Collaborative approach of Detection of DDOS attack on SDN Networks

MSc Research Project
Cybersecurity

Sankaran Selvam
Student ID: X21217467

School of Computing
National College of Ireland

Supervisor: Evgeniia Jayasekera

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	SANKARAN SELVAM		
Student ID:	X21217467		
Programme:	MSCCYB1	**Year:**	2022-2023
Module:	Research Project		
Supervisor:	Evgeniia Jayasekera		
Submission Due Date:	14/08/2023		
Project Title:	Collaborative approach of Detection of DDOS attack on SDN Networks		
Word Count:	5778		

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: Sankaran Selvam

Date: 13/8/2023

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):	

Collaborative approach of Detection of DDOS attack on SDN Networks

Sankaran Selvam
X21217467

Abstract

The project proposes a collaborative Network Anomaly Detection System using bleeding edge technologies like Software Defined Networking with the cutting-edge system like Snort Intrusion Detection System and Machine Learning. Our objective is to develop a strong and clever system that can recognize and counter potential security risks, particularly Distributed Denial of Service assaults on a network. The integration of the smart SDN controller with the Snort IDS forms the basis of our solution. We allow real time packet analysis and deep network traffic inspection by seamlessly combining these components, giving our system the ability to proactively detect malicious traffic patterns and potential threats. The strategy makes use of capabilities of the SDN controller that dynamically reroute malicious traffic to the Snort IDS for in-depth analysis. When examining the traffic and producing alerts for any unusual behaviour, the Snort IDS, known for its effectiveness and versatility in detecting network intrusions, is crucial. We have created a complex Decision Tree based Machine Learning model to improve the accuracy of anomaly detection. With the help of this ML model, which was trained using historical flow information, our system is able to distinguish between safe and unsafe traffic with astounding accuracy. SDN, Snort IDS and ML model are combined to provide a comprehensive network management system that intelligently responds to ever-changing threats. The proposed system has a capability to effectively distinguishing between legitimate traffic and attack traffic, enabling proactive response and protecting the network infrastructure from potential threats.

The project enables to improve network performance and security by combining SDN, Snort IDS and Machine Learning. The main aims are to protect the dependability and integrity of important network infrastructures from current and future cybersecurity threats by offering an innovative, scalable and adaptable solution.

Keywords: Software Defined Networking (SDN), Distributed Denial of Service (DDOS), Snort, Machine Learning, Network Traffic.

1 Introduction

Network administration has been transformed by Software Defined Networking decouple architecture , which takes a software centric approach to network control. This research study intends to thoroughly examine SDN's complexities while illuminating its significance, operational capabilities and benefits over traditional networking techniques. Network administrators now have better speed, flexibility and control over network traffic because of SDN's network control innovative capability.

Software defined networking and conventional network topologies are two fundamentally different ways to manage a network. In traditional network configurations, each networking device, such as switches and routers, has a close connection between the control plane and data plane. Inferring that each device decides on its own how to manage and route incoming communications, human setups are required, making network management a time consuming and labour-intensive operation. Instead, by dividing the control plane from the data plane, SDN decouple architecture is paradigm shift. The control plane in SDN is centralized and independent of the actual network hardware as shown in the below image (Figure 1). A centralized software entity known as the SDN controller serves as the network's central nervous system and interacts with network devices in the data plane via a southbound interface. Network provisioning and configuration modifications are made easier thanks to this division, which enables network managers to see and operate the whole network from a single control point.

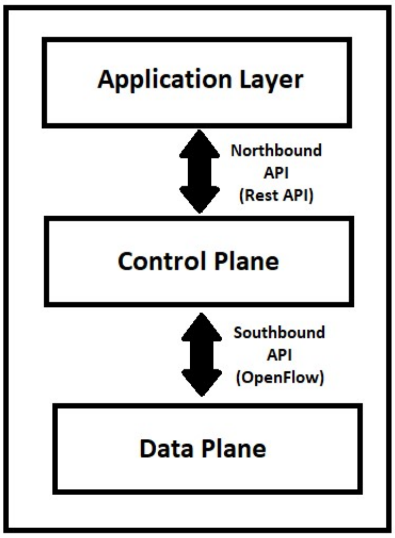


Figure 1 SDN Architecture

Let’s break down the SDN architecture into layers (Jenifa, 2023):

Application Layer: The top most layer where different applications are housed like virtual network overlays, security guidelines and traffic engineering tools that specify the desired network behaviour via the Northbound interface, they talk to the underlying SDN infrastructure.

Control Layer: The brain of the SDN is often a central controller that communicates with the network hardware in the Data Plane. The control layer configures and manages the behaviour of these devices through the Southbound interface to make sure that they abide by the network policies established by the applications.

Data Plane: It is also known as the Infrastructure Layer which comprises the physical network components like switches and routers that delivers network traffic based on the directives received from the Control Layer.

1.1 Motivation

The key capability of SDN is the ability to create network services and allocate virtual resources in real time via a centralized controller. This crucial feature helps administrators to enhance network speed and resource utilization, prioritize vital applications and optimize data flow. Additionally, the significant network visibility offered by SDN offers a thorough perspective of security flaws, enabling proactive mitigation efforts. To stop infections from spreading throughout the network, operators can quickly isolate infected devices. They can also create secure zones for devices with varied levels of security (VMware, 2021). However, due to its centralized architecture it is likely to be highly vulnerable to cyber-attacks like DDos, and other network related attack (Kaur *et al.*, 2021). Furthermore, SDN network's control plane and data plane can be independently targeted to compromise the network.

1.2 Contribution

The major objective of this project is to construct a collaborative system that is capable of identifying, evaluating and addressing any DDos attacks in real time. This system aims to combine the dynamic nature of SDN, the deep inspection capabilities of the Snort IDS and the precision of a ML model to give a solution to network anomaly detection and traffic control. The technical design of the system will be covered in detail , with a focus on how SDN and Snort IDS combine to allow real time packet inspection and network traffic analysis. The implementation and analysis of a sophisticated machine learning model for anomaly detection will also demonstrate how historical flow statistics are used to distinguish between safe and malicious traffic. The project's adaptability and scalability will also be highlighted, as well as how it might be used in and altered for use in other network infrastructures. This research looks closely at the Denial-of-service attack detection in the network which helps in taking an action to prevent any damage to the network. The project offers a strategy for reducing cyber risks and ensuring the smooth operation of crucial network infrastructures in a society that is becoming more linked. Proactive security measures and adaptive network management are heavily emphasized in this research.

2 Related Work

An emerging field termed Software-Defined Networking proposes is a network architecture that suits modern technology, the architecture decouples the control plane from the data plane to provide a centralized management capability. However, SDN is centralized and the controller oversees the entire network, it is extremely vulnerable to Distributed Denial of Service assaults (Kaur *et al.*, 2021). This literature review accentuates contemporary research endeavours that present an array of detection methodologies. The spotlight is directed towards defence mechanisms adept at recognizing DDoS attacks within SDN contexts. These strategic defences encompass an assortment of methodologies ranging from conventional approaches to cutting-edge machine learning-based techniques.

RYU Framework for SDN testbed:

The RYU framework is widely embraced by researchers for the researches related to SDN networks. It is primarily attributed as an open-source SDN controller that not only manages traffic routing but also helps in optimizing the utilization of network resources (Bhardwaj and Panda, 2022). The RYU controller demonstrates its utility in the assessment of crucial parameters within SDN architectures parameters like bandwidth, throughput, round-trip time, jitter, and packet loss (Islam, Islam and Refat, 2020). A remarkable attribute of the RYU framework is its scalability in presence of dynamic circumstances and for observing throughput of the controller and checking its performance in the dynamic networking conditions (Asadollahi, Goswami and Sameer, 2018). The main reason that researchers prefer using Ryu controller is because it has rich functionality, adaptability and cost effectiveness. It also has a function to visually represent the network using GUI functionality (Ha, Quan and Nguyen, 2022).

Comprehensive analysis of Machine Learning Techniques used for DDOS detection:

In recent years, the application of machine learning techniques in detecting network anomalies within SDN networks has gained considerable attention of the researchers. It is interesting how machine learning changed the threat detection, the Systematic Review was performed to analyse the machine learning and deep learning methods employed for identifying DDoS attacks in SDNs (Ali, Chong and Manickam, 2023). By delving into existing literature, the we identified gaps and research directions.

The research paper showcased bunch of ideas to detect the DDoS attacks in a software-defined networking environment through the application of diverse machine learning algorithms which resulted as effective techniques for DDoS detection, showcasing impressive accuracies of up to 99.993% in identifying malicious traffic (Prasad *et al.*, 2022). In another study, the author use four algorithms and analyse their performance on the CICDDoS2019 dataset, noted that Random Forest showcased 68.9% accuracy rate (Hamarshe, Ashqar and Hamarsheh, 2023). On searching further, we found another relevant research (Atul Sharma *et al.*, 2022), investigates the use of the Decision Tree machine learning technique for detecting DDoS attacks in SDN environments resulted in better accuracy in determine malicious traffics than other algorithms. We also noted that the datasets available online like CIC-DDoS2019, CNN-BI-LSTM and KDDCup99 which were used in the most of the study were outdated and it does cover the recent attack patterns resulted in less accuracy in the detection.

The machine learning techniques were widely utilized to effectively detect the DDoS attacks within SDN environments (Gupta and Grover, 2021). Even the classifiers like K-Nearest Neighbours (KNN) algorithm were used and achieves an impressive accuracy of 97% in detecting the attack (Madathi M *et al.*, 2022), with a proper feature selection method for detecting attack with a KNN classifier achieves an impressive accuracy rate of 98.3% in DDoS attack detection (Polat, Polat and Cetin, 2020). When they tested various machine learning techniques like Naïve Bayes, Decision Tree, k-nearest neighbours, Logistic Regression and Random Forest for detecting DDoS attacks in SDN network it seems that Random Forest and Decision Tree algorithms showcased best accuracy and decision rates results among the other (Ashodia and Makadiya, 2022). Also, we found that the use of Decision Tree and SVM machine learning techniques achieve better accuracy and detection rates in this context (Sudar *et al.*, 2021). The server study has bolstered that Decision tree and Random Forest algorithm were found to perform the best among the other algorithms and it accurately and quickly detecting attacks with minimal false alarms (Khashab *et al.*, 2021). Additionally other study also suggest that the detection accuracy of the Random forest algorithm was best in case if performance and accuracy (Santos *et al.*, 2019). On the other hand studies suggested SVM algorithm handle high-dimensional data and its a promising machine learning technique for robust classification performance for enhancing network security in SDN environment (Li *et al.*, 2018). The effectiveness of SVM in identifying and mitigating DDoS attacks in SDN environments is promising for detecting flood attacks (Kokila, Thamarai Selvi and Govindarajan, 2014). We found another interesting study showing that the use of deep learning models achieves remarkable accuracy above 99% in classifying unseen traffic in a simulated environment to detect transport and application layer DDoS attacks (Yungaicela-Naula, Vargas-Rosales and Perez-Diaz, 2021). and when compared to the best SVM machine learning algorithms with the Deep Neural Network it outperforms the Support Vector Machine in terms of accuracy (B. V., D. G. and S, 2018). Finally we found the article (Yadav *et al.*, 2021) and (Deepa, Muthamil and Deepalakshmi,

2018) which strongly suggest that hybrid approach of combining the two-machine learning model achieves a higher accuracy in detecting attacks in both the control and data planes of SDN while minimizing overhead.

The below table (Table 1) showcase the related research works

Table 1 Literature Review Overview

Title	Authors	Year	Abstract Summary	Gap
SDN-Based Architecture for Transport and Application Layer DDoS Attack Detection by Using Machine and Deep Learning	N. M. Yungaicela-Naula, C. Vargas-Rosales, J. A. Pérez-Díaz	2021	The research shows that using machine learning and deep learning models, on an SDN based architecture to identify DDoS attacks at the transport and application layer. In a simulated environment, it can classify unseen traffic with an accuracy of more than 99%.	However, it needs further research to determine the suggested SDN-based architecture's scalability and practical implementation.
DDoS Attack Detection and Mitigation in SDN using Machine Learning	Fatima Khashab, Joanna Moubarak, Antoine Feghali, C. Bassil	2021	A machine learning based model for DDoS attack detection and mitigation in SDN networks is proposed and explained in this paper. It compares the performance of six machine learning algorithms and concluded that Random Forest performs the best, reliably and effectively identifies assaults with a minimal likelihood of disrupting regular traffic.	It is important to look into the computational resources needed to apply the model on a sizable SDN network.

Detection of DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models	Hüseyin Polat, O. Polat, Aydın Çetin	2020	It examines DDoS attacks in Software-Defined Networks and uses machine learning based solution and feature selection methods for detection. The wrapper feature selection with a KNN classifier achieves the highest accuracy rate of 98.3% in DDoS attack detection.	The impact of different feature selection methods on detection performance should be further explored.
Machine learning algorithms to detect DDoS attacks in SDN	R. Santos, D. Silva, Walter E. Santo, Admilson R. L. Ribeiro, E. Moreno	2020	This paper investigates the implementation of four machine learning algorithms to classify DDoS attacks in an SDN simulated environment. It identifies Random Forest as the best-performing algorithm in terms of accuracy.	The robustness of the proposed model to adversarial attacks and variations in network conditions requires further investigation.
Detection of DDoS Attacks in Software Defined Networking	Karan B. V., N. D. G., P. S. Hiremath	2018	This paper proposes a DDoS attack detection system for SDN using Snort for signature-based attacks and machine learning algorithms for anomaly-based attacks. Deep Neural Network outperforms Support Vector Machine in terms of accuracy.	A comprehensive analysis of the trade-offs and overhead associated with combining Snort and machine learning algorithms is needed.
Using SVM to Detect DDoS Attack in SDN Network	Dong Li, Chang Yu, Qizhao Zhou, Junqing Yu	2018	This paper proposes a new model to detect DDoS attacks in SDN based on Support Vector Machine (SVM). The model extracts key features from	Further the research should be conducted to ascertain whether the concept can be applied to various SDN network topologies and configurations.

			packet-in messages and uses SVM to detect the attacks, achieving high efficiency.	
DDoS detection and analysis in SDN-based environment using support vector machine classifier	R. Kokila, S. Thamarai Selvi, K. Govindarajan	2014	The paper mentions the usage of SVM classifier for DDoS analysis and detection in SDN setups is examined. Due to SVM's high accuracy and low false positive rate, DDoS detection in SDN networks may be effectively accomplished.	It is important to look for the potential difficulties of adopting SVM in a dynamic and real world SDN environment.

Table 1 Literature Review Overview

3 Research Methodology

3.1 Snort-Based Signature Detection:

Implement the Snort system as the first line of defence to detect the DDoS attack signatures based on the predefined rules. It also enables to configure Snort rules to recognize common attack patterns and analyse its performance in terms of detection rate and false positives. This enables the network administrator to get alert if any surge in traffic is identified by snort. These alerts can be configured to generate for any kind of network anomalies patterns that can possibly happen in to the network environment. The Snort inspects the traffic flow in the SDN switches and alerts the SDN controller which is listening to snort alerts via Unix domain socket. The below image (Figure 2) represents the snort integration with RYU Controller.

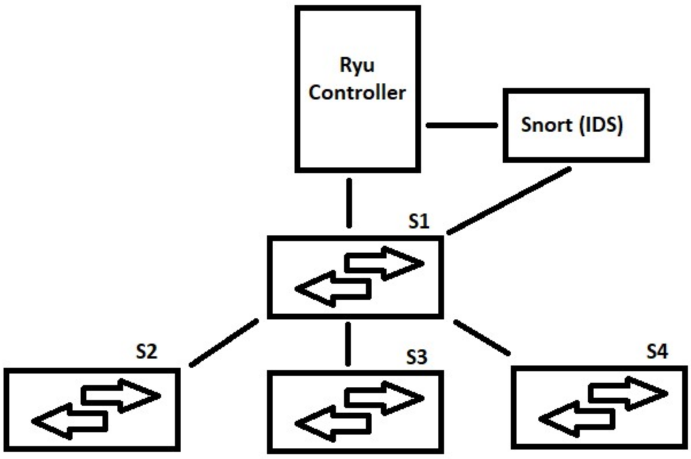


Figure 2 Snort SDN Controller

3.2 Machine Learning Model Selection:

The ML algorithms were used to identify the most suitable model for DDoS attack detection in SDN. The chosen model that demonstrates the best trade-off between accuracy, processing speed and resource utilization was used to integrate with SDN control for detection of network anomaly such as DDos attack. These models are trained is identify and differentiate the traffics flows inside the network. The SDN console will alert if it identifies any malicious patten in the network based on the trained dataset. The below image (Figure 3) shows the ML based RYU controller

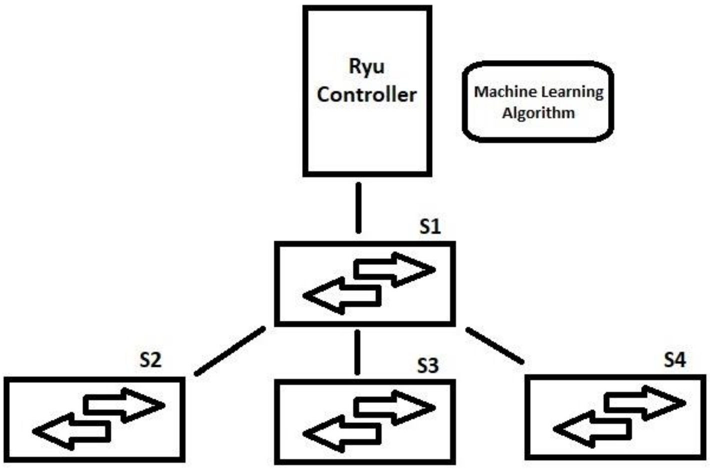


Figure 3 Machine Learning SDN Controller

3.3 Hybrid Detection System:

Finally, the hybrid model approach, where the SDN controller has a two line of defence where the first line of defence is the snort integration which classifies traffic based on signature detection. On the other hand, with the ML-based anomaly detection the SDN controller differentiate the traffic based on decision made by the ML module. This creates a hybrid detection system a mechanism to combine the outputs of both techniques to improve overall detection accuracy. The below image (Figure 4) shows the hybrid detection system.

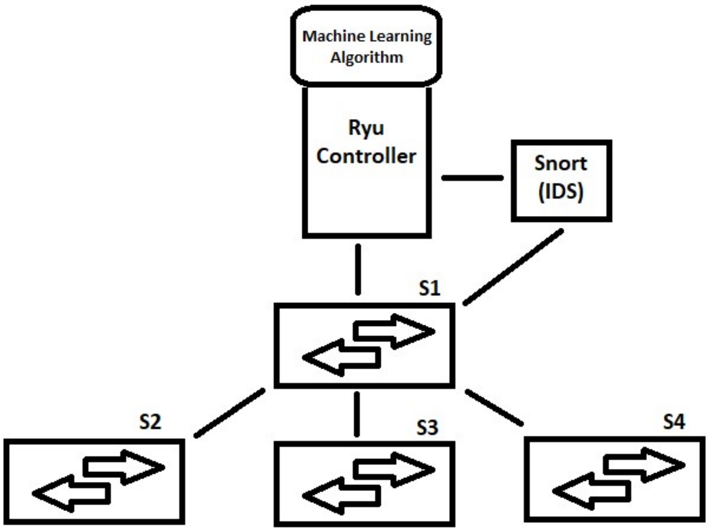


Figure 4 Hybrid Controller

4 Design Specification

The test bed setup was created using the virtual machines on a custom Laptop with Windows 11 which had Oracle VM Virtual Box installed.

Desktop Specification:

- Performance-oriented CPU: AMD Ryzen 7 5800H, stable clock at 3.20ghz
- 16 GB DDR4 ram.
- External GPU Nvidia 3060 with 6GB RAM.
- 2TB SSD storage

The experimental setup consists of two virtual machines with the below mentioned configuration

RYU Controller VM

- 1 Core Processor
- 4GB ram
- 30GB Storage
- OS: Ubuntu 64-bit

Mininet VM

- 2 Core Processor
- 4GB ram
- 30GB Storage
- OS: Ubuntu 64-bit

Software used:

- 64-bit Windows 11 operating system is running on the Host Machine.
- Python 3.10.5

- Mininet
- RYU SDN Controller
- SNORT

4.1 SDN Testbed experimental setup:

As represent in the below image (Figure 5) there are three main components which make up the SDN network testbed. The first component is comprised of Mininet which helps in creating simulate SDN network and the SDN controller, which filters malicious traffic using Machine learning Algorithms. Finally, the Snort IDS which detects network anomalies based on the rules defined on it.

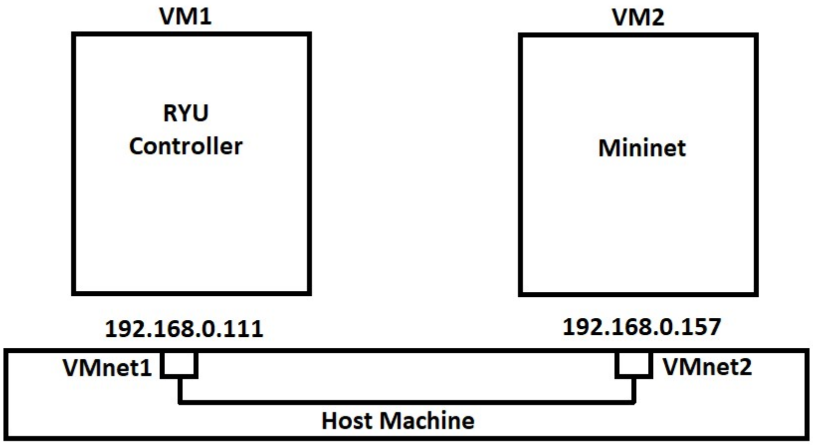


Figure 5 SDN Testbed experimental setup

This setup includes the integration of the SDN controller with the ML module and Snort IDS, which aids in monitoring and rerouting traffic based on its flow pattern. The ML logic assists in distinguishing between legitimate and DDoS traffic based on the trained dataset. On the other hand, Snort helps in monitoring events and sends the information to the SDN controller to make decisions. This hybrid approach effectively detects and proactively prevents any DDoS attacks.

5 Implementation

In this section, we provide a detailed description of the implementation process for the SDN controller which has an integrated Machine Learning module and Snort for Intrusion Detection System functionalities. The implementation procedure involves setting up the SDN test bed environment, data collection, ML model training, Snort integration, real-time traffic management and SDN controller deployment. The test bed consists of two virtual machines which has Ryu controller on one of the machines and Mininet for network simulation on the other machine as shown below (Figure 6).

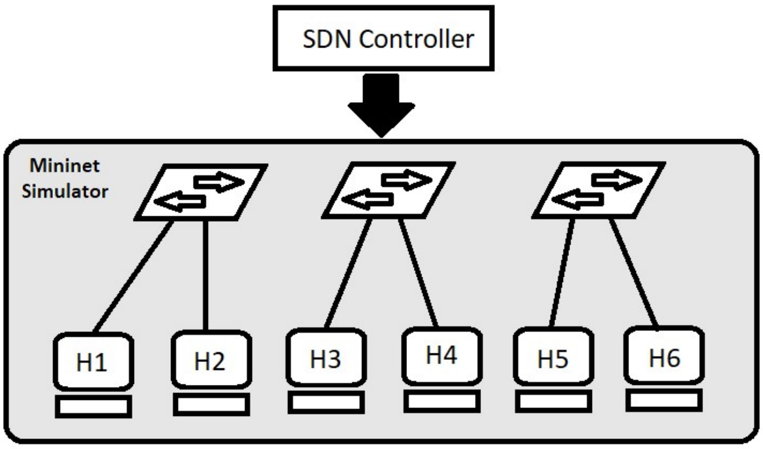


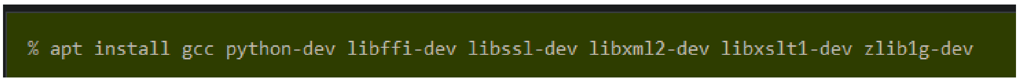
Figure 6 SDN Testbed Components

5.1 Development Environment Setup

In this first step of the implementation process the development environment was setup with two Virtual machines. The Ryu SDN controller framework was installed in one of the virtual machines and it's required some dependencies and python to run this controller. The Virtual machine has Ubuntu operating system.

In the Ryu virtual machines, the below prerequisite steps were performed to develop the Ryu controller machine.

1. The dependencies (Figure 7) were installed to resolve issues prior to installation of the Ryu controller.



```
% apt install gcc python-dev libffi-dev libssl-dev libxml2-dev libxslt1-dev zlib1g-dev
```

Figure 7 Dependencies

2. After resolving the dependencies, the Ryu controller was installed, there are several ways to install the Ryu Controller. We can use python pip module to install it or the source code is available on GitHub to install. We used pip module to install the Ryu controller.

3. The installation included sample switches for experimental purpose and it also supports modification and allows improvements. So, one of such script was used to improvise the controller to add the ML module and integrate snort with it.

4. For the integration purpose the snort application was installed in the same machine and configured to use for integration.

In the Mininet virtual machine, the following steps were performed to ensure the machine is configured for network simulation.

1. The Mininet tool was downloaded from GitHub.

2. The prerequisite components and dependencies were installed before the Mininet installation.

3. The Mininet was installed for network simulation purpose, it is a powerful tool which helps in creating a different network topology for testing the network controller.

Additionally, these two virtual machines were configured to communicate on the virtual network this allows the controller machine to serves as the brain of the simulated virtual network.

5.2 Data Collection and Preprocessing

Data collection is crucial and important step for training the ML model. We have gathered flow statistics from SDN-enabled switches using the OpenFlow protocol. These flow statistics, which include information such as IP addresses, transport protocols, packet counts and byte counts, etc were collected periodically from each switch and stored in a CSV file. Before training the ML model, we preprocess the flow statistics. This preprocessing involves feature selection, data cleaning to handle any missing values and converting IP addresses and ports into numerical representations suitable for ML algorithms. Table 2 represent the relevant attributes to be provide as input variable for the model.

In the data collection process, we have created a script to generate a legitimate traffics and malicious traffic

- In the script to generate which generates legitimate traffics such as file transfers, TCP request and reachability check using ping we have made the script to perform all legitimate network activities for a duration of time. These traffic data were collected and stored in the csv file for training the module.
- In the malicious traffic generation script, the malicious traffics like ICMP flooding, TCK Sync flood and other DDos related packet transfers were performed using hping3. These traffics were recorded in the CSV file.
- The feature was selected

Table 2 Features of the dataset

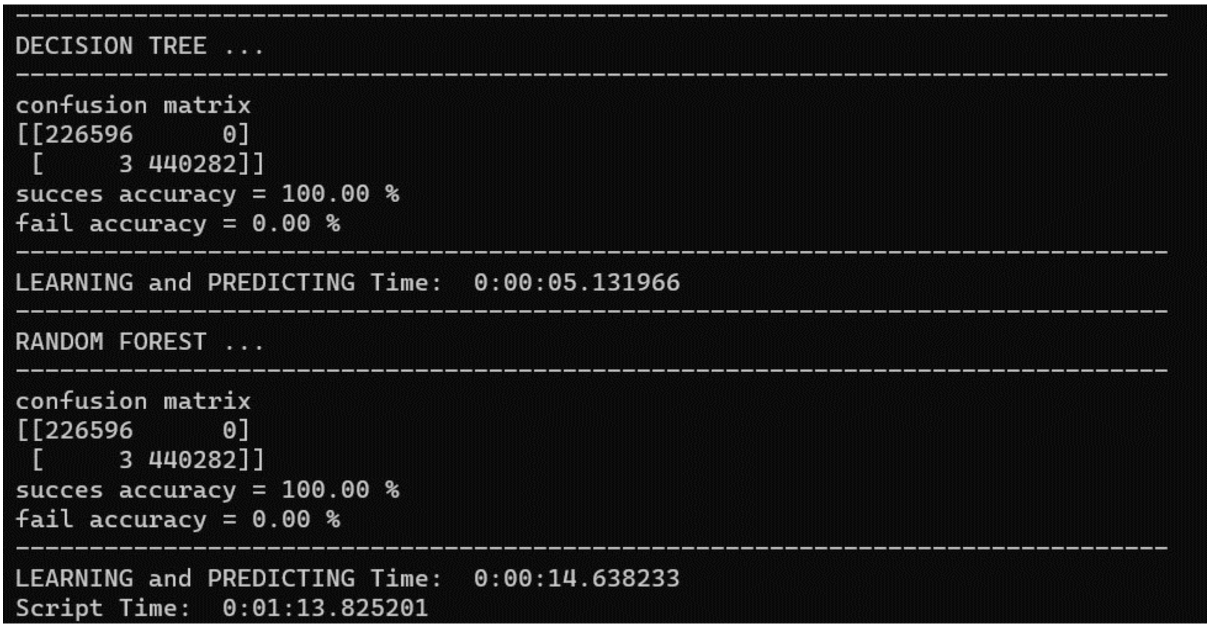
S.No	Features	Description
1	timestamp	Timestamp of the flow statistics
2	datapath_id	ID of the switch that observed the flow
3	flow_id	Unique identifier for the flow based on source IP, source port, destination IP, destination port and IP protocol
4	ip_src	Source IP address of the flow
5	tp_src	Source transport port (TCP/UDP) of the flow
6	ip_dst	Destination IP address of the flow
7	tp_dst	Destination transport port (TCP/UDP) of the flow
8	ip_proto	IP protocol number
9	icmp_code	ICMP code
10	icmp_type	ICMP type

11	flow_duration_sec	Duration of the flow in seconds
12	flow_duration_nsec	Duration of the flow in nanoseconds
13	idle_timeout	Idle timeout value for the flow
14	hard_timeout	Hard timeout value for the flow
15	flags	Flow flags
16	packet_count	Number of packets observed for the flow
17	byte_count	Number of bytes observed for the flow
18	packet_count_per_second	Packet count per second
19	packet_count_per_nsecond	Packet count per nanosecond
20	byte_count_per_second	Byte count per second
21	byte_count_per_nsecond	Byte count per nanosecond

After collecting this traffic data in the csv file, the data was pre-processed by cleansing the null values or corrupted values and changing the data format suitable for training the ML module.

5.3 ML Model Training

With the pre-processed data generated in the previous step, we proceeded to train the ML model. For this project, we have used classifiers like Linear regression, K-Nearest Neighbour, Naive Bayes, Decision Tree and Random Forest and choose a Decision Tree Classifier due to its accuracy and also based on the research work we noted that decision tree and random forest had higher accuracy and widely used for DDos detection in SDN network. Also, due to its simplicity and ability to handle both numerical and categorical data. The dataset is split into training and testing sets. The Decision Tree Classifier is trained on the training data using the fit method. The below image (Figure 8) shows the accuracy and performance of the models which were evaluated using the testing data.



```
------------------------------------------------------------------------------
DECISION TREE ...
------------------------------------------------------------------------------
confusion matrix
[[226596      0]
 [     3 440282]]
succes accuracy = 100.00 %
fail accuracy = 0.00 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:05.131966
------------------------------------------------------------------------------
RANDOM FOREST ...
------------------------------------------------------------------------------
confusion matrix
[[226596      0]
 [     3 440282]]
succes accuracy = 100.00 %
fail accuracy = 0.00 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:14.638233
Script Time:  0:01:13.825201
```

Figure 8 Machine Learning Modules Accuracy

We noted that the results show 66.02% accuracy using linear regression and 71.41% accuracy in Naive Bayes algorithm. Further noted K-Nearest Neighbour, Decision Tress and Random Forest has shown 100% accuracy

5.4 Snort Integration

In this step, we have integrated Snort IDS with the SDN controller, the Snort IDS generates alerts for potential security threats detected in the network and a function was implemented to listen for Snort alerts through a Unix domain socket. These alerts contain the information about the suspicious activities, such as type of attack and associated packet information. The SDN controller receives and processes these alerts, logging the alert messages and associated packet details.

5.5 Real-Time Traffic Management using ML

The core implementation lies in the real-time traffic management functionality. When a packet is received by the SDN controller, key features in the dataset such as source and destination addresses, transfer rate, duration and protocol are extracted. As shown in Figure 9 the ML model is used to predict if the traffic is legitimate or suspicious based on these extracted features. Depending on the ML prediction, appropriate actions are taken on the packets. Legitimate packets are forwarded to their destination, while suspicious packets may be dropped or further analysed by the Snort IDS for potential security threats.

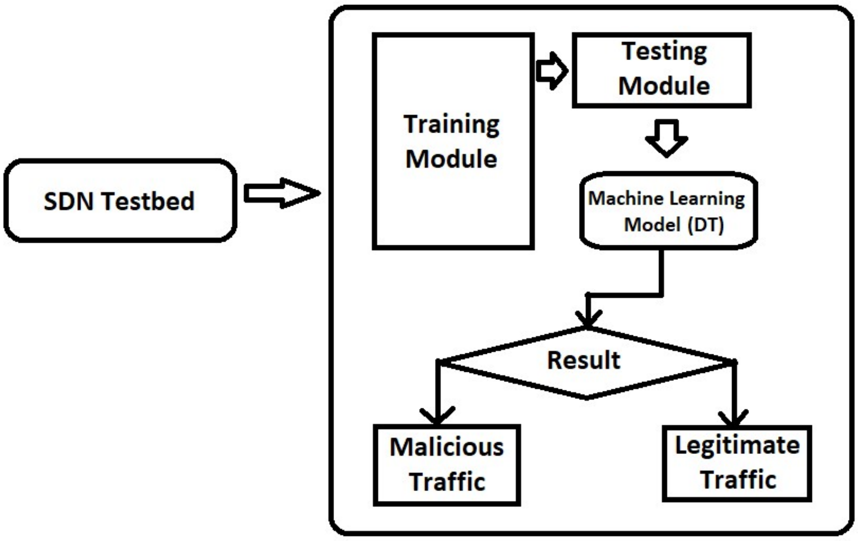


Figure 9 ML Module

5.6 SDN Controller Deployment

Once the hybrid model is implementation, we deploy the hybrid SDN controller on the network and configure the SDN switches to connect to the controller in OpenFlow mode. The SDN controller acts as the brain of the network, making real-time traffic decisions based on the integrated ML model and Snort IDS. Throughout the implementation process, the testing was conduct to ensure the proper functioning of the integrated system. Synthetic network traffic was generated to simulate various scenarios and test the ML-based traffic management system thoroughly. The controller response was monitored to validate the effectiveness of the integrated solution in detecting of the potential network anomalies and security threats.

6 Evaluation

The evaluation of the performance and effectiveness of the hybrid system that has the integration of Machine Learning and Snort Intrusion Detection System in to Software-Defined Networking controller is explained in detail in this section. The aim of the evaluation is to analyse effectiveness and accuracy of the detection of the hybrid solution on different network topology. We used the below topologies to evaluate our proposed system. (Figure 10-A and 10-B).

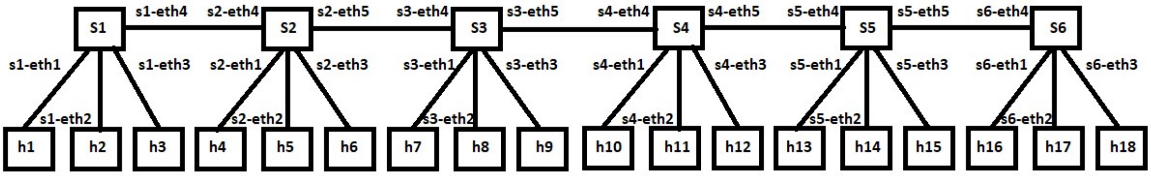


Figure 10-A Topology A

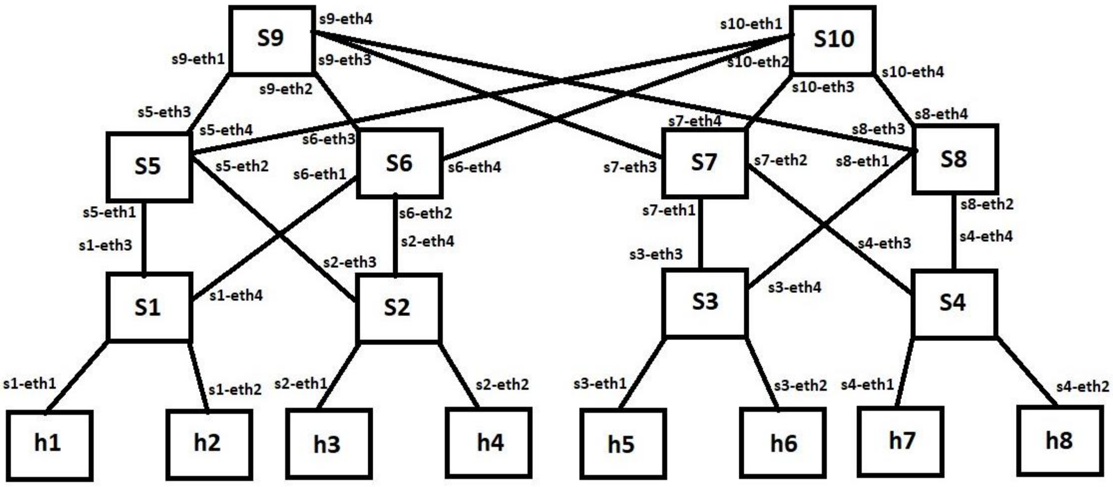


Figure 10-B Topology B

6.1 ML Model Accuracy Evaluation:

The ML model's accuracy was evaluation is explained in this section, we have used the synthetic dataset that we have generated with the standard environment. The ML model predicts the legitimacy of traffic based on dataset that we have generated with a single standard topology (Figure 10-A). The ML model predictions with the actual labels in the test dataset to compute metrics such as accuracy, precision, recall and F1-score. These metrics provide insights into the model's ability to correctly classify legitimate and suspicious traffic. A high accuracy and F1-score indicate a robust and reliable ML model.

Performed an hping3 attack in the Mininet network as shown in Figure 11

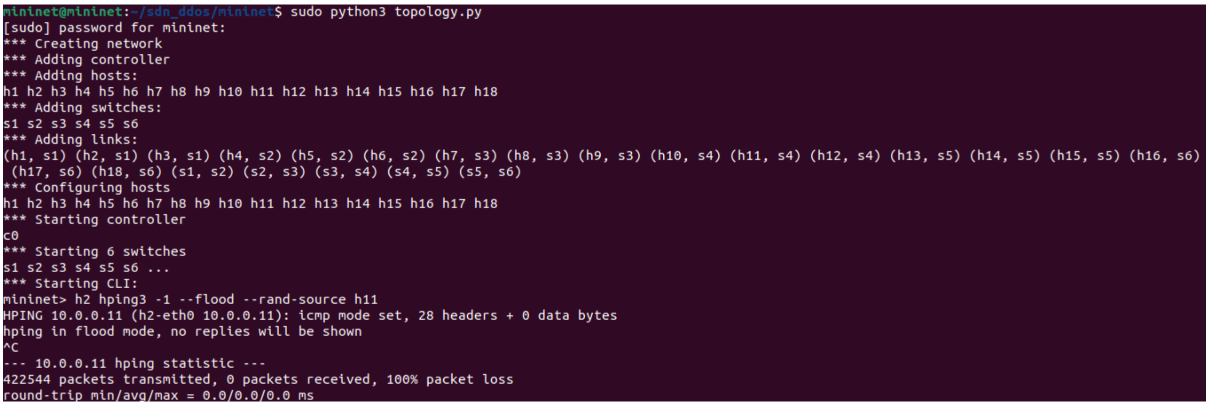
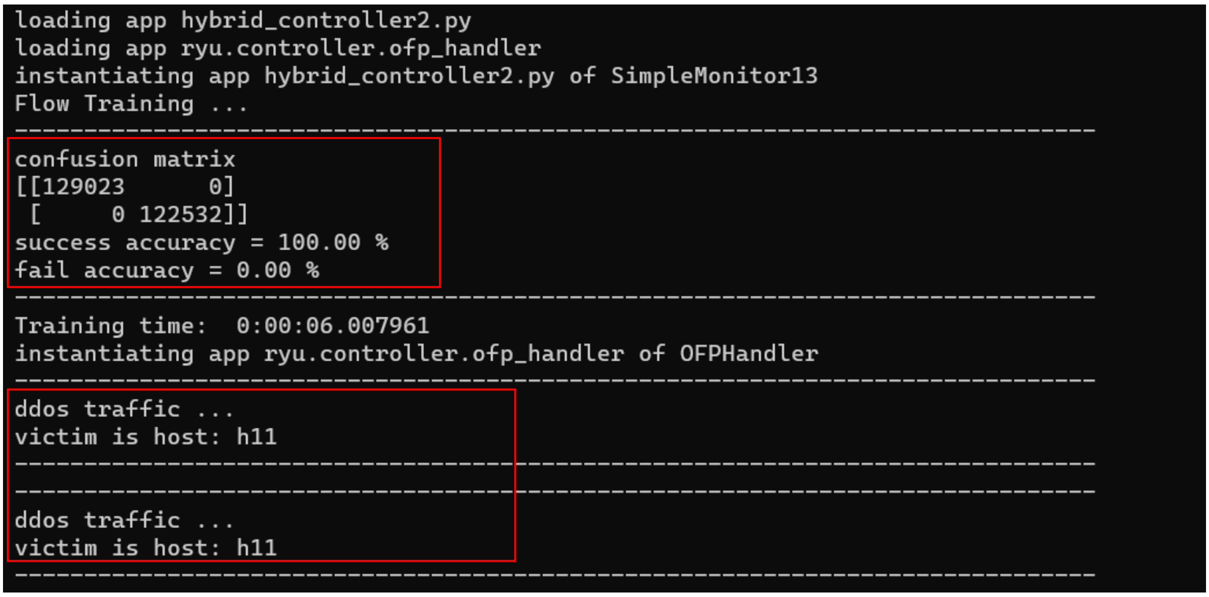

```
mininet@mininet:~/sdn_ddos/mininet$ sudo python3 topology.py
[sudo] password for mininet:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18
*** Adding switches:
s1 s2 s3 s4 s5 s6
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s2) (h5, s2) (h6, s2) (h7, s3) (h8, s3) (h9, s3) (h10, s4) (h11, s4) (h12, s4) (h13, s5) (h14, s5) (h15, s5) (h16, s6)
 (h17, s6) (h18, s6) (s1, s2) (s2, s3) (s3, s4) (s4, s5) (s5, s6)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18
*** Starting controller
c0
*** Starting 6 switches
s1 s2 s3 s4 s5 s6 ...
*** Starting CLI:
mininet> h2 hping3 -1 --flood --rand-source h11
HPING 10.0.0.11 (h2-eth0 10.0.0.11): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 10.0.0.11 hping statistic ---
422544 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Figure 11 Hping3 attack in Mininet

The Hybrid controller detected and created an alert message in the controller console as show in the Figure 12. Also, the accuracy of the model with the synthetic data is also shown in the console screen. However, we noted that the detection accuracy reduced when we changed the testing topology due to the limitation of data and the synthetic data that we created doesn’t cover all topological data.

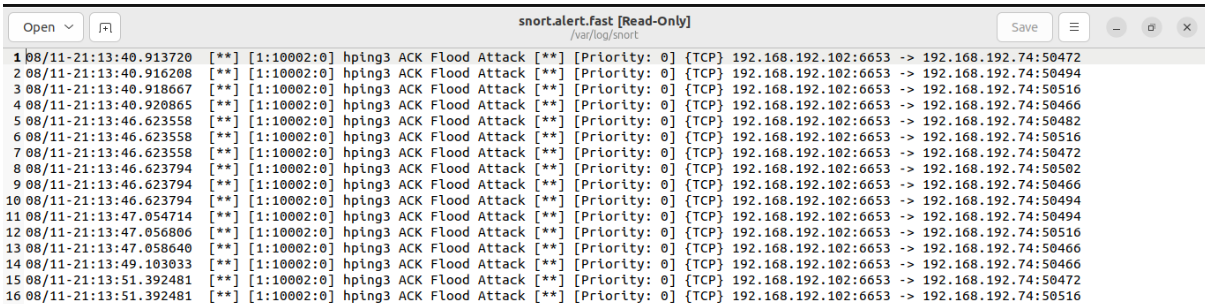


```
loading app hybrid_controller2.py
loading app ryu.controller.ofp_handler
instantiating app hybrid_controller2.py of SimpleMonitor13
Flow Training ...
-------------------------------------------------------------------------------
confusion matrix
[[129023      0]
 [     0 122532]]
success accuracy = 100.00 %
fail accuracy = 0.00 %
-------------------------------------------------------------------------------
Training time:  0:00:06.007961
instantiating app ryu.controller.ofp_handler of OFPHandler
-------------------------------------------------------------------------------
ddos traffic ...
victim is host: h11
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
ddos traffic ...
victim is host: h11
-------------------------------------------------------------------------------
```

Figure 12 Hybrid controller output

6.2 Snort IDS Performance Evaluation:

The Snort IDS is used to analyse the network anomaly based on its ability to detect various types of network attacks and intrusions. The attack traffic into the network that we have created made Snort to generate alert. The evaluation includes analysing the detection, time taken for alerts generation and false positive rate of the detection. The Snort IDS's performance is measured based on its effectiveness in identification. We noted that snort has effectively identified all the flood attacks that we performed and generated alerts even for different topologies as shown in Figure 13. Snort acted as a primary defence of our design and even when the ML model failed to detect during the dynamic environment.



```
08/11-21:13:40.913720  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50472
08/11-21:13:40.916208  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50494
08/11-21:13:40.918667  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50516
08/11-21:13:40.920865  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50466
08/11-21:13:46.623558  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50482
08/11-21:13:46.623558  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50516
08/11-21:13:46.623558  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50472
08/11-21:13:46.623794  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50502
08/11-21:13:46.623794  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50466
08/11-21:13:46.623794  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50494
08/11-21:13:47.054714  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50494
08/11-21:13:47.056806  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50516
08/11-21:13:47.058640  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50466
08/11-21:13:49.103033  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50466
08/11-21:13:51.392481  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50472
08/11-21:13:51.392481  [**] [1:10002:0] hping3 ACK Flood Attack [**] [Priority: 0] {TCP} 192.168.192.102:6653 -> 192.168.192.74:50516
```

Figure 13 Hybrid controller output

6.3 Discussion

Our proposed system has addressed several gaps identified in the literature review. The system's resilience against adversarial attacks during the testing phase and its adaptability to diverse network conditions require in-depth analysis to ensure robustness with no overheads. It also achieved a detection rate close to real-time detection. Note that this solution also provides layer of defence, one of the defences worked based on rules and the other learns from the network and improvise its defence based on the ML model. The effective detecting of DDos or Dos attacks with higher accuracy rate is achieved using this model but it requires further analysis and improvement in creating dataset that reflects the complete traffic data for different network structure. The limitation we noted in the dataset required huge amount of data and the dataset available online are outdated. Additionally, decision-based model is a continuously improving mechanism and the Snort is a traditional tool that has a huge predefined rules that was created based on the past cyber-attacks. However, there solution provides an effective and accurate detection results which helps in mitigating the threat proactively and take actions.

7 Conclusion and Future Work

In the evaluation, we identify that the proposed system has potential to detection the DDos attack with higher accuracy rate and effectiveness in detection is increased by the hybrid approach where both the defence mechanism complements each other in detection. When the network environment changes the model needs more data to classify the features and understand the traffic flow on the network. This affects the ML model efficiency in detection the traffic flow to some extent until the ML model prepares its knowledge base. The challenges implementation of the proposed system in real world is that it required more research work to improve the accuracy of the ML model or there might be better technique that can be used for detection and also the proposed system should be capable of handling threats in various situation and also defend network against future threat which is not event recorded in the knowledge data base of ML model. However, Snort is a powerful system which helps in providing protection to the network environment even in a dynamic scenario, being the first line of defence snort provides accurate detection result and alerts to the SDN controller even if the ML module is incapable of detecting the anomaly in some scenarios. Additionally, future enhancements of the hybrid solution's performance and security capabilities by creating an effective dataset which has all the recent DDos attack patterns and techniques. To conclude, the evaluation of the hybrid SDN with ML and Snort IDS solution provides an effectiveness result as an intelligent network security solution. With the hybrid solution in real-world network environments, it ensures defence in depth security feature and helping network administrators and security professionals make enhance network security and performance.

References

Ali, T.E., Chong, Y.-W. and Manickam, S. (2023) ‘Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review’, *Applied Sciences*, 13(5), p. 3183. Available at: https://doi.org/10.3390/app13053183.

Asadollahi, S., Goswami, B. and Sameer, M. (2018) ‘Ryu controller’s scalability experiment on software defined networks’, in *2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*. IEEE. Available at: https://doi.org/10.1109/icctac.2018.8370397.

Ashodia, N. and Makadiya, K. (2022) ‘Detection of DDoS attacks in SDN using Machine Learning’, in *2022 International Conference on Electronics and Renewable Systems (ICEARS)*. IEEE. Available at: https://doi.org/10.1109/icears53579.2022.9751879.

Atul Sharma *et al.* (2022) ‘Detection of Distributed Denial of Service Attack in SDN using a Machine Learning Technique’, *International Journal of Advanced Research in Science, Communication and Technology*, pp. 537–539. Available at: https://doi.org/10.48175/ijarsct-7514.

B. V., K., D. G., N. and S, H.P. (2018) *Detection of DDoS Attacks in Software Defined Networks*. IEEE. Available at: https://doi.org/10.1109/csitss.2018.8768551.

Bhardwaj, S. and Panda, S.N. (2022) 'Performance Evaluation Using RYU SDN Controller in Software-Defined Networking Environment', *Wireless Personal Communications*, 122(1), pp. 701–723. Available at: https://doi.org/10.1007/s11277-021-08920-3.

Deepa, V., Muthamil, S.K. and Deepalakshmi, P. (2018) *Detection of DDoS Attack on SDN Control plane using Hybrid Machine Learning Techniques*. IEEE. Available at: https://doi.org/10.1109/icssit.2018.8748836.

Gupta, S. and Grover, D. (2021) *A Comprehensive Review on Detection of DDoS Attacks using ML in SDN Environment*. IEEE. Available at: https://doi.org/10.1109/icais50930.2021.9395987.

Ha, N.V., Quan, D.D. and Nguyen, T.T.T. (2022) 'Graphical User Interface for RYU Software Defined Network Controller', in *2022 IEEE 8th Information Technology International Seminar (ITIS)*. IEEE. Available at: https://doi.org/10.1109/itis57155.2022.10010215.

Hamarshe, A., Ashqar, H.I. and Hamarsheh, M. (2023) 'Detection of DDoS Attacks in Software Defined Networking Using Machine Learning Models'. Available at: https://doi.org/10.48550/ARXIV.2303.06513.

Islam, Md.T., Islam, N. and Refat, Md.A. (2020) 'Node to Node Performance Evaluation through RYU SDN Controller', *Wireless Personal Communications*, 112(1), pp. 555–570. Available at: https://doi.org/10.1007/s11277-020-07060-4.

Jenifa, A. (2023) 'Software-Defined Networking (SDN) Explained in 5 Minutes or Less'. Geekflare. Available at: https://geekflare.com/software-defined-networking/ (Accessed: 28 July 2023).

Kaur, S. *et al.* (2021) 'A Comprehensive Survey of DDoS Defense Solutions in SDN: Taxonomy, Research challenges, and Future Directions', *Computers & Security*, 110, p. 102423. Available at: https://doi.org/10.1016/j.cose.2021.102423.

Khashab, F. *et al.* (2021) 'DDoS Attack Detection and Mitigation in SDN using Machine Learning', in *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*. IEEE. Available at: https://doi.org/10.1109/netsoft51509.2021.9492558.

Kokila, R.T., Thamarai Selvi, S. and Govindarajan, K. (2014) 'DDoS detection and analysis in SDN-based environment using support vector machine classifier', in *2014 Sixth International Conference on Advanced Computing (ICoAC)*. IEEE. Available at: https://doi.org/10.1109/icoac.2014.7229711.

Li, D. *et al.* (2018) 'Using SVM to Detect DDoS Attack in SDN Network', *IOP Conference Series: Materials Science and Engineering*, 466, p. 012003. Available at: https://doi.org/10.1088/1757-899x/466/1/012003.

Madathi M *et al.* (2022) 'Detection of DDoS attack in SDN environment using KNN algorithm'.

Polat, H., Polat, O. and Cetin, A. (2020) ‘Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models’, *Sustainability*, 12(3), p. 1035. Available at: https://doi.org/10.3390/su12031035.

Prasad, A. *et al.* (2022) ‘INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING Detection of DDoS Attack in Software-Defined Networking Environment and Its Protocol-wise Analysis using Machine Learning’.

Santos, R. *et al.* (2019) ‘Machine learning algorithms to detect DDoS attacks in SDN’, *Concurrency and Computation: Practice and Experience*, 32(16). Available at: https://doi.org/10.1002/cpe.5402.

Sudar, K.M. *et al.* (2021) *Detection of Distributed Denial of Service Attacks in SDN using Machine learning techniques*. IEEE. Available at: https://doi.org/10.1109/iccci50826.2021.9402517.

VMware (2021) ‘What Is Software-Defined Networking (SDN)? | VMware Glossary’. VMware. Available at: https://www.vmware.com/topics/glossary/content/software-defined-networking.html.

Yadav, A. *et al.* (2021) ‘A Hybrid Approach for Detection of DDoS Attacks using Entropy and Machine Learning in Software Defined Networks’, in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE. Available at: https://doi.org/10.1109/icccnt51525.2021.9580057.

Yungaicela-Naula, N.M., Vargas-Rosales, C. and Perez-Diaz, J.A. (2021) ‘SDN-Based Architecture for Transport and Application Layer DDoS Attack Detection by Using Machine and Deep Learning’, *IEEE Access*, 9, pp. 108495–108512. Available at: https://doi.org/10.1109/access.2021.3101650.