**Identification of Host in Software Defined Network**

**A Project Report submitted in partial fulfilment of the requirement for the award of the degree of B. Tech Computer Science and Engineering**

**Submitted By: Group No – 36**

**Under the supervision**

**of**

**Mr. Abhay Kumar**

****

**Department Of Computer Science and Engineering**

### DECLARATION

We students of the 6th semester hereby declare that this project entitled “Identification of host in software defined network” has been carried out by us in the Department of Computer Science and Engineering of National Institute of Technology Patna under the guidance of Mr. Abhay Kumar, Department of Computer Science and Engineering, NIT Patna.

Name Signature

1. Shivam …………………….
2. Rahul Jha …………………….
3. Deepak Kumar **…………………….**

Place: NIT Patna Date: …………………

### CERTIFICATE

This is to certify that Shivam Roll No. 1906070, Rahul Jha Roll No. 1906077, Deepak Kumar Roll No. 1906079has carried out the Minor project (CS6491) entitled as “Identification of host in software defined network” during their 6th semester under the supervision of Mr. Abhay Kumar, Assistant Prof., CSE Department in partial fulfilment of the requirements for the award of Bachelor of Technology degree in the Department of Computer Science and Engineering, National Institute of Technology Patna.

.............................. ……………………

**Mr. Abhay Kumar Dr. J.P. Singh**

Assistant Professor Head of Department

CSE Department CSE Department

NIT Patna NIT Patna

# Acknowledgement

We would like to express our gratitude to our project guide Mr. Abhay Kumar who gave us the golden opportunity to do this wonderful project on the topic, “Identification of host in software defined network” which helped us in learning a lot of new technology stacks and methodologies. Secondly, we would also like to thank all those people who helped us a lot in finalizing this project within the limited time frame.

Date: 29th April 2022

GROUP MEMBERS -

|  |  |
| --- | --- |
| SHIVAM | 1906070 |
| RAHUL JHA | 1906077 |
| DEEPAK KUMAR | 1906079 |

**Index**

1. Introduction …………………………………………………………………………………. 7
   1. Overview……………………………………………………………………………….. 7
   2. Existing system……………………………………………………………………… 7

1.3 Proposed system…………………………………………………………………….. 8

1.4 Technology stack used …………………………………………………………… 8

1. Methodology……………………………………………………………………………… 9
   1. Workflow……………………………………………………………………………… 9
   2. Dataset generation and refining……………………………………………… 9
   3. Framework modules……………………………………………………………… 10

2.4. Results………………………………………………………………………………… 15

1. Features …………………………………………………………………………………… 17
2. Conclusion………………………………………………………………………………… 17
3. Future scope and challenges……………………………………………………… 17
4. References………………………………………………………………………………… 18

## Abstract

Software defined networking (SDN) is an emerging network paradigm that decouples the control plane from the data plane. The data plane is composed of forwarding elements called switches and the control plane is composed of controllers. SDN is gaining popularity from industry and academics due to its advantages such as centralized, flexible, and programmable network management. The software-defined networking (SDN) market is expected to grow by double-digits from 2023, driven by the increasing mainstream adoption of this technology. And for good reason: SDN gives IT the ability to make changes through software code instead of reconfiguring each device, and manage data traffic from a central console to move it in any direction. IT departments can now overcome traditional networking issues such as performance bottlenecks and develop a more responsive network strategy.

But along with its increasing popularity threats on this system is also increasing. One of such threats include the attacks on the network via malicious hosts with processes such as spoofing, hacking etc. In our project we are trying to map hosts and information packets via the packet information captured without trusting its IP addresses and Mac addresses.

This will help in capturing the host info which is trying to do malicious activities on network.

## 1. Introduction

### 1.1 Overview

In this project we try to develop a machine learning model which maps unidentified host with the help of packet information captured.

### 1.2 Existing system

To make the most of the controller’s potential to positively impact network security and guard against the controller becoming the prime attack surface, requires Including security in your design plans right from the beginning to avoid problems down the road.

Just one of the enterprise network management and security possibilities with the SDN controller is its ability to push global security policy updates out centrally across the network. This makes network edge packet filtering possible via a virtualized switch for suspicious traffic redirection to other security devices for more analysis. SDN controller programmability gives engineers the power to install northbound interface security applications that create new ways for applying network security policies.

The major attack vulnerability in the controller requires that access be tightly controlled as a fundamental means of preventing unauthorized activity. To further secure the controller requires:

* Auditing, reviewing, monitoring, and updating role-based access policies
* A high-availability controller architecture for distributed denial-of-service (DDoS) attack prevention
* Encrypting northbound communication via TLS or SSH with secure coding of northbound applications
* Eliminating default application password use and implement application authentication for controller communication approval
* The use of TLS to authenticate endpoints for southbound communication
* Segregating control protocol traffic from the primary data flows via an out-of-band network.

SDN architecture is only as secure as the design, so IT groups should be prepared to use additional methods for securing the network, such as a holistic integration of third-party security solutions.

### 1.3 Proposed system

In this project we capture the information packets sent over the network and try to segregate useful information from those packets to train a machine learning model which predicts the sender host via given packet’s information.

### 1.4 Technology stack used

1. Mininet - Mininet is a software emulator for prototyping a large network on a single machine. Mininet can be used to quickly create a realistic virtual network running actual kernel, switch and software application code on a personal computer.
2. MiniEdit - MiniEdit is an experimental tool created to demonstrate how Mininet can be extended.
3. Scapy - Scapy is a library made in Python, with its own command line interpreter (CLI), which allows to create, modify, send and capture network packets.
4. Floodlight controller - Floodlight Controller is an SDN Controller developed by an open community of developers, many of which from Big Switch Networks, that uses with the OpenFlow protocol to orchestrate traffic flows in a software-defined networking (SDN) environment.
5. Wireshark - Wireshark is the world's foremost and widely-used network protocol analyser. It lets you see what's happening on your network at a microscopic level.

### 2.0 Methodology

#### 2.1. Workflow

First, we have simulated a virtual network in Mininet emulator which uses Floodlight controller as SDN controller and then generated traffic using Scapy which was captured via Wireshark and then refined and used in the following machine learning algorithms:

* Decision Tree Classifier- A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.
* Random Forest Classifier**-** The Random Forest classifier creates a set of decision trees from a randomly selected subset of the training set.
* Support Vector Machine-The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.

The model trained by these algorithms were then tested to predict sender hosts and the accuracy and classification report were then used to determine the optimal model.

##### 2.2. Dataset generation and refining

Firstly, we created our topology using MiniEdit tool which is provided by MiniEdit emulator, then we started our floodlight controller which controls this topology as SDN controller. Wireshark is used to capture the packets transmitted across our topology in pcap file format but as some information is not captured in this format we use csv file format in our machine learning algorithm hence we exported the generated pcap file to json format which is then converted to csv through an online json to csv converter. The dataset generated was huge so we split our dataset in small dataset to be converted via these online converters and then we merged all the generated csv files as a single csv file. As we analyse the packets we try to segregate the useful attributes which are relevant in prediction of our host ip.

**2.3. Framework Modules**

##### A. Data Validation

In order to use attributes of the captured data packets we should remove all redundant and irrelevant data. Data Validation is necessary in order to prevent our model from giving erroneous results.

**B. Basic Topology**

Topology defines the structure of the network of how all the components are interconnected to each other. The following is the topology simulated which was emulated in mininet:

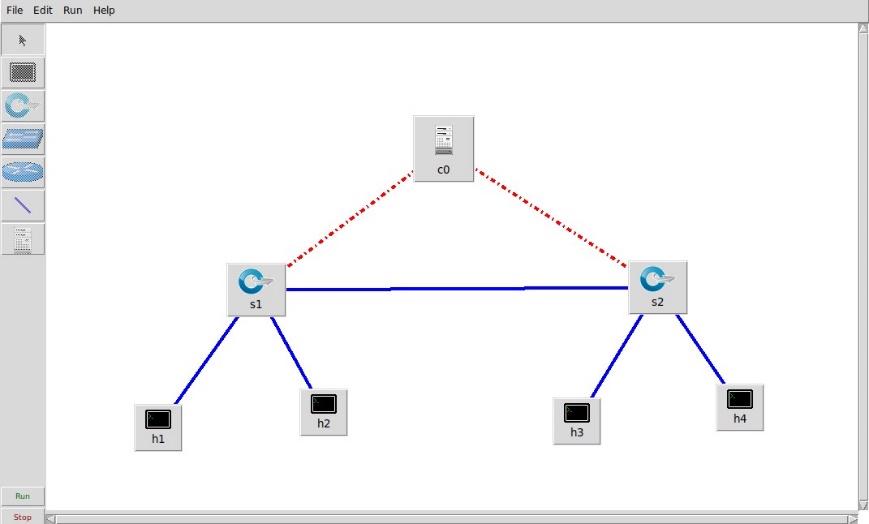


Fig-1 : Basic mininet topology

**C. Large Topology**

Topology defines the structure of the network of how all the components are interconnected to each other. The following is the topology simulated which was emulated in mininet:

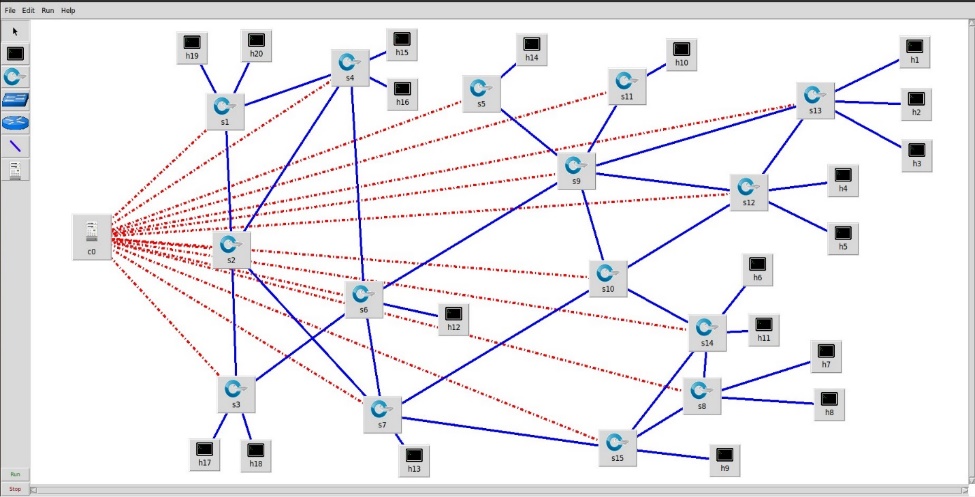


Fig 2: Complex mininet topology

**D. Decision Trees Classifiers**

* Decision Tree is a Supervised learning techniquethat can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
* In a Decision tree, there are two nodes, which are the Decision Node andLeaf Node**.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions***.***
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

****

Fig 3: Basic Structure of Decision Tree

**E. Random Forest Classifiers**

* Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,**which is a process *of*combining multiple classifiers to solve a complex problem and to improve the performance of the model.
* As the name suggests, **“Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.”**Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
* **The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.**
* The below diagram explains the working of the Random Forest algorithm:

****

Fig 4: Random Forest Classifier

**F**. **Support Vector Machines (SVM)**

* Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
* The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
* SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Fig 5 : Support Vector Machine

**2.4**. **Results**

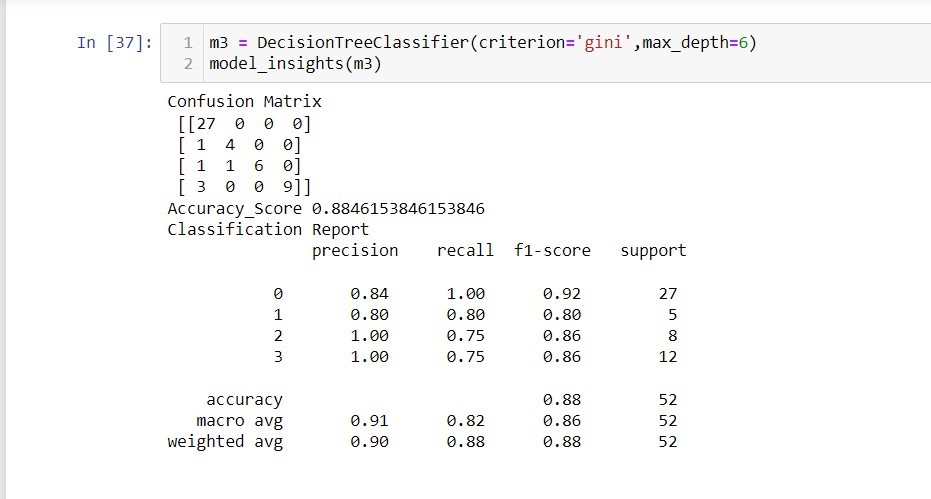


FIG 6 : Decision Tree Classification

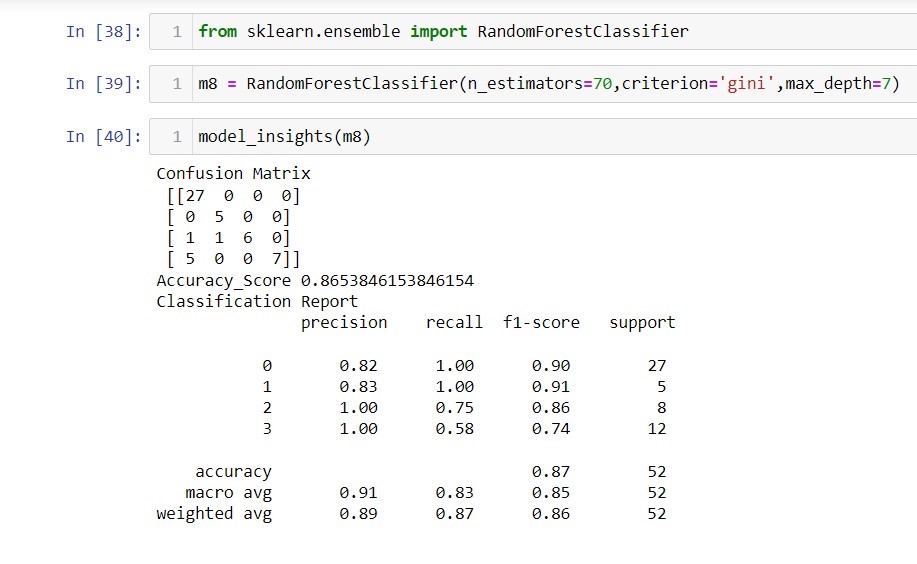
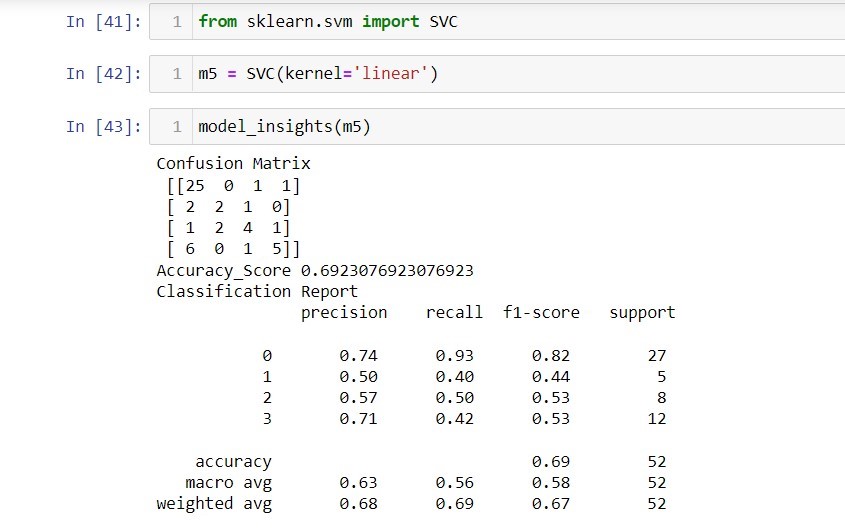


FIG 7 : Random Forest Classification



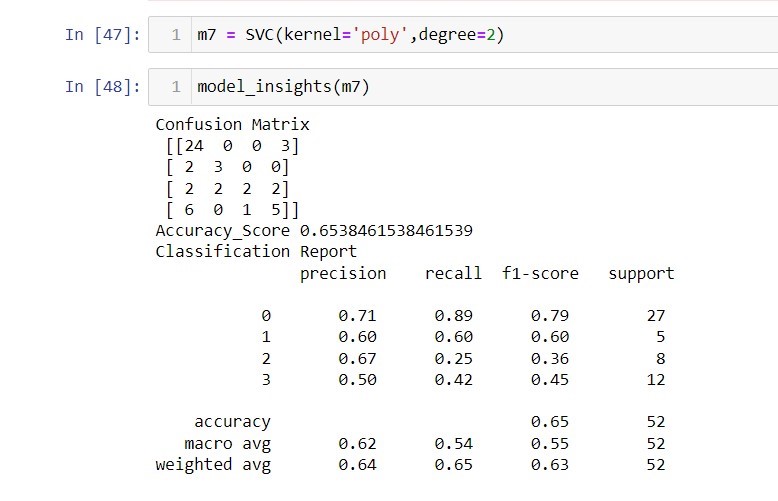


FIG 8 : Support vector machine for classification for linear and poly kernel

### 3.0. Features

1. It can identify any host on SDN network.
2. Our model can be applied on networks of different sizes.
3. It uses three different algorithms to predict accuracy and precision.
4. Our model has satisfactory accuracy.
5. Optimization of results with random forest classifier.

#### 4.0. Conclusion

Our trained machine learning algorithm is able to predict the sender host correctly with high accuracy.

#### 5.0. Future scope and challenges

1. Currently in the model we have only passed data sets which can’t be compared to real traffic which comes over a large network so we would need to validate the results with real data which comes over a SDN system.
2. We would have to upgrade our model such that it could cope up with real time traffic data. This would require use of different and customised machine learning models.
3. The ML model we trained could increase in complexity over a large network and thus would take a lot of time to compute the results. So we would need to make better models which have less time complexities.

**6.0. References**

1. <https://pypi.org/project/scapy/>
2. <https://github.com/topics/openflow>
3. [https://opennetworking ... openflow-spec-v1.3.0.pdf](https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf)
4. <https://opennetworking.org/sdn-definition/>
5. <https://towardsdatascience.com/>
6. [https://floodlight.atlassian ... floodlightcontroller/overview](https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview)
7. [GeeksforGeeks | A computer science portal for geeks](https://www.geeksforgeeks.org/)
8. [Online json to csv converter](https://www.convertcsv.com/json-to-csv.htm)