KLE Technological University

A Course Project Report on

Simulating DDoS Attacks to Test Server Resilience

Mini Project (15ECW301)

Submitted by

Team Number: A-8		
Name	**SRN**	**Roll no**
Mohd Qadir Ternikar	02FE22BCS053	15
Aisha Karigar	02FE22BCS009	04
Vanashree A N	02FE22BCS171	63
Niranjan D	02FE23BCS059	17

Under the guidance of

Prof.Vaishali Parab

Faculty in-charge:

Prof.Shankar Biradar

Department of Computer Science and Engineering

KLE Technological University's Dr. M. S. Sheshgiri College of Engineering and Technology, Belagavi – 590 008.

2024-2025

KLE
TECHNOLOGICAL UNIVERSITY
Creating Value, Leveraging Knowledge
Belagavi Campus

Dr.M.S.Sheshgiri College of Engineering & Technology
Department of Computer Science & Engineering

DECLARATION

We hereby declare that the matter embodied in this report entitled "***Simulating DDOS Attack To Test Server Resilience***" submitted to KLE Technological University for the course completion Mini Project(15ECSW301) in the 5th Semester of Computer Science and Engineering is the result of the work done by us in the Department of Computer Science and Engineering, KLE Dr. M. S. Sheshgiri College of Engineering,Belagavi under the guidance of Prof Vaishali Parab, Department of Computer Science and Engineering. We further declare that to the best of our knowledge and belief, the work reported here doesn't form part of any other project based on which a course or award was conferred on an earlier occasion on this by any other student(s). Also, the results of the work are not submitted for the award of any course, degree, or diploma within this or in any other University or Institute. We hereby also confirm that all of the experimental work in this report has been done by us.

Belagavi – 590 008
Date: 04th January 2025

Mohd Qadir Ternikar
(02FE22BCS053)

Vanashree A N
(02FE22BCS171)

Aisha Karigar
(02FE22BCS009)

Niranjan Desai
(02FE22BCS059)

KLE TECHNOLOGICAL UNIVERSITY
Creating Value, Leveraging Knowledge
Belagavi Campus
Dr.M.S.Sheshgiri College of Engineering & Technology
Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled "***Simulating DDOS Attack To Test Server Resilience"*** submitted to KLE Technological University's Dr. MSSCET, Belagavi for the partial fulfillment of the requirement for the course – Mini Project (15ECSW301) by Mohd Qadir, Aisha K , Vanashree A N, Niranjan D students in the Department of Computer Science and Engineering, KLE Technological University's Dr. MSSCET, Belagavi, is a bonafide record of the work carried out by them under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any other course completion.

Belagavi – 590 008
Date: 7th January 2025

Prof. Vaishali Parab	Prof. Shankar Biradar
(Course Guide)	(Course Coordinator)

Dr. Rajashri Khanai
(Head of the Department)

Abstract

Denial of Service (DoS) / Distributed Denial of Service (DDoS) attacks remain a persistent threat to network infrastructure, causing severe service disruptions and resource exhaustion. This study explores a range of DDoS attack methodologies across Layer 7 (Application Layer) and Layer 4 (Transport Layer), including advanced techniques such as Cloudflare Bypass (cfb), HTTP/2 request floods, and spoofing attacks, executed with custom Python scripts. By systematically launching these attacks on test systems, we assessed their impact on network latency, downtime, packet rates (PPS), bandwidth consumption (BPS), and resource utilization (CPU and memory). Comprehensive monitoring was conducted during the attacks, leveraging tools for real-time traffic analysis and resource tracking to capture critical performance metrics. This enabled a detailed understanding of how various attack vectors compromise system stability and degrade network performance. Our findings highlight distinct signatures and patterns associated with each attack type, providing valuable insights into their operational characteristics. This study underscores the importance of robust monitoring systems that detect and analyze attack behavior in real-time. The results serve as a reference for security practitioners to refine their detection mechanisms and response strategies against increasingly sophisticated DDoS threats.

Chapter 1 : Introduction

1.1 Background

DDoS attacks aim to flood a web service with malicious traffic, making it unavailable to legitimate users. These attacks can severely disrupt operations and cause significant financial losses. DDoS attacks use a distributed network of compromised devices (often called botnets) to send massive amounts of traffic to a web server, overwhelming its capacity. Attackers can use various techniques to make malicious traffic appear legitimate, complicating the detection and mitigation process.

1.2 Problem Statement

The increasing dependency on digital platforms across industries has amplified the risks associated with cyberattacks, particularly DoS and DDoS attacks. These attacks exploit the fundamental vulnerabilities of internet-connected systems, making them one of the most prevalent threats to cybersecurity today. Their simplicity, coupled with the significant damage they can inflict, makes them attractive to malicious actors ranging from individual hackers to organized groups and even nation-states.

1.2.1 Objectives

- To simulate DoS attacks on a web server in a controlled environment.
- To analyze the web server's response and resilience against various DoS attacks.
- To assess the effectiveness of existing defense mechanisms, including firewalls, Intrusion Detection Systems (IDS), and traffic monitoring tools.
- To identify vulnerabilities and suggest improvements to enhance web server security and availability.

Chapter 2: Literature Survey

.

1. **Comprehensive Review of DDoS Attacks on Web Platforms (Singh & Gupta [1])**

- **Focus Area:** This study examines Distributed Denial of Service (DDoS) attacks and their impacts on various web platforms.
- **Proposed Methods/Models:** The research analyzes different attack vectors used for launching DDoS attacks across platforms.
- **Advantages:** The paper provides a broad overview of DDoS scenarios, making its findings applicable across diverse industries.
- **Limitations/Gaps:** The study lacks practical implementation details and does not provide specific examples of the attack models discussed.

2. **DDoS Attack Taxonomy and Types (S. Kumar et al. [2])**

- **Focus Area:** This research categorizes DDoS attacks into distinct types.
- **Proposed Methods/Models:** It classifies DDoS attacks into three major categories: volumetric attacks, protocol-based attacks, and application-layer attacks.
- **Advantages:** The paper delivers a detailed breakdown of attack types and associated vulnerabilities.
- **Limitations/Gaps:** It does not address recent sophisticated techniques, such as AI-powered botnets, leaving a gap in the coverage of emerging threats.

3. **IoT-Based DDoS Attack Strategy (Huang et al. [3])**

- **Focus Area:** This paper explores IoT-based DDoS attack models.
- **Proposed Methods/Models:** The study proposes a botnet growth model and optimized attack strategies tailored for IoT devices.
- **Advantages:** The approach is low-cost and effective, particularly for resource-constrained attack scenarios.
- **Limitations/Gaps:** The proposed model assumes ideal attack conditions, which limits its applicability to real-world scenarios.

4. **DDoS Attacks in IoT Networks (Kumari & Jain [4])**:

- **Focus Area**: Discusses DDoS attacks targeting Internet of Things (IoT) devices.
- **Key Insights**: Explores the creation of botnets and the variations of DDoS attacks within IoT ecosystems.
- **Strengths**: Provides a detailed analysis of vulnerabilities specific to IoT environments.
- **Limitations**: Primarily focuses on existing attack types without delving deeply into novel attack vectors.

.

5. **Advanced DDoS Attack Methods (Aamir & Zaidi [5]):**

- **Focus Area**: Centers on application-layer DDoS attacks.
- **Key Insights**: Investigates methods such as HTTP floods that exploit legitimate-looking traffic to conduct attacks.
- **Strengths**: Highlights the challenges of detecting more sophisticated attack strategies.
- **Limitations**: Lacks empirical validation or testing of the proposed attack strategies.

6. **DoS Attacks on Control Systems (Cetinkaya et al. [6])**

- **Focus Area**: Discusses DoS attack models targeting control systems.
- **Key Insights**: Explores specific attack mechanisms such as jamming and packet-dropping within control system environments.
- **Strengths**: Provides an in-depth analysis of vulnerabilities in control systems, which are critical for infrastructure and industrial processes.
- **Limitations**: Primarily theoretical in nature, which may limit the practical applicability or real-world validation of the proposed models.

7. **NTP Amplification Attacks (Jimoh & Ahmed [7])**

- **Focus Area**: Investigates DDoS attacks that leverage Network Time Protocol (NTP) servers for amplification.
- **Key Insights**: Highlights how attackers exploit NTP servers to amplify traffic significantly, making attacks more impactful.

- **Strengths**: Demonstrates the high amplification factor and its potential to overwhelm targets effectively.
- **Limitations**: Focuses solely on NTP as an amplification vector without addressing other possible methods, reducing its comprehensiveness

.

8. **DDoS in IoT and SDN Environments (Gocher et al. [8])**:

- **Focus Area**: Investigates DDoS attacks targeting Internet of Things (IoT) and Software-Defined Networking (SDN) environments.
- **Key Insights**: Explores high-volume DDoS attacks that use compromised devices within IoT and SDN to flood networks.
- **Strengths**: Emphasizes emerging technologies like SDN, highlighting their vulnerability to DDoS attacks.
- **Limitations**: Limited scope as it does not address DDoS attacks in environments outside IoT and SDN.

9. **DoS/DDoS Attacks in IPv4/IPv6 Networks (Tripathi & Mehtre [9])**:

- **Focus Area**: Examines the impact of DoS/DDoS attacks on the availability of IPv4 and IPv6 networks.
- **Key Insights**: Provides a detailed classification of attack types in both IPv4 and IPv6 protocols.
- **Strengths**: Comprehensive analysis of attack vectors in both network types.
- **Limitations**: Does not explore modern attack strategies, such as multi-vector DDoS attacks, which combine several attack types for higher effectiveness.

Chapter 3: Design space

3.1 B P 1.1 Understanding The Problem

3.1.1Problem Space

The increasing dependency on digital platforms across industries has amplified the risks associated with cyberattacks, particularly DoS and DDoS attacks. These attacks exploit the fundamental vulnerabilities of internet-connected systems, making them one of the most prevalent threats to cybersecurity today. Their simplicity, coupled with the significant damage they can inflict, makes them attractive to malicious actors ranging from individual hackers to organized groups and even nation-states.

A major aspect of the problem lies in the accessibility of attack tools. With tools like LOIC (Low Orbit Ion Cannon), HOIC (High Orbit Ion Cannon), and botnets such as Mirai being readily available online, executing these attacks requires minimal technical expertise. Additionally, the advent of Ransom DDoS (RDDoS) attacks, where attackers demand payment to cease their operations, has made these attacks a lucrative avenue for cybercriminals.

DoS/DDoS attacks are not limited to targeting websites; they also extend to critical infrastructure. Sectors like healthcare, finance, energy, and transportation are particularly vulnerable, as even short downtimes can have catastrophic consequences. For example, a successful attack on a hospital's system could delay critical care delivery, endangering lives. Similarly, attacks on financial institutions can lead to widespread economic disruptions.

Another dimension of the problem is the collateral damage caused by these attacks. The interconnected nature of digital ecosystems means that an attack on one entity can indirectly impact others. The Dyn attack in 2016 is a prime example, where the targeting of a DNS provider caused outages across multiple unrelated platforms, including Twitter, Reddit, and Spotify. Such cascading effects demonstrate the far-reaching implications of DoS/DDoS attacks, making them a critical area of concern for businesses and governments alike**.**

3.1.2State-of-the-Art

The field of cybersecurity has witnessed significant advancements in countering DoS and DDoS attacks, but attackers have simultaneously evolved their techniques. The current stateof-the-art encompasses both offensive and defensive strategies, with a strong focus on leveraging automation and machine learning.

On the attack side, the use of botnets has reached unprecedented levels. Modern botnets are highly sophisticated, often comprising millions of compromised devices, including IoT devices with minimal security. The Mirai botnet, for example, exploited weak default credentials in IoT devices, orchestrating one of the largest DDoS attacks in history. Attackers also employ advanced evasion techniques, such as encrypting their traffic or mimicking legitimate user behavior, to bypass traditional detection mechanisms.

Defensive strategies, on the other hand, have increasingly incorporated real-time monitoring and predictive analytics. Machine learning models are trained on large datasets to identify anomalous traffic patterns indicative of an attack. Behavioral analytics, for instance, can distinguish between normal user activity and malicious traffic based on metrics like request rates, geographic origin, and device fingerprints.

Mitigation techniques have also become more dynamic. Solutions like rate limiting, content delivery networks (CDNs), and load balancers are now commonly used to distribute traffic and reduce the impact of an attack. Furthermore, cloud-based DDoS protection services, such as those offered by AWS Shield and Cloudflare, provide scalable defenses that adapt to the intensity of an attack.

Despite these advancements, challenges remain. High false-positive rates in detection systems can lead to unnecessary disruptions, while the cost of implementing advanced defenses may be prohibitive for smaller organizations. Moreover, the increasing use of AI by attackers to adapt their methods highlights the need for continuous innovation in defense strategies**.**

3.1.3 External and Internal Factors

External Factors

Externally, the global internet landscape provides attackers with numerous opportunities to exploit vulnerabilities. The prevalence of unsecured IoT devices is a significant factor. Many of these devices, from smart cameras to connected appliances, are deployed with weak or default credentials, making them easy targets for botnet recruitment. Attackers also take advantage of misconfigured servers and open resolvers, which can be used in amplification attacks to generate massive traffic volumes.

The availability of attack tools and services on the dark web has further lowered the barrier to entry for executing DoS/DDoS attacks. Botnets for hire, known as "DDoS-for-hire" services or booters, allow even unskilled individuals to launch sophisticated attacks for a fee. This commoditization of cybercrime has contributed to the growing frequency of attacks.

Internal Factors

Internally, organizational weaknesses often contribute to the success of DoS/DDoS attacks. Many organizations lack robust cybersecurity measures, such as updated software, properly configured firewalls, and intrusion detection systems. Even when such measures are in place, inadequate monitoring and response capabilities can allow an attack to escalate before it is mitigated.

Human factors also play a role. Employees may inadvertently compromise network security by clicking on phishing links or using weak passwords, giving attackers a foothold within the organization. Additionally, a lack of cybersecurity awareness and training can leave staff unprepared to recognize and respond to potential threats.

Another critical internal factor is the scalability of the organization's infrastructure. Systems with limited bandwidth or outdated hardware are more susceptible to being overwhelmed during an attack. Without adequate redundancy and load balancing, even small-scale attacks can cause significant disruptions.

To address these factors, organizations must adopt a multi-faceted approach. This includes implementing technical solutions, such as network segmentation and anomaly detection, as well as fostering a culture of cybersecurity awareness. Regular vulnerability assessments and penetration testing can also help identify and mitigate risks before they can be exploited by attackers..

3.2 BP 1.2 Problem Flow

3.2.1 Solution Space

The problem of executing and simulating Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks revolves around leveraging technical tools, frameworks, and architectures to study and analyze attack methodologies comprehensively. This approach enables researchers and cybersecurity professionals to better understand the complexities of attack vectors, bottlenecks in target systems, and weaknesses in network architectures.

The solution involves constructing a controlled environment where these attacks can be performed ethically and safely. The environment must emulate real-world conditions, including variable network topologies, bandwidth constraints, and diverse target systems, to provide insights into attack mechanism

Designing a Controlled Testbed:

A dedicated testbed is essential for safely deploying DoS/DDoS attacks without harming live systems. This testbed can be built using virtualized environments or isolated networks that mimic actual server-client architectures. Tools such as Mininet, VMWare, or Kubernetes clusters allow the creation of scalable environments where multiple bots or devices can generate attack traffic.

Within this space, the system must simulate a range of attack strategies, including volumetric attacks, protocol-based attacks, and application-layer attacks. This allows researchers to observe how different attack types exploit vulnerabilities in the target system. For example, a TCP SYN Flood attack overwhelms the TCP handshake mechanism, while an HTTP GET flood targets application-layer resources.

Customizable Attack Modules:

Developing modular tools for executing attacks allows for flexibility and scalability. Each module is designed to replicate specific aspects of an attack:

Traffic Generation Module: Simulates botnet behavior by sending high volumes of traffic with varying patterns (e.g., uniform, bursty, or randomized).

Payload Delivery Module: Customizes the content of attack packets to exploit specific vulnerabilities, such as malformed packets for crashing a target server.

Command-and-Control (C&C) Simulation: Mimics the behavior of real-world C&C servers to coordinate attack traffic from multiple sources.

The modular design ensures that different attack scenarios can be executed with minimal changes to the system, enabling researchers to study a wide range of attack vectors efficiently.

Real-Time Monitoring of Impact

To measure the efficacy of the attacks, the test environment must include monitoring tools that provide real-time metrics. These metrics may include:

Latency and Throughput: Observing how attack traffic affects the response time and data transmission rate of the target.

Resource Utilization: Tracking CPU, memory, and bandwidth usage on the target system to identify bottlenecks.

Service Availability: Measuring downtime and accessibility issues caused by the attack.

Tools like Wireshark, NetFlow, or Prometheus with Grafana can be integrated into the environment to visualize and analyze the impact in real-time.

Diagram Details

Attack Workflow: A detailed diagram showcasing the flow of traffic from the attacker (or botnet) to the target system. Include the role of C&C servers, traffic generators, and payload modifiers

3.2.2 Features:

Load Balancing:

Load Balancing mimics the distribution of attack traffic across multiple targets to prevent any single system from being overwhelmed. The Traffic Generation Module creates high volumes of traffic, and by distributing this traffic across various servers or network nodes, the system simulates how an attack might be handled in a real-world distributed environment. This feature ensures that the simulation accurately reflects how a real-world DDoS attack would overload the network bandwidth and resources. By redirecting traffic to multiple points, it provides insight into how an attacker might test the resilience of a distributed network or infrastructure

Layered Attack Strategy:

A Layered Attack Strategy tests how multiple attack vectors, targeting different layers of the OSI model, impact the target system. The Traffic Generation Module simulates volumetric flooding at the network layer, while the Botnet Simulation Module and Payload Customization Module extend the attack to exploit vulnerabilities at the transport and application layers. This approach enables the simulation of attacks like SYN floods at the transport layer or HTTP request floods at the application layer. The ability to combine different attack types within a single test ensures that multiple layers of the system are overwhelmed simultaneously, creating a comprehensive and realistic attack scenario. This is critical for evaluating how each layer of a system reacts to various attack types and how resilient the infrastructure is to multi-layered attacks

Scalability and Flexibility:

The Scalability and Flexibility of the attack simulation system allow it to handle a wide range of scenarios, from low-volume DoS attacks to massive DDoS attacks involving thousands of bots. By customizing key parameters in the Botnet Simulation Module and Traffic Generation Module, users can adjust the number of bots, the volume of traffic, and the type of attack to match real-world conditions. This flexibility enables the system to simulate large-scale attacks, where millions of requests or packets are generated from distributed nodes. The ability to scale the attack traffic helps users assess how systems would respond to extreme volumes of malicious activity and determine their threshold for handling high-impact DoS or DDoS attacks.

Performance Metrics and Logging:

Performance Metrics and Logging are crucial for understanding the impact of the simulated attack. The system tracks key metrics such as the number of requests sent, response times, server resource usage, and network bandwidth consumption during the attack. These metrics help assess how the target handles the traffic and where vulnerabilities may lie. The Traffic Generation Module provides real-time feedback on the attack's progress, allowing for adjustments if the attack is not having the desired effect. Logs capture detailed information on every stage of the attack, from traffic generation to the response of the target system. These logs can then be analyzed to identify weaknesses in the infrastructure and provide insights into the effectiveness of defense mechanisms

Integration with External Tools:

The system can be integrated with External Tools to enhance its attack capabilities. For example, integration with network monitoring tools allows users to track network traffic in real-time, providing additional insights into how the attack affects network performance. Additionally, the system can integrate with threat intelligence platforms to import updated attack patterns, ensuring that the simulations remain relevant and reflective of the latest DDoS tactics. This integration expands the scope of the attack simulations, providing a more comprehensive understanding of how current threats evolve over time.

Realistic Attack Simulation:

The system's ability to generate Realistic Attack Scenarios is essential for accurately testing how a target would react to different types of malicious traffic. For example, the Payload Customization Module allows users to generate malformed or exploitative packets, such as oversized ICMP packets or DNS amplification requests. These realistic attack patterns can then be used to simulate scenarios like the Ping of Death or DNS amplification. The Botnet Simulation Module also enables the creation of realistic botnets, mimicking how real-world attackers use large numbers of compromised devices to execute distributed attacks. The traffic generated in these simulations closely mirrors the behavior of actual DDoS attacks, allowing for more meaningful testing and analysis.

3.2.3 Modules

1. **Traffic Generation**

The Traffic Generation is at the heart of a DoS/DDoS attack simulation, responsible for generating a massive volume of traffic to target the victim system. It offers flexibility in crafting various types of attack patterns, such as volumetric flooding, where a large number of packets are sent to exhaust network bandwidth, or protocol exploitation, where specific vulnerabilities in protocols like TCP or UDP are leveraged to launch attacks like SYN floods. Additionally, the module can simulate application-layer attacks, such as flooding a server with HTTP or DNS requests to overload its processing capacity. The customization capabilities of this module allow for fine-tuning parameters such as packet size, payload, frequency, and even source IP addresses, making it highly versatile for different testing and simulation scenarios. By mimicking realistic attack conditions, it helps test the resilience of systems against a wide variety of DoS and DDoS tactics.

Command-and-Control (C&C)

The Command-and-Control (C&C) plays a crucial role in simulating DDoS attacks, where multiple compromised devices (bots) are coordinated to amplify the attack. This module functions by managing a network of virtual bots, assigning each one specific task such as targeting different ports or using different packet types. It ensures synchronization between these bots, allowing them to launch coordinated bursts of traffic, thus maximizing the attack's impact. This module is vital for testing DDoS scenarios that involve botnets, as it simulates real-world attack behaviours where large numbers of compromised machines collaborate in executing complex attacks.

Payload Unit

The Payload Customization allows for generating attack payloads that exploit specific weaknesses in target systems. By creating malformed packets with invalid headers or payloads, this module can destabilize network components and cause disruptions. Additionally, it can design protocol-specific payloads, targeting vulnerabilities in protocols like ICMP, HTTP, or DNS. For example, sending oversized ICMP packets often referred to as the "Ping of Death," can overload a system, while fragmented packets may cause issues in reassembly, further contributing to network instability. This module’s ability to tailor attack payloads enhances the ability to test specific vulnerabilities and assess the effectiveness of security systems in identifying and mitigating such attacks.

Wifi Target

The Wi-Fi DoS Attack targets vulnerabilities specific to wireless networks, taking advantage of the shared nature of the communication medium. This module can carry out de-authentication attacks by sending spoofed de-authentication frames to disconnect devices from access points (APs). It allows attackers to specify either individual devices or broadcast attacks to all connected devices, causing widespread disruption. Another attack technique, beacon flooding, generates rogue networks by sending fake beacon frames, confusing legitimate devices, and exhausting available channels. Finally, the module includes jamming attacks, where attackers broadcast signals on the same frequency as the target Wi-Fi network, causing interference and making the network unusable. These Wi-Fi-based attacks are highly effective in environments with high network density, where they can severely degrade the performance of wireless networks.

3.2.4 Relationship Between Features

The features of the solution are intricately interlinked, each complementing and enhancing the others to provide a comprehensive and realistic DoS/DDoS attack simulation framework. Here's how they relate to one another:

Load Balancing & Layered Attack Strategy

Load Balancing and Layered Attack Strategy work hand-in-hand to ensure that the attack is both realistic and manageable. The Traffic Generation Module creates high volumes of traffic, which is then distributed across various servers through Load Balancing. This mimics how DDoS attacks are typically handled in real-world distributed environments. Meanwhile, the Layered Attack Strategy ensures that traffic is not only volumetric but also targets multiple layers of the OSI model, including network, transport, and application layers. Together, these features enable the system to simulate the impact of large-scale, multi-layered attacks while ensuring that the target infrastructure is not overwhelmed by excessive traffic at any single point.

Scalability and Flexibility & Realistic Attack Simulation

Scalability and Flexibility enhance the ability of the system to handle different types of attack simulations, from small-scale to large-scale DDoS attacks. The flexibility allows users to define attack parameters like the number of bots or the traffic volume. When combined with Realistic Attack Simulation, which generates precise attack patterns (such as malformed packets or DNS amplification), the system can simulate a wide array of real-world DDoS attacks. This means users can replicate specific attacks with varying scales and complexities, offering a versatile environment for testing and improving network defence.

Performance Metrics and Logging & Realistic Attack Simulation

The data captured by Performance Metrics and Logging is essential for analyzing the effectiveness of the Realistic Attack Simulation. While the Payload Customization Module creates sophisticated attack vectors like DNS amplification or SYN floods, the Performance Metrics and Logging feature tracks how the system performs under these attacks. Key metrics like server resource usage, response time, and network bandwidth are monitored in real-time, giving a direct feedback loop on how the attack impacts the system. This relationship ensures that simulated attacks are not just theoretical but are actually tested against the target's performance, highlighting areas of weakness that need strengthening.

Integration with External Tools & Scalability and Flexibility

The Integration with External Tools and Scalability and Flexibility features are closely linked in terms of enhancing attack simulation capabilities. By integrating network monitoring tools, threat intelligence platforms, and other external resources, users can test their systems under conditions that reflect the latest attack patterns. The Scalability and Flexibility of the system allow it to adjust the scale and complexity of these external attack patterns, ensuring that the simulations can match the evolving nature of cyber threats. External tools provide the necessary data to fine-tune attack simulations, and the system's scalability ensures that it can adapt to these enhanced scenarios.

Layered Attack Strategy & Performance Metrics and Logging

The Layered Attack Strategy generates traffic at multiple layers of the OSI model, ensuring that all aspects of the target system are tested for vulnerabilities. Meanwhile, the Performance Metrics and Logging tracks the response of the system to these layered attacks. For example, when volumetric traffic targets the network layer, and protocol weaknesses are exploited at the transport or application layer, the logs and performance metrics capture how well the system handles each layer's specific vulnerabilities. This combined insight helps identify which layers are most susceptible to attacks and require further hardening.By interlinking these features, the system offers a holistic approach to DoS/DDoS attack simulation, providing a more realistic, flexible, and scalable environment to test and strengthen network defenses.

3.3 BP 1.3 Interface Designing

The DoS/DDoS project interface is designed to provide a seamless and efficient platform for conducting controlled experiments on denial-of-service attacks. This document outlines the layout and operations of the interface, emphasizing usability and integration with the project’s core tools, including Kali Linux, Ubuntu, and an Apache server running in Oracle VM.

Purpose of the Interface

The interface is a centralized control point for setting up, launching, monitoring, and analyzing DoS/DDoS attacks in a controlled environment. The goal is simplifying complex attack setups while providing clear visibility into the ongoing process.

Key Features

User-Friendly Navigation:

A clean and intuitive design enables users to access various functions like attack configuration, monitoring, and logs with minimal effort.

A navigation bar links all essential components, including the Home Page, Attack Configuration, Monitoring Dashboard, and Reports.

Tool Integration:

The interface connects directly with the Python script on a Kali Linux system, utilizing its capabilities to simulate real-world DoS/DDoS scenarios.

An Apache server hosted in Oracle VM on Ubuntu acts as the target, ensuring a secure and isolated environment for testing.

Customizable Options:

Users can define attack parameters such as the target URL/IP, thread count, and duration, directly from the interface.

Real-Time Feedback:The interface displays metrics like ongoing request counts and server response times, helping users monitor the attack's impact.

Operational Environment

The interface operates seamlessly within the following setup:

Operating Systems: Kali Linux for attack execution and Ubuntu for hosting the Apache server.

Virtualization: Oracle VM is used to create an isolated environment, ensuring experiments remain secure and do not impact external networks.

Scripting Tools: The primary attack tool is a Python-based script , which is invoked directly through the interface.

This setup ensures compatibility, security, and optimal performance during all stages of the experiment.

3.4 BP 1.4 Inclusive Designing

Introduction to Inclusive Digital Society

Inclusive design plays a critical role in fostering a digital society where everyone, regardless of their background, abilities, or resources, has equitable access to technology. In the context of cybersecurity, particularly in projects simulating DoS/DDoS attacks, inclusivity ensures that tools and interfaces are usable and accessible by a diverse audience. This project integrates inclusive design principles to create a controlled platform for simulating denial-of-service attacks, utilizing tools like Kali Linux, Ubuntu, and an Apache server in Oracle VM. The goal is to bridge technical barriers while ensuring that ethical and responsible practices are upheld.

The Role of Inclusive Design in Cybersecurity

The cybersecurity domain often involves complex technical processes that can be challenging for beginners. Inclusive design simplifies these processes by creating user-friendly interfaces that cater to different expertise levels. For the DoS project, principles like usability, accessibility, and adaptability are central. The interface offers intuitive navigation, clear terminology, and adaptable configurations to accommodate beginners and experts alike. This ensures that the tools are not only functional but also approachable for educational and research purposes.

Product Users and Stakeholders

The primary users of this project include cybersecurity students, researchers, and educators. For students, the interface provides a platform to learn about attack dynamics through practical simulations. Educators benefit from its ability to demonstrate real-world scenarios in a controlled environment. Secondary stakeholders include training institutions and organizations conducting security assessments. To meet diverse user needs, the interface incorporates features like guided tutorials for beginners and advanced settings for professionals, making it versatile and inclusive.

Designing for Diverse Product Users

Addressing the needs of diverse users is at the heart of inclusive design. The interface accommodates varying levels of technical expertise through its design. Beginners can utilize simplified workflows and step-by-step guides, while advanced users can access detailed configurations and direct integrations with tools like Hammer.py. Feedback mechanisms are built into the interface, allowing users to share their experiences and suggest improvements, thereby fostering a continuously evolving, user-centric platform.

Ethical Considerations in Data Sources:In projects involving cybersecurity, ethical handling of data is paramount. The DoS project operates within a controlled environment, avoiding real-world targets and relying on sandboxed systems like an Apache server on Oracle VM. Data collection is limited to essential information, such as logs for monitoring and analysis, ensuring no personal or sensitive data is involved. Transparency is maintained by informing users about the data collection process and offering anonymized options. This approach aligns with ethical research practices and reinforces trust among users.

3.4.1 Society

The project serves as a wake-up call for society about the growing prevalence of DoS/DDoS attacks and their potential to cripple critical infrastructure. These attacks often target sectors like healthcare, finance, education, and government services, impacting millions.

Promoting Cybersecurity Culture: By studying and addressing these attacks, the project fosters a culture of cybersecurity awareness among individuals, organizations, and policymakers. It emphasizes the importance of proactive defense strategies.

Economic Implications: The financial losses caused by service downtime and data breaches due to DDoS attacks are immense. This project indirectly supports economic resilience by helping businesses understand and prepare for such attacks.

Ethical Implications: The knowledge gained through this project carries a dual-use risk, stressing the need for ethical guidelines to ensure that it is only used for protective and educational purposes.

3.4.2 End Users

Website Owners and Admins: Gain insights into detecting and mitigating attack patterns, reducing downtime and losses.

Network Security Professionals: Enhance their ability to analyze and respond to network anomalies in real time.

End Users of Services: Indirectly benefit from improved website uptime and reduced disruption.

Educational Institutions and Researchers: Use this project as a learning tool for understanding attack mechanics and defenses.

Use Case Scenarios:

Real-time monitoring of traffic to identify potential threats.

Implementing traffic analysis tools to predict and prevent future attacks.

Training cybersecurity professionals in DoS/DDoS mitigation techniques.

Practical Benefits:

1. Reduced service interruptions.
2. Faster recovery from attacks.
3. Insights for designing more resilient systems.
4. Privacy and Security

Protecting User Data:

While monitoring traffic to detect anomalies, it is crucial to ensure that personal user data remains secure and anonymized to comply with privacy regulations like GDPR and CCPA.

Ethical Traffic Analysis:

The project incorporates methods that analyze patterns without infringing on the privacy of legitimate users. Monitoring tools are designed to detect malicious activity while maintaining transparency.

Building Secure Systems:

Insights gained from traffic monitoring and analysis help design robust systems that can withstand future attacks, ensuring long-term security for both service providers and users.

Avoiding Abuse of Knowledge:

Knowledge about DoS/DDoS attacks is a double-edged sword. While it equips defenders with the tools to protect systems, safeguards must be in place to prevent its misuse for malicious purposes.

Legal Compliance:

All traffic monitoring and mitigation efforts should align with cybersecurity laws and ethical standards, ensuring a balance between security and user rights.

Empowering End Users:

The project provides actionable recommendations for users to protect their online presence, such as avoiding suspicious links, using VPNs, and ensuring strong passwords.

3.4.3 Privacy and Security

Protecting User Data:

While monitoring traffic to detect anomalies, it is crucial to ensure that personal user data remains secure and anonymized to comply with privacy regulations like GDPR and CCPA.

Ethical Traffic Analysis:

The project incorporates methods that analyze patterns without infringing on the privacy of legitimate users. Monitoring tools are designed to detect malicious activity while maintaining transparency.

Building Secure Systems:

Insights gained from traffic monitoring and analysis help design robust systems that can withstand future attacks, ensuring long-term security for both service providers and users.

Avoiding Abuse of Knowledge:

Knowledge about DoS/DDoS attacks is a double-edged sword. While it equips defenders with the tools to protect systems, safeguards must be in place to prevent its misuse for malicious purposes.

Legal Compliance:

All traffic monitoring and mitigation efforts should align with cybersecurity laws and ethical standards, ensuring a balance between security and user rights.

Empowering End Users:

The project provides actionable recommendations for users to protect their online presence, such as avoiding suspicious links, using VPNs, and ensuring strong passwords.

Chapter 4: Requirement Engineering

4.1 Actors

In the context of this business-oriented system, the actors refer to all individuals or entities that interact with the system. These actors include both internal (company-specific) and external users, each with different roles and access requirements.

Primary Actors:

Security Engineers:

These are the core users who will interact with the system to simulate DoS/DDoS attacks against corporate infrastructure. They configure parameters for attacks (e.g., target IPs, attack types) and monitor the results. Their goal is to identify vulnerabilities in the system and improve security protocols.

System Administrators:

These actors maintain the underlying infrastructure, ensuring that the Apache server, virtual machines, and other components are operational. They may also configure the system to simulate attacks in various environments and monitor server performance during the attacks.

Secondary Actors:

Business Analysts:

These users access post-simulation reports to analyze how the simulated attacks impacted server performance. They provide insights into the system's overall resilience and recommend improvements

4.2 General Requirements

Infrastructure Requirements

The system must be designed to simulate DoS/DDoS attacks in a secure, isolated environment to ensure no disruption to live business systems.

Tools:

Kali Linux: The operating system used for running the python script tool to launch DoS/DDoS attacks.

Apache Server: A real server hosted on Oracle VM with Ubuntu as the operating system, simulating a target for the attack.

Oracle VM: Used to create a virtualized environment, ensuring that the simulated attacks do not affect live production systems.

Scalability: The system should be capable of supporting simulations for multiple concurrent users, each testing their business infrastructure against different attack scenarios without compromising performance.

Since businesses have a global presence, the system must support multiple languages and be customizable to accommodate various corporate cybersecurity policies and compliance standards.

4.2.1 Infrastructure

Hardware Requirements:

Network Interface: Integrated NIC capable of stable Wi-Fi or Ethernet connectivity for initiating and maintaining attack sessions.

Internet Connection: A stable and high-speed broadband connection to ensure low latency for real-time attack demonstrations.

Software Requirements:

Operating System: a Linux-based OS (e.g., Kali Linux,Ubuntu) for running the attack scripts and tools.

Attack Tools:

Preconfigured tools supporting Layer 4 and Layer 7 attacks, such as custom Python scripts or open-source software (e.g., Hping3 for Layer 4 and HTTPFlood for Layer

Browser-based tools for simple HTTP requests if needed.

Monitoring Tools: Wireshark or similar software to monitor outgoing attack traffic and confirm payload delivery.

Network Requirements:

Target Website: A local or isolated instance of the Dstat website, set up on a secured test environment, to avoid legal and ethical violations.

Proxy/VPN: Optional, for IP masking during the attack demonstration.

4.2.2 Competitor

Existing Tools and Techniques:

LOIC (Low Orbit Ion Cannon):

Strength: Simple and effective for Layer 4 attacks like UDP and TCP.

Weakness: No Layer 7 attack support or traffic variability.

Hping3:

Strength: Customizable for TCP and UDP-based attacks.

Weakness: Limited to Layer 4 and lacks high-level protocols like HTTP/2.

Slowloris:

Strength: Effective for keeping HTTP connections open to exhaust server resources.

Weakness: Only targets specific Layer 7 vulnerabilities, lacks flexibility for varied methods.

GoldenEye:

Strength: Generates Layer 7 GET/POST flood requests.

Weakness: Minimal customization and scalability.

Gaps in Competitor Tools:

Existing tools often focus on isolated attack layers (either Layer 4 or Layer 7).

Lack of hybrid frameworks capable of executing and analyzing both Layer 4 and Layer 7 attacks in a single environment.

Limited adaptability to evolving attack methods like HTTP/2 or advanced spoofing techniques.

Proposed Advantage:

A comprehensive framework supporting both Layer 4 (TCP/UDP) and Layer 7 (cfb, sky, http2, etc.) attacks.

Real-time analysis and metric collection for deeper insights into attack impacts.

Customizable attack scenarios to test robustness across diverse environments.

4.3 Functional and Non-Functional Requirements

4.3.1 Functional Requirements

Simulation of Attacks:

Input: Configuration of attack types and parameters (e.g., volume, rate, duration).

Output: Layer 4 (TCP, UDP), Layer 7 (get/post Attack, HTTPx).

Target Webserver Testing:

Input: Web server details and testing goals (e.g, performance benchmarks).

Output: Evaluation of server performance under attack

Traffic Monitoring:

Input: Network traffic captured during attack simulations (e.g., .pcapng files).

Output: Analyzed logs of incoming and outgoing traffic using Wireshark .

4.3.2 Non- Functional Requirements

1. **Performance**: Ability to handle up to X simulated requests per second.
2. **Reliability**: Ensure the simulation is stable without crashing.
3. **Scalability**: Support for simulating various attack sizes.
4. **Security**: Prevent unintended damage or actual server compromise.
5. **Compliance**: Ensure ethical boundaries and legal compliance

4.4 Use Case Diagram

A use case diagram illustrates the interactions between different actors (Security Engineers, Administrators, Analysts, and Regulators) and the core functionalities of the system. This includes attack simulations, monitoring, reporting, and system management.

Use Case Descriptions

Configure Attack

1. Actors: Security Engineer
2. Description: The user configures parameters like target IP, thread count, and attack duration.
3. Preconditions: The user is authenticated.
4. Success Scenario: Attack configuration is successfully stored and executed.

Generate Report

1. Actors: Security Engineer, Business Analyst

2. Description: After the simulation, a report is generated summarizing the impact of the attack, including server performance, downtime, and vulnerabilities.
3. Preconditions: The simulation has concluded successfully.
4. Success Scenario: The report is generated and available for analysis.

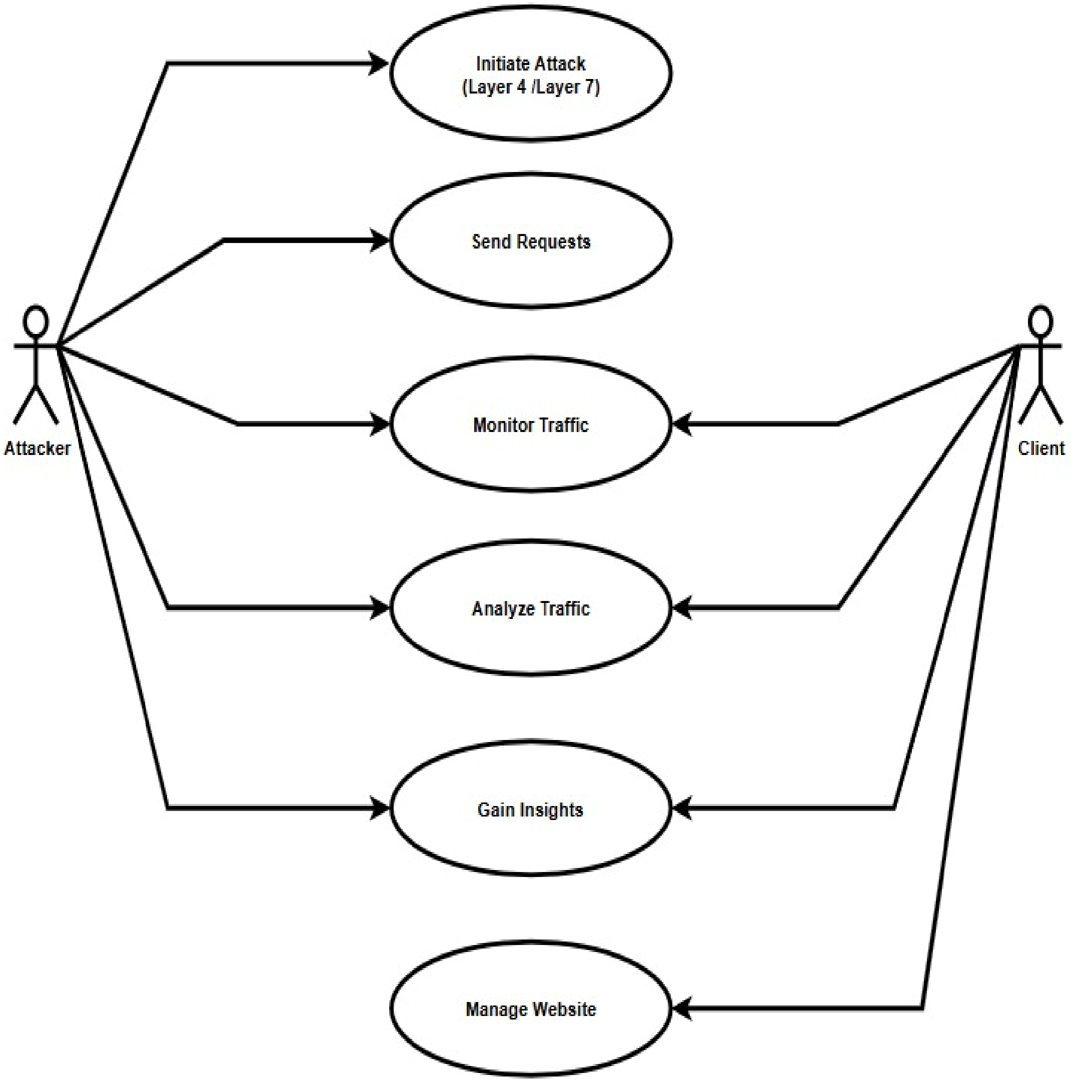


Fig 1. Use Case Diagram

Chapter 5: System Modeling

5.1 UML diagram

5.1.1 Component Diagram

The class diagram represents the structural and functional relationships between three key components: the attacker, the website, and the client. The attacker has an interface with methods LaunchAttack() and MonitorTraffic(), enabling it to initiate and observe attacks using protocols such as cfb, http, tcp, and udp. The website incorporates a RequestQueue for handling incoming traffic and a LoadBalancing mechanism to distribute requests efficiently. It also implements an interface with methods like ReceiveRequest(), ProcessRequest(), and ForwardTraffic() for traffic handling and communication with other components. The client contains a Dashboard and ControlPanel for traffic visualization and website management. It implements methods such as ViewTraffic() and ManageWebsite() for monitoring and administrative control. The diagram demonstrates a modular approach to managing website operations, traffic monitoring, and responding to potential attacks**.**

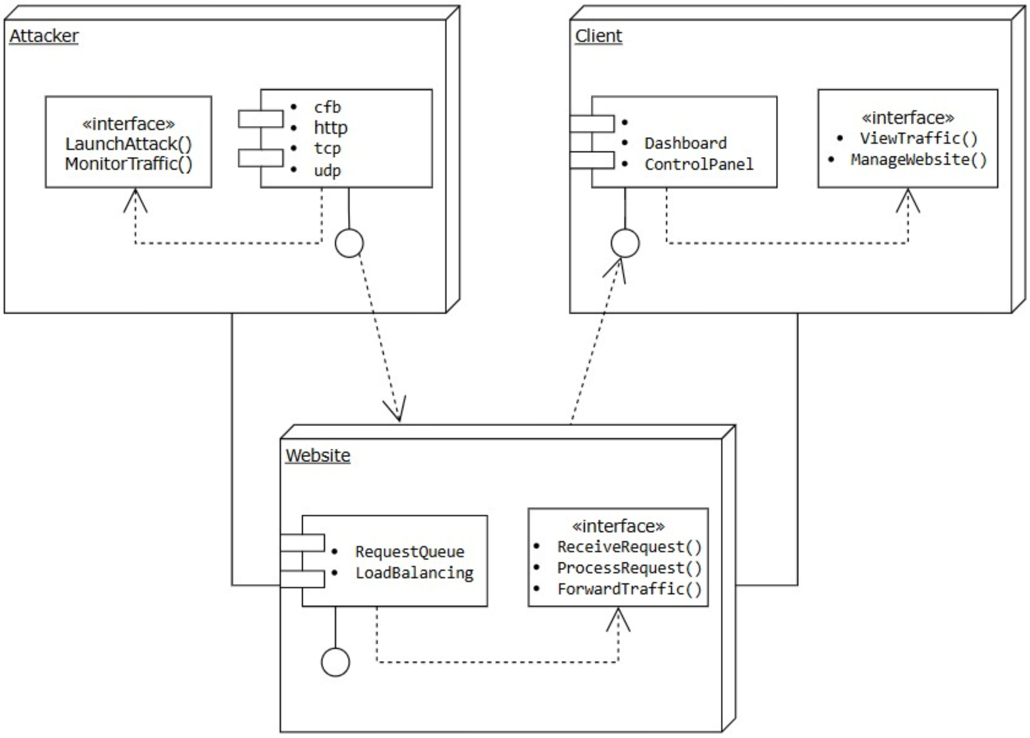


Fig.2 Component Diagram

5.1.2 Activity Diagram

The diagram illustrates a flow of events starting with the attacker initiating an attack. The attacker sends requests to the website, which then forwards the traffic data to a monitoring system. The monitoring system analyzes the traffic to detect malicious activities. Following this, the client monitors the website's status, views the analyzed traffic data, and takes necessary actions to manage the website. This process ensures that the website is monitored and managed effectively to mitigate any potential threats posed by the attacker.

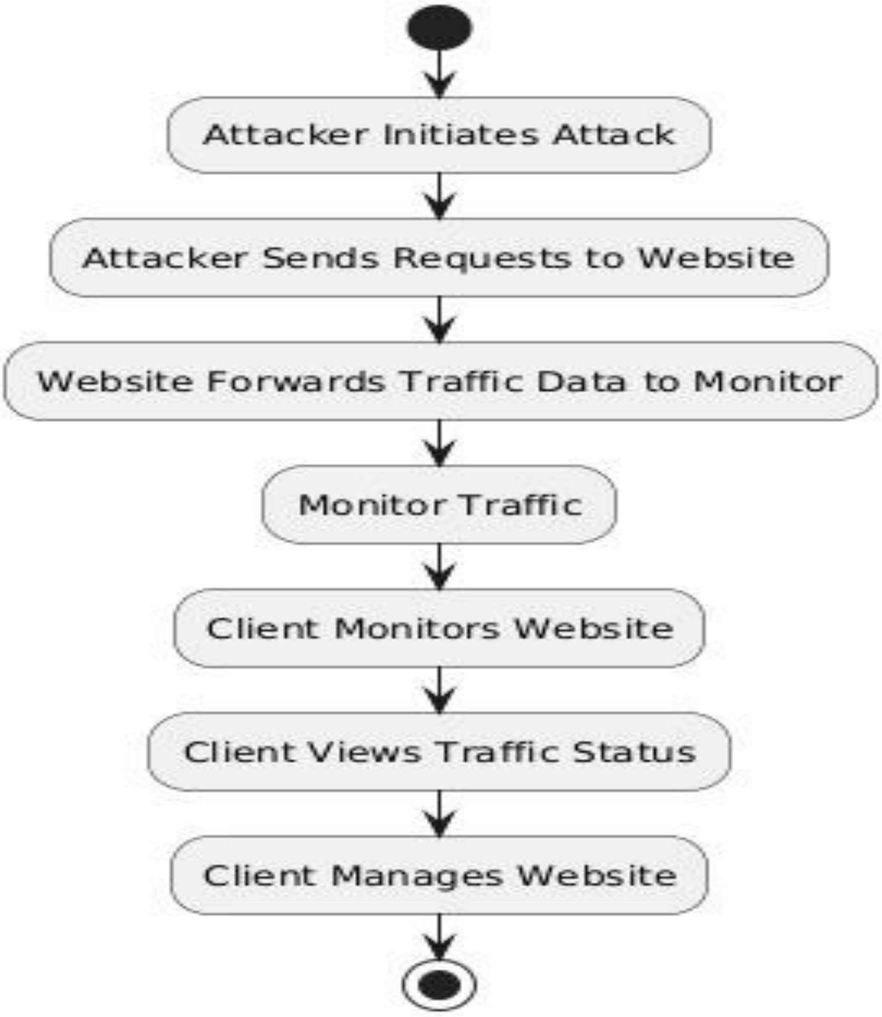

Attacker Initiates Attack
Attacker Sends Requests to Website
Website Forwards Traffic Data to Monitor
Monitor Traffic
Client Monitors Website
Client Views Traffic Status
Client Manages Website

Fig.3 Activity Diagram

5.1.3 Sequence Diagram

The sequence diagram illustrates the interaction between an attacker, a website, a traffic monitor, and a client in the context of monitoring and managing malicious activities. The attacker initiates an attack by sending multiple requests to the website, potentially as part of a Distributed Denial of Service (DDoS) attack. The website processes these requests and forwards the traffic data to the traffic monitor for analysis. The traffic monitor examines the incoming traffic to detect suspicious activities and relays the traffic status to the client, who can view and manage the website accordingly. This flow highlights the role of traffic monitoring tools in detecting and mitigating cyberattacks while keeping the client informed about the website's status.

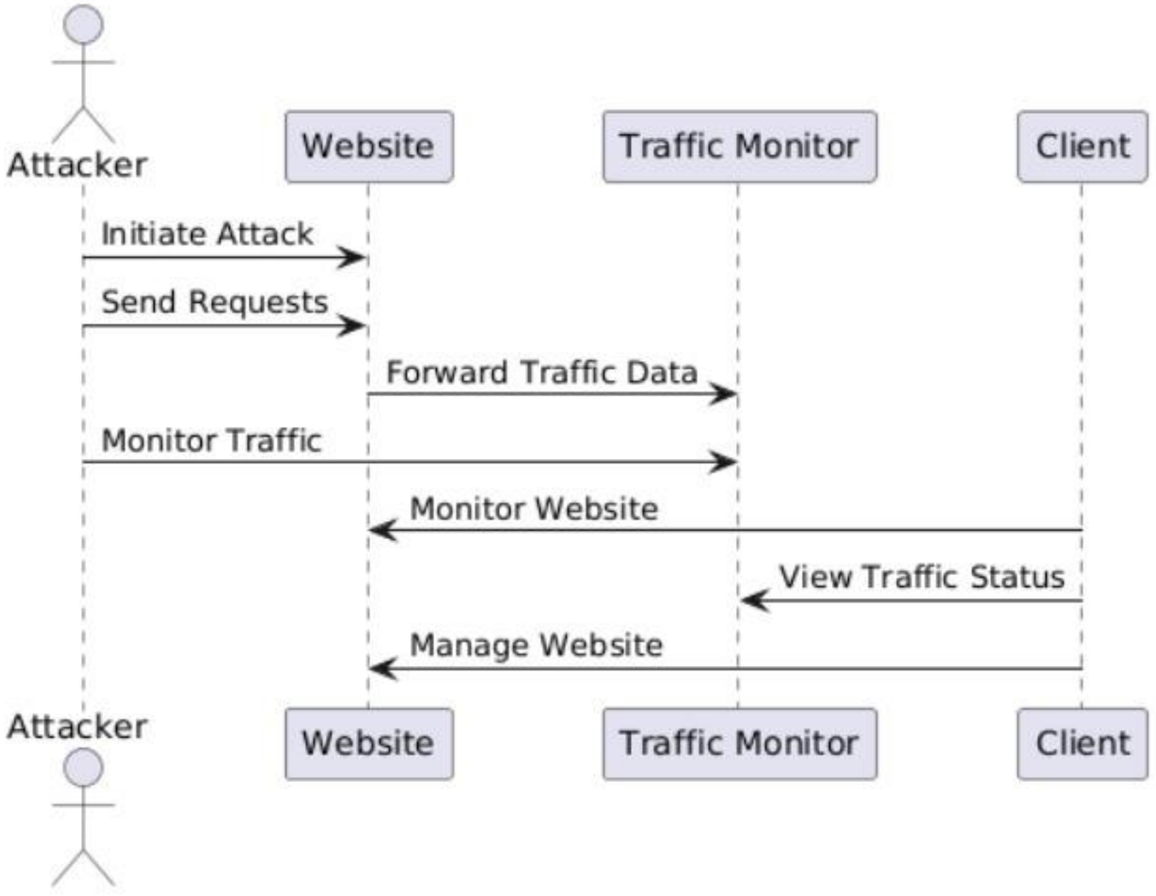


Fig.4 Sequence Diagram

Chapter 6: Implementation

6.1 Algorithm Development

Step 1: Configure the Target Server

- Deploy the server on a specific IP address and port.
- Enable logging to record incoming requests and server metrics.

Step 2 : Select Attack Vector

- Choose the attack type:
- Layer 4: UDP/TCP Floods.
- Layer 7: HTTP GET/POST Floods or HTTP/2.

Step 3 : Prepare the Attack

- Write attack scripts or configure tools:
- For Layer 4, craft raw packets.
- For Layer 7, generate HTTP requests with high frequency (e.g., using Python's requests library).

Set attack parameters:

- Number of requests/packets per second.
- Packet size and target IP/port.
- **Step 4: Execute the Attack**

Launch the attack from the attacker machine:

Start with a low intensity to observe behavior.

Gradually increase intensity (e.g., request rate or packet size).

- **Step 5: Monitor Server Response**

Use monitoring tools (Wireshark) to analyze:

Incoming traffic volume and patterns.

Anomalies in packet structure or behavior.

Track server performance:

CPU, memory usage, and response time logs.

- **Step 6: Analyze Results**

Measure the impact on the target:

Server downtime or slowdown.

Packet loss or connection failures.

Visualize metrics:

Create graphs/charts for traffic spikes, server load, or request failures.

6.2 User Interface Design

UI Features

1. Welcome and Initialization

- Upon launching the script, the terminal displays a welcome message with the project name and version.
- A brief description of the functionality is provided, along with a prompt for user input.

Help Command

Provides guidance on using the tool, including available commands and their syntax.

Example Command:

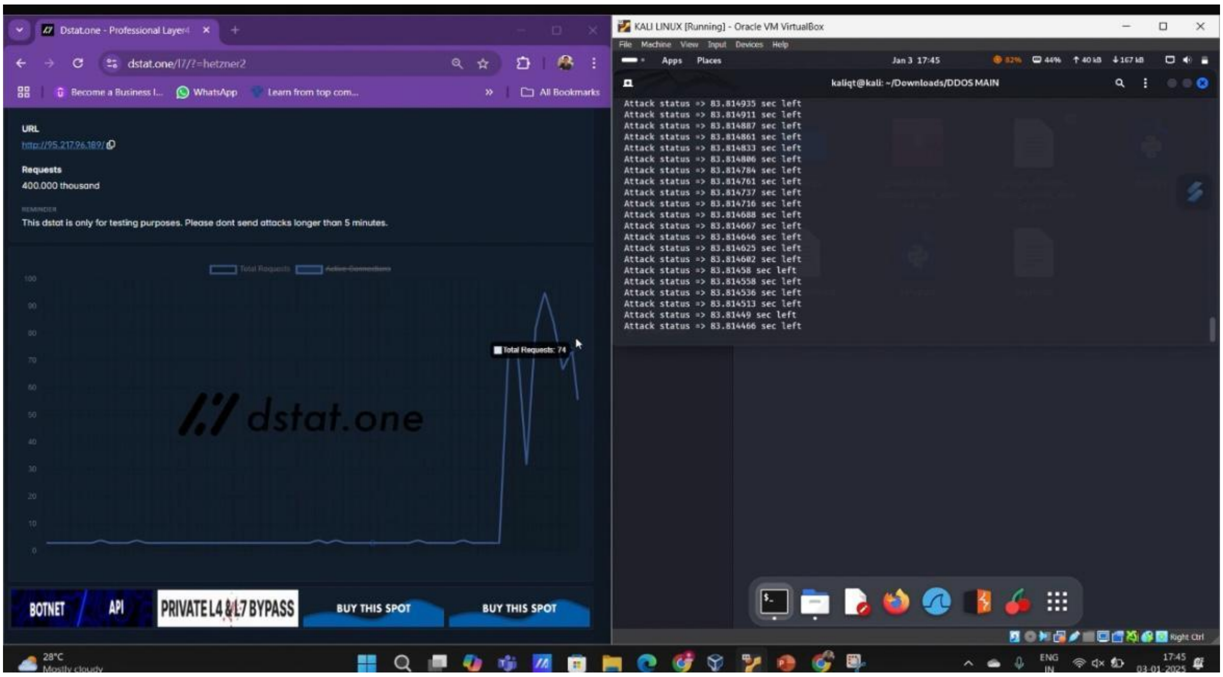
- bash
- Copy code
- Attack Execution
- Users specify the attack type, target, number of requests, and duration.

Example Command:

- Traffic Monitoring
- Monitors and analyzes traffic in real time, with Wireshark .

Example Command:

- Tools Integration
- Displays DNS and IP analysis tools available for the user
- Clear Terminal Display
- Clears terminal output to maintain a clean workspace



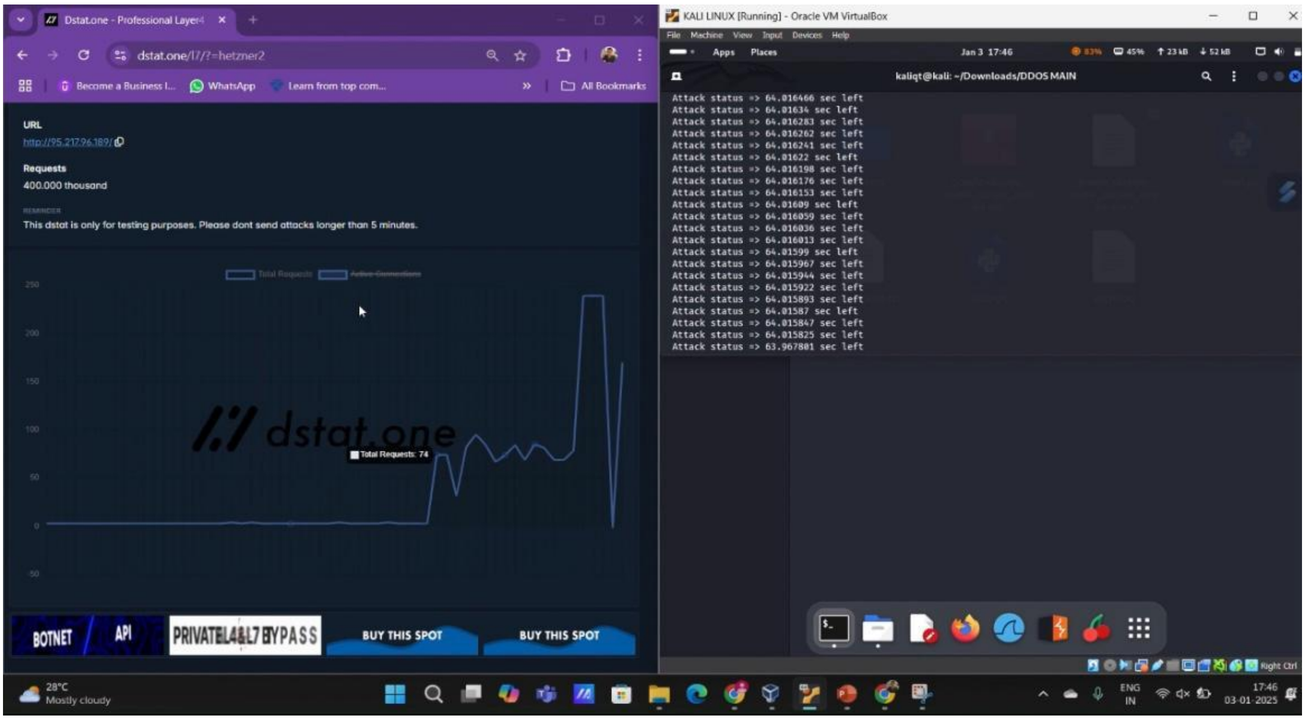
URL
http://95.217.96.189/

Requests
400.000 thousand

REMINDER
This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 83.814935 sec left
Attack status => 83.814911 sec left
Attack status => 83.814887 sec left
Attack status => 83.814861 sec left
Attack status => 83.814833 sec left
Attack status => 83.814806 sec left
Attack status => 83.814784 sec left
Attack status => 83.814761 sec left
Attack status => 83.814737 sec left
Attack status => 83.814716 sec left
Attack status => 83.814688 sec left
Attack status => 83.814667 sec left
Attack status => 83.814646 sec left
Attack status => 83.814625 sec left
Attack status => 83.814602 sec left
Attack status => 83.81458 sec left
Attack status => 83.814558 sec left
Attack status => 83.814536 sec left
Attack status => 83.814513 sec left
Attack status => 83.81449 sec left
Attack status => 83.814466 sec left
```

Fig.6 DDoS Attack Simulation Dashboard and Kali Linux Console



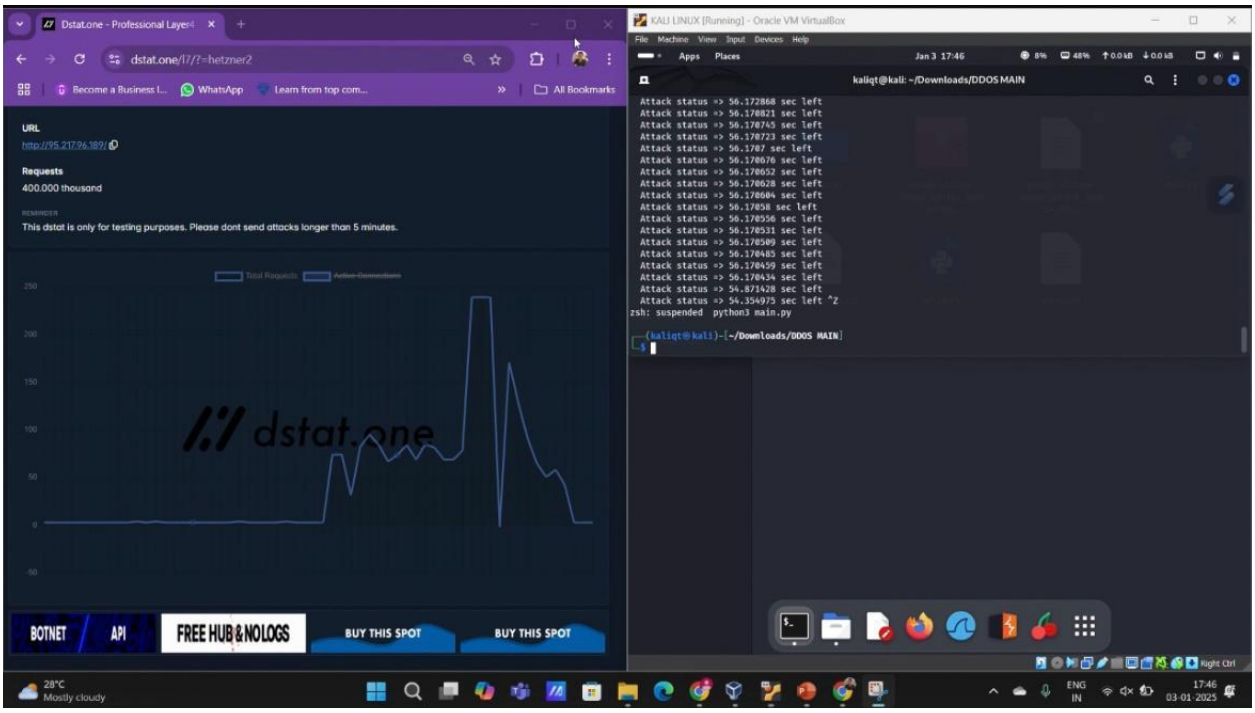
URL
http://95.217.96.189/

Requests
400.000 thousand

REMINDER
This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 64.016466 sec left
Attack status => 64.01634 sec left
Attack status => 64.016283 sec left
Attack status => 64.016262 sec left
Attack status => 64.016241 sec left
Attack status => 64.01622 sec left
Attack status => 64.016198 sec left
Attack status => 64.016176 sec left
Attack status => 64.016153 sec left
Attack status => 64.01609 sec left
Attack status => 64.016059 sec left
Attack status => 64.016036 sec left
Attack status => 64.016013 sec left
Attack status => 64.01599 sec left
Attack status => 64.015967 sec left
Attack status => 64.015944 sec left
Attack status => 64.015922 sec left
Attack status => 64.015893 sec left
Attack status => 64.01587 sec left
Attack status => 64.015847 sec left
Attack status => 64.015825 sec left
Attack status => 63.967801 sec left
```

Fig.7 DDoS Monitoring and Execution Interface - Updated View



URL

http://95.217.96.189/

Requests

400.000 thousand

REMINDER

This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

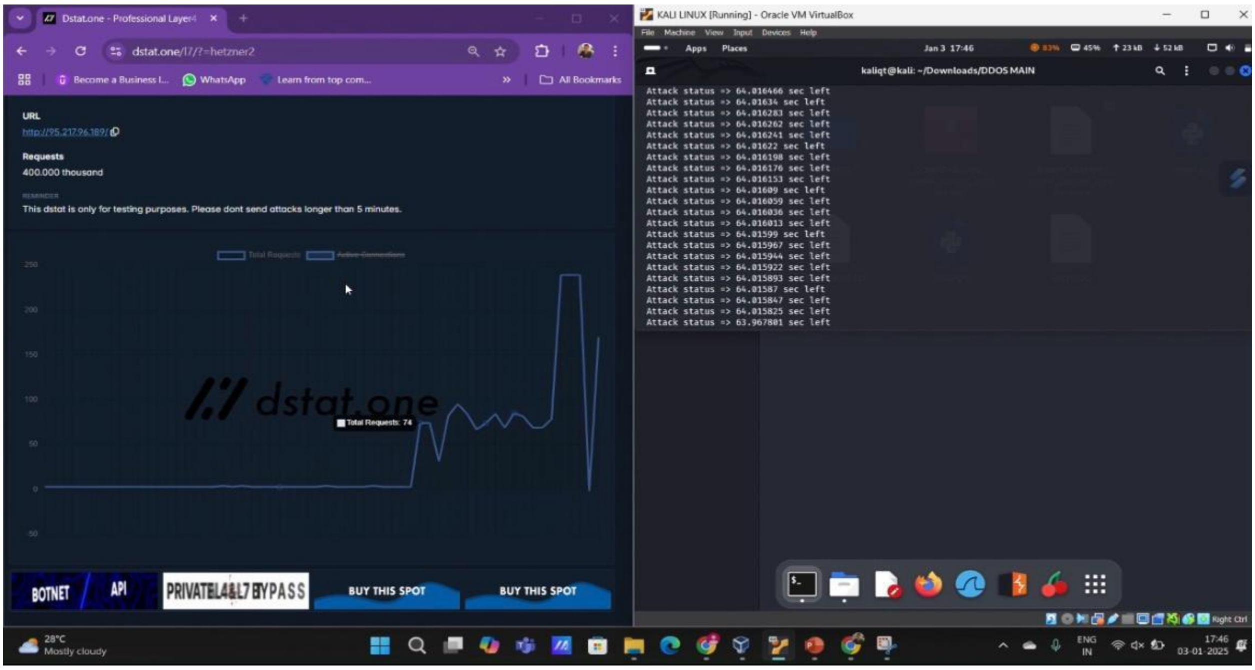
```
 Attack status => 56.172868 sec left
 Attack status => 56.170821 sec left
 Attack status => 56.170745 sec left
 Attack status => 56.170723 sec left
 Attack status => 56.1707 sec left
 Attack status => 56.170676 sec left
 Attack status => 56.170652 sec left
 Attack status => 56.170628 sec left
 Attack status => 56.170604 sec left
 Attack status => 56.17058 sec left
 Attack status => 56.170556 sec left
 Attack status => 56.170531 sec left
 Attack status => 56.170509 sec left
 Attack status => 56.170485 sec left
 Attack status => 56.170459 sec left
 Attack status => 56.170434 sec left
 Attack status => 54.871428 sec left
 Attack status => 54.354975 sec left ^Z
zsh: suspended  python3 main.py

┌──(kaliqt㉿kali)-[~/Downloads/DDOS MAIN]
└─$ 
```

Fig.8 DDoS Attack Testing with Suspended Execution

Chapter 7: Testing

7.1Results



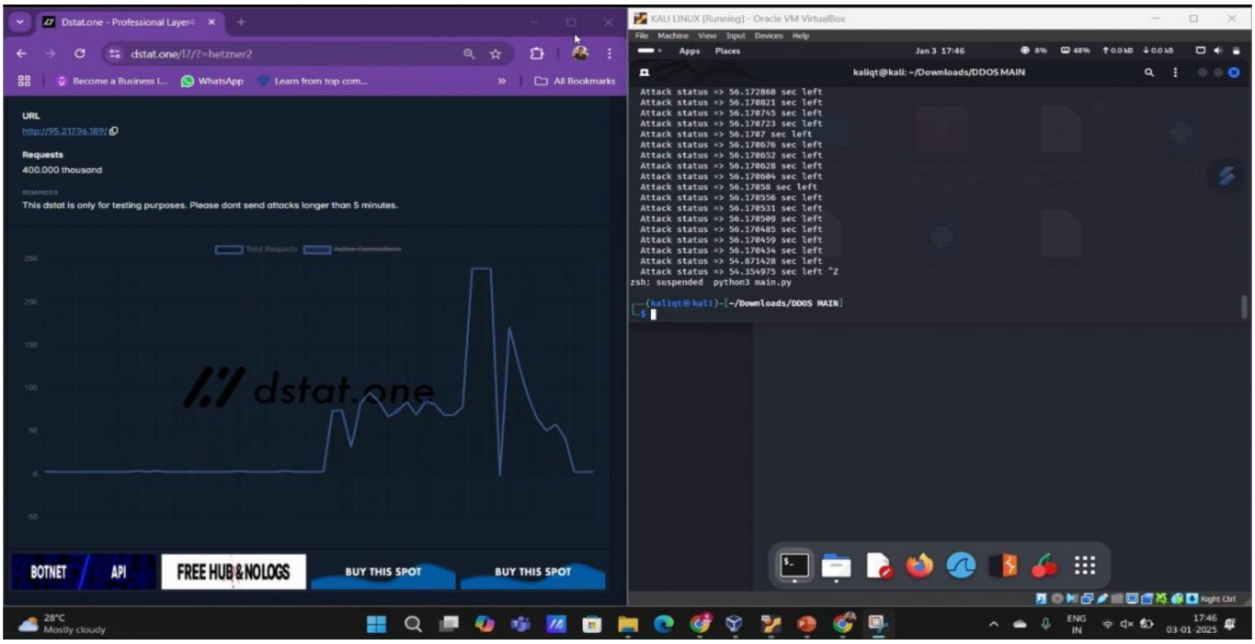
URL
http://95.217.96.189/

Requests
400.000 thousand

REMINDER
This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 64.016466 sec left
Attack status => 64.01634 sec left
Attack status => 64.016283 sec left
Attack status => 64.016262 sec left
Attack status => 64.016241 sec left
Attack status => 64.01622 sec left
Attack status => 64.016198 sec left
Attack status => 64.016176 sec left
Attack status => 64.016153 sec left
Attack status => 64.01609 sec left
Attack status => 64.016059 sec left
Attack status => 64.016036 sec left
Attack status => 64.016013 sec left
Attack status => 64.01599 sec left
Attack status => 64.015967 sec left
Attack status => 64.015944 sec left
Attack status => 64.015922 sec left
Attack status => 64.015893 sec left
Attack status => 64.01587 sec left
Attack status => 64.015847 sec left
Attack status => 64.015825 sec left
Attack status => 63.967801 sec left
```

Fig.9 DDoS Monitoring and Execution Interface - Updated View



URL
http://95.217.96.189/

Requests
400.000 thousand

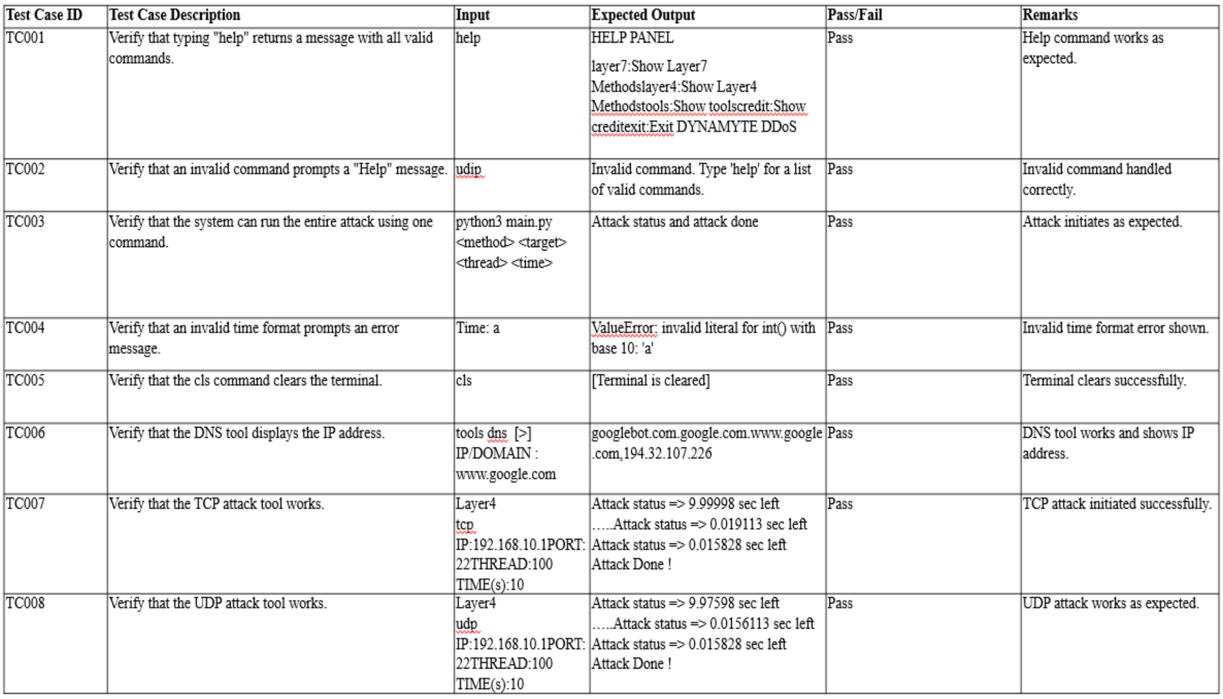
REMINDER
This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 56.172868 sec left
Attack status => 56.170821 sec left
Attack status => 56.170745 sec left
Attack status => 56.170723 sec left
Attack status => 56.1707 sec left
Attack status => 56.170676 sec left
Attack status => 56.170652 sec left
Attack status => 56.170628 sec left
Attack status => 56.170604 sec left
Attack status => 56.17058 sec left
Attack status => 56.170556 sec left
Attack status => 56.170531 sec left
Attack status => 56.170509 sec left
Attack status => 56.170485 sec left
Attack status => 56.170459 sec left
Attack status => 56.170434 sec left
Attack status => 54.871428 sec left
Attack status => 54.354975 sec left ^Z
zsh: suspended  python3 main.py

┌──(kaliqt㉿kali)-[~/Downloads/DDOS MAIN]
└─$ 
```

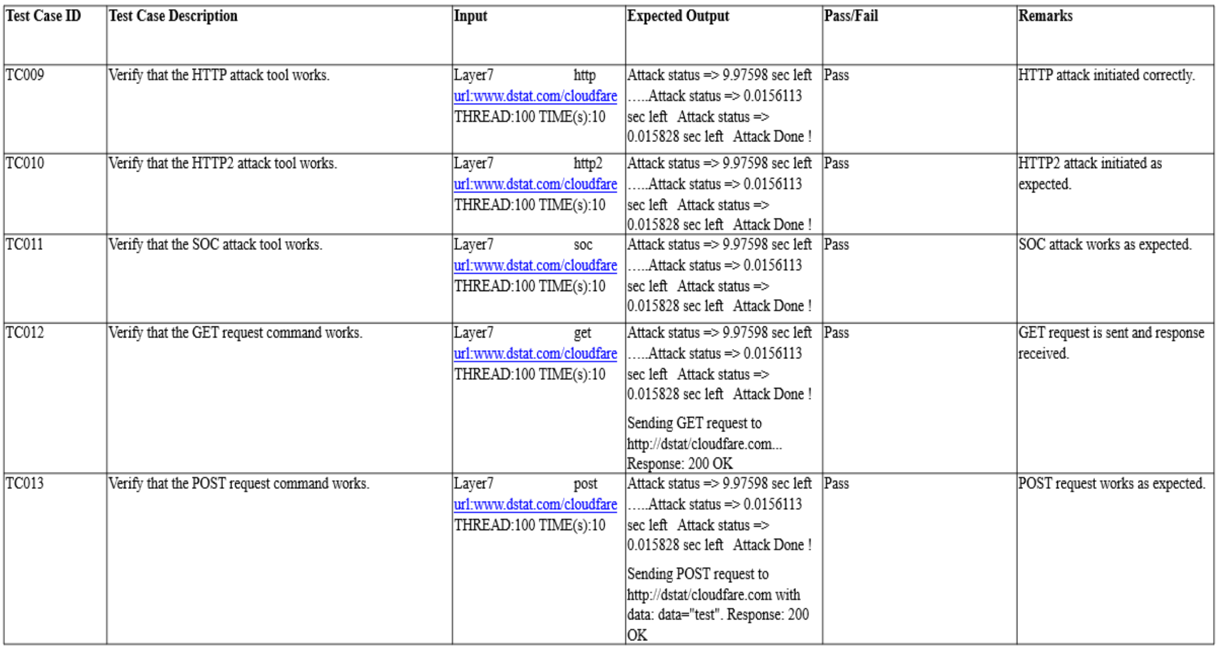
Fig.10 DDoS Attack Testing with Suspended Execution

7.2 Testing report



Test Case ID	Test Case Description	Input	Expected Output	Pass/Fail	Remarks
TC001	Verify that typing "help" returns a message with all valid commands.	help	HELP PANEL layer7:Show Layer7 Methodslayer4:Show Layer4 Methodstools:Show toolscredit:Show creditexit:Exit DYNAMYTE DDoS	Pass	Help command works as expected.
TC002	Verify that an invalid command prompts a "Help" message.	udip	Invalid command. Type 'help' for a list of valid commands.	Pass	Invalid command handled correctly.
TC003	Verify that the system can run the entire attack using one command.	python3 main.py <method> <target> <thread> <time>	Attack status and attack done	Pass	Attack initiates as expected.
TC004	Verify that an invalid time format prompts an error message.	Time: a	ValueError: invalid literal for int() with base 10: 'a'	Pass	Invalid time format error shown.
TC005	Verify that the cls command clears the terminal.	cls	[Terminal is cleared]	Pass	Terminal clears successfully.
TC006	Verify that the DNS tool displays the IP address.	tools dns [>] IP/DOMAIN : www.google.com	googlebot.com.google.com.www.google .com,194.32.107.226	Pass	DNS tool works and shows IP address.
TC007	Verify that the TCP attack tool works.	Layer4 tcp IP:192.168.10.1PORT:22THREAD:100 TIME(s):10	Attack status => 9.99998 sec left .....Attack status => 0.019113 sec left Attack status => 0.015828 sec left Attack Done !	Pass	TCP attack initiated successfully.
TC008	Verify that the UDP attack tool works.	Layer4 udp IP:192.168.10.1PORT:22THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done !	Pass	UDP attack works as expected.

Fig.11 Test cases



Test Case ID	Test Case Description	Input	Expected Output	Pass/Fail	Remarks
TC009	Verify that the HTTP attack tool works.	Layer7 http url:www.dstat.com/cloudfare THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done !	Pass	HTTP attack initiated correctly.
TC010	Verify that the HTTP2 attack tool works.	Layer7 http2 url:www.dstat.com/cloudfare THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done !	Pass	HTTP2 attack initiated as expected.
TC011	Verify that the SOC attack tool works.	Layer7 soc url:www.dstat.com/cloudfare THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done !	Pass	SOC attack works as expected.
TC012	Verify that the GET request command works.	Layer7 get url:www.dstat.com/cloudfare THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done ! Sending GET request to http://dstat/cloudfare.com... Response: 200 OK	Pass	GET request is sent and response received.
TC013	Verify that the POST request command works.	Layer7 post url:www.dstat.com/cloudfare THREAD:100 TIME(s):10	Attack status => 9.97598 sec left .....Attack status => 0.0156113 sec left Attack status => 0.015828 sec left Attack Done ! Sending POST request to http://dstat/cloudfare.com with data: data="test". Response: 200 OK	Pass	POST request works as expected.

Fig.12 Test Cases

7.3 Testing Tool

For evaluating the performance and resilience of the web server under DDoS attacks, two essential tools were utilized: **Dstat** and **Wireshark**.

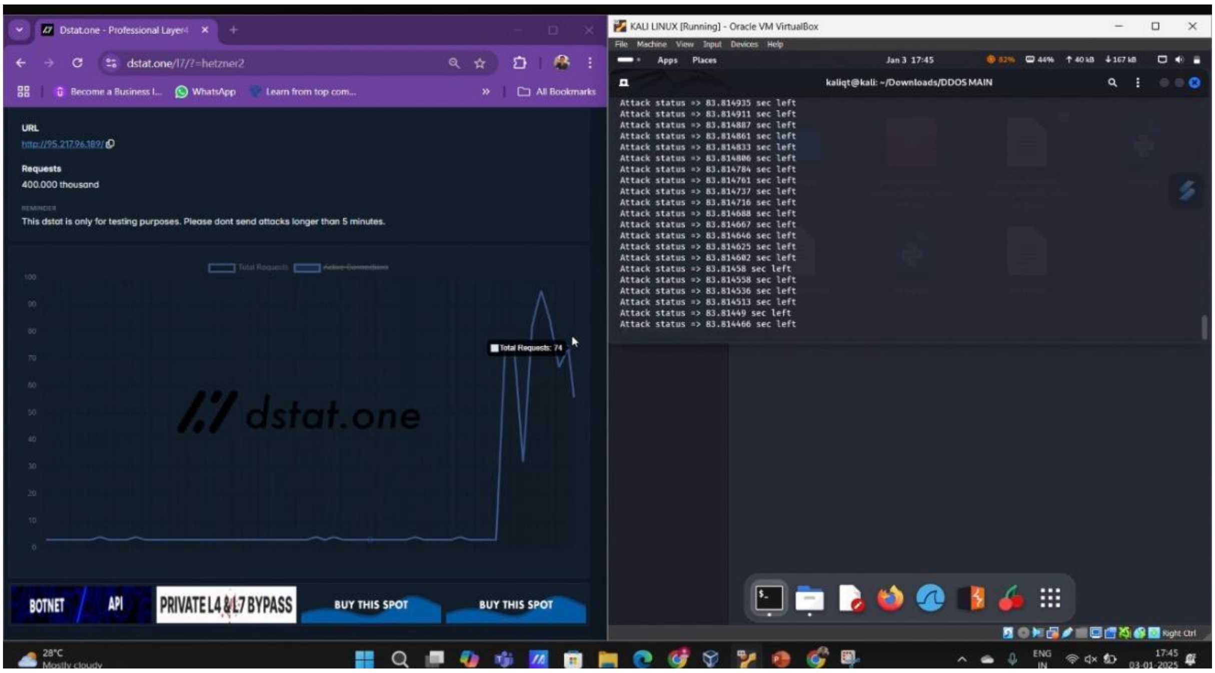
Dstat is a versatile resource monitoring tool that provides real-time insights into various system metrics such as CPU usage, memory consumption, network throughput, and disk activity. It offers a comprehensive view of how the web server handles the simulated DDoS attack, enabling a detailed analysis of system performance and identifying potential bottlenecks or inefficiencies.

Wireshark, on the other hand, is a powerful packet capturing tool used to analyze network traffic at a granular level. During the DDoS simulations, Wireshark was employed to capture and inspect the incoming and outgoing packets, allowing for the identification of traffic patterns, anomalies, and the effectiveness of implemented mitigation strategies. Together, these tools provided a robust framework for monitoring and analyzing the impact of DDoS attacks on the web server, ensuring a thorough evaluation of its performance and security posture.

Conclusion

This study examines the impact of various Distributed Denial of Service (DDoS) attacks on server performance, focusing on latency, downtime, CPU and memory utilization, and network traffic. Results indicate that Layer 4 attacks, such as UDP Flood and TCP Flood, cause significant spikes in packets per second (PPS), bandwidth consumption, and latency, often leading to service outages. Layer 7 attacks, including HTTP Flood and HTTP/2 Requests, though less intense in traffic volume, still degrade service performance over time. Proxy-based attacks, leveraging intermediary servers to obscure their origins, proved particularly effective in evading detection and causing prolonged disruptions, with methods like Proxy Socket and Proxy Request attacks being the most impactful. Mitigation requires a multi-faceted approach, including rate limiting, traffic filtering, server scaling, and continuous monitoring of server metrics for early detection. Future research should focus on enhancing detection techniques for sophisticated proxy-based attacks and improving resource allocation strategies to better manage large-scale traffic surges. This study offers valuable insights for improving DDoS defense strategies and building resilient server architectures.

Photos of Project Execution



URL

http://95.217.96.189/

Requests

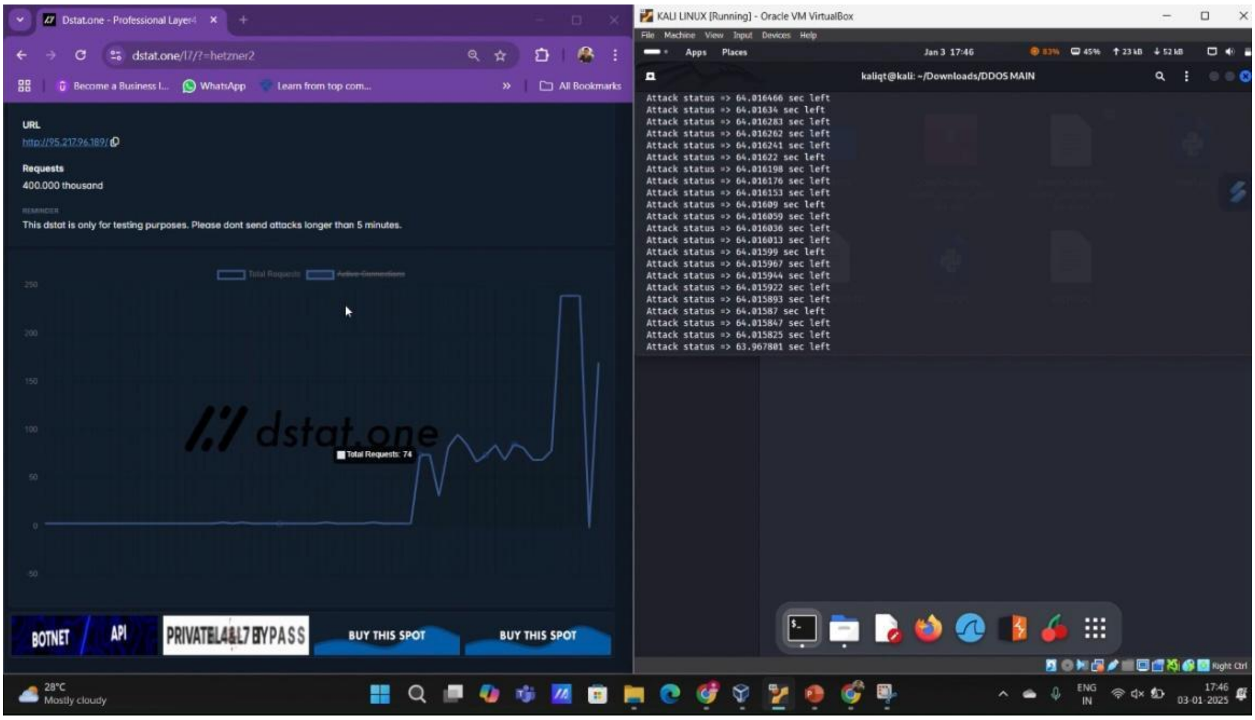
400.000 thousand

REMINDER

This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 83.814935 sec left
Attack status => 83.814911 sec left
Attack status => 83.814887 sec left
Attack status => 83.814861 sec left
Attack status => 83.814833 sec left
Attack status => 83.814806 sec left
Attack status => 83.814784 sec left
Attack status => 83.814761 sec left
Attack status => 83.814737 sec left
Attack status => 83.814716 sec left
Attack status => 83.814688 sec left
Attack status => 83.814667 sec left
Attack status => 83.814646 sec left
Attack status => 83.814625 sec left
Attack status => 83.814602 sec left
Attack status => 83.81458 sec left
Attack status => 83.814558 sec left
Attack status => 83.814536 sec left
Attack status => 83.814513 sec left
Attack status => 83.81449 sec left
Attack status => 83.814466 sec left
```

Fig.13 DDoS Attack Simulation Dashboard and Kali Linux Console



URL

http://95.217.96.189/

Requests

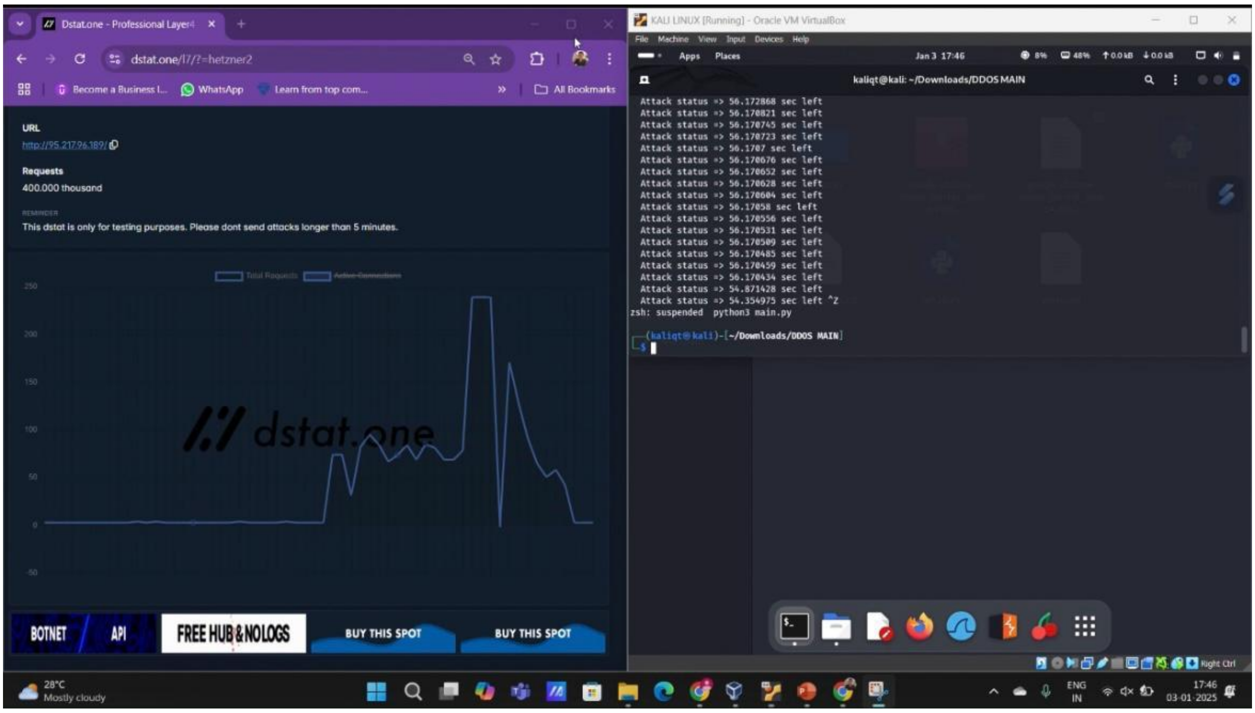
400.000 thousand

REMINDER

This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
Attack status => 64.016466 sec left
Attack status => 64.01634 sec left
Attack status => 64.016283 sec left
Attack status => 64.016262 sec left
Attack status => 64.016241 sec left
Attack status => 64.01622 sec left
Attack status => 64.016198 sec left
Attack status => 64.016176 sec left
Attack status => 64.016153 sec left
Attack status => 64.01609 sec left
Attack status => 64.016059 sec left
Attack status => 64.016036 sec left
Attack status => 64.016013 sec left
Attack status => 64.01599 sec left
Attack status => 64.015967 sec left
Attack status => 64.015944 sec left
Attack status => 64.015922 sec left
Attack status => 64.015893 sec left
Attack status => 64.01587 sec left
Attack status => 64.015847 sec left
Attack status => 64.015825 sec left
Attack status => 63.967801 sec left
```

Fig.14 DDoS Monitoring and Execution Interface - Updated View



URL

http://95.217.96.189/

Requests

400.000 thousand

REMINDER

This dstat is only for testing purposes. Please dont send attacks longer than 5 minutes.

```
 Attack status => 56.172868 sec left
 Attack status => 56.170821 sec left
 Attack status => 56.170745 sec left
 Attack status => 56.170723 sec left
 Attack status => 56.1707 sec left
 Attack status => 56.170676 sec left
 Attack status => 56.170652 sec left
 Attack status => 56.170628 sec left
 Attack status => 56.170604 sec left
 Attack status => 56.17058 sec left
 Attack status => 56.170556 sec left
 Attack status => 56.170531 sec left
 Attack status => 56.170509 sec left
 Attack status => 56.170485 sec left
 Attack status => 56.170459 sec left
 Attack status => 56.170434 sec left
 Attack status => 54.871428 sec left
 Attack status => 54.354975 sec left ^Z
zsh: suspended  python3 main.py

┌──(kaliqt㉿kali)-[~/Downloads/DDOS MAIN]
└─$ 
```

Fig.15DDoS Attack Testing with Suspended Execution