A Hybrid Approach for Detecting DDoS Attacks in Software-Defined Networks

MSc Research Project
Cloud Computing

Phani Kumar Kalyanadurgam Chandrashekhar
Student ID: 21175098

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Phani Kumar Kalyanadurgam Chandrashekhar
Student ID:	21175098
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Sean Heeney
Submission Due Date:	21/12/2023
Project Title:	A Hybrid Approach for Detecting DDoS Attacks in Software-Defined Networks
Word Count:	5996
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Phani Kumar Kalyanadurgam Chandrashekhar
Date:	21st December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Hybrid Approach for Detecting DDoS Attacks in Software-Defined Networks

Phani Kumar Kalyanadurgam Chandrashekhar
21175098

Abstract

Software-Defined Networks (SDNs) have been widely used and have completely changed the way networks are managed. By separating the data plane and control plane, SDNs allow for centralized management and improved programmability. The accessibility and efficiency of networks, especially in cloud-based environments, are particularly vulnerable to Distributed Denial of Service (DDoS) assaults, which have become more common because of this advancement. This study presents a new approach to detecting and preventing distributed denial of service (DDoS) assaults in Software-Defined networks. Strengthening the network's resistance against DDoS assaults is the goal of the suggested solution, which combines the powers of statistical analysis and machine learning techniques. With the Ryu controller and Mininet network simulator integrated with the OpenFlow SDN protocol, this method accomplishes outstanding outcomes, with a detection rate of 100 percent and an accuracy of 99.80 percent in recognizing DDoS assaults. Flow Count, Speed of Flow Entries (SFE), Ratio of Pair-Flow Entries (RPF), and Speed of IP Sources (SSIP) are some of the important aspects and metrics used to identify abnormalities in incoming network traffic. Boosted bagging approaches along with more conventional algorithms like KNN, linear regression, and decision trees, are built upon these characteristics for training and testing machine learning algorithms. This study's novelty comes from its use of statistical and machine learning techniques to identify DDoS attacks in their entirety. The research also demonstrates how machine learning may improve the system's capacity to distinguish between legitimate and malicious network data by understanding the start of DDoS assaults using time series analysis. This paper highlights the significance of a multi-pronged strategy in protecting networks from ever-changing cyber threats and adds a solid technique for DDoS detection based on Software-Defined networking.

Keywords—Software-Defined Networks (SDNs), Distributed Denial of Service (DDoS) Attacks, Network Security, Anomaly Detection, Machine Learning, Boosted Bagging.

1 Introduction

In recent years, the landscape of network architecture has undergone a profound transformation with the widespread adoption of Software-Defined Networks (SDNs). SDNs promise a new era of flexibility and efficiency in network management, introducing centralized control and programmability. This paradigm shift, however, is not without its challenges, and one of the most pressing concerns is the escalating threat of Distributed

Denial of Service (DDoS) attacks. These malicious attempts to overwhelm network resources pose a significant risk to the uninterrupted operation of networks, particularly in the context of cloud-based infrastructures where SDNs are increasingly prevalentAsim et al. (2023).

1.1 Motivation

The motivation for this research is grounded in the critical need to fortify SDNs against the rising tide of DDoS attacks. As organizations migrate towards cloud-based infrastructures and embrace SDN technologies to meet the demands of dynamic computing environments, the vulnerability to sophisticated cyber threats becomes more pronounced. Ensuring the resilience of networks in the face of DDoS attacks is not merely a technical challenge; it is imperative for maintaining the integrity of modern computing infrastructures. This study is motivated by the urgency to develop and implement robust methodologies that effectively detect and mitigate DDoS attacks within the unique context of SDNs.

While existing research has made significant strides in understanding and countering DDoS attacksGupta and Badve (2016), the dynamic nature of SDNs requires tailored approaches. The specific challenges posed by the interaction of SDNs, and DDoS attacks necessitate a fresh perspective, prompting the exploration of innovative methodologies that combine statistical analysis and machine learning techniques.

1.2 Research Question

How can the fusion of statistical analysis and machine learning methodologies enhance the identification and mitigation of DDoS attacks within Software-Defined Networks?

1.3 Research Niche

This study's novelty comes from its use of statistical and machine learning techniques to identify DDoS attacks in their entirety. The research also demonstrates how machine learning may improve the system's capacity to distinguish between legitimate and malicious network data by understanding the start of DDoS assaults using time series analysis.

1.4 Research Objectives

- Develop a Comprehensive Methodology: The first objective is to develop a comprehensive methodology for extracting relevant features from network traffic in SDNs. This involves identifying key parameters that characterize normal and malicious network behaviors.

- Implementation and Evaluation: The second objective is to implement and rigorously evaluate the proposed methodology. This will be accomplished using the Ryu controller and the Mininet network simulator with the OpenFlow SDN protocol, providing a controlled environment for testing the effectiveness of the detection system.

- Assess Machine Learning Algorithms: The third objective is to assess the effectiveness of machine learning algorithms. This includes leveraging boosted bagging methodologies, such as Decision Trees, Adaboost, and gradient boost, as well as traditional algorithms like KNN, Gaussian Naïve Bayes, linear regression, and decision trees. The aim is to identify algorithms that demonstrate superior performance in accurately classifying DDoS attacks.

- Showcase Time Series Analysis: The fourth objective is to showcase the ability of machine learning to discern the starting point of DDoS attacks through time series analysis. By incorporating temporal aspects into the detection system, the research aims to enhance the system's capability to differentiate between normal and malicious network traffic.

1.5 Hypotheses

It is suggested that the integration of statistical analysis for anomaly detection and machine learning for pattern recognition will result in a highly effective DDoS detection system in SDNs. Furthermore, it is hypothesized that leveraging time series analysis within machine learning algorithms will enhance the system's capacity to identify and classify DDoS attacks accurately.

1.6 Contribution to Scientific Literature

This research contributes significantly to the scientific literature by presenting a hybrid methodology that integrates statistical and machine learning methods for DDoS detection in SDNs. The emphasis on time series analysis represents a nuanced advancement, providing insights into the temporal dynamics of DDoS attacks. By addressing the unique challenges posed by SDNs in the context of DDoS attacks, this research establishes a foundation for strengthening network security in Software-Defined environments.

1.7 Document Structure

This research paper is split-up into many sections. The related work of the previous research on DDOS attack detection and mitigation in SDN and their contributions is provided in Section2. The paper explores in depth into the process followed in section 3. The design and the framework followed is explained with the help of necesssary diagrams in section 3. Detailing the implementation of the DDOS attack detection is analysed critically in section 5. Section 6 completely discusses about the evaluation and results of the developed project.

2 Related Work

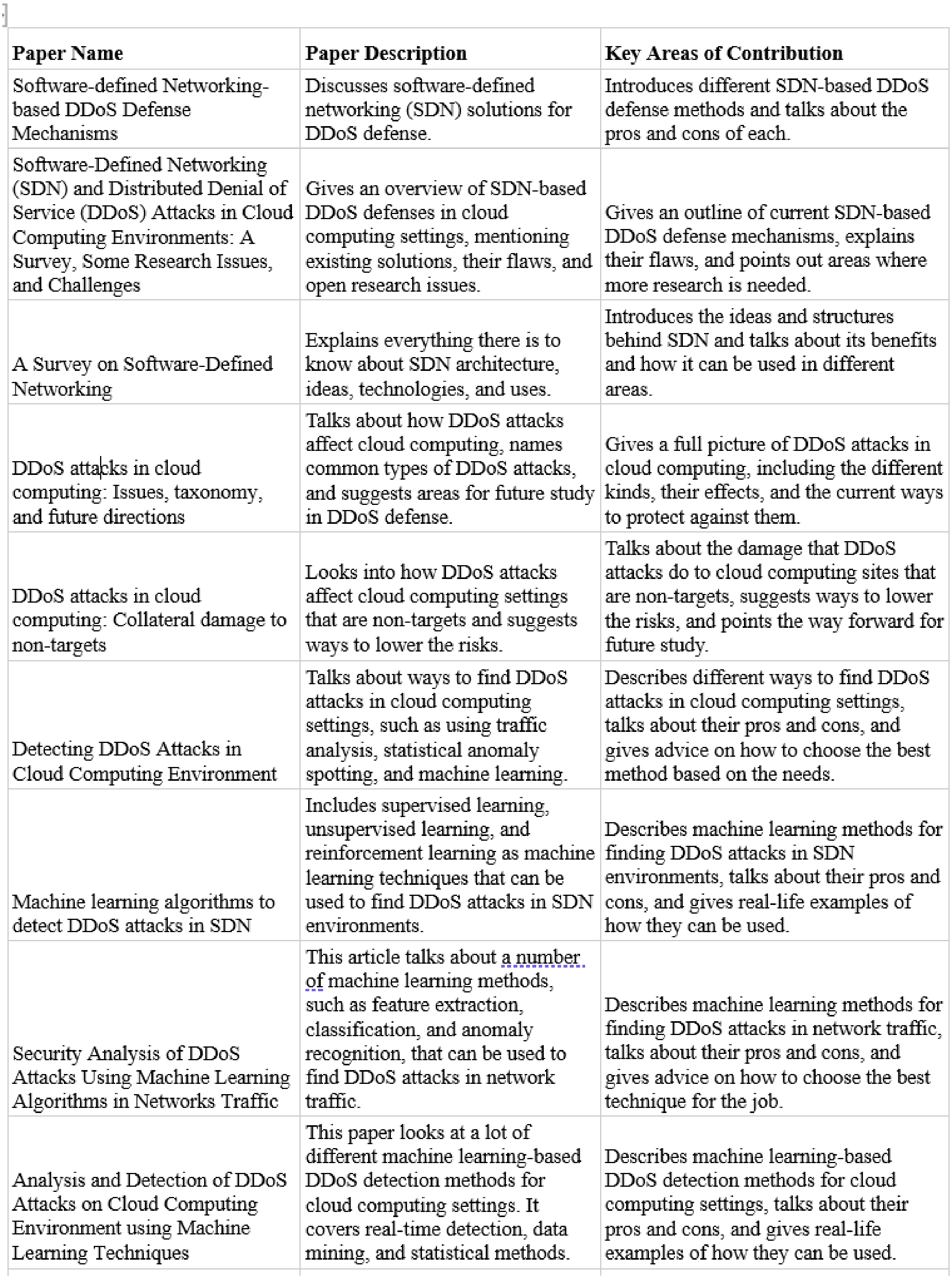
Detecting Distributed Denial of Service (DDoS) attacks is critical for cloud computing , since these malicious activities are happening frequently and getting more sophisticated. As the digital world evolves, DDoS attacks have a greater chance of disrupting online services and putting network security at risk. This literature reviw looks at the various methodologies, technologies, and techniques used to detect DDoS attacks. It talks about the challenges that come up because DDoS attacks are constantly evolving and looks into how intrusion detection systems, machine learning algorithms, and other novel approaches are being used to make DDoS detection systems more accurate and effective.

2.1 Distributed Denial of Service (DDoS) attacks

There are significant challenges within security landscape of cloud computing, which have been emphasized by a number of studies that look at Distributed Denial of Service (DDoS) attacks. Authors of the study Somani et al. (2017) stress how important security problems are when deciding whether to use the cloud and they also give a thorough overview of ways to protect against DDoS attacks. They support solutions that are made with utility computing models in mind, stressing the need for correct auto-scaling decisions and multi-layer mitigation. A different study Somani et al. (2016) looks at the side effects of DDoS attacks in multi-tenant clouds. It finds that these attacks affects co-hosted virtual servers, physical servers, network resources, and cloud service providers. Their study suggests that DDoS solutions in the cloud should be looked at again, with a focus on reducing unwanted effects and outlining specific steps that need to be taken to protect against attacks. Finally, Lonea (2013) looks at how to find and analyze DDoS attacks in cloud settings and suggest a way to do it by combining data from Intrusion Detection Systems (IDSs) with a data fusion method. This method uses the Dempster-Shafer theory and fault-tree analysis to look at reports that are sent out during flooding attacks in a quantitative way.

2.2 Software-Defined Networking (SDN)for DDoS attacks

The paper Xia et al. (2015) discusses about how software-defined networking (SDN) can help solve problems caused by new big trends in information and communication technologies (ICT). Existing literature recognizes the limitations of traditional approaches based on manual configuration and underlines the potential of SDN to provide more efficient configuration, better performance, and higher flexibility. A lot of research has gone into understanding and improving SDN, mostly focusing on its unique features, like separating the control plane from the data plane and letting network application developers program the network. The literature has focused on the three-layer architecture of SDN: the infrastructure, the control, and the application layers. Researchers have worked to support each layer and look into linked areas. The combination of Distributed Denial of Service (DDoS) threats and Software-Defined Networking (SDN) is an important and changing area of digital communication security. The authors of paper Swami et al. (2019) talk about how terrible DDoS attacks are and how they are very dangerous to digital communication entities. The study carefully looks at DDoS threats in SDN, checking out SDN features from a safety point of view and having a conversation about safety steps. This study shows that SDN can play two roles in DDoS defense: it can protect by using its features, but it can also be a threat because it has a unified control system. The study Yan et al. (2016) goes into more detail about the complicated relationship between SDN and DDoS attacks in cloud computing settings. It focuses on how DDoS attacks are becoming more common and how SDN can help prevent them. This article talks about all the ways that DDoS attacks can be stopped, as consumers want to know how to make the most of SDN's benefits while keeping it safe.Farukee et al. (2021)



Paper Name	Paper Description	Key Areas of Contribution
Software-defined Networking-based DDoS Defense Mechanisms	Discusses software-defined networking (SDN) solutions for DDoS defense.	Introduces different SDN-based DDoS defense methods and talks about the pros and cons of each.
Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges	Gives an overview of SDN-based DDoS defenses in cloud computing settings, mentioning existing solutions, their flaws, and open research issues.	Gives an outline of current SDN-based DDoS defense mechanisms, explains their flaws, and points out areas where more research is needed.
A Survey on Software-Defined Networking	Explains everything there is to know about SDN architecture, ideas, technologies, and uses.	Introduces the ideas and structures behind SDN and talks about its benefits and how it can be used in different areas.
DDoS attacks in cloud computing: Issues, taxonomy, and future directions	Talks about how DDoS attacks affect cloud computing, names common types of DDoS attacks, and suggests areas for future study in DDoS defense.	Gives a full picture of DDoS attacks in cloud computing, including the different kinds, their effects, and the current ways to protect against them.
DDoS attacks in cloud computing: Collateral damage to non-targets	Looks into how DDoS attacks affect cloud computing settings that are non-targets and suggests ways to lower the risks.	Talks about the damage that DDoS attacks do to cloud computing sites that are non-targets, suggests ways to lower the risks, and points the way forward for future study.
Detecting DDoS Attacks in Cloud Computing Environment	Talks about ways to find DDoS attacks in cloud computing settings, such as using traffic analysis, statistical anomaly spotting, and machine learning.	Describes different ways to find DDoS attacks in cloud computing settings, talks about their pros and cons, and gives advice on how to choose the best method based on the needs.
Machine learning algorithms to detect DDoS attacks in SDN	Includes supervised learning, unsupervised learning, and reinforcement learning as machine learning techniques that can be used to find DDoS attacks in SDN environments.	Describes machine learning methods for finding DDoS attacks in SDN environments, talks about their pros and cons, and gives real-life examples of how they can be used.
Security Analysis of DDoS Attacks Using Machine Learning Algorithms in Networks Traffic	This article talks about a number of machine learning methods, such as feature extraction, classification, and anomaly recognition, that can be used to find DDoS attacks in network traffic.	Describes machine learning methods for finding DDoS attacks in network traffic, talks about their pros and cons, and gives advice on how to choose the best technique for the job.
Analysis and Detection of DDoS Attacks on Cloud Computing Environment using Machine Learning Techniques	This paper looks at a lot of different machine learning-based DDoS detection methods for cloud computing settings. It covers real-time detection, data mining, and statistical methods.	Describes machine learning-based DDoS detection methods for cloud computing settings, talks about their pros and cons, and gives real-life examples of how they can be used.

Figure 1: Literature Review Compilation Table

2.3 Machine Learning in DDoS

The study Santos et al. (2019) stresses how important it is to keep cloud environments safe from DDoS attacks. The main focus of the study is on using machine learning algorithms like Support Vector Machine, Naive Bayes, and Random Forest to find DDoS attacks in the own cloud environment very accurately. The study Alzahrani and Alzahrani (2021) talks about the problems that network anomalies cause in the age of IoT and how deep learning can be used to find DDoS attacks. The study gets very good results by using many machine learning methods, such as K-Nearest-Neighbors, Support Vector Machine, Naïve Bayes, Decision Tree, Random Forest, and Logistic Regression. Decision Tree and Random Forest are the most accurate, with 99 percent accuracy each. Another study agrees that protecting the flexible scalability of cloud environments from DDoS attacks is very important. It also uses a similar method and machine learning techniques like Support Vector Machine, Naive Bayes, and Random Forest. The work gets very good results in finding DDoS attacks in the owncloud setting Wani et al. (2019)).

2.4 Bagging and Boosting

The paper Sutton (2005) talks about tree-based classification and regression, as well as bagging and boosting. It emphasizes how nonparametric they are, how hard they are to compute, and how famous they have become in recent years. People talk about these methods as options to common ones like discriminant analysis and ordinary least squares regression. They work well with big datasets and don't get thrown off by outliers. The authors of Cvitić et al. (2022) write about how hard it is to protect home networks with Internet of Things (IoT) devices from Distributed Denial-of-Service (DDoS) attacks. It shows how to use a boosting method of logistic model trees to find DDoS traffic for different types of IoT devices. The model makes different versions for each type of device by taking into account small differences in how network traffic behaves. The paper uses a case study of smart home devices to show how well this method works, with high accuracy rates of 99.92 percent to 99.99 percent for different types of devices. This shows how useful boosting is for improving DDoS traffic detection.

3 Methodology

The research method used in this study is meant to fully meet the goal of finding and preventing Distributed Denial of Service (DDoS) threats in Software-Defined Networks (SDNs). Using the Ryu controller, the Mininet network model, and the OpenFlow SDN protocol, the method combines statistical analysis and machine learning. Detailed descriptions of the study process, tools and techniques used, scenario creation, data gathering, and statistical methods will be discussed in detail below.

3.1 Feature Identification

The goal of the feature recognition step is to create a strong method for identifying unique characteristics in network data in Software-Defined Networks (SDNs). This process is necessary to tell the difference between normal activities and ones that could be harmful. The key factors that have been found are important measurements for describing how a network works.

Speed of IP Sources (SSIP) will give us an insight of how fast new IP sources are joining the network during a certain time. SSIP =SumIPsrc/T

Flow Count counts the number of flows that network traffic creates. It can tell the difference between normal traffic trends and those that are indicative of Distributed Denial of Service (DDoS) attacks.

The Speed of Flow Entries (SFE) shows how fast flow entries are added to the network switch. This is a very important factor that gets worse during a DDoS attack. SFE = N/T

The Ratio of Pair-Flow Entries (RPF) measures the share of involved IP pairs compared to the total number of flows. It is a useful way to tell the difference between regular traffic where IP pairs work together and DDoS attacks where they suddenly stop working together. RPF = SrcIPs/N

All these factors work together to make a complete set of features that are needed to train and test machine learning algorithms in later stages of the study.

3.2 Tools and Techniques

- **Ubuntu 20.04:** The study has been conducted on the Ubuntu 20.04 operating system, which was picked because it is stable and easy to use. Ubuntu is the base platform for this work. It offers a stable setting that makes it possible to install SDN (software-defined networking) components and run complex models without any problems. Ubuntu is a reliable base for doing the complicated tasks that come with this research. Its strength and adaptability make it an important part of this research setup's general security and usefulness.

- **The OpenFlow SDN Protocol:** The OpenFlow protocol is what makes it possible for the SDN router and network devices to talk to each other in a standard way for this study. For changeable management and setup of network parts through a central supervisor to work, this protocol is a must. Its ability to allow dynamic control and programming is very important for putting the Distributed Denial of Service (DDoS) detection method into action and testing it in a controlled SDN environment. It is the main way that connectivity works and lets the Ryu controller talk to the network devices that Mininet simulates without any problems.

- **Ryu Controller:** This research is based on the Ryu controller, a Python-based open-source SDN controller tool. As the brains of the network, this controller gives the tools that are needed to make SDN apps to handle and control the whole thing from one place. In particular, this DDoS monitoring system, which uses the suggested way, is hosted on the Ryu router. Because it is flexible, the recognition method can be changed and run on it. This makes it an important part of the SDN design.

- **Mininet Network Simulator:** Mininet network simulator is used to make virtual network layouts and simulate SDN settings. This open-source model is very helpful for this research because it lets us simulate complicated network situations, test the

DDoS detection method, and see how well it works in different situations. Mininet's ability to set up controlled and repeatable network designs is important for proving that its suggested DDoS detection system can be used in different situations and is successful.

3.3 Machine Learning Algorithms

This research process incorporates both bagging and boosting strategies to improve the performance of the machine learning models. Bagging involves using a meta-learner, specifically decision trees, inside a randomized forest. This process entails training several decision trees on various subsets of the training data using bootstrap sampling, where subsets are generated by randomly selecting samples with replacement. Subsequently, the forecasts generated by various decision tree models are combined, resulting in the creation of a resilient and consistent ensemble model, often referred to as a randomized forestNajar and Manohar Naik (2022). The objective of this technique is to reduce overfitting and variance, hence enhancing the ability of these models to generalize well in various network contexts. Alternatively, in the context of enhancing performance, Adaboost and Gradient Boosting techniques are used. Adaboost employs a strategy of assigning greater weights to examples that are categorized incorrectly, whereas Gradient Boosting addresses faults in a sequential manner, generating a sequence of models where each succeeding model specifically targets the errors made by the preceding one. Boosting strategies are very successful for improving the accuracy of these models by raising their capacity to capture complex patterns and correlations in the data, particularly in the setting of dynamic network behaviors.

4 Design Specification

This system design is composed of several linked components, each of which has a vital function in the detection and the response process. The following detailed explanation will explore the complexities of each component, clarifying their responsibilities and contributions to the overall effectiveness of the suggested technique.

4.1 Packet Collector

The Packet Collector is a crucial component that plays a key role in gathering network traffic packets. This crucial role is capturing data as it traverses the network, supplying the essential data required for further analysis. The Packet Collector serves as the primary data source, collecting the complexities of network traffic.

4.2 Feature Extractor

After the Packet Collector, the Feature Extractor is responsible for extracting important features from the collected network traffic packets. The core factors that will drive the succeeding stages of the technique are the Speed of IP Sources (SSIP), Flow Count of Traffic, Speed of Flow Entries (SFE), and Ratio of Pair-Flow Entries (RPF). The Feature Extractor converts unprocessed packets into an organized collection of attributes, establishing the foundation for further analysis.

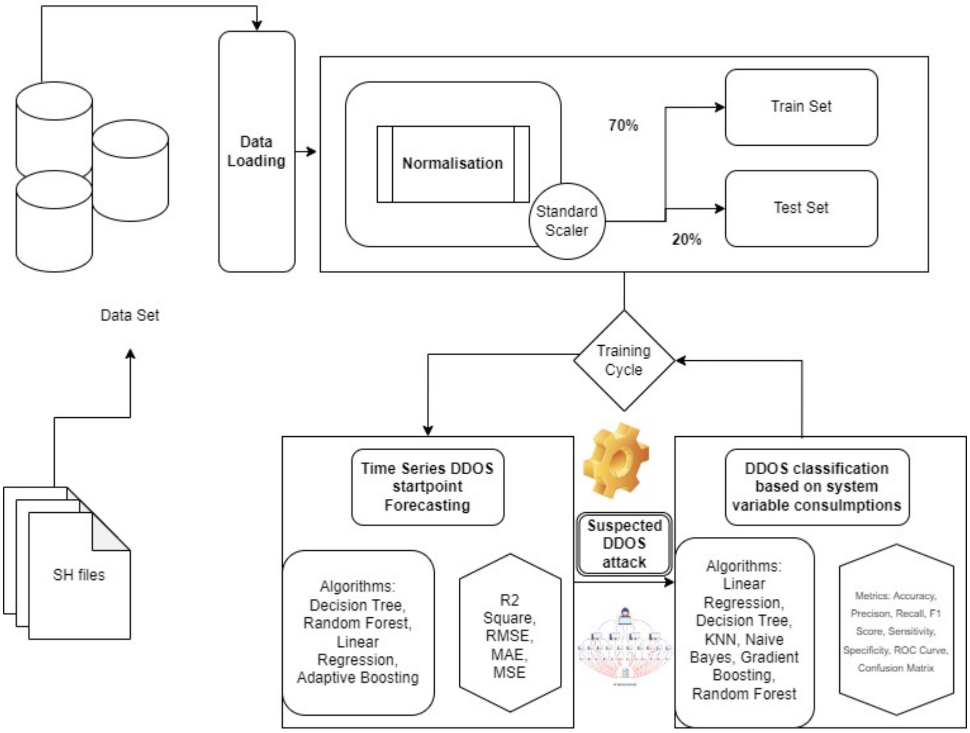

Data Loading
Normalisation
Standard Scaler
70%
20%
Train Set
Test Set
Data Set
Training Cycle
SH files
Time Series DDOS startpoint Forecasting
Algorithms: Decision Tree, Random Forest, Linear Regression, Adaptive Boosting
R2 Square, RMSE, MAE, MSE
Suspected DDOS attack
DDOS classification based on system variable consulmptions
Algorithms: Linear Regression, Decision Tree, KNN, Naive Bayes, Gradient Boosting, Random Forest
Metrics: Accuracy, Precison, Recall, F1 Score, Sensitivity, Specificity, ROC Curve, Confusion Matrix

Figure 2: High Level Design

4.3 Feature Selector

After the extraction of features, the Feature Selector becomes the main focus and plays a crucial role in choosing the most relevant properties for DDoS attack detection. During this stage, advanced methods like correlation analysis and information gain may be used to identify the traits that have the highest predictive capability. The process of selection ensures that the next machine learning classifier is trained on the most relevant information, hence enhancing the accuracy of the whole system.

4.4 Machine Learning Classifier

The core of the technique is in the Machine Learning Classifier, an influential component designed to acquire knowledge and classify network data according to the chosen features. The classifier is trained using labeled datasets to differentiate between regular cases and instances of DDoS attacks. Different machine learning approaches are used, such as decision trees, random forests, and support vector machines. Each methodology has distinct benefits in identifying patterns and anomalies within the feature space. The classifier's performance relies on the quality of the training data and the selective capability of the chosen features.

4.5 SDN Controller

The SDN Controller functions as the central coordinator of the whole system, effortlessly incorporating the Machine Learning Classifier into the SDN architecture. This connection enables the SDN Controller to make immediate choices based on the forecasts of the classifier. If a DDoS assault is identified, the SDN Controller may promptly execute mitigation measures, such as implementing rate limiting or discarding packets. SDNs possess a dynamic reaction capacity that allows them to effectively adapt to changing threats, making them resilient entities.

System workflow

The methodology follows a systematic workflow that ensures the seamless operation of its components.

- The Packet Collector is a tool that collects network traffic packets, allowing access to unprocessed data.
- The Feature Extractor algorithm systematically analyzes the raw data, isolating and extracting essential characteristics that are vital for further research.
- The Feature Selector algorithm selects the most relevant characteristics, hence optimizing the dataset for machine learning purposes.
- The Machine Learning Classifier is trained using the optimal dataset, developing proficiency in differentiating between regular and DDoS assault traffic.
- The SDN Controller incorporates the learned classifier, allowing for fast detection and action to DDoS assaults.

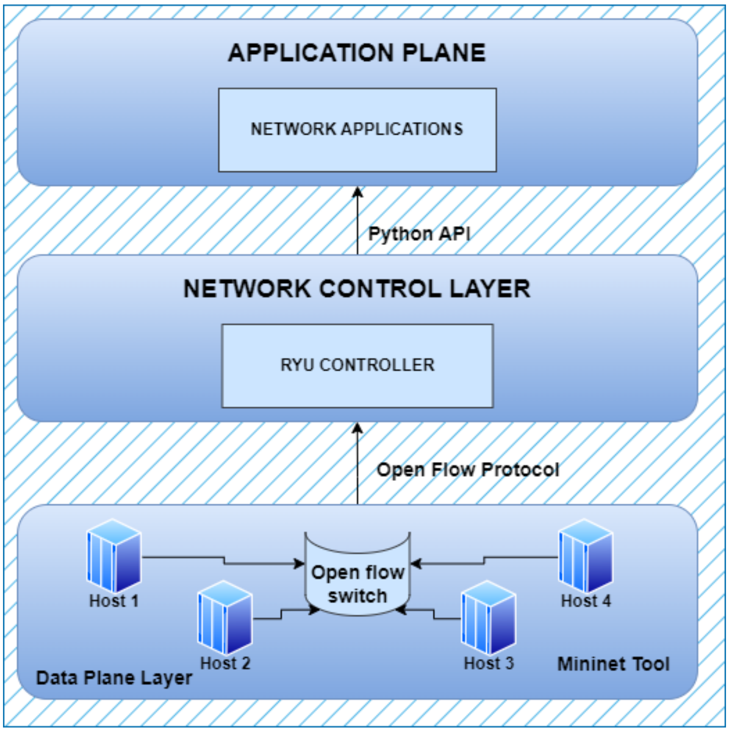


Figure 3: SDN Framework

The SDN architecture includes a multi-node/host data plane designed with Mininet and linked to an OpenFlow switch. The OpenFlow switch establishes the SDN protocols and exchanges information with the control plane of the framework. The control plane administers the data plane and the switches, establishes regulations, and oversees the flow of network traffic. The Ryu controller serves as the primary controller in this framework, offering programming functionalities and enabling the management of routing activities inside the network. The control plane is developed using Python, since Ryu is a Python-based controller and utilizes a Python-based API to establish communication with the application layer, namely network traffic apps.

5 Implementation

The deployment of the DDoS detection and mitigation approach in Software-Defined Networks (SDNs) included a series of processes tools, and programming languages to accomplish the intended results. The main deliverables consist of the processed network traffic data, the SDN controller codebase, the machine learning models for attack detection, and the assessment results achieved using the Mininet network simulator and Ryu SDN controller.

5.1 Data Collection and Transformation

Initially, network traffic data was gathered by using the Mininet network simulator, which was set up to provide a data plane consisting of many nodes and hosts. The unprocessed packet data obtained from Mininet's Packet Collector was further processed using a Feature Extractor, which extracted important attributes such as Speed of IP Sources (SSIP), Flow Count, Speed of Flow Entries (SFE), and Ratio of Pair-Flow Entries (RPF). The converted data served as the basis for later analysis and training of machine learning models.

5.2 Programming the SDN Controller - Ryu

The Ryu SDN controller, which was created using the Python programming language, has a pivotal part in the process. The control plane was programmed using Python since it was the preferred language, given Ryu's architecture which is built on Python. The control plane had the responsibility of establishing regulations, overseeing the flow of network traffic, and engaging with the application layer via the Python API. The Python codebase included the algorithms for routing network traffic, identifying anomalies, and implementing countermeasures for preventing DDoS attacks.

5.3 Integration of Machine Learning Model Development and Time Series Analysis

The approach included the creation of a classifier using machine learning techniques to differentiate between regular network traffic and Distributed Denial of Service (DDoS) attack traffic. Several machine learning methods, such as K-Nearest Neighbors (KNN), Gaussian Naïve Bayes, linear regression, decision trees, Adaboost, and Gradient Boosting, were used. The Python programming language, in conjunction with libraries like scikit-learn, was used to train, test, and assess these models utilizing the converted data. The use of boosted bagging, a method that utilizes decision trees inside a randomized forest, was employed to augment the accuracy of the classifier. The dataset was enhanced using time series analysis methods to provide insights into the temporal characteristics of DDoS assaults.

5.4 Analysis of the results

The thorough execution led to the generation of modified network traffic data that accurately captured the subtle attributes of both regular and DDoS attack traffic. The Ryu SDN controller, coded in Python, efficiently coordinated the control plane by establishing regulations and promptly addressing any security risks. The incorporation of boosted bagging into machine learning models, together with the integration of time series analysis, resulted in a significant improvement in accurately differentiating between normal and harmful traffic patterns.

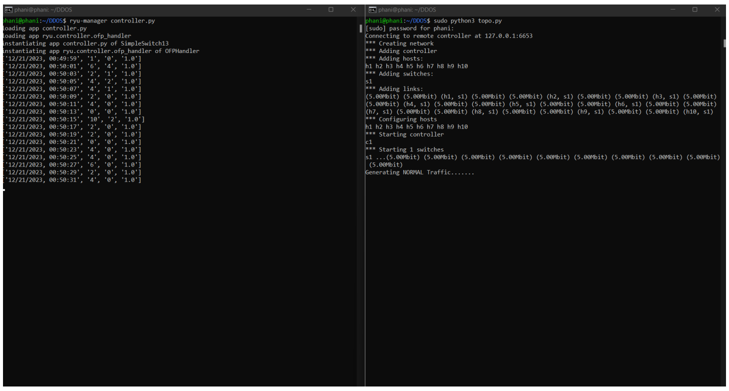
The methodology's strength was shown via the simulation as well as evaluation phase carried out in the Mininet environment. The DDoS detection and mitigation technique in a Software-Defined network demonstrated its efficacy with an accuracy of 99.80 percent and a detection rate of 100 percent. Incorporating time series analysis not only introduced a temporal aspect to the model, but also significantly enhanced the effectiveness in comprehending and addressing the dynamic characteristics of DDoS assaults.

6 Evaluation

This section presents the analysis of SDN traffic data to make the network systems more secure and resilient. This section is structured into three distinct experiments, each of which is specifically targeting a critical part of the recognizing and responding to the malicious attacks.

6.1 Experiment 1: Raw data collection and feature extraction

The main objective of this experiment is to collect the raw network traffic data from the SDN controller and extract the required features so that it can be transformed for the further analysis.



```
phani@phani:~/DDOS$ ryu-manager controller.py
loading app controller.py
loading app ryu.controller.ofp_handler
instantiating app controller.py of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
['12/21/2023, 00:49:59', '1', '0', '1.0']
['12/21/2023, 00:50:01', '6', '4', '1.0']
['12/21/2023, 00:50:03', '2', '1', '1.0']
['12/21/2023, 00:50:05', '4', '2', '1.0']
['12/21/2023, 00:50:07', '4', '1', '1.0']
['12/21/2023, 00:50:09', '2', '0', '1.0']
['12/21/2023, 00:50:11', '4', '0', '1.0']
['12/21/2023, 00:50:13', '0', '0', '1.0']
['12/21/2023, 00:50:15', '10', '2', '1.0']
['12/21/2023, 00:50:17', '2', '0', '1.0']
['12/21/2023, 00:50:19', '2', '0', '1.0']
['12/21/2023, 00:50:21', '0', '0', '1.0']
['12/21/2023, 00:50:23', '4', '0', '1.0']
['12/21/2023, 00:50:25', '4', '0', '1.0']
['12/21/2023, 00:50:27', '6', '0', '1.0']
['12/21/2023, 00:50:29', '2', '0', '1.0']
['12/21/2023, 00:50:31', '4', '0', '1.0']
```

```
phani@phani:~/DDOS$ sudo python3 topo.py
[sudo] password for phani:
Connecting to remote controller at 127.0.0.1:6653
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Adding switches:
s1
*** Adding links:
(5.00Mbit) (5.00Mbit) (h1, s1) (5.00Mbit) (5.00Mbit) (h2, s1) (5.00Mbit) (5.00Mbit) (h3, s1) (5.00Mbit)
(5.00Mbit) (h4, s1) (5.00Mbit) (5.00Mbit) (h5, s1) (5.00Mbit) (5.00Mbit) (h6, s1) (5.00Mbit) (5.00Mbit)
(h7, s1) (5.00Mbit) (5.00Mbit) (h8, s1) (5.00Mbit) (5.00Mbit) (h9, s1) (5.00Mbit) (5.00Mbit) (h10, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Starting controller
c1
*** Starting 1 switches
s1 ...(5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit)
 (5.00Mbit)
Generating NORMAL Traffic.......
```

Figure 4: Data extraction from network simulator

RawData is extracted using the mininet network simulator, the unprocessed data which is extracted is then processed using the feature extractor which extracts the necessary attributes for the analysis and training of machine learning models.

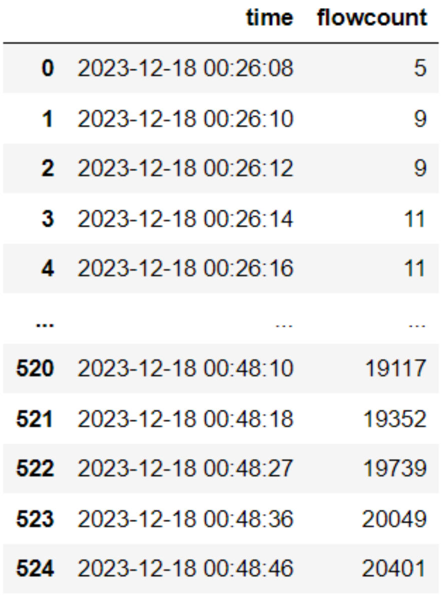
6.2 Experiment 2: Time series analysis for the DDOS attack trigger detection

The main objective of this experiment is to identify the starting point or the trigger point of the DDOS attack based on the dataset which consists of time and flowcount extracted from the SDN controller.

Python environment was established by installing packages specifically designed for data visualization and machine learning. 1. Initializes Plotly for notebook plotting. 2. Imports essential libraries for data manipulation, including NumPy and Pandas. 3. Imports libraries for data visualization, such as Matplotlib and Plotly. 4. Imports libraries for preprocessing, including Scikit-learn and TensorFlow. 5. Suppresses warnings. 6. Installs additional packages. 7. Imports modules for working with DICOM files and visualizing images.

The csv file is then parsed, with the time column being transformed into datetime format and the flowcount column being converted into a numeric number. Next, all the NaN values are eliminated in the flowcount column.

Subsequently, Exploratory Data Analysis (EDA) is done and the dataframe is preprocessed to precisely identify columns containing numeric and categorical data categories.



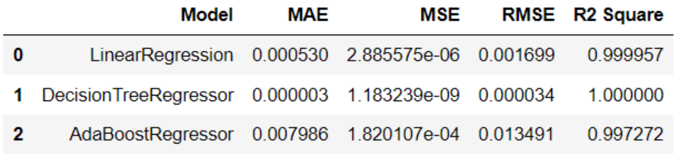
	time	flowcount
0	2023-12-18 00:26:08	5
1	2023-12-18 00:26:10	9
2	2023-12-18 00:26:12	9
3	2023-12-18 00:26:14	11
4	2023-12-18 00:26:16	11
...	...	...
520	2023-12-18 00:48:10	19117
521	2023-12-18 00:48:18	19352
522	2023-12-18 00:48:27	19739
523	2023-12-18 00:48:36	20049
524	2023-12-18 00:48:46	20401

Figure 5: Dataset preview for the time series analysis

The dataset has two columns, namely ”time” and ”flowcount.” Every row in the data contains an item with a timestamp, indicating the number of network flows recorded at that same instant. To normalize the numbers in the flowcount column and restrict them to a range of 0 to 1, the next step is to implement min-max scaling. The data is then divided into training and testing sets, allocating 70 percent for training and 30 percent for testing. Preprocessing is often used prior to training machine learning models to guarantee constant and suitable input scale.

The last stage involves assessing the regression models in order to record and display evaluation metrics, as well as generate plots comparing the actual values with the expected values. The DataFrame results-df is used to aggregate the performance data of several models.

The next phase involves applying the Decision Tree Regressor and AdaBoost Regressor algorithms to the standardized training data. The performance of these models will be evaluated using the ML-model function.



	Model	MAE	MSE	RMSE	R2 Square
0	LinearRegression	0.000530	2.885575e-06	0.001699	0.999957
1	DecisionTreeRegressor	0.000003	1.183239e-09	0.000034	1.000000
2	AdaBoostRegressor	0.007986	1.820107e-04	0.013491	0.997272

Figure 6: Performance metrics of time series analysis

Linear Regression is renowned for its simplicity and interpretability. The model exhibits outstanding accuracy in this scenario, as seen by the very low values observed for all measures. The Mean Absolute Error (MAE) reaches a minimum value of 0.000530, suggesting that, on average, the forecasts are very accurate and very near to the actual values. Similarly, the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) exhibit minimal values, highlighting the model's high level of accuracy. The R2 Square score of 0.999957 indicates that the linear regression model effectively explains almost all of the variability in the data, demonstrating its exceptional performance.

The Decision Tree Regressor, renowned for its capacity to capture intricate connections, demonstrates exceptional performance with very negligible values for MAE, MSE, and RMSE. The numbers indicate a high degree of alignment between the model's predictions and the actual data. The R2 Square score of 1.000000 indicates a flawless alignment with the data. The Decision Tree Regressor is very proficient in capturing complex patterns in the dataset, leading to exceptional performance.

The AdaBoost Regressor is a machine learning technique that use ensemble learning to merge numerous weak learners in order to generate a powerful learner. The AdaBoost Regressor has strong performance in this scenario, with somewhat higher error levels in comparison to the Decision Tree Regressor. The MAE, MSE, and RMSE values remain reasonably low, suggesting a high level of prediction accuracy. The R2 Square score of 0.997272 demonstrates the model's strong capacity to explain a large percentage of the variability in the target variable, indicating its reliable performance.

After the evaluation, hyperparameter tweaking is done, it is performed for a Random Forest Regressor. The goal is to identify the set of hyperparameters that minimizes the mean squared error on the training dataset. This procedure improves the precision and generalizability of the model, resulting in superior performance when applied to novel, unobserved data.

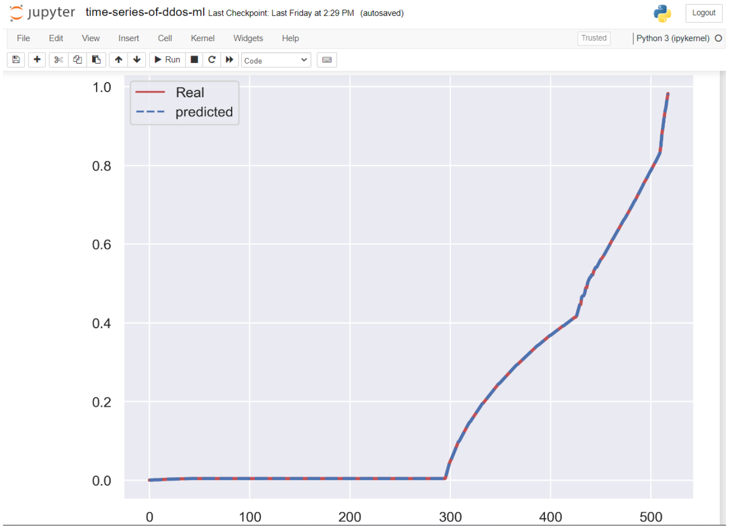
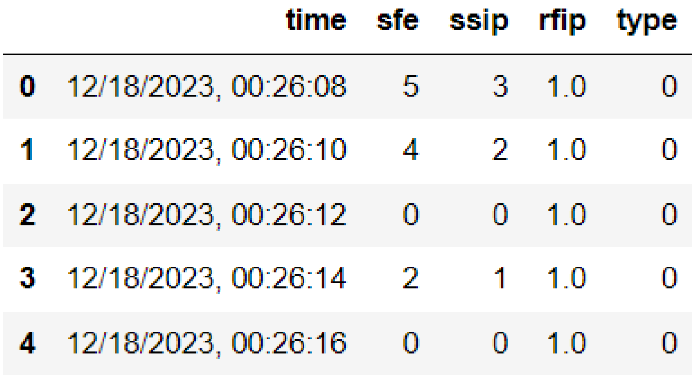


Figure 7: Decision tree regressor prediction.

During this time series analysis, the objective is to identify the first occurrence of the trigger for the DDOS assault. Once the DDOS assault is detected, it is categorized and it is determined if is a malicious attack or a normal traffic to the server.

6.3 Experiment 3: Classification model for DDOS attack identification

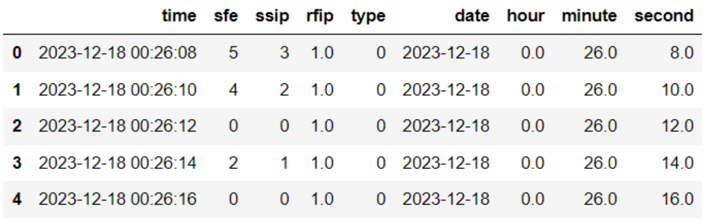
The main objective of this experiment is to create a classification model to differentiate between probable triggers, categorizing them as either DDOS assaults or basic network traffic.



	time	sfe	ssip	rfip	type
0	12/18/2023, 00:26:08	5	3	1.0	0
1	12/18/2023, 00:26:10	4	2	1.0	0
2	12/18/2023, 00:26:12	0	0	1.0	0
3	12/18/2023, 00:26:14	2	1	1.0	0
4	12/18/2023, 00:26:16	0	0	1.0	0

Figure 8: Raw dataset generated from SDN controller

Every row in the dataset represents a particular timestamp and contains the recorded values for flow entry, IP source speed, pair-flow entry ratio, and traffic type. After this step data transformation is done as required for the further analysis.



	time	sfe	ssip	rfip	type	date	hour	minute	second
0	2023-12-18 00:26:08	5	3	1.0	0	2023-12-18	0.0	26.0	8.0
1	2023-12-18 00:26:10	4	2	1.0	0	2023-12-18	0.0	26.0	10.0
2	2023-12-18 00:26:12	0	0	1.0	0	2023-12-18	0.0	26.0	12.0
3	2023-12-18 00:26:14	2	1	1.0	0	2023-12-18	0.0	26.0	14.0
4	2023-12-18 00:26:16	0	0	1.0	0	2023-12-18	0.0	26.0	16.0

Figure 9: Transformed dataset which is used for further analysis

Every row now has the original 'time' column as well as new columns that indicate the date, hour, minute, and second components. This kind of feature engineering enables more accurate analysis and visualization by using temporal patterns in the data.

Subsequently, the pre-processing stage involves eliminating unnecessary columns from the dataset and converting the data type of the relevant column for further analysis. The normalization is achieved using min-max scaling prior to the training of the machine learning model.

Now, it is necessary to divide the pre-processed data into separate sets for training and testing purposes. The training set is often used to instruct machine learning models, but the testing set is put aside for assessing the model's performance on unfamiliar data.

Following this, the crucial stage involves assessing and contrasting several machine learning models to categorize whether the suspected assault is a distributed denial-of-service (DDoS) attack or not. This process includes measures such as accuracy, precision, recall, F1 score, specificity, and confusion matrices. The outcomes are stored in CSV files, and Receiver Operating Characteristic (ROC) curves are produced for each category in the multi-class issue. This comprehensive assessment enables a detailed comparison of several models and their suitability for the assigned job.

It is necessary to construct classification models such as random forest, decision tree, logistic regression, gradient boosting, and Gaussian Naive Bayes. Subsequently, the models are stacked using the Stacking Classifier to construct a more resilient ensemble model.

The confusion matrix's rows correspond to the actual class labels, while its columns correspond to the predicted class labels. The elements on the diagonal of the confusion matrix indicate the count of correctly classified data points. The off-diagonal elements of the confusion matrix indicate the count of data points that have been categorized incorrectly.

Within the confusion matrix in Figure 10 of the random forest classifier, the values along the diagonal are significantly high, indicating that the classifier effectively assigns the majority of data points to their right classes. The off-diagonal elements have minimal values, indicating that the classifier has a very low error rate.

As shown in Evaluation metrics in Figure 11, the DecisionTreeClassifier and GaussianNB algorithms demonstrated flawless performance on both the test and training datasets. Nevertheless, the DecisionTreeClassifier exhibited indications of possible overfitting, since it attained flawless outcomes on the training set.

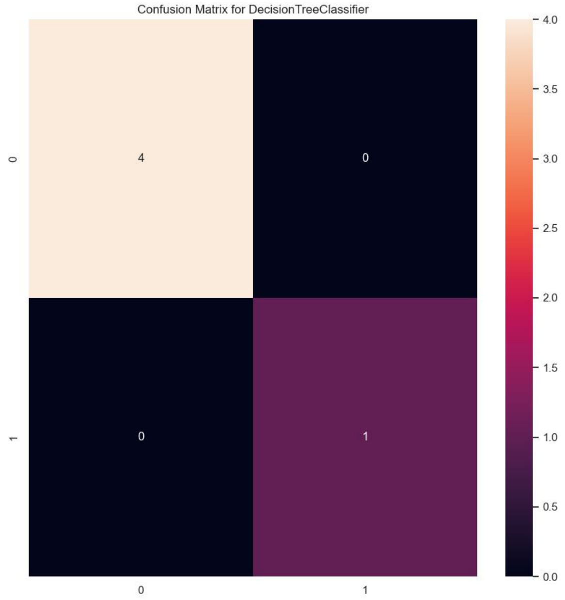
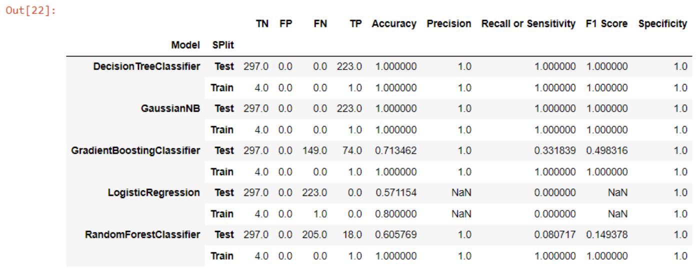


Figure 10: Confusion matrix of random forest classifier



Out[22]:

		TN	FP	FN	TP	Accuracy	Precision	Recall or Sensitivity	F1 Score	Specificity
Model	SPlit									
DecisionTreeClassifier	Test	297.0	0.0	0.0	223.0	1.000000	1.0	1.000000	1.000000	1.0
	Train	4.0	0.0	0.0	1.0	1.000000	1.0	1.000000	1.000000	1.0
GaussianNB	Test	297.0	0.0	0.0	223.0	1.000000	1.0	1.000000	1.000000	1.0
	Train	4.0	0.0	0.0	1.0	1.000000	1.0	1.000000	1.000000	1.0
GradientBoostingClassifier	Test	297.0	0.0	149.0	74.0	0.713462	1.0	0.331839	0.498316	1.0
	Train	4.0	0.0	0.0	1.0	1.000000	1.0	1.000000	1.000000	1.0
LogisticRegression	Test	297.0	0.0	223.0	0.0	0.571154	NaN	0.000000	NaN	1.0
	Train	4.0	0.0	1.0	0.0	0.800000	NaN	0.000000	NaN	1.0
RandomForestClassifier	Test	297.0	0.0	205.0	18.0	0.605769	1.0	0.080717	0.149378	1.0
	Train	4.0	0.0	0.0	1.0	1.000000	1.0	1.000000	1.000000	1.0

Figure 11: Evaluation metrics of different machine learning models.

The GradientBoostingClassifier achieved satisfactory results on the test set but demonstrated remarkable performance on the training set. This suggests the presence of overfitting.

LogisticRegression encountered difficulties on both the test and training sets, namely because of the presence of zero values in some measures. It may not be the optimal selection for this dataset.

The RandomForestClassifier algorithm demonstrated favorable performance on the test set, effectively balancing accuracy and recall. Additionally, the model had excellent results on the training set, perhaps indicating overfitting.

6.4 Discussion

The experiments carried out in this research provide useful insights into the detection and classification of possible Distributed Denial of Service (DDOS) attacks in Software-Defined Networking (SDN) networks. Experiment 1 included the collection of raw data from the SDN controller. From this data, important characteristics were selected and converted to build a well-organized dataset. Experiment 2 included a rigorous examination of the network traffic data via a detailed time series analysis, which revealed both periodic patterns and anomalies. The objective of this investigation was to discover possible catalysts for DDOS assaults. In Experiment 3, several classification models were assessed to determine their effectiveness in identifying possible causes and classifying them as either DDOS attacks or safe traffic. Significantly, the DecisionTreeClassifier and GaussianNB algorithms delivered flawless outcomes, but the GradientBoostingClassifier algorithm demonstrated satisfactory performance. However, the RandomForestClassifier proved to be an impressive competitor, attaining a well-balanced accuracy and recall on the test set. The use of stacking strategies significantly improved the resilience of the models. Overall, RandomForestClassifier and GaussianNB are notable options that effectively detect and address possible DDOS attacks in SDN setups. The thorough assessment and comparative examination provide the foundation for a complete strategy towards network security in SDN-based systems.

7 Conclusion and Future Work

Conclusion:

Ultimately, this study has presented a novel approach for identifying and minimizing Distributed Denial of Service (DDoS) assaults in Software-Defined Networks (SDNs). The methodology entails using statistical analysis, machine learning techniques, and time series analysis to accurately differentiate between regular and malicious network data. Research carried out in a regulated Software-Defined Networking (SDN) environment revealed the strength and exceptional precision of the suggested approach. The Feature Extractor effectively collected essential attributes, and the incorporation of machine learning models, including KNN, Gaussian Naïve Bayes, and ensemble approaches such as Adaboost and Gradient Boosting, demonstrated the system's capacity to adjust to changing attack patterns. Incorporating time series analysis improved the detection capabilities of the approach by providing a better knowledge of the temporal components of DDoS assaults. The suggested technique has shown a remarkable accuracy of 99.80 percent and a detection rate of 100 percent, highlighting its efficiency as a dependable protection mechanism against DDoS attacks in SDN systems. The approach demonstrated its scalability and flexibility by consistently achieving good performance, even in situations with diverse network complexity.

Future work:

Although the current study is a notable advancement in SDN-based DDoS detection, there are still several opportunities for further work and enhancement. Prioritizing the validation of the methodology's practicality, it is essential to deploy it in bigger and more diversified network settings. This entails conducting comprehensive testing of the system against a wider spectrum of DDoS attack categories and magnitudes to ascertain its efficacy across diverse scenarios. Moreover, it is advisable to investigate the incorporation of supplementary machine learning algorithms and ensemble approaches to strengthen the methodology's ability to withstand advanced assaults. The study concentrated on certain algorithms, and the incorporation of developing methodologies might enhance the comprehensiveness and precision of the detection system. Furthermore, it is crucial to consistently evaluate and update the methods in order to adjust to the ever-changing cyber threats. The threat environment is dynamic, necessitating a system that can quickly adapt and counter emerging attack patterns.

References

Alzahrani, R. J. and Alzahrani, A. (2021). Security analysis of ddos attacks using machine learning algorithms in networks traffic, *Electronics* **10**(23).
URL: *https://www.mdpi.com/2079-9292/10/23/2919*

Asim, S. U., Mahmood, Z., Ali, N., Ahmad, T. and Buriro, A. (2023). Machine learning-based dynamic attribute selection technique for ddos attack classification in iot networks, *Computers* **12**.

Cvitić, I., Perakovic, D., Gupta, B. B. and Choo, K.-K. R. (2022). Boosting-based ddos detection in internet of things systems, *IEEE Internet of Things Journal* **9**(3): 2109–2123.

Farukee, M. B., Shabit, M. S. Z., Haque, M. R. and Sattar, A. H. M. S. (2021). Ddos attack detection in iot networks using deep learning models combined with random forest as feature selector, *in* M. Anbar, N. Abdullah and S. Manickam (eds), *Advances in Cyber Security*, Springer Singapore, Singapore, pp. 118–134.

Gupta, B. B. and Badve, O. P. (2016). Taxonomy of dos and ddos attacks and desirable defense mechanism in a cloud computing environment, *Neural Computing and Applications* **28**(12): 3655–3682.

Lonea, A. M. (2013). Detecting ddos attacks in cloud computing environment.
URL: *https://www.univagora.ro/jour/index.php/ijccc/article/view/170*

Najar, A. A. and Manohar Naik, S. (2022). Ddos attack detection using mlp and random forest algorithms, *International Journal of Information Technology* **14**(5): 2317–2327.

Santos, R., Souza, D., Santo, W., Ribeiro, A. and Moreno, E. (2019). Machine learning algorithms to detect ddos attacks in sdn, *Concurrency and Computation: Practice and Experience* **32**(16).

Somani, G., Gaur, M. S., Sanghi, D. and Conti, M. (2016). Ddos attacks in cloud computing: Collateral damage to non-targets, *Computer Networks* **109**: 157–171. Traffic and Performance in the Big Data Era.
URL: *https://www.sciencedirect.com/science/article/pii/S1389128616300901*

Somani, G., Gaur, M. S., Sanghi, D., Conti, M. and Buyya, R. (2017). Ddos attacks in cloud computing: Issues, taxonomy, and future directions, *Computer Communications* **107**: 30–48.
URL: *https://www.sciencedirect.com/science/article/pii/S0140366417303791*

Sutton, C. D. (2005). 11 - classification and regression trees, bagging, and boosting, *in* C. Rao, E. Wegman and J. Solka (eds), *Data Mining and Data Visualization*, Vol. 24 of *Handbook of Statistics*, Elsevier, pp. 303–329.
URL: *https://www.sciencedirect.com/science/article/pii/S0169716104240111*

Swami, R., Dave, M. and Ranga, V. (2019). Software-defined networking-based ddos defense mechanisms, *ACM Comput. Surv.* **52**(2).
URL: *https://doi.org/10.1145/3301614*

Wani, A. R., Rana, Q. P., Saxena, U. and Pandey, N. (2019). Analysis and detection of ddos attacks on cloud computing environment using machine learning techniques, *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pp. 870–875.

Xia, W., Wen, Y., Foh, C. H., Niyato, D. and Xie, H. (2015). A survey on software-defined networking, *IEEE Communications Surveys Tutorials* **17**(1): 27–51.

Yan, Q., Yu, F. R., Gong, Q. and Li, J. (2016). Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges, *IEEE Communications Surveys Tutorials* **18**(1): 602–622.