International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 04 | April – 2025 DOI: 10.55041/ISJEM02959
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

AUTOMATED DDOS SIMULATION IN A VIRTUAL ENVIRONMENT USING PYTHON AND TELEGRAM BOT

Dhilip sankar U1, Gokul raj M2 , Shridhar M. J3 , Mrs .B. Durga devi 4 , Mrs. K. Saranya5 1, 2, 3.B.SC ISCF students, Dr.M.G.R Educational and Research Institute Deemed to be University, Chennai.
Corresponding Email ID: viswaboy0904@gmail.com
4. Assistant Professor, Dr.M.G.R Educational and Research Institute, Deemed to be University, Chennai.
5. Assistant Professor, Dr.M.G.R Educational and Research Institute, Deemed to be University, Chennai.

ABSTRACT

This project presents an automated system for simulating and mitigating DDoS attacks in a virtual environment. The system mimics DDoS attacks based on TCP, UDP, and ICMP using Python scripts and the hping3 tool. Incoming traffic is watched for harmful patterns by a real- time packet sniffing mechanism. An automated reaction that blocks the attacker's IP address is triggered when threats are detected. Remote management and immediate alerts are made possible by a Telegram bot integration. The system has a web dashboard that shows blocked IPs and attack logs. It guarantees practical experience without endangering infrastructure in the real world. Because it makes use of WSL and Virtual Box, the setup is accessible and portable. This framework is perfect for experimentation and education in cybersecurity. It provides an efficient, automated, and reasonably priced DDoS defense strategy.

1. INTRODUCTION

Distributed Denial-of-Service (DDoS) attacks continue to be one of the most disruptive threats to network security in the current digital era. By overloading a target system with traffic, these attacks seek to disrupt services or bring about a total system failure. Intelligent, real-time defense systems that can react quickly and efficiently are desperately needed as cyber threats become more complex.

The goal of this project, "Automated DDoS Simulation in a Virtual Environment Using Python and Telegram Bot," is to use scripting to automatically mitigate DDoS attacks in a controlled virtual environment.

The target system, a Kali Linux virtual machine, runs a Python-based detection and response engine, while the attacker system uses Windows Subsystem for Linux (WSL) to initiate traffic floods via hping3.

A Telegram bot is incorporated for remote IP management and real-time alerts to improve accessibility and control. A web dashboard also provides visual information About attack occurrences and system reactions. In addition to showcasing useful Cybersecurity abilities, this system acts as a teaching tool for scholars and students investigating network defense tactics.

2. REQUIREMENT ANALYSIS

2.1 OBJECTIVE OF THE PROJECT

This project's goal is to create an automated system that can efficiently mimic, identify, and counter Distributed Denial-of-Service (DDoS) attacks in a virtual setting. It seeks to establish a secure and regulated environment in which the hping3 tool can be used to simulate different kinds of DDoS attacks, including TCP, UDP, and ICMP floods. Python-based packet sniffing techniques are used in the system's design to monitor network traffic in real-time, and anomalous traffic patterns are used to detect malicious activity. To lessen the threat, the system automatically blocks the source IP address when it detects an attack. Users can respond to incidents even from mobile devices thanks to the integration of a Telegram bot that enables remote firewall management and real- time alerts.

To improve usability and monitoring, a web-based dashboard also logs all blocked IPs and provides a visual timeline of attacks. The goal of this project is to offer a useful, affordable way to learn, experiment, and develop defense strategies in cybersecurity.

2.2 EXISTING SYSTEM

Nowadays DDoS detection and prevention systems usually rely on manual use of packet generators like hping3 or more conventional tools like firewalls and intrusion detection systems (IDS). Although malicious traffic can be detected by tools like Snort or Suricata, they frequently require manual configuration and lack automated response capabilities. These systems typically rely on preset rules and lack remote control capabilities and real- time alerts. Furthermore, a lot of the solutions currently in use are intricate, costly, or not made for educational settings. They are therefore unsuitable for researchers or students who need a platform that is lightweight, safe, and simple to set up in order to study and test network security risks such as DDoS attacks.

Disadvantages

- The majority of systems need human intervention to prevent attacks, which slows down reaction times.
- Long-term attacks are more likely when users are not immediately alerted to threats.
- Configuring traditional IDS/IPS tools can be challenging, particularly for novices or students.
- Bot control and other mobile or remote management features are frequently absent from current systems.

2.3 PROPOSED SYSTEM

The proposed system is a user-friendly, automated framework made to mimic and protect against DDoS attacks in a virtual setting. It employs Python scripts to track network activity and identify attack trends instantly. Using firewall rules, the system automatically blocks the attacker's IP address when malicious traffic is detected. The user can monitor and manage the system remotely from any location thanks to the integration of a Telegram bot that offers real-time alerts and remote access for IP management. A web-based dashboard also shows blocked IPs and attack timelines, providing visual information about network activity. The entire setup is inexpensive, lightweight, and perfect for small-scale experimentation and cybersecurity learners.

Advantages

- detects and stops DDoS attacks automatically without human assistance.
- Users can manage the system and get alerts from any location thanks to an integrated Telegram bot.
- For convenient monitoring, a web interface shows a timeline of attacks And blocked IPs.
- It is perfect for researchers and students because it makes use of open-source tools in a virtual environment.

3. REQUIREMENT SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- Processor: Intel i5 / AMD Ryzen 5 or higher
- RAM: Minimum 8 GB (16 GB recommended for smooth virtualization)
- Storage: At least 50 GB of free disk space
- System Type: 64-bit operating system with virtualization support enabled
- Graphics: Integrated graphics (no dedicated GPU required)
- Keyboard : Standard keyboard
- Monitor : 15 inch color monitor

3.2 SOFTWARE REQUIREMENTS

- Operating System (Host): Windows 10/11 (64-bit) with WSL enabled
- Virtualization Tool: Virtual Box (version 6.1 or higher)
- Target OS (Virtual Machine): Kali Linux (latest stable release)
- Attacker Environment: WSL (Windows Subsystem for Linux)
- Programming Language: Python 3.13.2
- Network Attack Tool: hping3
- Telegram API Integration: Telegram Bot via BotFather and HTTP API
- Web Browser: For accessing the dashboard interface and log

4.MODULES

- Packet Sniffing & Detection Module
- Firewall Control Module
- Telegram Bot Module
- Web Dashboard Module
- Attack Simulation Module
- Logging and Auditing Module
- Configuration and Threshold Management Module

Packet Sniffing & Detection Module - This module uses Python and Scapy-like libraries to keep an eye on incoming network traffic. To identify possible DDoS attacks, it examines packets for unusual activity like frequent requests or flood patterns.

Firewall Control Module -This module uses Python scripts to dynamically update the firewall rules to block malicious IP addresses when it detects an attack. It keeps track of blocked IPs and guarantees prompt response.

Telegram Bot Module - This module includes a Telegram bot to provide real- time alerts to the administrator. It also enables remote commands such as log viewing or IP unblocking, so improving accessibility and control.

Web Dashboard Module - The dashboard shows a timeline of identified attacks, justifications for blocking, and blocked IP addresses. Built with HTML and Flask, it offers a simple way to track the status of the system.

Attack Simulation Module - Using the hping3 tool from the attacker system in a virtual environment, simulates DDoS attacks (TCP, UDP, ICMP floods).

Logging and Auditing Module - Logs IP addresses, attack type, detection time. Read by the dashboard and Telegram bot modules for live display. Helps in creating datasets for future ML model training.

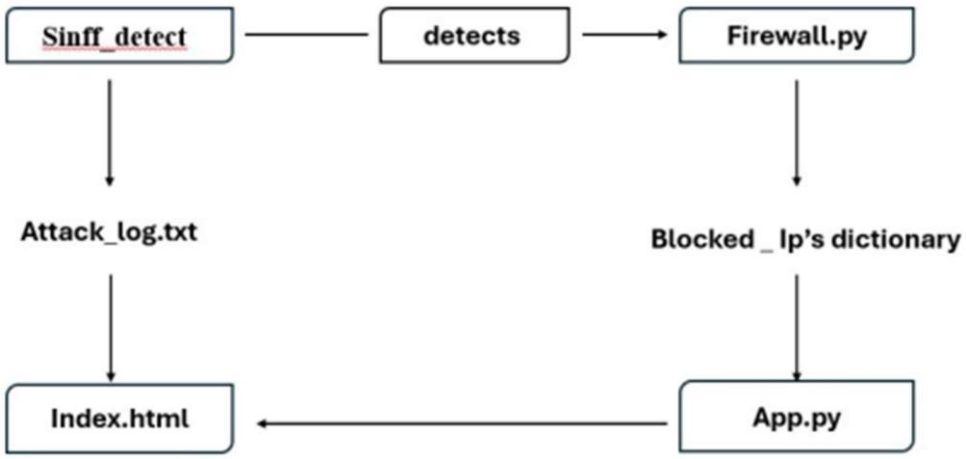
Configuration and Threshold Management Module - Adjustable thresholds for SYN, UDP, ICMP traffic. Secure storage of bot token, chat ID, and Gemini AI key. Ensures flexibility and easy updates to the system's behavior.

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 04 | April – 2025 DOI: 10.55041/ISJEM02959
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

5. ARCHITECTURE DIAGRAM

It can provide a plan is made to mimic and react to DDoS attacks in a safe virtual setting. Using Windows Subsystem for Linux (WSL), the attacker's system is configured to simulate DDoS traffic, including TCP, UDP, and ICMP floods, using tools like hping3. The target system, which is running Kali Linux within VirtualBox, is the destination of this traffic. A Python-based detection script that examines traffic patterns for indications of an attack is used by the target machine to continuously monitor network packets.

Fig no.1 ARHITECTURAL DIAGRAM



6. IMPLEMENTATION

In order to replicate actual DDoS attacks and assess an automated defense mechanism, the suggested system, "Automated DDoS Simulation in a Virtual Environment Using Python and Telegram Bot," was implemented in a controlled and modular fashion. A dual-system environment makes up the setup. The target system is hosted on Kali Linux in Virtual Box, while the attacker system runs Ubuntu on Windows Subsystem for Linux (WSL). In order to enable smooth traffic exchange for attack simulation, both machines are set up on a shared virtual network. The tool hping3 is used on the attacker's computer to mimic DDoS attacks. This tool can create a variety of attack traffic types, such as ICMP ping floods, UDP floods, and TCP SYN floods.

- App.Py
- Firewall.Py
- Sniff_Detect.Py
- Reporter.Py
- Config.Py
- Index.Html

Using the Scapy library, an intrusion detection script written in Python continuously watches incoming network packets on the target side. This script, which runs as sniff_detect.py, the various types of traffic (SYN, UDP, and ICMP) and

Performs real-time packet header inspections. A single IP address is marked as malicious if its traffic surpasses a predetermined threshold, which is 100 for SYN, 200 for UDP, or 150 for ICMP.

CONCLUSION

An efficient and user-friendly method for simulating and preventing DDoS attacks in a controlled environment is provided by the project "Automated DDoS Simulation in a Virtual Environment Using Python and Telegram Bot." The system allows for automated firewall responses and realistic attack simulations without endangering real-world infrastructure by utilizing tools like iptables, hping3, and Python scripting in a virtual environment.

Important features like automatic IP blocking, real-time traffic monitoring, and remote control through Telegram give users a practical grasp of both offensive and defensive cybersecurity strategies. Usability and situational awareness are improved with the addition of a web-based dashboard and AI-integrated Telegram alerts.

This project provides a useful framework for experimentation and education in cybersecurity.

7. REFERENCES

[1] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," ACM SIGCOMM Computer Communication Review, vol. 34, no. 2, pp. 39–53, Apr. 2004. doi: 10.1145/997150.997156

[2] C. Douligeris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: classification and state-of-the-art," Computer Networks, vol. 44, no. 5, pp. 643–666, Apr. 2004. doi: 10.1016/j.comnet.2003.10.003

[3] H. Wang, C. Jin, and K. G. Shin, "Defense against spoofed IP traffic using hop-count filtering," IEEE/ACM Transactions on Networking, vol. 15, no. 1, pp. 40–53, Feb. 2007. doi: 10.1109/TNET.2006.890134

[4] R. Kumar and A. Vasan, "Detection of DDoS attacks using machine learning algorithms," in Proc. 2020 4th Int. Conf. Comput. Methodologies Commun. (ICCMC), Erode, India, 2020, pp. 730–735. doi: 10.1109/ICCMC48092.2020.ICCMC-000134

[5] S. Behal, K. Kumar, M. Sachdeva, and K. Singh, "Trends in detection of DDoS attacks," Defence Science Journal, vol. 67, no. 6, pp. 562–567, Nov. 2017. doi: 10.14429/dsj.67.11023

[6] B. B. Gupta, R. C. Joshi, and M. Misra, "Cloud computing vulnerabilities: Taxonomy, solutions and challenges," in Proc. 2013 Int. Conf. Adv. Comput., Commun. Informatics (ICACCI), Mysore, India, 2013, pp. 365–373. doi: 10.1109/ICACCI.2013.6637215

[7] B. Choudhury and R. Shukla, "An approach to detect and prevent DDoS attacks in SDN using machine learning," J. Inf. Secur. Appl., vol. 55, p. 102582, Mar. 2020. doi: 10.1016/j.jisa.2020.102582

[8] M. Sarhan and N. Z. Jhanjhi, "Machine learning approaches for detection of DDoS attacks: 40 A survey," J. King Saud Univ. - Comput. Inf. Sci., Apr. 2021. doi: 10.1016/j.jksuci.2021.04.004

[9] S. Ho, S. Chatterjee, T. Chakraborty, H. Wong, and L. Hui, "A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network," IEEE Open J. Comput. Soc., vol. 2, pp. 14–25, Jan. 2021. doi: 10.1109/OJCS.2021.3051501