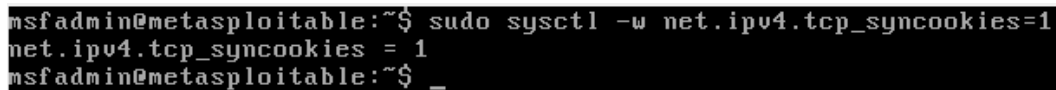


Figure 16: Ping from two machine and packets were dropped

### 4.3. Enabling SYN Cookies

During my research, I found that one of the most efficient ways to prevent SYN flood attacks is by enabling SYN cookies. This technique allows the server to avoid allocating resources too early in the connection process. Basically, the system holds off on committing any memory or processing power until it confirms that the client is genuine. It's like saying, "Prove you're real before I give you attention." This smart delay helps filter out malicious connection attempts without affecting real users.

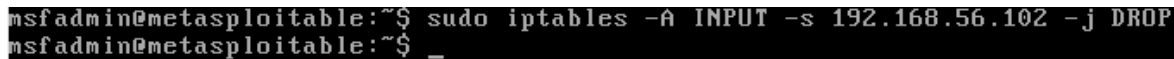
A terminal window with a black background and white text. The prompt is 'msfadmin@metasploitable:~\$'. The command entered is 'sudo sysctl -w net.ipv4.tcp\_syncookies=1'. The output is 'net.ipv4.tcp\_syncookies = 1'. The prompt changes to 'msfadmin@metasploitable:~\$ \_' after a space.

```
msfadmin@metasploitable:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
msfadmin@metasploitable:~$ _
```

Figure 17: Enabling SYN cookies

### 4.4. Blocking a Malicious IP

In a practical scenario, if an attacker's IP address can be identified (especially during repeated scans or flooding), the simplest yet powerful defence is to block it completely. It's like cutting off a toxic person from your life you prevent any more damage by just



```
msfadmin@metasploitable:~$ sudo iptables -A INPUT -s 192.168.56.102 -j DROP
msfadmin@metasploitable:~$ _
```

Figure 18: Blocking malicious IP

refusing to let them reach you. This can be done using basic firewall rules and is a direct way to stop incoming threats immediately.

#### **4.5. Limiting Number of Connections Per IP**

Another method I explored was setting limits on how many simultaneous connections one IP address can make. Imagine someone hammering your door 100 times in a minute it's overwhelming. By saying "I'll only respond to 10 knocks at a time," you reduce the load and protect your system from being flooded by a single source. This is especially useful when bots or scripts are trying to exhaust server resources.

#### **4.6. Using Fail2Ban to Ban Repeat Offenders**

Fail2Ban is like having a personal guard for your system. It watches for suspicious patterns like failed login attempts or rapid port scans and then automatically bans the offender for a while. I tested this in a simulated environment and found it really handy almost like setting up a security camera that locks the gate if it catches something fishy. It adds an automated layer of defence, saving time and effort.

#### **4.7. Dropping Fragmented Packets**

Some attackers use oddly broken-up packets (fragmented packets) that confuse the target system kind of like sending a letter in 10 different envelopes with no instructions. Older systems might crash while trying to piece them together. So, I learned that by dropping these unusual packets, we can block such trickery early on. It's a clean way to handle attacks like Teardrop.

#### **4.8. Using NGINX as a Reverse Proxy with Rate Limiting**

Deploying NGINX in front of a server is like putting a smart bouncer at the door of a party. I found that NGINX can be set up to only let in a limited number of guests (requests) per second. So even if someone tries to flood the entry, they'll be held back. It's great for managing HTTP-level traffic and filtering out the noisy crowd before they even get near your main system.

## **5. Evaluation and Analysis**

### **5.1. Effectiveness of Implemented Techniques**

The impact of simulated DoS attacks was mitigated or stopped entirely by the mitigation techniques used throughout the experiment, such as ICMP echo suppression, rate limitation with iptables, and turning on SYN cookies. For example, a ping flood attack was instantly neutralised by turning off ICMP echo answers, as seen by the attacker machines' failed ping attempts. In a similar vein, rate-limiting rules effectively throttled ping requests, guaranteeing that the system remained operational for actual diagnostics and impervious to misuse.

Enabling SYN cookies helped lessen the resource fatigue typically brought on by half-open connections in the instance of the SYN flood attack carried out using hping3. In order to avoid memory and CPU overload, it made sure the server postponed allocating resources until the handshake was over.

### **5.2. Comparison and Practicality**

Out of every strategy investigated, it was discovered that restricting the number of concurrent connections per IP and blocking known malicious IPs were the most simple and successful in situations involving a single attacker. These strategies might, however, be less successful against widespread DDoS attacks that use spoof or rotating IPs. Fail2Ban provided a dynamic and automatic solution for such situations. It reduced the need for manual intervention by automatically enforcing prohibitions when it recognised repeated suspicious behaviour, like port scanning or quick connection attempts.

By screening excessive HTTP requests before they reach the backend, the use of NGINX as a reverse proxy added an application-layer defence mechanism. It is a good defence against HTTP flood attacks, but it isn't immediately relevant to lower-level SYN attacks.

### **5.3. Advantages and Disadvantages**

#### **Advantages**

- Immediately stops basic DoS attacks like ping and SYN floods.
- Built-in tools consume minimal system resources.
- Easy to set up without needing extra software.

- Rules can be tailored to fit specific environments.
- Supports automation through tools like Fail2Ban.
- Enables multi-layered protection when combined.
- Widely used in real-world servers and networks.

#### **Disadvantages**

- Less effective against large-scale distributed attacks.
- May block legitimate users due to strict rules.
- Hard to manage across multiple machines manually.
- Offers limited visibility without logging enabled.
- Needs technical knowledge to avoid misconfiguration.
- Not sufficient as a standalone protection method.
- Doesn't provide advanced detection like IDS or firewalls.

#### **5.4. Real-World Applicability**

These mitigation strategies are frequently used in cloud environments, enterprise networks, and web servers. A multi-layered defence approach includes tools like iptables, SYN cookies, Fail2Ban, and reverse proxies like NGINX. In order to continuously detect, analyse, and respond to threats, organisations frequently integrate these tools with Intrusion Detection and Prevention Systems (IDS/IPS) and Security Information and Event Management (SIEM) platforms.



## 6. Conclusion

The study successfully demonstrated how Denial-of-Service (DoS) attacks, such as SYN floods and ping floods, can compromise system performance and availability. Through a controlled lab environment, the effectiveness of several defensive measures was tested and verified. Techniques such as disabling ICMP, enabling SYN cookies, and using tools like Fail2Ban proved to be simple yet powerful methods to mitigate DoS threats. The findings revealed that although these methods are effective against small-scale attacks, they have limitations when facing more sophisticated, distributed attacks. Therefore, it is crucial to implement a layered security strategy that integrates both host-based and network-based defences. Real-world applicability of the tested methods was also confirmed, as many are used in enterprise networks, cloud infrastructures, and web server environments. Moving forward, organizations should combine these traditional approaches with advanced detection systems and automated responses to ensure comprehensive protection against evolving cyber threats.

## 7. Bibliography

(2024, 04 24). Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/what-is-botnet/>

(2025). Retrieved from Cloudflare: <https://www.cloudflare.com/en-gb/learning/ddos/syn-flood-ddos-attack/>

*fortinet*. (2025). Retrieved from <https://www.fortinet.com/resources/cyberglossary/what-is-cyber-attack#:~:text=A%20cyber%20attack%20is%20the%20process%20of%20attempting,computers%20and%20networks%20using%20one%20or%20more%20computers.>

GarySackrider. (2023, 06 9). *NetScout*. Retrieved from <https://www.netscout.com/blog/ddos-attack-motivations-abound#:~:text=Here%20are%20some%20possible%20motivations%20behind%20DDoS%20attacks%3A,to%20their%20social%20or%20political%20causes.%20More%20items>

*geeksforgeeks*. (2024, 07 1). Retrieved from <https://www.geeksforgeeks.org/types-of-dos-attacks/>

*geeksforgeeks*. (2025, february 24). *geeksforgeeks*. Retrieved from *geeksforgeeks*: <https://www.geeksforgeeks.org/http-flood-attack/>

Lutkevich, B. (2025, May 28). *TechTarget*. Retrieved from TechTarget: <https://www.techtarget.com/searchsecurity/definition/SYN-flooding>

Markova, V. (2024, November 21). *CloudDNS*. Retrieved from CloudDNS: <https://www.cloudns.net/blog/what-is-teardrop-attack-and-how-to-protect-ourselves/>

Medium. (2023, april 10). *Medium*. Retrieved from Medium: <https://medium.com/%40d21dcs151/a-case-study-on-mirai-botnet-attack-of-2016-4b66630e6508>

Raza, M. (2025, april 14). *Splunk Blogs*. Retrieved from Splunk Blogs: [https://www.splunk.com/en\\_us/blog/learn/dos-denial-of-service-attacks.html?](https://www.splunk.com/en_us/blog/learn/dos-denial-of-service-attacks.html?)

smarterMSP. (2025, April 14). *smartermsp*. Retrieved from smartermsp: <https://smartermsp.com/tech-time-warp-why-the-panix-attack-was-a-wake-up-call/>

Twingate Team. (2024, August 1). *Twingate*. Retrieved from Twingate:  
<https://www.twingate.com/blog/glossary/icmp%20flood>