PREVENTING NETWORK ATTACKS THROUGH SVM AND SDN INTEGRATION

MINI PROJECT REPORT

Submitted by

Dinakar Laxmi Viswanath Bodapati

(208W1A1201)

Sotsava Skandhaa Baji

(208W1A1202)

Under the Guidance of

S. Kranthi

Assistant Professor

DEPARTMENT OF INFORMATION TECHNOLOGY

V R SIDDHARTHA ENGINEERING COLLEGE

(AUTONOMOUS - AFFILIATED TO JNTU-K, KAKINADA)

Approved by AICTE &Accreted by NBA

KANURU, VIJAYAWADA-520007

ACADEMIC YEAR

(2023)

V.R.SIDDHARTHA ENGINEERING COLLEGE

(Affiliated to JNTUK: Kakinada, Approved by AICTE, Autonomous)

(An ISO certified and NBA accredited institution)

Kanuru, Vijayawada – 520007

CERTIFICATE

This is to certify that this project report titled **"Preventing Network Attacks through SVM and SDN Integration"** is a bonafide record of work done by **Bodapati Dinakar Laxmi Viswanath (208W1A1201), Baji Sotsava Skandhaa (208W1A1202)** under my guidance and supervision is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, **V.R. Siddhartha Engineering College** (Autonomous under JNTUK) during the year **2023**.

(S. Kranthi)	**(Dr. M. Suneetha)**
Assistant Professor	**Professor & Head**
Dept. of Information Technology	Dept. of Information Technology

EXTERNAL EXAMINER SIGNATURE

Date of examination**:**

ACKNOWLEDGEMENT

First and foremost, I sincerely salute our esteemed institution **V.R SIDDHARTHA ENGINEERING COLLEGE** for giving me this opportunity to fulfilling my project. I am grateful to our principal **Dr. A.V. RATNA PRASAD**, for his encouragement and support all through the way of my project.

On the submission of this project report, I would like to extend my honor to **Dr. M. Suneetha**, Head of the Department, of IT for her constant motivation and support throughout my work.

I feel glad to express my deep sense of gratefulness to my project guide **S. Kranthi**, Assistant Professor for his guidance and assistance in completing this project successfully.

I would also like to convey my sincere indebtedness to all faculty members, including supporting staff of the Department, friends, and family members who bestowed their great effort and guidance at appropriate times without which it would have been very difficult on my part to finish the project work.

DEPARTMENT OF INFORMATION TECHNOLOGY
VELAGAPUDI RAMAKRISHNA SIDDHARTHA ENGINEERING COLLEGE

PROJECT SUMMARY

S.NO	ITEM	DESCRIPTION
1	Project Title	Preventing Network Attacks through SVM and SDN Integration
2	Student Names& Numbers	Bodapati Dinakar Laxmi Viswanath (208W1A1201) Baji Sotsava Skandhaa (208W1A1202)
3	Name of The Guide	S. Kranthi
4	Research Group	Network Security
5	Application Area	Cyber Security
6	Aim of the Project	The aim is to prevent attacks on network using Machine Learning Model
7	Project Outcomes	Ensures security and accurate attack detection on networks

Student Signatures

1. Bodapati Dinakar Laxmi Viswanath -
2. Baji Sotsava Skandhaa -

Signature of the Guide

S. Kranthi -

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The continued growth of connected devices and increasing reliance on digital infrastructure will lead to an increasing number of cyber-attacks on networks. Traditional security solutions such as firewalls and intrusion detection systems will become less effective in defending against these attacks. In this project, we will use a deep learning model-based solution for detecting and preventing network attacks in a software-defined networking (SDN) environment. We will use a popular deep learning framework, to train a deep learning model based on a DDOS attack network security dataset collected from Kaggle. The model we will use is a Linear Support Vector Machine (SVM). The trained model will be integrated into the SDN environment, and using Mininet we build a network and we perform attack generation on the network and the data is tested with the trained model to predict the attack. This will be used to control the behavior of one or more SDN controllers to prevent attacks. The performance of the proposed solution will be evaluated using a simulated network environment and real-world network security datasets. The results will demonstrate that our solution is effective in detecting and preventing network attacks, and has the potential to significantly enhance network security.

Keywords: SDN Environment, DDOS attack, SVM Model, and RYU Controller.

CHAPTER – 1
INTRODUCTION

This chapter discusses the origin of the problem, objectives and outcomes.

1.1 Origin of the Problem:

The field of network security is a critical aspect of modern computing systems, with software-defined networking (SDN) offering new opportunities for effective security. The objective of this project is to explore the potential of deep learning algorithms for intrusion detection in SDN environments. The project involves selecting a DDOS attack network security dataset from Kaggle and using deep learning techniques to train a model that can detect and classify different types of attacks. The trained model will be integrated with a software-defined network (SDN) environment, which will use controllers to block the detected attack. This project aims to demonstrate the feasibility and effectiveness of deep learning for intrusion detection in SDN and contribute to the advancement of network security research.

1.2 Basic definitions and Background

The increasing number of cyber-attacks on networks has become a major concern for organizations and individuals. Traditional security solutions, such as firewalls and intrusion detection systems, are becoming less effective in defending against these attacks. This project aims to address this problem by using a deep learning model-based solution for detecting and preventing DDOS network attacks in a software-defined networking (SDN) environment. The solution uses a trained linear Support Vector Machine (SVM) algorithm model to control the behavior of one or more SDN controllers to prevent attacks

1.3 Problem Statement with Objectives and Outcomes

Problem Statement:

The aim of the project is to address the problem of cyber attacks by using a deep learning model-based solution for detecting and preventing DDOS network attacks in a software-defined networking (SDN) environment. The solution uses a trained linear Support Vector Machine (SVM) algorithm model to control the behavior of one or more SDN controllers to prevent attacks.

Objectives:

- To develop an efficient machine learning model that can classify the given network traffic dataset to various attacks with maximum accuracy.
- Using SDN controllers to stop traffic from a host based on its Mac address.

Outcomes:

- A working machine learning-based solution for network security that can detect and prevent network attacks in a software-defined networking environment
- Improved understanding of the integration of deep learning models into a software-defined networking environment.

CHAPTER –2
REVIEW OF LITERATURE

This chapter describes the review of literature that we have taken from various papers and considered all the points mentioned in the papers.

2.1 Description of Existing Systems:

Table 2.1: Literature Review

S.no	**Paper Title**	**Authors**	**Publishing year**	**Review**
1.	A Flow-Based Anomaly Detection Approach With Feature Selection Method Against DDoS Attacks in SDNs	Mahmoud Said El Sayed Nhien-An Le-Khac , Marianne A. Azer and Anca D. Jurcut	2020	Our approach provides a high detection rate and presents a more efficient better time to build the model. We further tested the trained model on the performance of the SDN controller to evaluate how the used dataset can impact on the performance of the SDN controller. The results showed that the proposed approach does not deteriorate the network performance
2	Deep Neural Networks for Intrusion Detection in Software-Defined Networking	Wang et al.	2019	The authors evaluate the performance of their proposed solution using both simulated and real-world network security datasets and show that deep neural networks can significantly improve the accuracy of intrusion detection in SDN environments.

3.	End-to-end intrusion detection in software-defined networks using deep reinforcement learning	Qin et al.	2019	The proposed solution can effectively detect various types of network attacks in real-time and provide a flexible and scalable solution for SDN security.
4.	Anomaly-based Intrusion Detection in Software-Defined Networks: A Deep Learning Approach	Zhang et al.	2019	The proposed method uses an autoencoder to learn the normal behavior of the network and identify anomalies, which are then classified as either benign or malicious using a deep neural network.

2.1 Summary of literature:

The aim of this work is to reduce the redundant or irrelevant features without any significant impact on the classification accuracy. We have selected 10 features out of available 48 features using two common feature selection methods IG and RF. The approach provides a high detection rate and presents a more efficient better time to build the model. We further tested the trained model on the performance of the SDN controller to evaluate how the used dataset can impact on the performance of the SDN controller. The results showed that the proposed approach does not deteriorate the network performance.

CHAPTER-3

PROPOSED METHOD

3.1 Design Methodology: As the number of cyber-attacks on networks continues to increase, organizations and individuals are becoming increasingly concerned about their security. Traditional security solutions such as firewalls and intrusion detection systems are becoming less effective in defending against these attacks. To address this problem, this project proposes using a machine learning model-based solution for detecting and preventing DDOS network attacks in a software-defined networking (SDN) environment. The solution involves using a trained linear Support Vector Machine (SVM) algorithm model to control the behavior of one or more SDN controllers to prevent attacks. The design methodology for this solution involves first analyzing the network traffic and identifying the features that are indicative of a DDOS attack. These features are then used to train the SVM model to accurately detect such attacks. Once the model is trained, it is deployed to one or more SDN controllers to monitor network traffic and take action to prevent attacks.

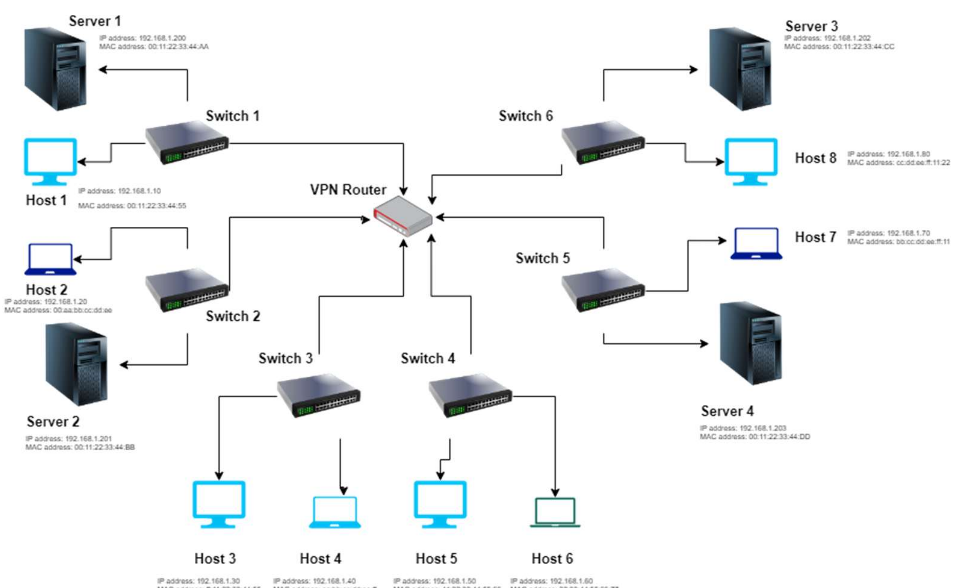


Figure-1: Block Diagram

3.2 System Architecture Diagram

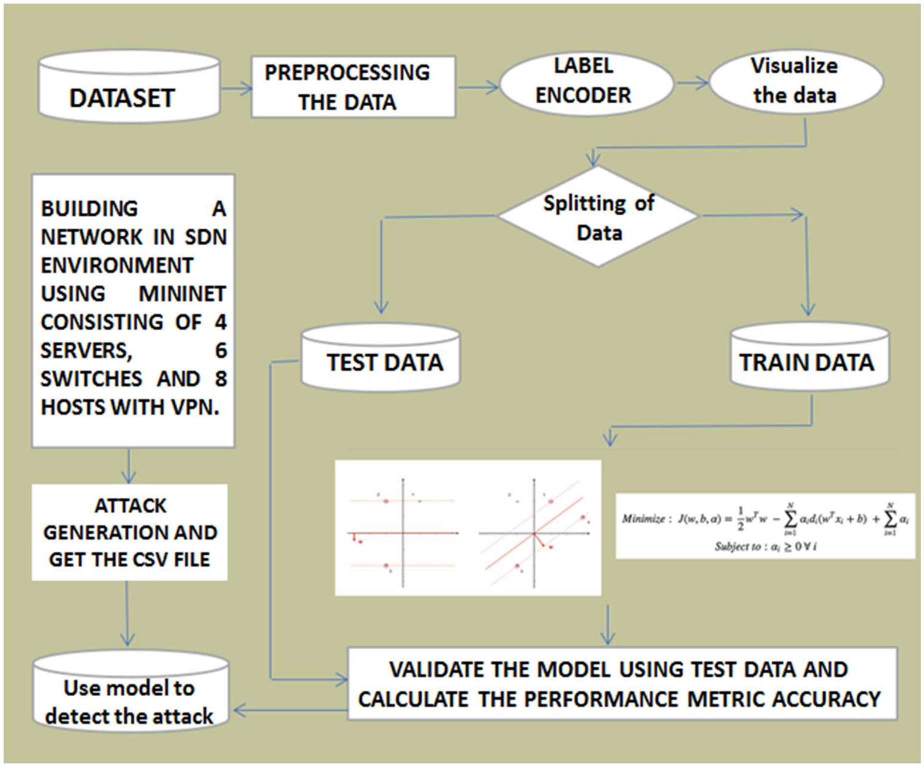


Figure-2: Architecture Diagram

The design methodology also includes testing the effectiveness of the solution on a test network and refining the model as needed. Overall, this solution offers a more effective approach to detecting and preventing DDOS attacks in an increasingly complex and challenging cybersecurity environment.

3.3 Software Design: Module level

In the above architecture diagram, the illustration goes as follows: The information dataset is collected in csv format from internet. Now we preprocess the data in R Environment convert the categorial data to our required input and we visualize the data. Split the data into training and testing and train the SVM model and calculate the accuracy. Now we build a network in SDN environment with the help of mininet package. the network consists of 4 servers, 6 switches, and 8 hosts with an integration with open vpn. We develop the ryu controller to control the attack i.e flood from that attacker host. we generate icmp flood traffic using hping3 and recorded with the help of ITGrec to save it as the required file. Now using the deep learning model i.e linear SVM we test the new csv attack file and detect the attack. With the help of controller we block the attack.

3.4 Datasets:

The Training Dataset is collected from Kaggle from the internet.

We have a huge amount of data entries (867523 Observations)

This is a snapshot of the sample data with column names.

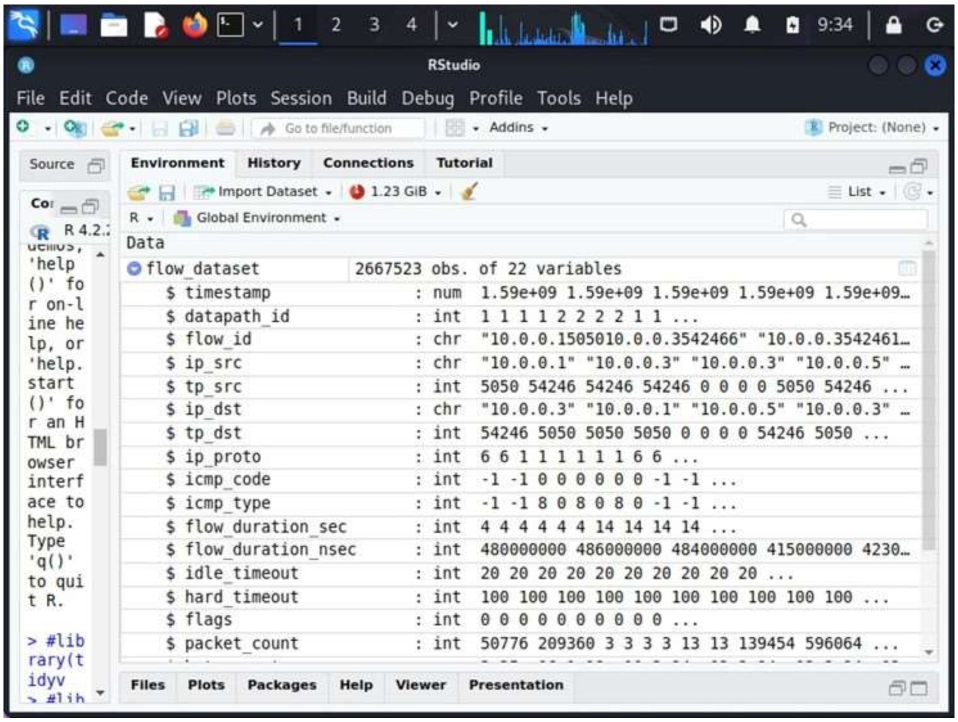


Figure-3: Flow Dataset Diagram

Requirements:

User Interface:

This system's user interface is the Linux os, which is a user-friendly interface.

Hardware Interfaces:

Oracle Virtual Machine, Kali Linux, and Kaggle

Software Interfaces:

Required modules (tidyverse, e1071, caret, graphics, ggplot2, class, KNN, SVM)

Hardware Requirements:

1. Processor – Pentium-IV
2. RAM – 4GB (Minimum)
3. HDD/SSD – 256GB (Minimum)

Algorithms

- Algorithm K-Nearest-Neighbor (KNN) :
- Input: 02152020-threats-02-15-2020.csv Dataset
- Output: Success Accuracy of KNN trained Model on test Data

Pseudocode for K Nearest Neighbor (classification):

This is pseudocode for implementing the KNN algorithm from scratch:

1. Load the training data.
2. Prepare data by scaling, missing value treatment, and dimensionality reduction as required.
3. Find the optimal value for K:
4. Predict a class value for new data:
    1. Calculate distance(X, Xi) from i=1,2,3,….,n.
       where X= new data point, Xi= training data, distance as per your chosen distance metric.
    2. Sort these distances in increasing order with corresponding train data.
    3. From this sorted list, select the top 'K' rows.
    4. Find the most frequent class from these chosen 'K' rows. This will be your predicted class.

Figure-4: K-Nearest Neighbor Algorithm

Algorithms

- Algorithm Linear Support Vector Machine (SVM) :
- Input: 02152020-threats-02-15-2020.csv Dataset
- Output: Success Accuracy of SVM trained Model on test Data

Find the optimal values for the tuning parameters of the SVM model;
Train the SVM model;
$p \leftarrow p^*$;
while $p \geq 2$ **do**
  $SVM_p \leftarrow$ SVM with the optimized tuning parameters for the p variables and observations in **Data**;
  $w_p \leftarrow$ calculate weight vector of the SVM_p $(w_{p1}, \ldots, w_{pp})$;
  $rank.criteria \leftarrow (w_{p1}^2, \ldots, w_{pp}^2)$;
  $min.rank.criteria \leftarrow$ variable with lowest value in $rank.criteria$ vector;
  Remove $min.rank.criteria$ from **Data**;
  $Rank_p \leftarrow min.rank.criteria$;
  p ← p - 1 ;
end
$Rank_1 \leftarrow$ variable in **Data** $\notin (Rank_2, \ldots, Rank_{p^*})$;
return $(Rank_1, \ldots, Rank_{p^*})$

Figure-5: Linear SVM Model Algorithm

Implementation Steps:

- Data Preprocessing
- Label Encoding
- Data Reshaping
- Removal of Null values
- Splitting of data for training and testing
- Reshaping the data for SVM
- Run the SVM model using a deep learning function
- Visualization of Results
- Build the Network and perform attack and generate the file
- Use model to detect the attack and use SDN controllers to prevent the attack.

Network Topology Steps:

Set up a network topology using Mininet or other network emulation tools:

Use the provided code to create a custom topology using the Mininet Python API.
Define the switches, hosts, and servers based on your requirements.
Configure traffic generators to mimic the behavior of a DDoS attack:

Within the Mininet environment, configure traffic generators on selected hosts.
Use tools like D-ITG (Distributed Internet Traffic Generator) or hping3 to generate synthetic attack traffic.
Specify the desired attack parameters, such as source IP addresses, source ports, traffic rates, and packet sizes, to simulate different types of DDoS attacks.
Capture the network traffic during the simulation:

Install and configure tools like Wireshark or tcpdump on the hosts or capture traffic directly from the switches using OpenFlow-based packet capture mechanisms.
Set the capture filters to capture the desired traffic (e.g., based on IP addresses, ports, protocols).
Start the capture process to collect the network traffic data.

Save the captured traffic data:

Once the desired duration of the attack simulation is complete, stop the traffic generators and the network capture process.
Save the captured traffic data to a CSV file or any other desired format.
Extract relevant features from the captured packets, such as source IP, source port, destination IP, destination port, protocol, packet size, and timestamps.
Assign labels to the captured traffic indicating whether it represents an attack or normal traffic.
Use the collected data for training and testing the machine learning model:

Load the captured traffic data from the CSV file into a DataFrame using a library like pandas.
Preprocess the data if necessary, such as scaling numerical features or encoding categorical variables.
Split the data into training and testing sets.
Train your machine learning model, such as a linear SVM, using the training data.
Evaluate the model's performance on the testing data, considering metrics like accuracy, precision, recall, and F1-score.
Utilize the trained model to predict whether new traffic represents an attack or normal traffic:

Obtain new traffic data or capture real-time network traffic using the same capture setup.
Preprocess the new data in the same way as the training data.
Use the trained SVM model to make predictions on the new data.
Analyze the predictions and identify whether the traffic is classified as an attack or normal based on the assigned labels.

CHAPTER 4

Results & Observations

4.1 Description of Results:

TRAINED KNN MODEL

```
set.seed(0)
train_index <- createDataPartition(y_flow, p = 0.75, list = FALSE)
X_flow_train <- X_flow[train_index, ]
X_flow_test <- X_flow[-train_index, ]
y_flow_train <- y_flow[train_index]
y_flow_test <- y_flow[-train_index]

flow_model <- knn(X_flow_train, X_flow_test, y_flow_train, k = 5)

y_flow_pred <- flow_model

print("-----------------------------------------------------------

cm <- table(y_flow_test, y_flow_pred)
print("confusion matrix")
print(cm)
```

Figure-6: KNN Model

		Predicted Class	
		0	1
Actual	0	50	20
Class	1	30	50

Figure-7: KNN Model Confusion Matrix

TRAINED SVM MODEL:

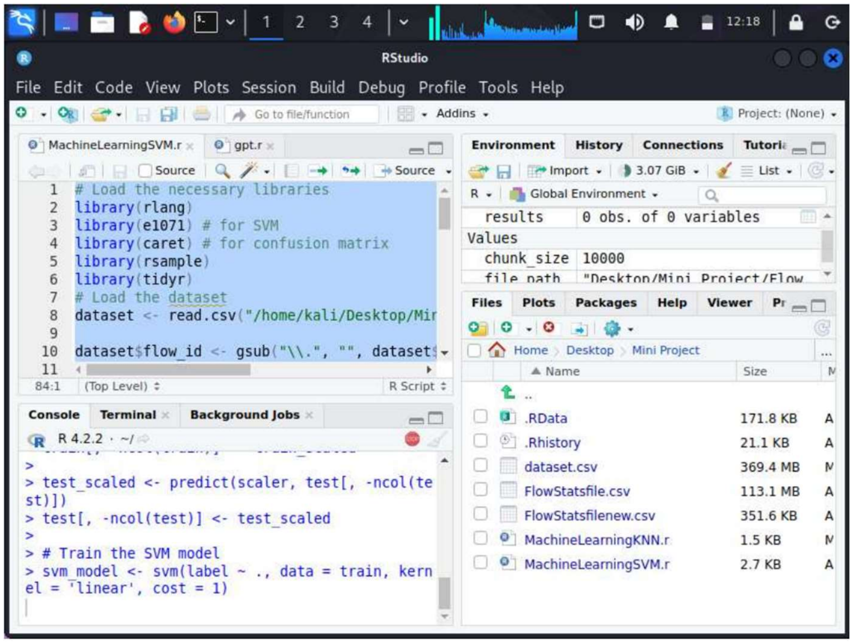


Figure-8: Linear SVM Model

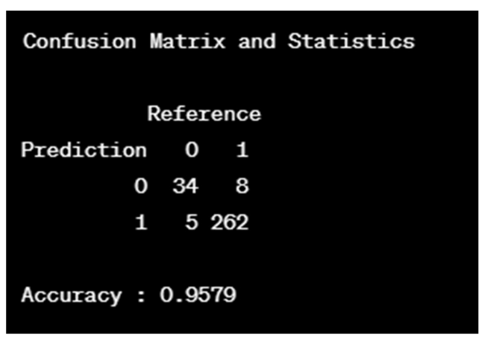


Figure-9: SVM Model Confusion Matrix

Final Result:

The Linear SVM model is successfully evaluated with the validation dataset and got an accuracy of 95.79%.

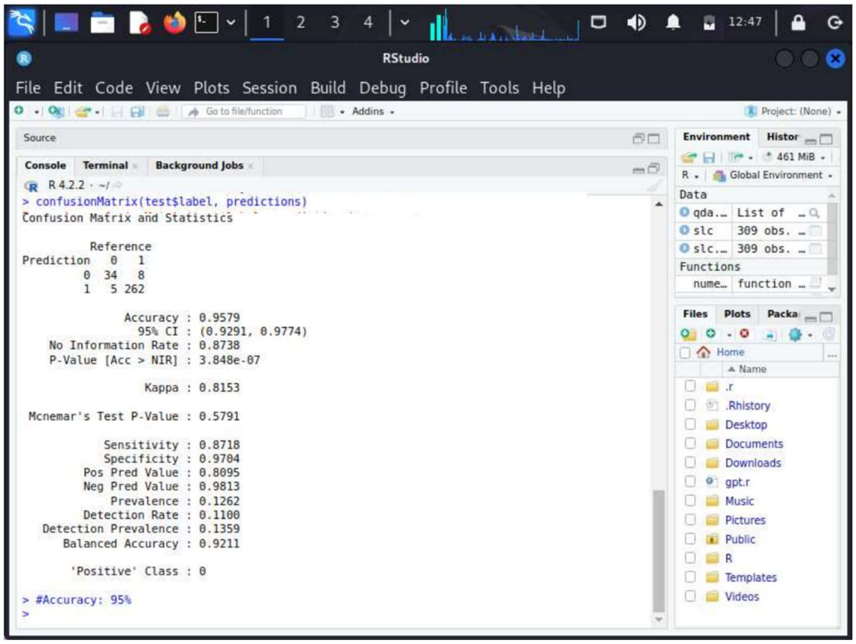


Figure-10 : Linear SVM Model Accuracy

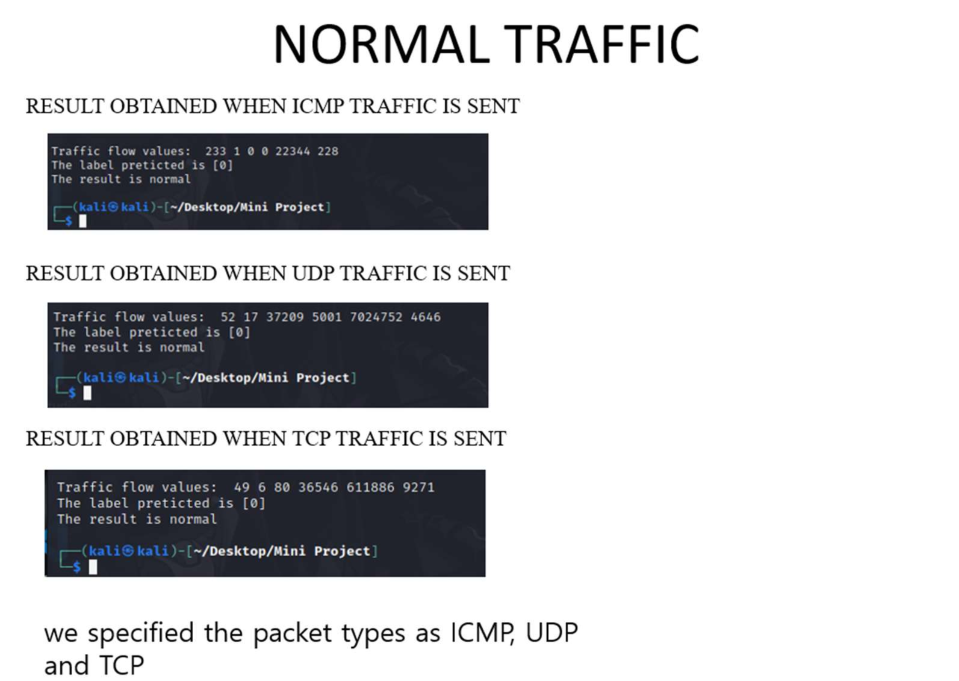


Figure-11 : Normal Traffic Results

ATTACK TRAFFIC

PERFORMING ICMP FLOOD ATTACK USING HPING3

```
# hping3 -1 -V -d 120 -w 64 -p 80 --rand-source --flood 10.0.0.12
using h1 -eth0, addr: 10.0.0.03, MTU: 1500
HPING 10.0.0.12 (h13-eth0 10.0.0.12): icmp mode set, 28 headers + 120 data bytes
hping in flood mode, no replies will be shown
```

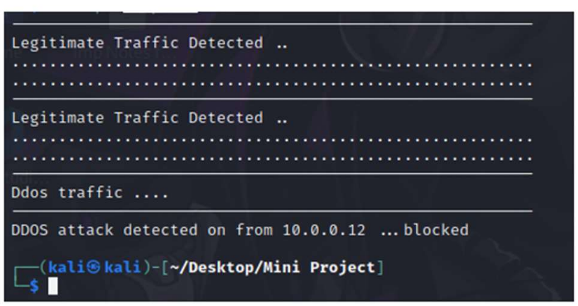
- -1: Specifies the ICMP protocol for the flood ping attack.
- -V: Enables verbose output, providing more detailed information during the attack.
- -d 120: Sets the data size of the ICMP packet to 120 bytes.
- -w 64: Sets the IP TTL (Time To Live) value to 64.
- -p 80: Sets the destination port to 80.
- --flood: Floods the target with ICMP packets at a high rate, generating a large volume of traffic.

Overall, this command launches a flood ping attack to the specified destination IP address, generating a high volume of ICMP traffic.

Figure-12 : ICMP FLOOD USING HPING3

ATTACK IDENTIFIED AND BLOCKED

ICMP packets flood stopped by RYU Controller



```
Legitimate Traffic Detected ..
.....................................................
.....................................................
Legitimate Traffic Detected ..
.....................................................
.....................................................
Ddos traffic ....
DDOS attack detected on from 10.0.0.12 ... blocked
┌──(kali㉿kali)-[~/Desktop/Mini Project]
└─$
```

RESULT OBTAINED WHEN ICMP FLOOD IS SENT

```
Traffic flow values:  246 2 10060 23898 0 0
The label preticted is [1]
The result is attack
┌──(kali㉿kali)-[~/Desktop/Mini Project]
└─$
```

```
# check IP Protocol and create a match for IP
if eth.ethertype == ether_types.ETH_TYPE_IP:
ip = pkt.get_protocol(ipv4.ipv4)
srcip = ip.src
dstip = ip.dst
protocol = ip.proto

# if ICMP Protocol
if protocol == in_proto.IPPROTO_ICMP:
t = pkt.get_protocol(icmp.icmp)
match = parser.OFPMatch(eth_type=ether_types.ETH_TYPE_IP,
ipv4_src=srcip, ipv4_dst=dstip,
ip_proto=protocol,icmpv4_code=t.code,
icmpv4_type=t.type)
```

Here the attack is performed from host-4 with IP address : 10.0.0.12, so the attacker host is blocked from sending the further traffic.

Figure-13 : Attack Identified and Blocked

CHAPTER-5

Conclusion and Future work

5.1 Conclusion:

The proposed model has been developed successfully. Its performance is measured using the Accuracy metric and the result obtained is 95.79%. Also the model successfully detects the traffic and the attacker host is blocked based on the IP address.

Multiple attacks on the network will be performed and using more than one controller to prevent the attack. The model efficiency will also be increased by analysing more features in the future development.

Future Study: Future research for the project will include the expansion Multiple attacks on the network will be performed and using more than one controller to prevent the attack. The model efficiency will also be increased by analysing more features in the future development in network security field.

References:

[1] UllaZ and Q. H. Mahmoud, "Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks," in IEEE Access, vol. 9, pp. 103906-103926, 2021, doi: 10.1109/ACCESS.2021.3094024

[2] Z. Chkirbene, A. Erbad, R. Hamila, A. Mohamed, M. Guizani and M. Hamdi, "TIDCS: A Dynamic Intrusion Detection and Classification System Based Feature Selection," in IEEE Access, vol. 8, pp. 95864-95877, 2020, doi: 10.1109/ACCESS.2020.2994931.

[3] Alqahtani H., Sarker I.H., Kalim A., Minhaz Hossain S.M., Ikhlaq S., Hossain S. (2020) Cyber Intrusion Detection Using Machine Learning Classification Techniques. In: Chaubey N., Parikh S., Amin K. (eds) Computing Science, Communication and Security. COMS2 2020. Communications in Computer and Information Science, vol 1235. Springer, Singapore. https://doi.org/10.1007/978-981- 15-6648Kettle, K. L., &Häubl, G.. (2011). The Signature Effect: Signing Influences Consumption- Related Behavior by Priming Self-Identity. Journal of Consumer Research, 38(3), 474–489.http://doi.org/10.1086/659753.

[4] F. Jiang et al., "Deep Learning Based Multi-Channel Intelligent Attack Detection for Data Security," in IEEE Transactions on Sustainable Computing, vol. 5, no. 2, pp. 204-212, 1 April-June 2020, doi: 10.1109/TSUSC.2018.2793284.-6_10Grace T.R. Lin Chia-Chi Sun, (2009),"Factors influencing satisfaction and loyalty in online shopping: an integrated model", Online Information Review, Vol. 33 Iss 3 pp. 458 – 475.

[5] Dhama, K., Khan, S., Tiwari, R., Sircar, S., Bhat, S., Malik, Y.S., Singh, K.P., Chai-cumpa,W., Bonilla-Aldana, D.K. and Rodriguez-Morales, A.J. (2020) Coronavirus Disease 2019—COVID-19. Clinical Microbiology Reviews, 33, e00028-20. https://journals.asm.org/doi/10.1128/CMR.00028-20

[6] Awojide, Simon, I. M. Omogbhemhe, O. S. Awe, and T. S. Babatope, "Towards the digitalization of Restaurant Business Process for Food Ordering in Nigeria Private University: The Design Perspective. A Study of Samuel Adegboyega University Edo State Nigeria," Int. J. Sci. Res. Publ., vol. 8, no. 5, pp. 46–54, 2018.