MINI PROJECT REPORT

DDOS attack Detection System in Cloud Environments
using Machine Learning Algorithms

by

Name : Apiksha
Entry. **No**.: 22BCS020
and
Name : Soumitra Rai
Entry. **No**.: 22BCS090

Under the Guidance of
Dr. Naveen Gondhi

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE & ENGINEERING

SHRI MATA VAISHNO DEVI UNIVERSITY, KATRA
(School of Computer Science & Engineering)

JAMMU & KASHMIR – 182 320
Session 2024-25

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to **Shri Mata Vaishno Devi University**, School of Computer Science, for providing us the platform and resources to undertake this project titled ***"DDoS Attack Detection in Cloud Environments Using Machine Learning Algorithms."***

We are deeply thankful to our project guide, **Dr. Naveen Gondhi**, for his invaluable support, guidance, and feedback throughout the project. Their expertise and insightful suggestions helped us overcome challenges and refine our work.

This project would not have been possible without the collaborative efforts of our team members, whose dedication, innovation, and teamwork were critical to its success. Each contribution was instrumental in achieving our objectives and ensuring the quality of the work presented here.

We also acknowledge the contributions of our institution for providing access to essential resources and a supportive learning environment, enabling us to explore innovative ideas and solutions.

This project has been a tremendous learning experience, and we are grateful to have completed it under the guidance and support of such a nurturing academic environment.

Apiksha
22BCS020, B. Tech, 5 th Sem
School of Computer Science and Engineering
Shri Mata Vaishno Devi University, Katra

Soumitra Rai
22BCS020, B. Tech, 5 th Sem
School of Computer Science and Engineering
Shri Mata Vaishno Devi University, Katra

DECLARATION

We, the undersigned solemnly declare that the project report "***DDoS Attack Detection in Cloud Environments Using Machine Learning Algorithms***" is based on our own work carried out during the course of our study under the supervision of **Dr. Naveen Gondhi**.

We assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

I. The work contained in the report is original and has been done by me under the supervision of my supervisor.
II. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad.
III. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

Apiksha
22BCS020, B. Tech, 5 th Sem
School of Computer Science & Engineering
Shri Mata Vaishno Devi University, Katra

Soumitra Rai
22BCS020, B. Tech, 5 th Sem
School of Computer Science & Engineering
Shri Mata Vaishno Devi University, Katra

I endorse the above declaration of the Student.

Dr. Naveen Gondhi
Date: 6/12/2024

ABSTRACT

We created a machine learning-based method in this research to identify Distributed Denial of Service (DDoS) threats. An Apache server was set up to mimic a real-world setting, and DDoS attacks were launched against its IP address using hping3. Wireshark was used to record traffic data both during regular server operations and during the attack. In order to ensure a balanced representation of various circumstances, this method produced a dataset that included both malicious and regular network traffic patterns.

To extract features pertinent to DDoS detection, the dataset underwent preprocessing and analysis. We developed a number of machine learning models to categorize network traffic into normal and attack groups using this dataset. The algorithms' accuracy in anticipating and differentiating DDoS attack patterns from normal traffic was assessed.After training, the models were used to monitor server traffic, providing predictions to prevent prospective DDoS attacks.

The effectiveness of machine learning in thwarting the escalating threat of DDoS attacks is demonstrated by this study. Data-driven categorization techniques and traffic monitoring tools are combined in the system to offer a scalable and adaptable attack detection method. The experiment highlights the advantages of combining cutting-edge machine learning algorithms with traditional network analysis methods to strengthen cybersecurity defenses. The results highlight how continuous data collection and model modifications are necessary to guarantee robustness in dynamic network environments.

Keywords: DDoS, Cloudsim, Cloud Security, Network Traffic Classification

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND NOMENCLATURE

DDoS: Distributed Denial of Service

UDP: User Datagram Protocol

TCP: Transmission Control Protocol

SYN: Synchronize

ICMP: Internet Control Message Protocol

HTTP: HyperText Transfer Protocol

SVM: Support Vector Machine

XGBoost: Extreme Gradient Boosting

CHAPTER 1 : INTRODUCTION

1.1 PURPOSE

Distributed Denial of Service (DDoS) attacks pose a persistent threat to servers and cloud infrastructures in the continually evolving landscape of cybersecurity. By flooding systems with traffic, these attacks interfere with systems and prevent authorized users from accessing them. The need for strong, scalable, and adaptable solutions to mitigate such risks is increased by the growing dependence on cloud computing platforms for vital applications. A viable method for real-time DDoS detection and prevention is machine learning, which gives computer systems the ability to proactively recognize harmful traffic patterns.

1.2 SYSTEM OVERVIEW

In this project, a DDoS detection system on an Apache server was developed and assessed that relies on machine learning. First, an Apache server was set up in a controlled setting and hping3, a malicious traffic generation program, was used to simulate DDoS attacks. Wireshark was used to record and examine network traffic, enabling us to gather both attack and normal data. For the purpose of training machine learning models, this traffic data was preprocessed and organized into an extensive dataset.

Key characteristics of DDoS attacks, including unusual packet frequency, erratic traffic patterns, and variations in source IP behavior, were identified by training the models. After the training phase, we assessed the models' accuracy in identifying attacks. The outcomes showed how well machine learning works to detect malicious traffic and distinguish it from legitimate ones. Although an Apache server was used for testing, our ultimate objective is to expand the implementation to cloud-based platforms. Because of its distributed architecture and dynamic resource allocation, cloud platforms provide specific challenges for DDoS detection, such as the

need for scalability and increasing traffic complexity. We aim to overcome these obstacles and create a scalable detection system that smoothly interfaces with cloud infrastructure by modifying our machine learning models for the cloud environment.

1.3 GOALS AND VISION

The main goals of this project are:
Simulating DDoS Attacks: Using Hping3 to generate realistic attack scenarios.
Traffic Data Collection: Using Wireshark to record and examine both normal and malicious traffic data.
Model Training: Creating machine learning models to accurately identify DDoS attacks.
Future Cloud System Implementation: Address the scalability and complexity of cloud traffic by expanding the solution to cloud environments.

The methods, outcomes, and future directions for incorporating machine learning-based DDoS detection systems into cloud infrastructure are described in this report. We aim to develop a flexible and scalable protection system that can adjust to the ever-changing landscape of contemporary cyberthreats by moving from a server-based testbed to a cloud platform.

Our project emphasizes the importance of machine learning in developing resilient cloud systems, particularly its ability to protect cloud infrastructures from DDoS attacks and assure continuous service availability.

CHAPTER 2 : REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATIONS

2.1 Requirement Analysis

The development of a machine-learning-based DDoS detection system necessitates a disciplined approach to defining and meeting the project's requirements. The following is an analysis of the functional and non-functional requirements of the system:

2.1.1 Functional Requirements:

Data Collection :

Set up Apache servers on several computers to serve as targets for DDoS attacks. Use Wireshark to capture network traffic, both normal and malicious.

Simulate attack :

Hping3 can simulate a variety of DDoS attack types (UDP Flood, HTTP Flood, TCP SYN Flood, and ICMP Flood).

Feature Extraction:

Analyze the collected traffic data to find relevant features (for example, packet size, source/destination IP behavior, protocol type, and traffic frequency).
Use Python in Google Colab to preprocess data and extract useful characteristics for machine learning models.

Model training and testing:

The machine learning models were trained on the retrieved features using Python-based modules available in Colab and Visual Studio Code (VS Code).
The models are evaluated on how well they recognize DDoS attack patterns.

Prediction and detection:
The trained models were used to predict ongoing or potential DDoS attacks in real time.

Scalable and adaptable:
The system is designed so that it can be used in cloud-based environments in the future, while also accommodating the dynamic nature of cloud traffic.

2.1.2 Non-functional Requirements:

Accuracy
High precision and recall is to be maintained in detecting DDoS attacks while avoiding false positives and negatives.

Performance:
The system is optimized to provide effective traffic monitoring and real-time detection with minimal delay.

Scalability:
The system should be able to handle increased traffic loads and adapt to distributed cloud architectures.

2.2 System Specifications

2.2.1 Hardware Requirements

1. **Systems for Apache Servers:**
   The Apache servers will be hosted on three Linux and/or Windows-based platforms.

Minimum requirements: 20 GB of storage, 4 GB of RAM, and a dual-core processor.

2. **Network Configuration:**
   A virtual private network (VPN) or local area network (LAN) configuration for regulated traffic flow.

2.2.2 Software Requirements

1. **Apache Server**
   set up on three systems to serve as DDoS attack targets.

2. **Wireshark:**
   used to record and examine network traffic information both during regular and malicious attacks.

3. **Hping3:**
   A tool for creating network traffic that mimics different DDoS assaults (UDP Flood, HTTP Flood, TCP SYN Flood, and ICMP Flood).

4. **Libraries and Python:**
   Python environment for training models, extracting features, and preparing data. Pandas, numpy, scikit-learn, matplotlib, and tensorflow/keras are important libraries.

5. **Google Colab:**
   For easy collaboration and GPU support when training machine learning models.

6. **Visual Studio Code:**
   For locally creating, testing, and debugging Python scripts.

2.2.3 Specifications for the Dataset

Source: Traffic data is captured using Wireshark.

Data Composition:

1. **Normal traffic data:** Generated by the Apache servers during normal operation.
2. **Attack traffic data:** Generated by using hping3 to simulate multiple DDoS attacks.

Features include traffic frequency, packet intervals, source/destination IP, protocol type, and packet size.

2.2.4 Models for Machine Learning

C.4.5, Random Forest, Support Vector Machines (SVM), Naïve Bayes, and XGBoost are the training algorithms.

Evaluation metrics include F1 Score, Accuracy, Precision, and Recall.

CHAPTER 3 : SYSTEM DESIGN

The DDoS detection system is designed to create an organized pipeline for successfully capturing, analyzing, and detecting harmful traffic patterns. The system is made up of multiple interconnected modules, each of which performs a specific purpose, ranging from data collecting to machine learning-based attack detection. The sections that follow detail the system's primary components and their roles.

3.1 Apache Server Setup

Three Apache servers were installed on separate PCs to imitate a real-world scenario in which multiple servers run concurrently. These servers were used as DDoS targets, allowing us to investigate the impact of various attack methods, including UDP Flood, HTTP Flood, TCP SYN Flood, and ICMP Flood, under controlled settings.

3.2 Traffic Data Collection

Wireshark was used to monitor and capture network data from Apache servers during both normal and attack conditions. This phase was critical for generating a complete dataset that included both legitimate and hostile traffic. The recorded data comprised packet size, protocol type, source and destination IP addresses, and traffic frequency.

3.3 Attack Simulation

To simulate realistic DDoS traffic, the hping3 tool was utilized to launch a variety of attacks. These attacks were intended to stress the servers and guarantee that the dataset included a variety of attack scenarios. Four attack types were simulated:

1. **UDPFlood Attack:** The server was flooded with UDP packets.
2. **HTTP Flood Attack:** Overloading the server with HTTP requests.
3. **TCP SYN Flood Attack:** Send excessive SYN packets to drain server resources.
4. **ICMP Flood Attack:** To overwhelm the server, send multiple ICMP Echo Request packets (pings).

3.4 Data Preprocessing and Feature Extraction

Python was used to handle and analyze the traffic data gathered using Wireshark. Key features collected included packet size, traffic patterns, packet intervals, and protocol usage. This phase guaranteed that the dataset was properly structured and informative for training machine learning models. Google Colab was used for feature extraction since it supports GPUs and allows for easy collaboration.

3.5 Mode Training and Testing

Machine learning models were created and tested on the extracted data. The models were implemented using Python and packages like scikit-learn, pandas, and numpy. Training was provided in both Google Colab and Visual Studio Code. The models were tuned to detect unusual traffic patterns suggestive of DDoS attacks. Random Forest, SVM, Naive Bayes, and XGBoost algorithms were evaluated using accuracy, precision, recall, and F1 scores.

3.6 Future Implementation for cloud setups

Although currently tested in a local server environment, the system is being further enhanced to support cloud deployment. The transition to cloud systems presents new issues, such as managing large-scale traffic, assuring real-time detection, and managing remote network monitoring.

3.7 Summary

This chapter describes the current system design and demonstrates its capabilities in a local server context. The system successfully simulates and detects DDoS attacks because of its

modular architecture. Moving forward, the focus will be on modifying the system for cloud deployment, so that it can handle the complexities of distributed and dynamic cloud networks. The proposed system assures scalability, real-time detection, and robustness, laying the groundwork for a full cloud-based DDoS mitigation solution.

CHAPTER 4 : IMPLEMENTATION AND TESTING

The implementation and testing phases transform the design into a functional solution and assess its performance under diverse scenarios. This chapter shows how to build up the DDoS detection system, execute attacks, collect data, train machine learning models, and verify their effectiveness. The testing phase also evaluates the system's capacity to identify various DDoS attacks, assuring robustness and dependability.

4.1 Implementation

4.1.1 Setting up the Environment

- **Apache Server Configuration :** Three Apache servers were set up on separate systems to serve as targets for DDoS attacks. Standard server setups were used to emulate actual web hosting settings.
- **Tools and Platforms :**
  Wireshark: Installed on each machine to capture live network traffic during both normal and attack stages.
  hping3: Used on a separate attacker machine to generate various sorts of DDoS attacks.
  Google Colab and VS Code: used to preprocess data, extract features, and train machine learning models. Python was the main programming language.

4.1.2 Data Collection

The network traffic data was acquired under two conditions:

Normal traffic is generated by performing normal browsing, file uploads, and downloads to reflect legitimate network activities.

Attack Traffic: Simulated DDoS traffic with hping3.

Four attack types were used:

- UDP Flood Attack
- HTTP Flood Attack
- TCP Flood Attack
- ICMP Flood Attack

Wireshark was used to capture packet information, such as timestamps, protocols, packet sizes, and source/destination IP addresses.

4.1.3 Feature Extraction and Database Creation

The raw data obtained from wireshark was exported as CSV files.
Packet size, traffic frequency, and protocol usage were all retrieved using Python packages such as pandas and numpy.
A labeled dataset was generated to categorize traffic as "Normal" or "Attack" based on the packet source.

4.1.4 Model Training

The dataset was split into training (70%) and testing (30%) sets. Machine learning models, including Random Forest, SVM, Naive Bayes, and XGBoost, were implemented in Python using ***scikit-learn*.**

4.2 Testing

The following conventional metrics were used to assess the system's effectiveness :
Accuracy : percentage of correct predictions (for both regular and attack traffic).
Precision: The ability to correctly identify attack traffic while avoiding misclassifying routine traffic.
Recall : the ability to detect all attack cases in the dataset.
F1 Score: The harmonic mean of precision and recall.

CHAPTER 5 : RESULTS AND DISCUSSION

In this project, the DDoS attack detection system was trained and tested with five different machine learning algorithms: **Naive Bayes, C4.5, SVM, Random Forest, and XGBoost**. The outcomes for each model were assessed using four important metrics:

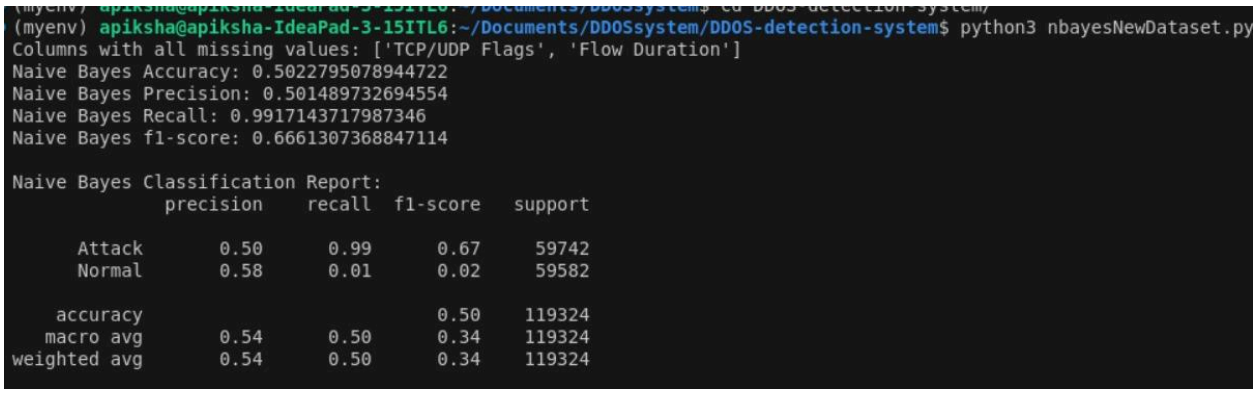
- **Accuracy**
- **Precision**
- **Recall**
- **F1 Score**

The comprehensive categorization reports and metrics for each model are shown below, along with a comparative analysis.

5.1 Model-specific results:

5.1.1 Naive Bayes

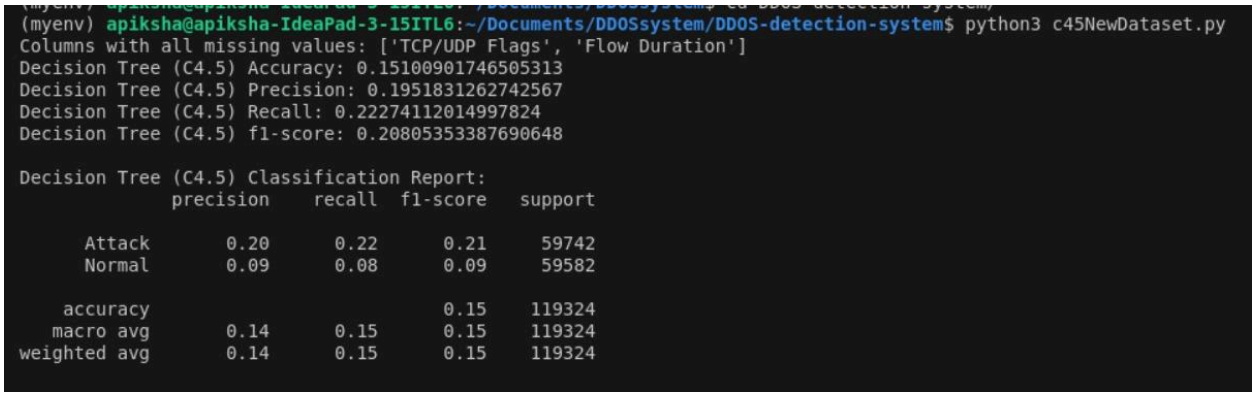
Performance: It had the best Recall (99%) for detecting attacks, as seen in the classification report (Figure 1). However, it had modest accuracy (50.2%) and precision (50.1%), yielding an F1 Score of 66.6%.



The model's high recall shows that it is effective in identifying the majority of DDoS attacks. However, its comparatively low precision indicates a high proportion of false positives.

5.1.2 **C4.5 (Decision Tree)**

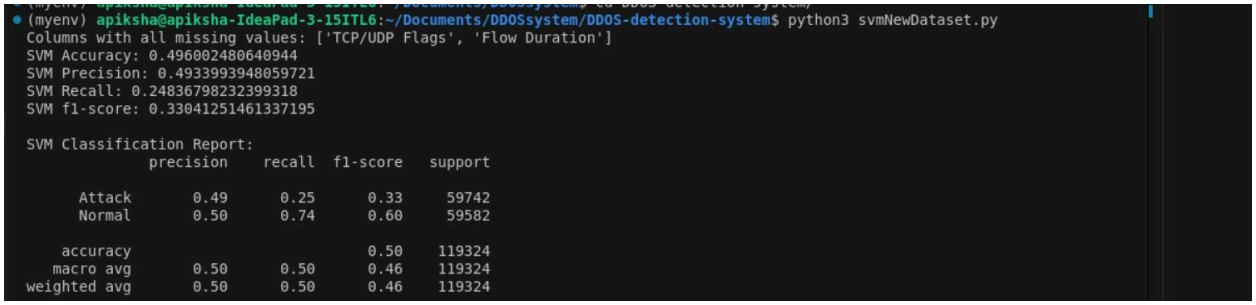
Performance: The C4.5 model had a poor overall accuracy of 15.1% and an F1 score of 20.8% (Figure 2). Precision (19.5%) and Recall (22.2%) were also low.



Discussion: This suggests that the C4.5 algorithm struggled to distinguish between attack and normal traffic in this dataset, most likely due to its complexity or feature qualities.

5.1.3 Support Vector Machine (SVM)

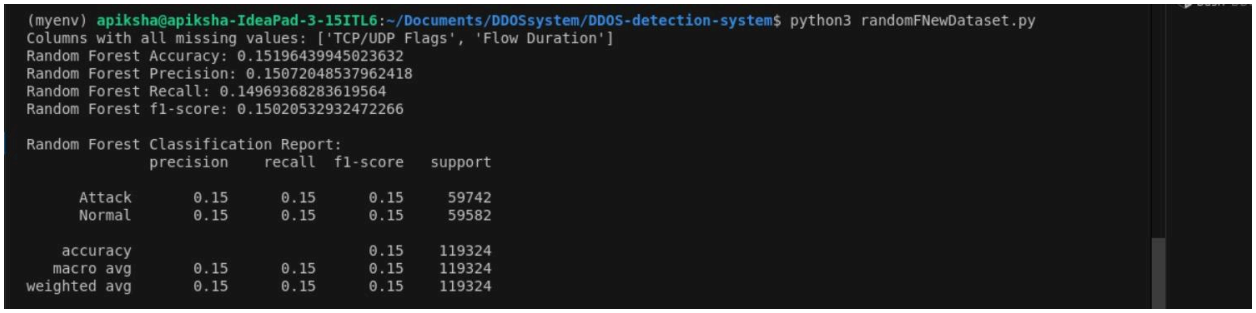
Performance: SVM had a modest accuracy of 49.6%, precision of 49.3%, and F1 Score of 33.0% (see Figure 3). Recall for attacks was relatively poor, at 24.8%.



Discussion: While SVM performed well overall, it suffered with recall, showing difficulty in spotting attacks. This may be due to the dataset's high dimensionality and the inherent limits of SVM for large datasets.

5.1.4 Random Forest

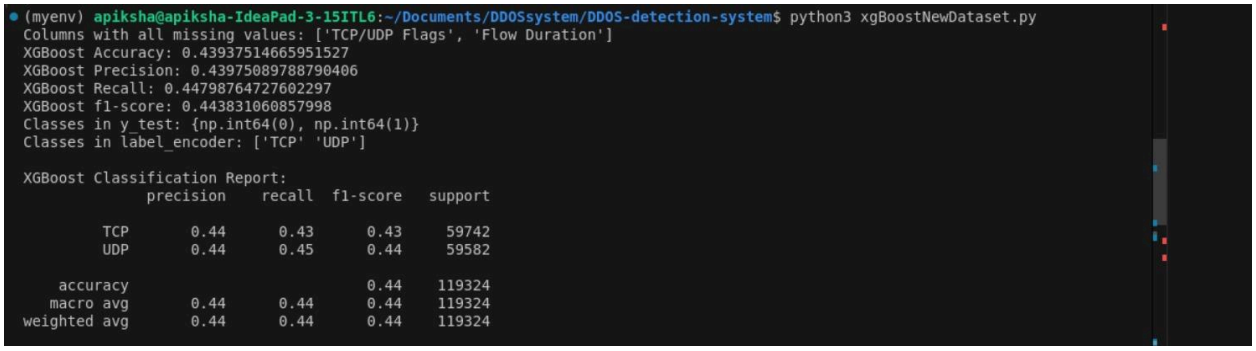
Performance: It performed poorly overall, with an accuracy of 15.2%, precision of 15.0%, and F1 Score of 15.0% .



Discussion: The poor results indicate that Random Forest was unsuitable for this dataset, maybe due to imbalanced classes or the nature of the features.

5.1.5 XGBoost

Performance: With an F1 Score of 44.3%, Accuracy of 43.9%, Precision of 43.9%, and Recall of 44.7%, XGBoost showed the best balanced performance.



Discussion: The XGBoost model demonstrated its robustness for this dataset by outperforming other models in terms of consistency across all measures. However, feature engineering or hyperparameter adjustment could improve its overall performance even more.

5.2 Comparative Analysis

Two comparative graphs (Figures 6 and 7) are presented to help visualize the relative performance of these models. The variations in Accuracy, Precision, Recall, and F1 Score among the models are displayed in these charts.

Important Findings: Naive Bayes had the highest recall, which makes it appropriate in situations when identifying as many attacks as possible is crucial, even if doing so results in false positives.

The best balanced performance was shown by XGBoost, which makes it a reliable option for general-purpose DDoS detection.

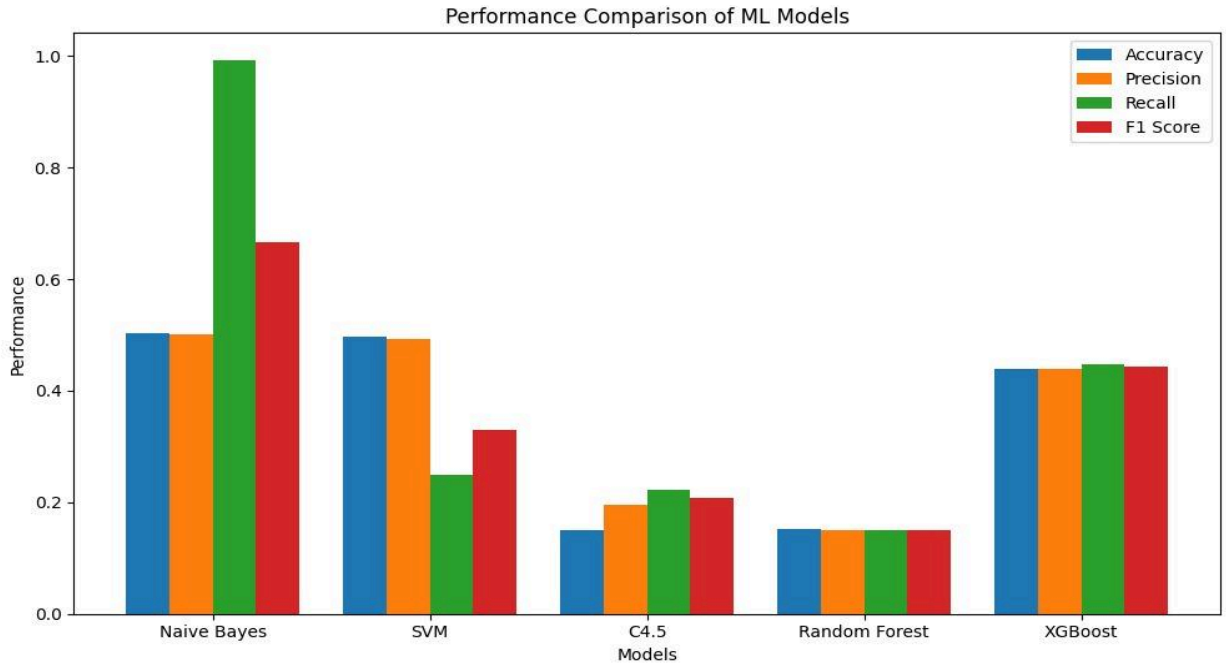
SVM's relevance in this situation is limited by its moderate accuracy and recall issues.

The poor performance of Random Forest and C4.5 suggests that they are less useful for this dataset.

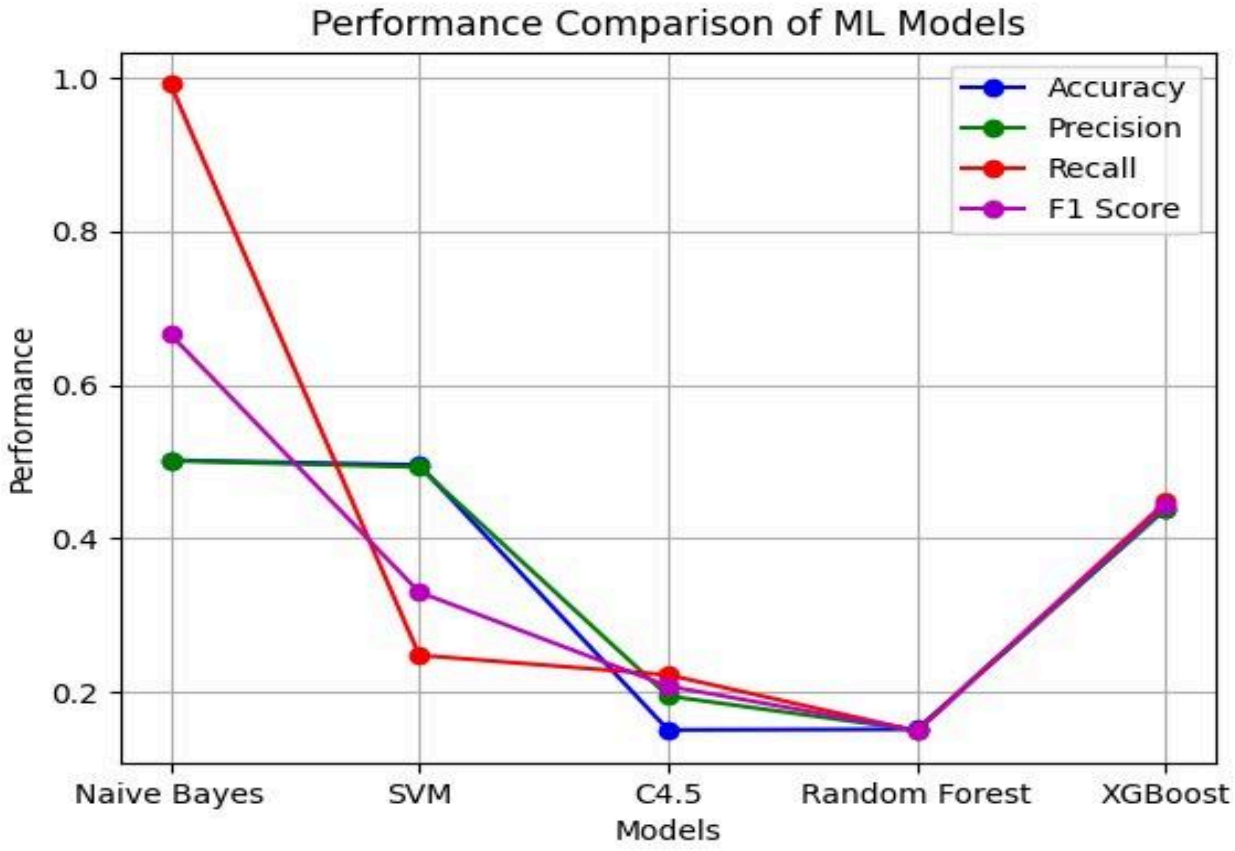
5.2.1 Performance Comparison Table

Model	Accuracy	Precision	Recall	F1 score
Random Forest	0.15	0.15	0.15	0.15
SVM	0.50	0.50	0.50	0.46
XG Boost	0.44	0.44	0.44	0.44
C.4.5	0.15	0.14	0.15	0.15
Naive Bayes	0.50	0.54	0.50	0.34

5.2.2 Performance Comparison Chart:



5.2.3 Detailed Performance Breakdown:



CHAPTER 6 : CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The research used five different algorithms—Naive Bayes, C4.5, SVM, Random Forest, and XGBoost—to successfully develop a machine-learning-based DDoS detection system. Important conclusions include:

With the highest recall, Naive Bayes is appropriate for situations where the majority of DDoS attacks must be detected at the risk of some false positives.
XGBoost is a reliable option for DDoS detection in dynamic contexts because it demonstrated the most balanced performance across all criteria.
SVM's usefulness was limited by its moderate accuracy and recall issues.
The low performance of Random Forest and C4.5 suggests that they are inappropriate for this dataset.
The whole research highlights the necessity of customized solutions, including XGBoost for balanced performance and Naive Bayes for use cases with a high detection burden.

6.2 Future Scope

1. **Feature Engineering**: To increase the accuracy and resilience of the model, apply sophisticated feature extraction techniques.
2. **Hyperparameter Optimization**: Adjust hyperparameters to improve XGBoost and other models' performance.
3. **Integration with Cloud**: Extend the detection system to manage scaled and dispersed cloud infrastructures.
4. **Real-time Detection**: Use streaming data processing frameworks to put real-time detection pipelines into place.

5. **Advanced Models**: Investigate deep learning models to identify temporal relationships in network traffic, such as recurrent neural networks (RNNs).

REFERENCES

1. DDoS attack detection using machine learning techniques in cloud computing environments - https://ieeexplore.ieee.org/document/8284731
2. DDoS Attack Detection Based on Random Forest
3. Prevention and detection of DDOS attack in virtual cloud computing environment using Naive Bayes algorithm of machine learning
4. An Evolutionary SVM Model for DDOS Attack Detection in Software Defined Networks
5. XGBoost Classifier for DDoS Attack Detection and Analysis in SDN-Based Cloud
6. Apache Web Server - https://httpd.apache.org/
7. Wireshark - https://www.wireshark.org/
8. Hping3 DDoS Attack Script - https://www.kali.org/tools/hping3/
9. Google Colab - https://colab.research.google.com/
10. BlackBox AI - https://www.blackbox.ai/
11. Stack Overflow - https://stackoverflow.com/
12. Youtube - http://youtube.com
13. PentestGPT-3.5 https://pentestgpt.ai/

APPENDIX

Self generated data by attacking web servers on target machines using hping3 script.

Different Datasets each for Normal and Attack traffic in various kinds of attack including:

1. UDP Flood
2. TCP Reset
3. ICMP Ping of Death
4. HTTP Flood

CV : APIKSHA (22BCS020)

Email : panditaapiksha@gmail.com

Phone: +91 9103393521

LinkedIn: linkedin.com/in/apiksha-22941a256/

GitHub: github.com/Apiksha

Education

Shri Mata Vaishno Devi University
B.Tech in Computer Science (Oct 2023 – May 2026)

- **CGPA**: 8.44
- Relevant Courses: Data Structures, Algorithms, Operating Systems, AI, Software Defined Networking, DBMS

Technical Skills

- **Languages: C++, C, Java, Python, JavaScript, Shell Scripting**
- **FrontEnd Development: HTML5, CSS3, Bootstrap, Javascript, ReactJS, JQuery**
- **BackEnd Development: NodeJS, ExpressJS, API**
- **Development tools and Environments: Git, Github, VSCode, PyCharm, MySQL, Google Colab, Docker**
- **Database Language: SQL**
- **Operating System: Windows, Ubuntu**

Experience

- **Research Intern, IIT Jammu (June – July 2024)**
  Conducted research on low RCS antenna design and blockchain-enabled federated learning.
- **Frontend Developer, NAT (Dec 2023 – Jan 2024)**
  Built a React-based website to enhance the NGO's digital presence.

- **Junior Web Developer, GDSC SMVDU (Jan 2023 – Jan 2024)**
  Designed user-friendly web solutions while fostering collaboration and skill development.

Projects

- **DDoS Attack Detection System**: Developed a scalable cloud system with 80% attack detection accuracy using CloudSim and Java.
- **Cafe Management System**: Designed a real-time management tool for ordering and employee data handling using Java and SQL.

Achievements

- **I2C2 Hackathon Winner (2023)**: Developed SEEKHOsphere, a blockchain-based learning platform.
- **Microsoft for Startups Founders Hub**: Gained mentorship and resources for innovative projects.

CV : SOUMITRA RAI (22BCS090)

Email: raisoumitra123@gmail.com
Phone: +91 6393469030
LinkedIn: linkedin.com/in/soumitrarai

Education
Shri Mata Vaishno Devi University, Katra, JK
B.Tech in Computer Science and Engineering (Oct 2022 – May 2026)
CGPA: 7.21
Relevant Courses: Data Structures, Algorithms, Operating Systems, DBMS, Software Defined Networking, AI

Technical Skills
Languages: GoLang, C++, C, Java, SQL, JavaScript, Python, Bash
Cybersecurity Tools: Wireshark, OSINT, Metasploit, Burp Suite, Nessus, Nmap
Cloud Computing: Google Cloud, Kubernetes, Docker, CloudSim, Ansible
Development Tools: Git, Github, Packet Tracer

Experience
Summer Intern, IIITDM Kanchipuram (June – Aug 2024)

- Researched "Password Attack-Based Vulnerabilities."
- Developed a parser for cryptographic vulnerabilities, improving security by 30%.

Summer Intern, NIELIT Shillong (June – July 2024)

- Developed an IoT-based air quality and climate control system.
- Achieved 15% energy efficiency and 35% cost reduction.

Winter Intern, NEHU Shillong (Jan 2024)

- Built an image model using OpenCV and TensorFlow, achieving 85% accuracy for medical applications.

Projects
ChatBot for Food Delivery App: A Python and DialogflowCX chatbot with search, recommendation, and order management.
Security Operations Center Lab: Created a SOC lab for threat detection using Kali Linux and Wireshark.

Achievements

- Microsoft for Startups Founders Hub participant for "Smart Water Management Systems."
- 3rd Prize in BARC Outreach Program – Nuclear Science Quiz.

Certifications

- Google Cybersecurity Professional Certificate
- Introduction to Cloud Computing (IIT Kharagpur)
- Introduction to Critical Infrastructure Protection by OPSWAT Academy
- Python & ML Bootcamp (GDSC SMVDU)