Controlled DDoS Attack Simulation for Cybersecurity Analysis

FROM THEORY TO MITIGATION – A HANDS-ON APPROACH

Jatin Navani D12/B 34

Nitika Nagdevani D12/B 33

Sanika Ingale D12/B 19

Project Overview

Objective:

- Simulate a realistic SYN flood DDoS attack in a safe, isolated environment.
- Analyze attack patterns, traffic behavior, and system impact.
- Develop and test an automated mitigation system using Python.

Why This Project?

- DDoS attacks are a top threat (e.g., 2023 Cloudflare report: 7.9M attacks/year).
- Hands-on experience with virtualization, networking, and scripting.

Tools Used:

- Simulation: GNS3, VirtualBox
- Attack: Kali Linux, hping3, Python
- Defense: Ubuntu Server, tshark, iptables, Flask (Dashboard)

Tech Stack:

- Backend: Flask + WebSockets.
- Frontend: HTML/CSS, Socket.IO, Chart.js.

Understanding DDoS & SYN Flood Attacks

What is DDoS?

Malicious traffic floods a target to disrupt services.

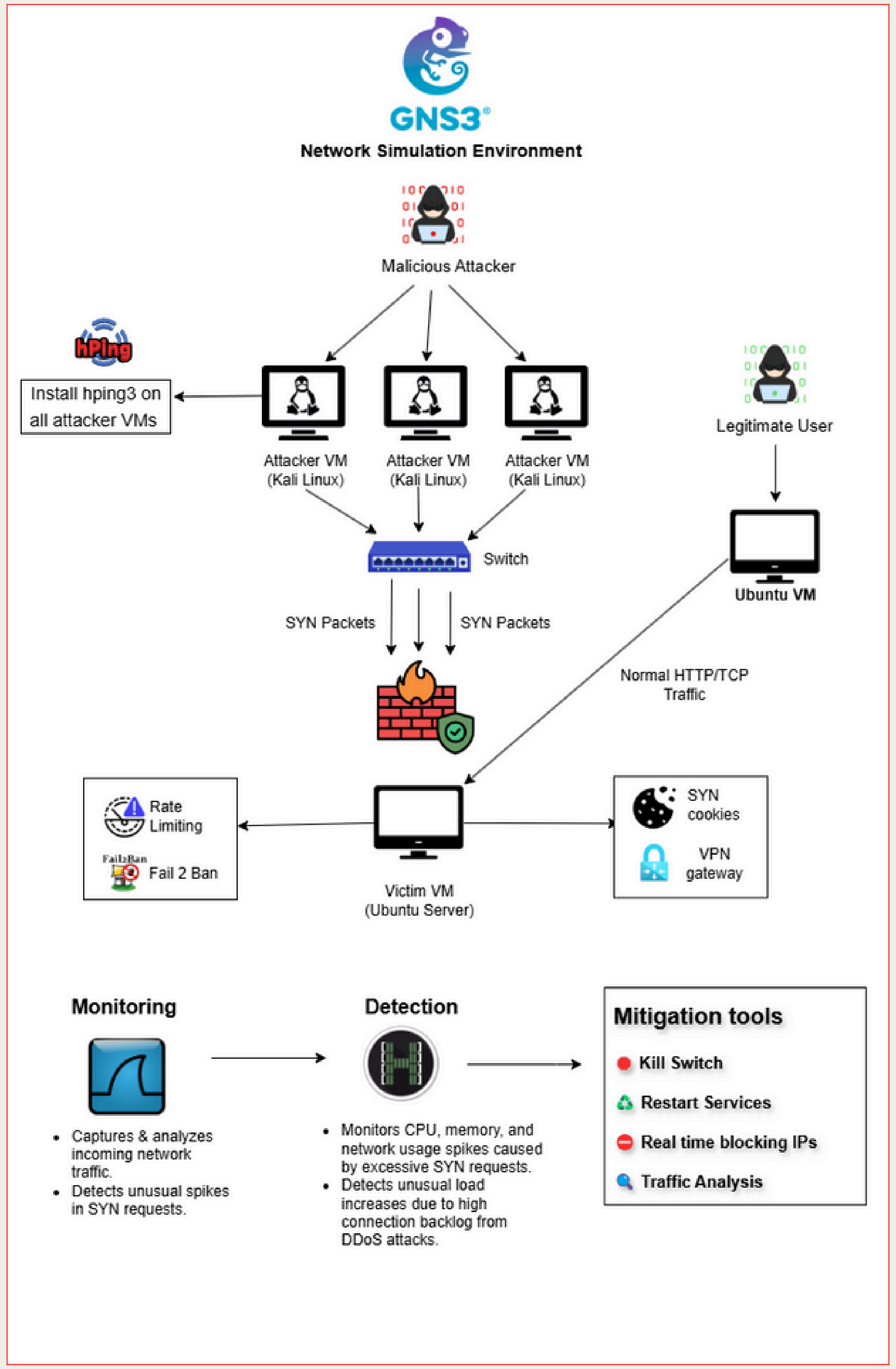
SYN Flood Mechanics:

- Attacker sends TCP SYN packets to victim.
- Victim allocates resources for half-open connections.
- Server exhausts memory/CPU, denying legitimate users.

Attack Design:

- Single Attacker → Botnet Simulation: One C2 server controls bots (scaled-down model).
- Controlled Variables: Packet rate, duration, victim specs.

System Design



- Captures & analyzes incoming network traffic.
- Detects unusual spikes in SYN requests.

- Monitors CPU, memory, and network usage spikes caused by excessive SYN requests.
- Detects unusual load increases due to high connection backlog from DDoS attacks.

System Architecture

Attacker VM (Kali Linux)

- Runs C2 server (c2_server.py).
- Commands bots to launch attacks.

Bot Machine (Python Script)

- Simulates multiple bots with hping3.
- Connects to C2 for attack triggers.

Victim VM (Ubuntu Server)

- Hosts a dummy web service (e.g., Apache).
- Runs tshark to monitor traffic.

Network Setup:

- Bridged Adapter: Isolated but mimics real-world IP routing.
- GNS3 Integration: Links VMs logically.

Step-by-Step Attack Simulation

Phase 1: Setup

Deploy VMs in VirtualBox (Kali + Ubuntu).

Configure IPs (Victim Server and C2 server)

Phase 2: Launch Attack

C2 Server Code: Listens on port 9000, sends ATTACK_START.

Bot Script:

```
attack_command = "hping3 -S --flood -p 80 192.168.1.20" //according to configured IP
subprocess.run(attack_command, shell=True)
```

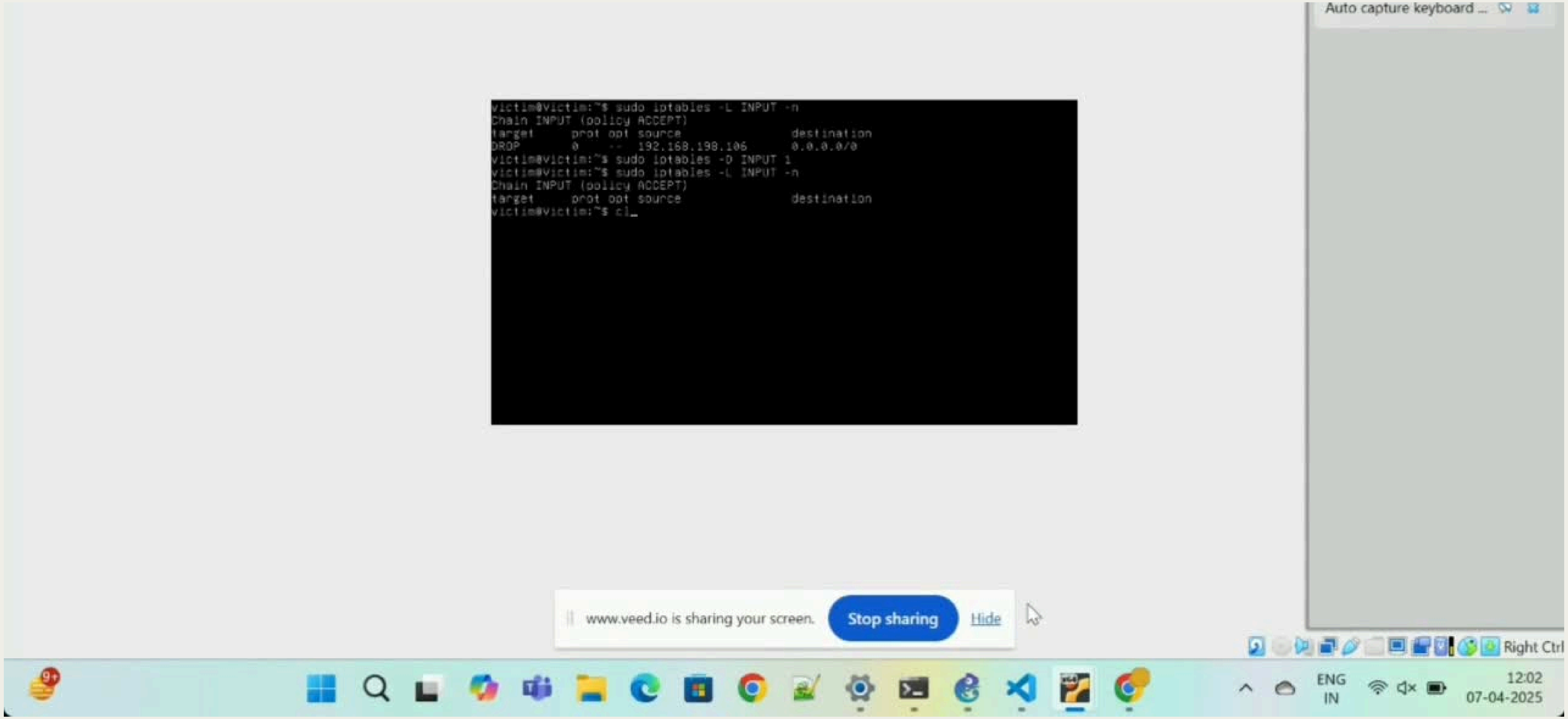
Step-by-Step Attack Simulation

Phase 3: Monitor Traffic

Victim terminal:

```
sudo tshark -i enp0s3 -Y "tcp.flags.syn==1"
```

Video Demonstration



```
victim@victim:~$ sudo iptables -L INPUT -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       0    --  192.168.198.106      0.0.0.0/0
victim@victim:~$ sudo iptables -D INPUT 1
victim@victim:~$ sudo iptables -L INPUT -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
victim@victim:~$ cl_
```

www.veed.io is sharing your screen.

DDOS Simulation

Defense Mechanism Design

Problem: Manual mitigation is too slow.

Solution: Automated Python Monitor (monitor.py)

Detection:

tshark filters SYN packets.

Counts SYNs per IP (threshold: 100 packets/10 sec).

Mitigation:

Bans IPs via iptables:

os.system(f"sudo iptables -A INPUT -s {malicious_ip} -j DROP")

Alerting:

Real-time dashboard updates via WebSockets.

Real-Time Dashboard Demo

Features:

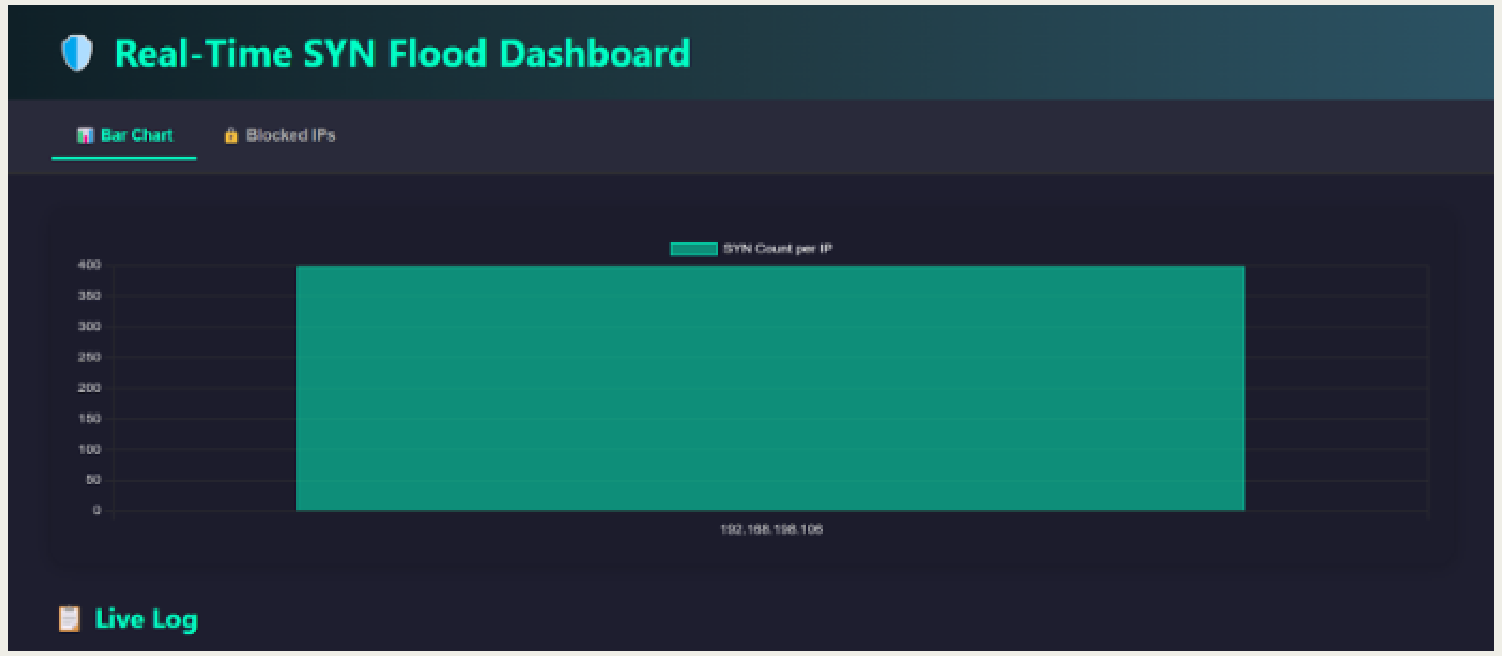
- Live SYN Packet Graph:
- Chart.js visualizes traffic per IP.

Banned IP List:

- Red-highlighted entries with timestamps.

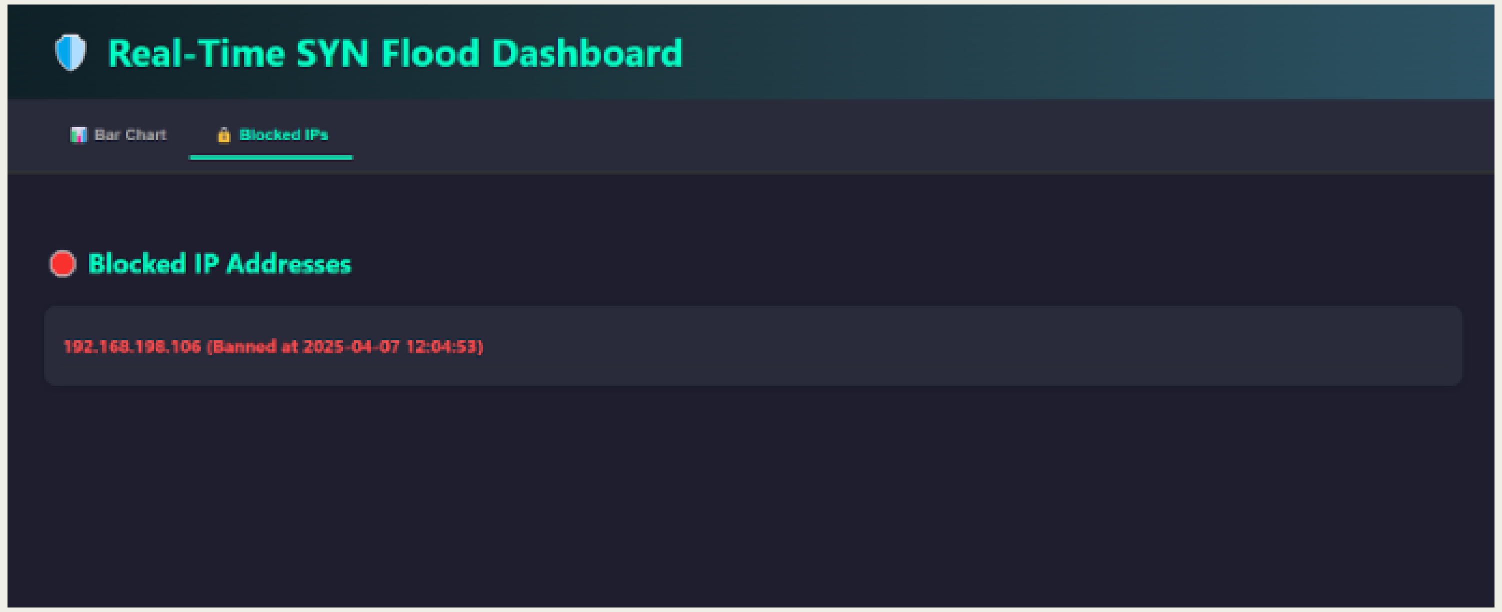
Popup Alerts:

"🚨 IP 192.168.1.15 banned!"



Real-Time SYN Flood Dashboard

Live Log



Real-Time SYN Flood Dashboard

Blocked IP Addresses

192.168.198.106 (Banned at 2025-04-07 12:04:53)

Conclusion & Future Work

- **Key Takeaways:**
- DDoS attacks are simple to launch but devastating.
- Automation (Python + iptables) can mitigate attacks in seconds.

Future Enhancements:

- Integrate machine learning for adaptive thresholds.
- Test with larger botnets (e.g., 50+ VMs).

Thank you!