A PROJECT REPORT ON

DETECTION OF DDoS IN SDN ENVIRONMENT USING SVM, ENTROPY BASED DISCRETIZATION AND FUZZY C-MEANS CLUSTERING.

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING
(Computer Engineering)

BY

Achyuth Rao	B120314254
Akib Shaikh	B120314257
Arun Pottekat	B120314203
Pranav Tale	B120314249

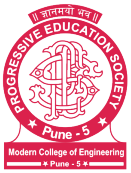
Under The Guidance of

Prof. Ms. Aparna Junnarkar

DEPARTMENT OF COMPUTER ENGINEERING

P.E.S's MODERN COLLEGE OF ENGINEERING
SHIVAJINAGAR, PUNE 411005.

SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE
2016 - 2017



P.E.S's MODERN COLLEGE OF ENGINEERING

Department of Computer Engineering

CERTIFICATE

This is to certify that the project entitled

DETECTION OF DDoS IN SDN ENVIRONMENT USING SVM, ENTROPY BASED DISCRETIZATION AND FUZZY C-MEANS CLUSTERING.

Submitted by

Achyuth Rao	B120314254
Akib Shaikh	B120314257
Arun Pottekat	B120314203
Pranav Tale	B120314249

is a bonafide work carried out by them under the supervision of Prof. Ms. Aparna Junnarkar and it is approved for the partial fulfillment of the requirement of Savtribai Phule Pune university, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering).

Prof. Ms. Aparna Junnarkar
Guide
Dept. of Computer Engg.

Prof. Dr. Mrs. S. A. Itkar
Head
Dept. of Computer Engg.

Signature of Internal Examiner

Signature of External Examiner

PROJECT APPROVAL SHEET
A Project Report Titled as

DETECTION OF DDoS IN SDN ENVIRONMENT USING SVM, ENTROPY BASED DISCRETIZATION AND FUZZY C-MEANS CLUSTERING.

Is verified for its originality in documentation, problem statement, proposed work and implementation successfully completed by

Achyuth Rao	B120314254
Akib Shaikh	B120314257
Arun Pottekat	B120314203
Pranav Tale	B120314249

at

DEPARTMENT OF COMPUTER ENGINEERING

PES MODERN COLLEGE OF ENGINEERING

SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE

ACADEMIC YEAR 2016-2017

Prof. Ms. Aparna Junnarkar
Guide
Dept. of Computer Engg.

Prof. Dr. Mrs. S. A. Itkar
H.O.D.
Dept. of Computer Engg.

Acknowledgement

It gives us pleasure in presenting the project report on **'Detection of Distributed Denial of service attack in Software Defined Network using Support Vector Machine, Entropy Based Discretization and Fuzzy C Means Clustering'**.

Firstly, we would like to express our indebtedness appreciation to our internal guide **Ms. Aparna A. Junnarkar**. Her constant guidance and advice played very important role in making the execution of the report. She always gave us her suggestions, that were crucial in making this report as flawless as possible.

We would like to express our gratitude towards **Prof. Dr. Mrs. S. A. Itkar** Head of Computer Engineering Department, PES Modern College of Engineering for her kind co-operation and encouragement which helped us during the completion of this report.

Also, we would like to thank **Mr. Kunal Khadke, Ms. Yogita Narwadkar, Mr. B. D. Phulpagare, Ms. Pallavi Baviskar, Ms. Deipali V. Gore, Ms. Renuka Kajale** and all **Technical assistants** for providing time to time guidance and various resources such as laboratory with all needed software platforms and continuous Internet connection for our Project.

In the end special thanks to all our classmates for helping us out during the entire documentation process.

Achyuth Rao
Akib Shaikh
Arun Pottekat
Pranav Tale
(B.E. Computer Engineering)

Contents

Abstract i

List of Figures ii

List of Tables iii

List of Abbreviations iv

1 Synopsis 1
- 1.1 Project Title . . . 2
- 1.2 Project Option . . . 2
- 1.3 Internal Guide . . . 2
- 1.4 Technical Keywords . . . 2
- 1.5 Problem Statement . . . 2
- 1.6 Abstract . . . 3
- 1.7 Goals and Objective . . . 3
- 1.8 Relevant mathematics associated with the Project . . . 4
- 1.9 Names of Conferences / Journals where papers can be published . . . 5

2 Technical Keywords 6
- 2.1 Area of Project . . . 7
- 2.2 Technical Keywords . . . 7

3 Introduction 8
- 3.1 Purpose . . . 9
- 3.2 Project Idea . . . 9
- 3.3 Literature Survey . . . 9
  - 3.3.1 Deciding the Project Topic . . . 9
  - 3.3.2 Understanding Industrial Requirements . . . 10
  - 3.3.3 What is SDN? . . . 10
  - 3.3.4 What is DDoS? . . . 10
  - 3.3.5 Types of DDoS . . . 11
  - 3.3.6 DDoS Detection Methods . . . 11
  - 3.3.7 Support Vector Machine . . . 12
  - 3.3.8 Entropy Based Discretization . . . 12
  - 3.3.9 Fuzzy C Means Clustering . . . 12

4 Problem Definition and Scope 13
- 4.1 Problem Statement . . . 14
  - 4.1.1 Goals and Objectives . . . 14
  - 4.1.2 Statement of Scope . . . 14
- 4.2 Major Constraints . . . 15
- 4.3 Methodoligies of Problem Solving and Efficiency Issues . . . 15
- 4.4 Outcome . . . 15

4.5 Applications . . . 15
4.6 Hardware Resources Required . . . 16
4.7 Software Resources Required . . . 16

5 Project Plan **17**
5.1 Project Estimates . . . 18
5.1.1 Reconciled Estimates . . . 18
5.1.2 Project Resources . . . 18
5.2 Project Schedule . . . 18
5.2.1 Project Task Set . . . 18
5.3 Team Organisation . . . 19
5.3.1 Team Structure . . . 19
5.3.2 Management reporting and communication . . . 19

6 Software Requirement Specification **20**
6.1 Purpose . . . 21
6.2 Project Scope . . . 21
6.3 Usage Scenario . . . 21
6.3.1 User profiles . . . 21
6.3.2 Use-cases . . . 22
6.3.3 Use Case Diagram . . . 22
6.4 Product Features . . . 23
6.5 Functional Model and Description . . . 24
6.5.1 Data Flow Diagram . . . 24
6.5.2 Activity Diagram . . . 25
6.5.3 Non Functional Requirements . . . 25
6.5.4 State Diagram . . . 26

7 Detailed Design Document **27**
7.1 Architectural Design . . . 28
7.2 Data Design . . . 29
7.2.1 Dataset Description . . . 29

8 Project Implementation **32**
8.1 Introduction . . . 33
8.2 Methodologies . . . 33

9 Software Testing **36**
9.1 Type of Testing Used . . . 37
9.2 Test Cases and Test Results . . . 37

10 Results **39**
10.1 Screenshots . . . 40
10.2 Output . . . 41

11 Deployment and Maintenance **44**
11.1 Installation and un-installation . . . 45
11.1.1 Installation . . . 45
11.1.2 Uninstallation . . . 46

12 Conclusion and Future Scope **47**

Abstract

The current networking paradigm involves switches, routers and gateways where these networkingdevices constitute both logical thinking as well as routing of packets. Traditionally the network administrator is responsible for configuring and managing these devices manually and at all times, which makes it a tedious task.

With the onset of the Software Defined Network, this task of manually managing the devices reduces to some extent, as it seperates the control plane from the data plane i.e. Forwarding of packets is done in the data plane and intelligence of the entire network resides in the control plane.

Data plane constitutes the network devices which comprise switches known as dumb terminals and the control plane constitutes a central controller which keeps track of all switches in the network.

Due to the centralized nature of SDN i.e. the controller at the center of SDN taking the logicaldecisions, there arises a threat of malicious users launching cyber attacks on this central component thereby, dislodging the entire network. Some attacks include Application level attacks, Brute Force attack, man in the middle attacks, DDoS attack etc.

Distributed Denial of Service attack involves a single malicious user controlling different users known as bots to launch an attack against a single entity in the network without even the victim being aware of the attack. DDoS attacks result in direct financial losses along with damage to company reputation and loss of the customers trust.

As a part of solution to this problem, three algorithms can be used for the detection of DDoS attack i.e. Support Vector Machine, a machine learning algorithm and Entropy based Discretization and Fuzzy C-Means Clustering, two Data Mining algorithms.

Support Vector Machine takes the rate of incoming packets as the input and classifies them as normal traffic or attack traffic. Whereas Entropy based Discretization monitors the entropy of the network i.e. measure of randomness and if it falls below a threshold value then it is classified as attack traffic and Fuzzy C-Means Clustering clusters the incoming packets into normal or attack classes.

Once the attack is detected by either of the applications, an entry will be made in the log files constantly monitored by network monitoring tools and thus report the incident back to the network adminstrative team.

The network adminstrator would be provided a monitoring application which would display the details of the attack detected such as the source IP and victim IP etc.

As a part of the project, comparison studies will also be done related to both algorithms displaying the rate at which attacks are detected along with their accuracies.

List of Figures

6.1 Use Case Diagram 22
6.2 Level 0 Data Flow Diagram 24
6.3 Level 1 Data Flow Diagram 24
6.4 Activity Diagram 25
6.5 State Diagram 26

7.1 System Architecture 28

10.1 Kibana Dashboard 40
10.2 POX Controller Output 40
10.3 Flask - List of Devices 41
10.4 Flask - List of Switches 41
10.5 Flask - SDN Topology 42
10.6 SVM Algorithm Output 42
10.7 Entropy Based Discretization Algorithm Output 43
10.8 Fuzzy C-Means Algorithm Output 43

15.1 Use Case Diagram 64
15.2 State Diagram 64
15.3 Activity Diagram 65
15.4 Sequence Diagram 65
15.5 Dataflow Level 0 Diagram 66
15.6 Dataflow Level 1 Diagram 66

16.1 Gantt Chart 68

18.1 Plagiarism Report 72
18.2 Plagiarism Report 72

List of Tables

6.1 Use cases 22

7.1 Dataset Fields - Support Vector Machine 29
7.2 Attack Dataset Fields - Support Vector Machine 30
7.3 Normal Dataset Fields - Support Vector Machine 30
7.4 Attack Dataset Fields - Fuzzy C Means clustering 30
7.5 Normal Dataset Fields -Fuzzy C Means clustering 30

14.1 IDEA Matrix 53

16.1 Project Planner 68

List of Abbreviations

SDN ..... Software Defined Network

SVM ..... Support Vector Machine

DDoS ..... Distributed Denial of Service

ONOS ..... Open Network Operating Switch

OVSDB ..... Open vSwitch Database

EBD ..... Entropy based Discretization

NP ..... Non Polynomial Time

P ..... Polynomial Time

FCM ..... Fuzzy C Means

HTTP ..... Hyper Text Transfer Protocol

UDP ..... Uniform Datagram Protocol

TCP ..... Transmission Control Protocol

ICMP ..... Internet Control Message Protocol

FTP ..... File Transfer Protocol

QP ..... Quadratic problem

1.

Synopsis

1.1 Project Title

Detection of DDoS in SDN environment using SVM, Entropy based Discretization and Fuzzy C-mean Clustering.

1.2 Project Option

Internal Project

1.3 Internal Guide

Prof. Mrs. Aparna Junnarkar

1.4 Technical Keywords

1. C. Computer Systems Organization
   (a) C.2 COMPUTER-COMMUNICATION NETWORKS
      i. C.2.0 General
         A. Security and protection
      ii. C.2.1 Network Architecture and Design
         A. Centralized networks
2. H. Information Systems
   (a) H.2 DATABASE MANAGEMENT
      i. H.2.8 Database Applications
         A. Data mining
   (a) H.3 INFORMATION STORAGE AND RETRIEVAL
      i. H.3.3 Information Search and Retrieval
         A. Clustering
         B. Information filtering
3. I. Computing Methodologies
   (a) I.5 PATTERN RECOGNITION
      i. I.5.3 Clustering
         A. Algorithms
         B. Similarity measures

1.5 Problem Statement

To provide a solution for the detection of DDoS attack in SDN environment using SVM, Entropy based Discretization, Fuzzy C-mean Clustering and monitoring OpenFlow statistics.

1.6 Abstract

The current networking paradigm involves switches, routers and gateways where these networkingdevices constitute both logical thinking as well as routing of packets. Traditionally the network administrator is responsible for configuring and managing these devices manually and at all times, which makes it a tedious task.

With the onset of the Software Defined Network, this task of manually managing the devices reduces to some extent, as it seperates the control plane from the data plane i.e. Forwarding of packets is done in the data plane and intelligence of the entire network resides in the control plane.

Data plane constitutes the network devices which comprise switches known as dumb terminals and the control plane constitutes a central controller which keeps track of all switches in the network.

Due to the centralized nature of SDN i.e. the controller at the center of SDN taking the logicaldecisions, there arises a threat of malicious users launching cyber attacks on this central component thereby, dislodging the entire network. Some attacks include Application level attacks, Brute Force attack, man in the middle attacks, DDoS attack etc.

Distributed Denial of Service attack involves a single malicious user controlling different users known as bots to launch an attack against a single entity in the network without even the victim being aware of the attack. DDoS attacks result in direct financial losses along with damage to company reputation and loss of the customers trust.

As a part of solution to this problem, three algorithms can be used for the detection of DDoS attack i.e. Support Vector Machine, a machine learning algorithm and Entropy based Discretization and Fuzzy C-Means Clustering, two Data Mining algorithms.

Support Vector Machine takes the rate of incoming packets as the input and classifies them as normal traffic or attack traffic. Whereas Entropy based Discretization monitors the entropy of the network i.e. measure of randomness and if it falls below a threshold value then it is classified as attack traffic and Fuzzy C-Means Clustering clusters the incoming packets into normal or attack classes.

Once the attack is detected by either of the applications, an entry will be made in the log files constantly monitored by network monitoring tools and thus report the incident back to the network adminstrative team.

The network adminstrator would be provided a monitoring application which would display the details of the attack detected such as the source IP and victim IP etc.

As a part of the project, comparison studies will also be done related to both algorithms displaying the rate at which attacks are detected along with their accuracies.

1.7 Goals and Objective

1. Detect DDoS and generate an alert for the network administrative team.
2. Compare three solutions by integrating the concepts of Software Defined Networking with Support Vector Machine, Entropy based Discretization and Fuzzy C-Means.
3. Monitoring OpenFlow Statistics using sFlow and log monitoring using Elasticsearch, Logstach and Kibana (ELK Stack).

1.8 Relevant mathematics associated with the Project

- Input: Network Statistics.
- Output: DDoS detected or not detected.
- Data Structure: Python dictionary
- Function Relation: DDoS alert will be generated if one or more functions detects the attack traffic.
- As, our DDoS detection mechanism will work on the switches distributed throughout the network. Hence, we are using distributed processing.
- Mathematical formulas:
  - SVM:

$$y = \overline{\omega} * x + b \tag{1.1}$$

    This Formula is used for calculating the classification value for a given data point.
  - Entropy based Discretization:

$$\varepsilon = \sum_{i=0}^{n} -P_i log P_i \tag{1.2}$$

    This Formula is used to calculate the Entropy value of Network.
  - Fuzzy C-mean Clustering:

$$C_i = \frac{\sum_{j=1}^{n} u_{ij}^m X_{ij}}{\sum_{n}^{j=1} u_{ij}^m} \tag{1.3}$$

    Formula to calculate centroid values.

$$u_{ij} = \frac{\frac{-2}{d_{ij}^{m-1}}}{\sum_{i=1}^{c} d_{ij}^{m-1}} \tag{1.4}$$

    Formula to calculate membership values.
- Success Conditions:
  - If DDoS attack is launched then an alert is generated.
  - If DDoS attack is not launched then an alert is not generated.
- Failure Conditions:
  - Network not established.
  - Controller is not functioning.
  - Packet capturing unsuccessful.

1.9 Names of Conferences / Journals where papers can be published

- IEEE / Journal of Security and Privacy.
- IEEE / Conference on Computer Network.
- ACM / ACM Transactions on Network Security.
- ELSEVIER / Journal of Network Security.
- Springer / Journal of Communication Network.

2.

Technical Keywords

2.1 Area of Project

- Networking
- Machine Learning
- Data Mining

2.2 Technical Keywords

1. C. Computer Systems Organization
   (a) C.2 COMPUTER-COMMUNICATION NETWORKS
      i. C.2.0 General
         A. Security and protection
      ii. C.2.1 Network Architecture and Design
         A. Centralized networks
2. H. Information Systems
   (a) H.2 DATABASE MANAGEMENT
      i. H.2.8 Database Applications
         A. Data mining
   (a) H.3 INFORMATION STORAGE AND RETRIEVAL
      i. H.3.3 Information Search and Retrieval
         A. Clustering
         B. Information filtering
3. I. Computing Methodologies
   (a) I.5 PATTERN RECOGNITION
      i. I.5.3 Clustering
         A. Algorithms
         B. Similarity measures

3.

Introduction

3.1 Purpose

The SDN architecture involves the separation of the control plane and the data plane. However this architecture brings to light many security issues. SDN market is expected to reach 12.5 billion dollars by the year 2020. As the number of SDN deployments increase it becomes a necessity to solve these security issues and threats. A successful Distributed Denial of Service (DDoS) attack can make the resources unavailable and cripple the entire network.

By bringing down the complete operation of the network and exhausting the resources a single DDoS attack can cause a huge loss to an organization. Therefore effective solutions are required to detect the DDoS attack as early as possible with high accuracy. For the prevention of any attack it's early detection is very necessary. On detecting the DDoS attack effective measures can be taken to prevent it so as to protect the network from the malicious activity initiated by the attackers. This project aims to solve the security issues so as to accelerate the adoption of SDN.

3.2 Project Idea

Software Defined Networking(SDN) is a promising approach that enables superior network control by providing the ability to change, manage and control, the behaviour of the network and the network devices in a dynamic manner. The separation of control plane (responsible for taking networking decisions) from the data plane (responsible for forwarding the packets) introduces programmability into the network.

This project combines the concepts of next-generation networking along with machine learning and data mining approaches in order to solve the security issues in SDN. Our aim is to detect the DDoS attack using either Support Vector Machine (SVM) Classifier, Entropy Based Discretization or Fuzzy C Means Clustering and generate an alert so that later it can be effectively mitigated. The acceptance of a solution in an industry depends on its successful functioning in a real world scenario. We propose and compare three solutions for detection of DDoS attack. Support Vector Machine Classifier is a solution based on Machine Learning whereas Entropy-Based Discretization and Fuzzy C Means Clustering are based on data mining. The final solution will be such that it can actually be used in the SDN deployed environments.

3.3 Literature Survey

3.3.1 Deciding the Project Topic

Initially, deciding the project title was a big task in itself. In recent years there has been a huge increase in the use of open source software and technologies, and contributing some work to the open source community would be a great achievement. Everything is future is going to be virtualized as it has already been achieved for computing and storage. In these recent years it was time that traditional networking which was into operation since past 20 years gets changed with the help of research and innovation.

Software Defined Networking[1][2][3] is a new and promising approach that introduces programmability into the network and is changing the way networking is done since years. But the adoption of SDN by various enterprises is limited due to security issues. This project focusses on the security issues in SDN architecture which separates the data and control plane from each other. From a few studies it is known that a DDoS attack on an enterprise or data center can cost them millions of dollars. This was a serious problem for which an effective solution was needed and little work was done related to DDoS in SDN[4][5].

3.3.2 Understanding Industrial Requirements

Software Defined Networking has received a lot of attention and some great work is done by the research community. As technology matures, the number of SDN deployments in enterprise, service provider, data center and Wide Area Networks will increase over a couple of years. It is found that though SDN has a lot of benefits because it provides programmability in networks, security is a very important question that arises. Currently network security is what is limiting the rate of increse in real world SDN deployments. Although the SDN is adopted widely by large web scale providers like Google, Amazon, Facebook, Microsoft and communication service providers like CenturyLink, AT&T, NTT and similar, it is not being adopted on a large scale by enterprises due to lack of security solutions, standardization and low level maturity of SDN. Using the granular control provided by SDN the security solutions need to be developed so as to encourage the adoption and use of SDN.

3.3.3 What is SDN?

Software Defined Networking is all about bringing programmability, automation and superior control in the network in order to increase the scalability and flexibility. The processing of the packets is not done by the switches. Thus SDN architecture decouples the network control from the data or forwarding plane which consists of network devices forwarding traffic based on the control-plane policy. This separation of the control plane and the data plane simplifies the design of new protocols and implementation of new network services.

OpenFlow is one of the protocols that can be used for communication between the control plane and the data plane. The controller is a software running on a server which acts as the operating system of the SDN. The devices in the data plane use secure transport layer to communicate securely with the controller using the OpenFlow protocol. Whenever a packet arrives at a switch the header of the packet is checked with the fields in the flow entries and if a match is found then the corresponding action associated with the flow entry is executed. If a match is not found by checking all the entries in the flow table then the packet header is forwarded to the controller for further processing. It then processes the packet and makes the decision of whether the packet will be forwarded by the switch or will be dropped and appropriately makes a flow entry in the flow table of the concerned switch.

The logically centralized controller can lead to many security challenges as if the controller becomes unavailable then the whole SDN architecture is lost. One of the reasons for the controller to become unavailbale is the occurrence of a DDoS attack on the network. If a DDoS attack is launched in which the source IP addresses are spoofed then every packet will be sent to the controller for processing which will exhaust the computing resources, where the controller is running and thus the controller can become unavailable for processing of legitimate packets. Moreover multiple flow entries will be made in the flow table which will be a processing overhead for the switches.

Also the effectiveness of attack detection of both these solutions will be studied and the best solution for a specific environment and network traffic will be proposed. After the attack is detected any appropriate mitigation technique can be applied.

3.3.4 What is DDoS?

Distributed Denial of Service[6] (DDoS) attack is an attempt to make network or server resources unavailable to its intended users such that some or all the legitimate requests are prevented from being fulfilled. The attacker accomplishes the DDoS attack by flooding the targeted machine with huge amount of requests, often thousands, using spoofed IP addresses. For achieving this, the attacker infects multiple compromised systems with Trojans and further they are used to target a single or multiple victims. Moreover the attack packets contain spoofed source IP addresses which makes it impossible to block the traffic based on the source IP addresses. In Software Defined Network a DDoS attack can cause problems to switches as well as the controller which acts as central point of contact.

3.3.5 Types of DDoS

1. UDP Flood:-
   UDP Flood is a type of DDoS attack in which the attacker floods the random ports on a remote host with numerous UDP packets. This causes the host to repeatedly check for the application listening at that port and when no application is found, reply with ICMP Destination unreachable packet. This exhausts the resources of the victim and can lead to inaccessibility.

2. ICMP or PING Flood:-
   In ICMP Flood attack a huge amount of ping requests are sent by the attacker to the remote machine continuously without even waiting or bothering about the ICMP reply messages. Because of this the remote machine gets involved in sending the reply messages for the requests received which can exhaust its resources.

3. SYN Flood:-
   SYN Flood is a type of DDoS attack which exploits the weakness of TCP three way handshake mechanism. When a SYN-request is sent to the host for a TCP connection, a SYN-ACK message is sent in response by the host, which is then confirmmed by an ACK response by the requester.

   In SYN Flood attack multiple SYN-request messages are sent to the host, usually with spoofed IP address which causes the host to send multiple SYN-ACK responses. But the final ACK response is not sent by the requester and the host resources are utilized since it is waiting for a response. Also the maximum number of simultaneous connections that can be made is reached causing a denial of service.

4. HTTP Flood:-
   HTTP Flood is a type of DDoS attack in which the attacker exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. HTTP floods do not use malformed packets, spoofing or reflection techniques, and require less bandwidth than other attacks to bring down the targeted site or server. The attack is most effective when it forces the server or application to allocate the maximum resources possible in response to each single request.

3.3.6 DDoS Detection Methods

1. Statistical Methods:
   In this approach the statistical property of the normal and attack traafic can be used. A statisticalmodel for normal traffic is developed and then a statistical inference test is applied to check if some other instance belongs to this model. If it does not belong to this model then it can be considered as an anomaly. Change Aggregation Trees (CAT) and D-WARD are some methods of DDOS detection that use statistical analysis.

2. Soft Computing Methods:
   Soft Computing can be described as a set of optimization and processing techniques which are tolerant to imprecision an uncertainty. Neural Networks, Radial Basis Functions and Genetic algorithms can be used in DDOS detection since they can classify intelligently. Algorithms such as Radial Basis Function (RBF) neural networks can be used to classify the data to normal or attack traffic.

3. Knowledge Based Methods:
   In this approach , the network status and the events are checked against predefined states, rules or patterns of attack. The general representations of known attacks are formulated to identify actual occurrences of the attack. MULTOPS (Multi Level Tree for Online Packet Statistics) is one of the methods that monitors the network traffic to detect the DdoS attack.

4. Machine Learning and Data Mining Methods:
   This approach can be used effectively for proactive detection of DDOS attacks by continuous

monitoring of DDOS attacks and legitimate applications. This approach can be tailored especially for DDOS flood attacks. Methods like Cluster analysis can be used. Support Vector Machine (SVM), k-means clustering and k-NN classifier are some of the machine learning techniques that can be used for DDOS detection.

3.3.7 Support Vector Machine

A Support Vector Machine[7][8] is a Supervised Algorithm used for classificaion and regression. From a set of Training Data points, the SVM model represents them in a space, it maps them by categories and divides them by the seperating Hyper-Planes. When new Data Points arrive based on the nature of the point it categorises the data into the clusters previously formed. Since, SVM maximizes the classification margin the achievable accuracy is very high also using Kernel functions SVM can act as a multi-dimentional non linear classifier.

Since detection of DDoS is a decision problem, a classifier is a very good approach to do this. Thus, we use Support Vector Machine Classifier on the Flow table entries in the aforementioned Software Defined Network. We will first train the Support Vector machine based on the network scenario and environment. Thus, the SVM will know the exact nature of the flow of traffic in both normal and DdoS scenarios. We will be using the advantages of the OpenFlow protocol to collect flow information and then classifying the traffic into normal or attack traffic.

3.3.8 Entropy Based Discretization

Entropy based discretization[9] is a lightweight DDoS attack detection module, which will be running on the edge switches to reduce the flow collection overhead to the controller. Here, first we initialize λ i.e., threshold entropy and ΔT i.e., time interval. For each ΔT we will calculate entropy of network and compare it with the threshold entropy to conclude whether the traffic is normal traffic or attack traffic. If the traffic is normal traffic then we will update the threshold entropy and next time when the ΔT is over, we will use this updated threshold entropy. To calculate the entropy of network we will take flow entries from swiches where the module will be running. The efficiency will depends on the ΔT. As ΔT decreases the efficiency will increase but decrease in ΔT will cause an increase in calculation overhead. Thus, by appropriately setting the value of ΔT, the accuracy of DDoS detection can be significantly increased.

3.3.9 Fuzzy C Means Clustering

Fuzzy C Means clustering is an unsupervised algorithm, which involves assigning fuzzy values to data points, such that a single data point can belong to more than one cluster. Initially, the total number of clusters and its respective centroid values are generated using the subtractive clustering algorithm. Using these centroid values, the membership values are generated for each data point. Following this the centroid values are recalculated. These steps are repeated till the centroid values do not change. A threshold membership value for the attack cluster is decided and compared against all data points during the detection phase. The percentage of data points belonging to the attack cluster is calculated and compared with a threshold, if found greater then it can be predicted that a DDoS scenario has risen.

4.

Problem Definition and Scope

4.1 Problem Statement

4.1.1 Goals and Objectives

The next generation network i.e. Software Defined Network will be classified as a revolutionary step in the networking paradigm, owing to the seperation of control plane and data plane which results in better network management and increased throughput. However, such a change comes at a huge cost, wherein due to centralization of the entire network, there is a risk to the network from network attacks such as DDoS etc. Hence, we propose a solution for detection of one such network attack i.e. Distributed Denial of Service attack in a Software Defined Network using Support Vector Machine, Entropy Based Discretization and Fuzzy C Means Clustering algorithms. The objectives of the project are as listed below:

1. We propose and compare three solutions for detection of DDoS attack. Support Vector Machine Classifier is a solution based on Machine Learning whereas Entropy-Based Discretization and Fuzzy C Means Clustering are based on data mining.
2. The SVM, Entropy or Fuzzy application will be running on edge switches. By analyzing the continuously changing data of the flow table and the network statistics the attack will be detected.
3. Log monitoring and analysis tools like Elasticsearch, Logstash and Kibana (ELK) is used and configured such that as soon as the attack is detected an alert is generated and sent to the network administrator so that further steps can be taken for effectively mitigating the attack.

4.1.2 Statement of Scope

The scope of the project is limited to the following points:

1. Deploying a custom topology of Software Defned Network for experimentation in a virtual environment using a network simulation tool like Mininet and configuring Linux machines to function as OpenFlow switches using Open vSwitch.
2. Training the SVM classifier to detect the attack using an environment-specific data set.
3. Selecting the threshold value based on the environment traffic and comparing the calculated entropy with the threshold to detect the attack packets.
4. Using Fuzzy C Means Clustering to dynamically cluster the incoming packets and distinguish between attack traffic cluster and normal traffic cluster.
5. Running the SVM, Entropy or Fuzzy application on the edge switches so that the run-time data can be analyzed by extracting the flow table information or packets from the switch and classifying or performing calculation to analyze the packets.
6. To generate the attack alert by using a network monitoring tool such as ELK stack so that the network administrator can further take the appropriate measure for mitigation of attack.
7. To monitor the usage and load on the switches using sFlow protocol. Host sFlow Daemon will be installed on the Linux machines which have been converted to switches. Sflow-RT will be running on the monitoring server which collects the real time statistics of the switches.
8. To compare the performance and the overhead of all the solutions and suggest the best solution for a specific environment traffic or a specific DDoS attack type.
9. To take a step towards solving the security issues in SDN architecture.
10. Implement a complete DDoS detection solution which can be used in real world SDN deployments by integrating multiple technologies.

The project does not involve the mitigation of the attack after its detection. This task is currently being assigned to the network administrator to take effective measures such as blocking the traffic from specific IP's or installing an appropriate flow in the switches.

4.2 Major Constraints

The project has certain constraints from the implementation point of view. Some of the constraints are as listed below:

1. The SDN must be deployed using programmable switches and should be functioning.
2. The network connections should be reliable and required topology should be setup before hand.
3. The switches must support the OpenFlow protocol. It may be integrated in the hardware or using a software such as Open vSwitch.
4. The flow entries in the flow table must have IP information which is required for analysis.
5. The application must have root privileges while running on the edge switches which is required for capturing the packets.
6. The switches must have considerable computing capability for running the application. Else it may slow down the packet forwarding activity on the switch.
7. Required softwares along with their dependencies must be installed and tested before the actual use of our application.

4.3 Methodoligies of Problem Solving and Efficiency Issues

4.4 Outcome

The basic aim of the project is the detection of DDoS attack. When the packets or flow entries are analyzed and the traffic appears to be malicious then the entry of the switch ID along with the MAC ID will be made in the log file - /var/log/ddoslog. This file will be monitored by an agent by tailing the file continously at regular intervals of few seconds. As soon as a new entry appears in this file this data along with the timestamp will be sent to central logging server consisting of Elasticsearch, Logstash and Kibana. Logstash will do the filtering of this data and store it in Elasticsearch in the form of indexes.
On applying the filter of the keyword DDoS it can be known that the attack has been detected at the particular switch for which the entry has been made recently. Accordingly the alert can be generated in the form of e-mail to the administrator giving the details of the switch ID for further investigation and mitigation.

4.5 Applications

Security is a major concern for any enterprise and along with the adoption of a particular technology. Security measures need to be taken so as the prevent any loss to the organization. Our DDoS detection mechanism is focussed on SDN and can be used in real world SDN deployments. Any deployment where there is a constant threat of an attack by malicious users can use this application. Moreover the deployment where the load on the controller is a mojor concern is a perfect candidate for using our application. This mechansim works in modular way where the application is individually running on each of the edge switches. The packets are captured and the flow entries are analyzed which requires

computing capability. The controller itself has a huge load for processing the new packets arrived and making appropriate flow entries in flow table of the switches. This computation for attack detection can be an overhead for the controller because of which we assign this task to the switches.

4.6 Hardware Resources Required

For the SDN deployment we needed switches which support the OpenFlow protocol. These switches are very expensive and such an investment for project purpose was not feasible. There was an alternative where we can convert Linux machines into OpenFlow enabled switches using Open vSwitch. In order to create multiple ports on the switch we used USB to Ethernet adaptors. The following hardware resources are required:

1. Two Linux Machines with Ubuntu 15.04 having at least 4 logical cores, 8 GB RAM and 30 GB storage capacity.
2. Two Linux Machines with Ubuntu 15.04 having at least 2 logical cores, 4 GB RAM.
3. Four RJ45 cables each at least 1 metre long.
4. Four USB to Ethernet adaptors with at least 100Mbps capacity.

4.7 Software Resources Required

As stated in the architecture for the demonstartion purposes we made use of two Linux machines as OpenFLow switches which had two hosts each in the form of virtual machines. The following softwares need to be installed:

1. Oracle VirtualBox
2. Open vSwitch
3. tshark
4. Ubuntu VMs
5. Apache Web Server
6. Ostinato
7. Elasticsearch
8. Logstash
9. Kibana
10. hsflowd
11. sFlow-RT
12. POX Controller
13. Python Flask
14. Filebeat Agent

5.

Project Plan

5.1 Project Estimates

Prototyping model has been used in this project. Initially, the model of our project was developed using mininet, a SDN simulator software to check the feasibility of the project. The algorithms were deployed on the simulated switches with network traffic generated manually using an open-source tool Ostinato. During the later stages the project was migrated from a simulation environment to a production environment.

5.1.1 Reconciled Estimates

1. **Cost Estimate :** No hardware cost involved, the only constraint is that SDN should already be set up by the orgranization.
2. **Time Estimate :** The project can be divided into two phases Design phase and the Implementation phase.
   (a) The design phase which started in the month of June and extended till the month of September. This phase included documenting the project and listing out the modules and programming interfaces required to implement the project.
   (b) The implementation started in the month of December through February.

   The detailed time line for our project has been given in the Annexure C.

5.1.2 Project Resources

Project resources that will be required are:

- A dedicated team of developers.
- A network setup with specifications as mentioned in hardware specification.
- Softwares with specifications as mentioned in software specification.

5.2 Project Schedule

5.2.1 Project Task Set

1. Domain Selection(Networking, Machine Learning, Data Mining, Physics).
2. Topic Finalization.
3. Feasibility Study and Market Potential Analysis.
4. Abstract Submission.
5. System Architecture Design.
6. Preparing Synopsis.
7. Project Report Submission for Stage I.
8. Discussion about Platform Selection.
9. System Design and Coding.
10. Testing.
11. Final Report Preparation.

5.3 Team Organisation

5.3.1 Team Structure

The project team consists of four people:

- Achyuth Rao - Responsible for developing the Support Vector Machine algorithm for the project.
- Akib Shaikh - Responsible for the deployment of the Software Defined Network.
- Arun Pottekat - Responsible for developing the Fuzzy C-Means algorithm for the project.
- Pranav Tale - Responsible for developing the Entropy based algorithm for the project.

5.3.2 Management reporting and communication

The two guides that we will be reporting to are:

1. External Guide
2. Internal Guide

- Any progress is to be first reported to the internal co-guide.
- Upon approval, reporting is done to the internal guide.
- Any changes or suggestions are made/adhered to.
- The presentation is given to the external guide and any further changes are made.

6.

Software Requirement Specification

6.1 Purpose

The number of Cyber Attacks is increasing day by day and has become a major concern for enterprises. Such attacks include application level attack, man in the middle attack, brute force, DDoS etc. Attackers are waging an asymmetric battle against these enterprises on their networks, assets and data. One major type of attack would be Distributed Denial of Service attack which is launched by flooding the targeted machine with multiple requests to overload the system and prevent the legitimate packets. In the next generation networking, when Software Defined Networks would be deployed in data centers, enterprises or campus networks, security will prove to be a critical issue against such attacks that needs to be taken care of, which is the aim of our project.

In Software Defined Networks, the network intelligence and state are logically centralized which gives programmability, better network control, flexibility and scalability. But in this new architecture the controller which is responsible for controlling the whole network can become a single point of attack. In our proposed method, we will be using Support Vector Machine, Entropy Based Discretization as well as Fuzzy C Means clustering in order to effectively detect any DDoS attack that takes place in a Software Defined Network.

6.2 Project Scope

The scope of the project extends to setting up of a Software Defined Network using OpenFlow configured switches and also use three algorithms i.e. Entropy based discretization, Fuzzy C Means Clustering and Support Vector Machine for the detection of DDoS attacks. As a part of this project we would also be using a network monitoring tool to monitor the software defined network and raise a notification as soon as any of the three algorithms reports a DDoS attack by marking an entry into the log files which could be utilized for future needs.

6.3 Usage Scenario

6.3.1 User profiles

1. **Network Administrative team :** Every software or non-software industry now-a-days involves its own network establishment. A network administrative team is delegated the task of monitoring the network activities and troubleshooting the network in case of any issues.

   Currently this team faces a tough task of manually monitoring the network for any attacks at all times. Deployment of our product in an enterprise would reduce the burden of the network team as they would receive a notification whenever DDoS attack occurs thereby allowing them to take precautionary measures.

6.3.2 Use-cases

Sr. No.	Use Case	Description	Actors	Assumptions
1	DDoS Notification raised	Raising DDoS notification as soon as DDoS is detected by any of the three algorithms i.e. Support Vector Machine, Entropy based discretization and Fuzzy C-Means Clustering	Network Administrator	SDN is established within the environment and DDoS attack is launched on the enterprise.
2	Network Monitoring	Monitoring the network statistics for any anomalies using sFlow, ElasticSearch Logstash Kibana (ELK Stack), and Flask based application	Network administrator	The link between switches and the monitoring system is maintained throughout.

Table 6.1: Use cases

6.3.3 Use Case Diagram

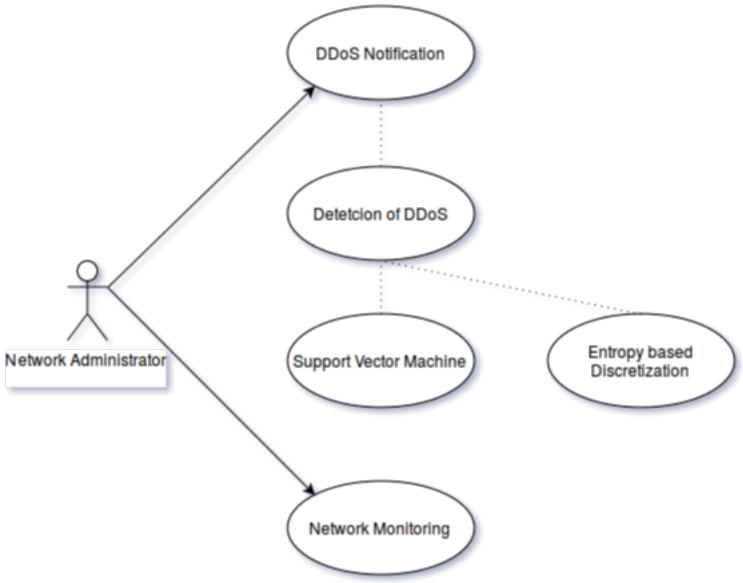


Figure 6.1: Use Case Diagram

6.4 Product Features

1. **Runtime Detection of DDoS attack :-**
   In the current network scenario, there aren't many utilies for efficient DDoS detection. Due to this, the network administrator has to monitor the network continuously. Hence, this product will be step towards the automation of this process wherein the network administrator will have to verify the attack, only when the notification is generated.

2. **Network Environment Specific Product :-**
   The training of the Support Vector Machine algorithm and Fuzzy C Means Clustering is done using an environment specific conditions or depending on the normal network traffic conditions in your network. Thus these algorithms are trained in such a way so as to reduce the false positive rate of detection. Also the threshold for Entropy Based mechanism is decided depending on the normal traffic conditions.

3. **Automated Alert Mechanism :-**
   As soon as the attack is generated the alert can be sent to a specific network monitoring team or can send an email along with the information of where the attack has occurred specifying the IP addresses.

4. **Suitable for all Software Defined Networks :-**
   Independent of what type of Software Defined Network like carrier, service provider, enterprise, data center or campus network, our solution will be effective in all these environments.

5. **Open Source to encourage contribution and research :-**
   As SDN itself is an open standard of next generation networking, our solution will be made open source along with the source code and the method of detection, describing the step-by-step complete procedure. This will help us to encourage the open source community to contribute their ideas and ultimately improve the effectiveness.

6. **SDN Controller Independent :-**
   Since our application will the running on the edge switches, irrespective of which SDN controller is in use our solution can still be used.

6.5 Functional Model and Description

6.5.1 Data Flow Diagram

1. **Level 0 Data Flow Diagram :**

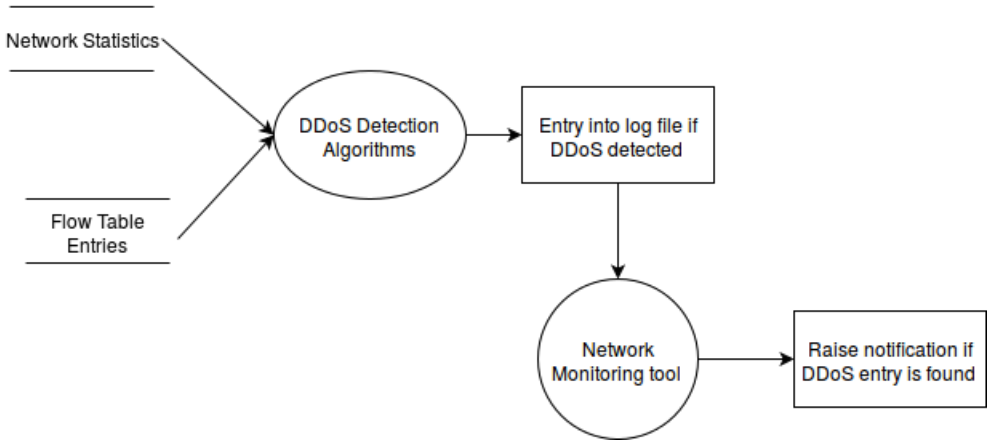


Figure 6.2: Level 0 Data Flow Diagram

2. **Level 1 Data Flow Diagram :**

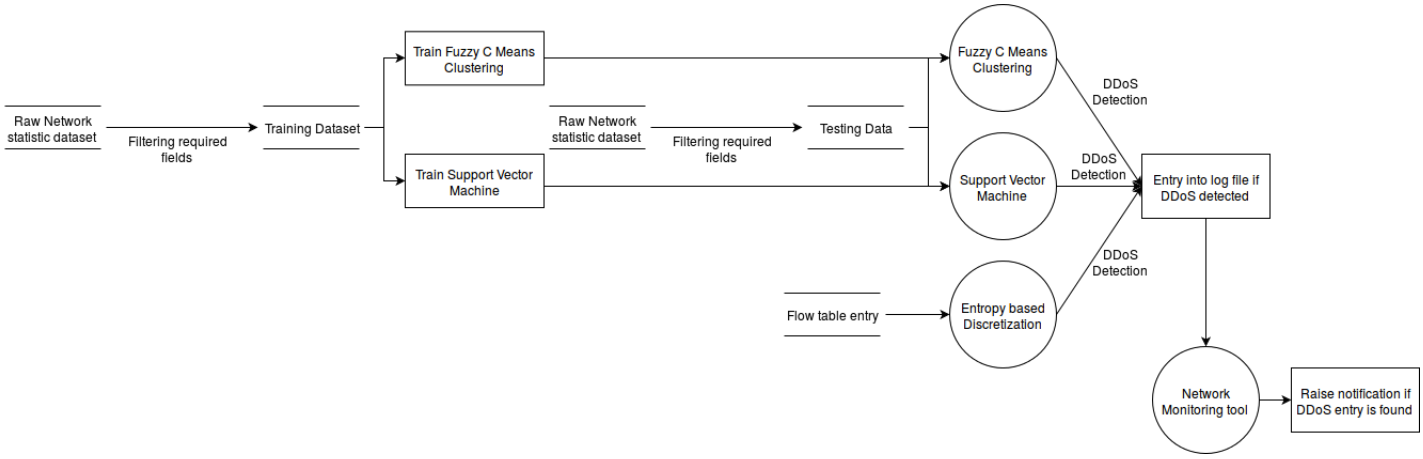


Figure 6.3: Level 1 Data Flow Diagram

6.5.2 Activity Diagram

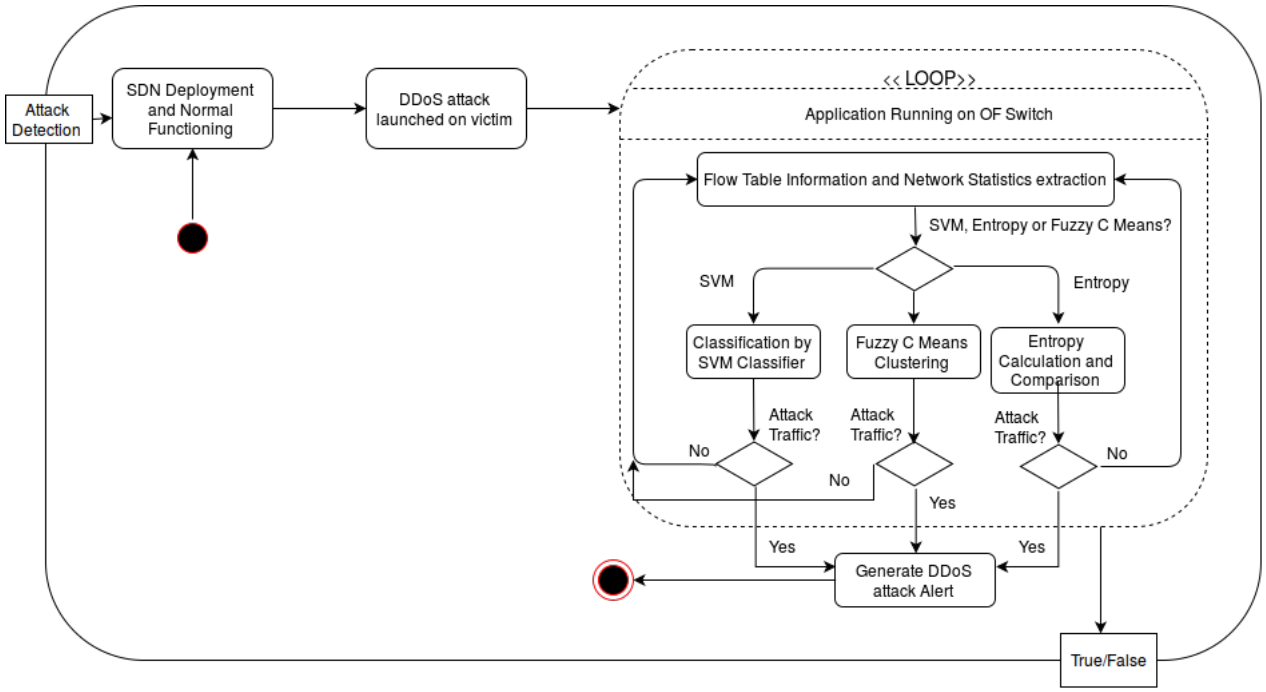


Figure 6.4: Activity Diagram

6.5.3 Non Functional Requirements

1. **Interface Requirements**

   (a) OpenFlow Protocol :
   Provides a medium for the SDN controller to direct traffic along the switches within the network.

   (b) Northbound API's :
   Allow the SDN controller to communicate between services and applications running over the network owing to programmatic nature of SDN.

   (c) Southbound API's :
   Allow the SDN controller to communicate between the network devices within the network.

2. **Software Quality Attributes**

   (a) Correctness:
   The correctness of our product would depend upon the accuracy with which either of the application would detect the attack. For this purpose both applications must be trained in the network which they would monitor to generate the necessary constant values needed for their execution.

   (b) Availability:
   As long as the network is up and running, this service would be also be available continuously monitoring the network for any anomalies.

(c) Usability:
As and when a DDoS attack is detected, a simple notification is sent to the network administrative team. It does not disrupt their normal work flow or involve using complex application for monitoring purposes. Hence no specific training of the user is needed for using this system.

(d) Portability:
Portability is restricted i.e. both applications must be given prior training before being deployed in a network. Thus the system being used in one enterprise cannot be used in another enterpise owing to the fact that the network traffic would vary from enterprise to enterprise such that normal traffic in one could be classified as attack traffic in another enterprise. Hence training in the network is of utmost importance.

6.5.4 State Diagram

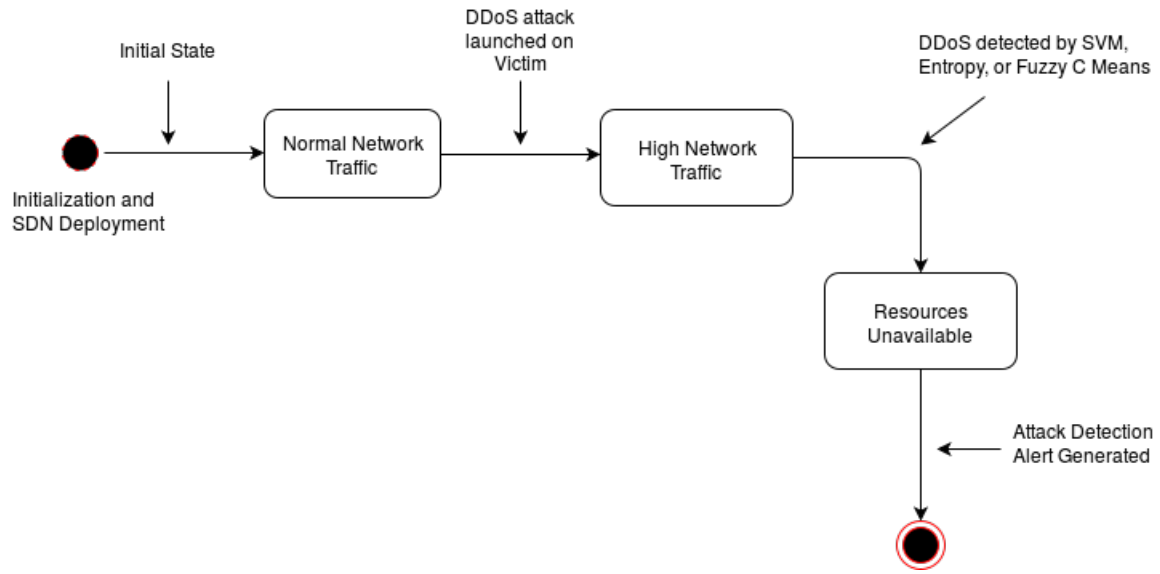


Figure 6.5: State Diagram

7.

Detailed Design Document

7.1 Architectural Design

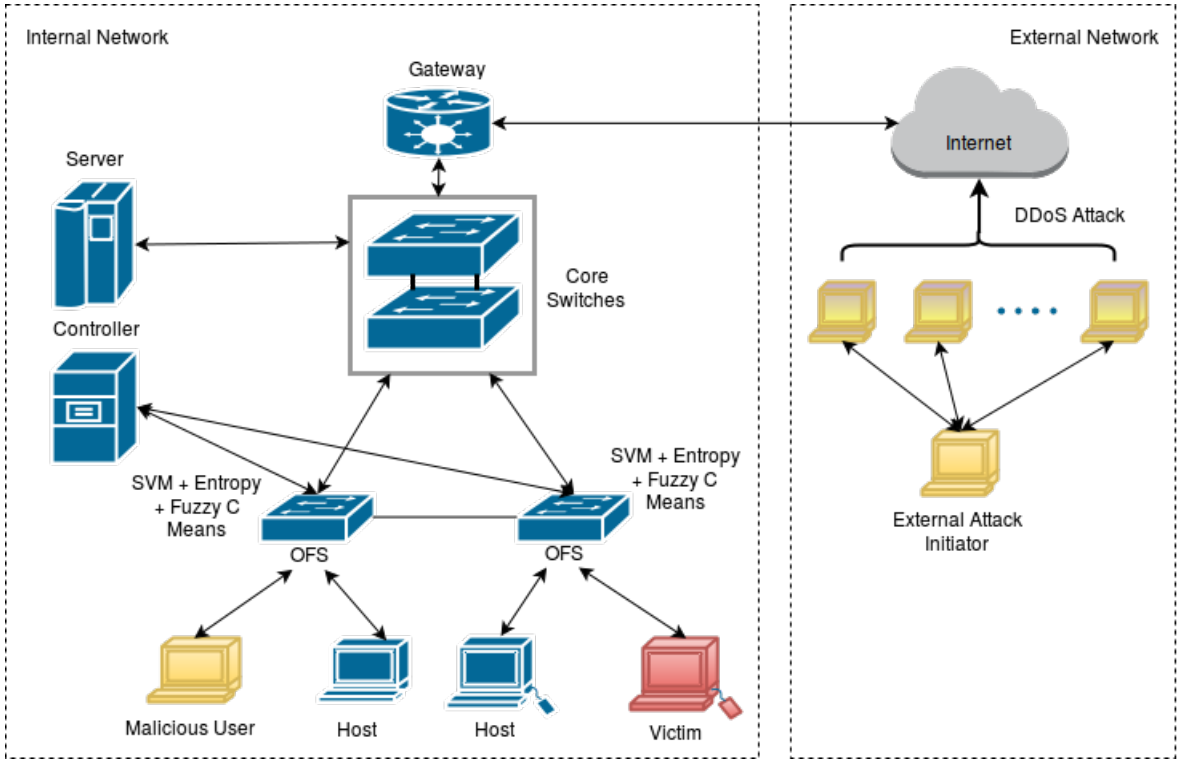


Figure 7.1: System Architecture

During normal state of communication between two users in the network, the first packet is transmitted from the host to it's nearby switch. Initially the switch does not have any entry in its flow table as to where it should forward the incoming packet, hence the default route is to forward the packet to the controller. The controller then dictates the switch to forward the packet to the respective destination port. Along with this the controller also makes an entry in the flow table of the switch for the particular source and destination such that the future packets with the same source and destination are forwarded directly without any intervention of the controller.

This helps in reducing the time taken for communication between source and destination systems. After a particular interval of time the flow table entries are flushed and the process is repeated all over again.

The system is designed to detect two types of DDoS attacks namely :-

1. Internal DDoS attack
2. External DDoS attack

In the first case, i.e. internal DDoS attack, the malicious user and the victim are located within the software defined network. whereas in external DDoS attack the host machine is from a different network. In both cases losses generated are immense.

The malicious machine can now launch attacks at two different points, one being the controller which would result in downfall of entire enterprise network incase there aren't any secondary controllers available, or the second point of attack could be one of host machines.

In case of the attack point being the controller, the attack is generated by spoofing the source and destination ip address so that it does not match any flow table entry and hence must make a trip to the controller, this would burden the controller and thereby render its services unavailable. To overcome this issue, SVM monitors the rate of incoming packets and classifies it as normal traffic or attack traffic, similarly Entropy based mechanism would find a drop in the entropy in the network owing to the large

number of same class of packets hence would be able to detect the DDoS attack, and Fuzzy C Means clustering would assign a membership value closer to ”1” near the core switch since all packets are being directed towards the controller and thus would be able to detect a DDoS attack.

Another method of generating an attack would be by generating a multitude of requests at the host thereby rendering the system void of performing any other operations. This type of attack is also solved along similar guidelines such that SVM would monitor the rate of incoming packets and Entropy based mechanism would monitor the entropy of the network which would drop due to large number of same class of packets, similarly Fuzzy C Means, running on the edge switches would be able to place the incoming packets in the attack cluster thereby detecting a DDoS attack.

Hence, a comparison can be deduced as to which method would generate a result instantaneously and with higher accuracy.

Once the attack is detected by any of the applications an entry is made into the log file of the respective switch, which is constantly monitored by the network monitoring tool which on finding the entry raises a notification to the network admnistrative team. Further it is the responsibility of the network administrative team to take preventive measures to eliminate any damage that could be caused by an impending full fledged DDoS attack.

7.2 Data Design

7.2.1 Dataset Description

For the purpose of detection, all three algorithms use some form of data on which computation is performed and results are predicted. In case of Support Vector Machine the dataset consists of the network statistics, which have been converted to a ”csv” file with limited fields using a terminal based tool ”tshark”.

The fields present in the dataset have been shown in table 2.1

Packet Interval	Source IP	Destination IP	Protocol
0.000300000	192.168.1.14	192.168.1.11	1

Table 7.1: Dataset Fields - Support Vector Machine

1. **Packet Interval :**
   Time interval between two incoming packets
2. **Source IP :**
   The source ip address of the sender machine.
3. **Destination IP :**
   The destination ip address of the receiver machine.
4. **Protocol**
   Protocol of the incoming packet.

This dataset is again categorized into attack traffic dataset and normal traffic dataset, which can be shown in table 2.2 and 2.3, wherein the most standout variation is the packet interval field between normal traffic and attack traffic.

Packet Interval	Source IP	Destination IP	Protocol
0.000300000	192.168.1.14	192.168.1.11	1
0.000100000	192.168.1.12	192.168.1.11	1
0.000200000	192.168.1.13	192.168.1.11	6

Table 7.2: Attack Dataset Fields - Support Vector Machine

Packet Interval	Source IP	Destination IP	Protocol
0.02000000	192.168.1.14	192.168.1.11	1
0.01000000	192.168.1.14	192.168.1.11	1
0.01000000	192.168.1.12	192.168.1.11	1

Table 7.3: Normal Dataset Fields - Support Vector Machine

Similar datasets are created during runtime using the packet capturing tool ”tshark” and the classification is carried out on the same.

In case of Fuzzy C Means clustering, the nature of datasets used are the same. The only difference is the number of fields used. For Fuzzy C Means only the packet interval field is used to generate clusters and assign membership values to each packet. Also like SVM, Fuzzy C Means uses two datasets i.e. attack traffic and normal traffic for training the model, using the packet capturing tool ”tshark”, again which is stored in a csv file format. The training dataset appears to be like the one showed in table 2.4 and 2.5

Packet Interval
0.000300000
0.000100000
0.000200000

Table 7.4: Attack Dataset Fields - Fuzzy C Means clustering

Packet Interval
0.02000000
0.01000000
0.01000000

Table 7.5: Normal Dataset Fields -Fuzzy C Means clustering

However, unlike SVM and Fuzzy C Means that use packet level statistics for classification and clustering, Entropy based discretization uses the flow table entries residing in the switch to perform its computation. The flow table entries are dumped into a file during each time interval, which is then looked up by the entropy based application to detect what is the entropy of the network and accordingly detect whether a DDoS attack has been launched or not.

Sample flow entry :
$cookie = 0x0, duration = 240.478s, table = 0, n\_packets = 21975265, n\_bytes = 2153575970, idle\_timeout = 60, idle\_age = 0, priority = 65535, icmp, in\_port = 2, vlan\_tci = 0x0000, dl\_src = 4a:06:64:e9:ef:f0, dl_dst = f2:57:bd:4c:19:28, nw\_src = 10.0.0.2, nw\_dst = 10.0.0.1, nw\_tos = 0, icmp\_type = 0, icmp\_code = 0actions = output : 1$

wherein the fields used during processing are:

1. *n_packets* **:**
   The total number of packets transitted by the switch having the same flow details.

2. *protocol* **:**
   The protocol of the incoming packets.

3. *nw_src* **:**
   Source IP address

4. *nw_dst* **:**
   Destination IP address

8.

Project Implementation

8.1 Introduction

There was always an option to implement the project by creating a Software Defined Network in a single system using network simulator called Mininet. But doing this would have decreased the quality of the project and our commitment that the solution can be used in real world deployment. In order to validate the effectiveness of the solution in real time environment we decided to deploy our own Software Defined Network on a small scale. This deployment was a challenge in itself as it is a new technology which is not well matured.

8.2 Methodologies

Algorithm 1 Entropy Based Discretization

function COLLECT_FLOW
    Collect the flows from switch and store it in a text file.
    Read this text file line by line and for every line do,
    Seperate $dest\_ip$ and no_of_packet
    Calculate total number of packets for every $dest\_ip$ in current interval.
end function
function CALC_ENTROPY'
    First, calculate the probabilily using, $P_i = \frac{X_i}{\sum_{i=1}^{N} X_i}$
        where,
        P_i =Probability of $i^{th} dest\_ip$
        X_i =Packet count on $i^{th} dest\_ip$
        N =Total number of $dest\_ip$
    Now, Calculate entropy of network using, $H(S_j) = -\sum_{i=1}^{N} p_i log p_i$
        where,
        $H(S_j)$ =Entropy of j^{th} switch
    Calculate the difference between above calculated entropy value and normal entropy value and stored it in $diff$
end function
function DDOS_DETECTION
    Compare the above calculated $diff$ with predefined $threshold$ value.
    if $diff > threshold$ **then**
        Increment $ddos\_detected\_count$
        if $ddos\_detected\_count > min\_ddos\_detected$ in perticular window **then**
            Generate alert of DDoS attack
        end if
    else
        Increment program counter.
    end if
end function

Algorithm 2 Support Vector Machine

```
function PREDICT(features, w, b)
    classification ← sign(dot(features, w)) + b)
    return classification
end function
function TRAIN(Data)
    optimized_sv ← dict()
    for y_i in data do
        for featureset in data[y_i] do
            for feature in featureset do
                all_data.append(feature)                    ▷ all_data is a list
            end for
        end for
    end for
    all_data ← None
    step_size ← [max(all_data) * 0.1]
    b_range_multiple, b_multiple ← 2, 5
    latest_optimum ← max(all_data) * 10
    range_value ← max(all_data) * b_range_multiple
    for step in step_size do
        w ← [latest_optimum, latest_optimum]
        optimized ← False
        while not optimized do
            for b in range(−range_value, range_value, step * b_multiple) do
                for transformation in transforms do
                    w_t ← w * transformation
                    found_option ← True
                    for i in data do
                        for x_i in data[i] do
                            y_i ← I
                            if not y_i * (dot(w_t, x_i) + b ≥ 1) then
                                found_option ← False
                            end if
                        end for
                    end for
                    if found_option then
                        normalize(optimized_sv[w_t]) ← [w_t, b]
                    end if
                end for
            end for
            if w[0] < 0 then
                optimized ← True
            else
                w ← w − step
            end if
        end while
        norms ← sort(optimized_sv)
        w ← optimized_sv[norms[0]][0]
        b ← optimized_sv[norms[0]][1]
        final_optimum ← optimized_sv[norms[0]][0]
    end for
end function
```

Algorithm 3 Fuzzy C Means Clustering

function CALCULATE_MEMBERSHIP($cluster\_count, centroids, dataset$)
    $membership[i,j] \leftarrow \frac{\frac{-2}{d_{ij}^{m-1}}}{\sum_{i=1}^{c} d_{ij}^{m-1}}$
        where,
        $m = Fuzzifier(default\ 2)$
        $c = cluster\ count$
        $i = centroid\ point, j = data\ point$
        $d = euclidean\ distance\ between\ i\ and\ j$
    return *membership*
end function
function CALCULATE_CENTROID($cluster\_count, centroids, dataset$)
    $centroid_i \leftarrow \frac{\sum_{j=1}^{n} u_{ij}^{m} X_{ij}}{\sum_{n}^{j=1} u_{ij}^{m}}$
        where,
        $m = Fuzzifier(default\ 2)$
        $c = cluster\ count$
        $u = membership$
        $i = centroid\ point, j = data\ point$
        $d = euclidean\ distance\ between\ i\ and\ j$
    $centroids = [centroid_1, \ldots, centroid_c]$
    return *centroids*
end function
function FUZZY($cluster\_count, centroids, training\_dataset$)
    Let, $normal\_centroids[]$, $attack\_centroids[]$ be 1-dimensional lists and,
    membership be a 2-dimensional list of order n x c
        where,
        n = number of data points
        c = number of centroids
    $membership\_df$ be a dataframe representing the 2-dimensional *membership* list
    $training\_data \leftarrow dataframe\ for\ training\ data$
    $membership \leftarrow calculate\_membership(cluster\_count, centroids, training\_data)$
    $centroid \leftarrow calculate\_centroid(cluster\_count, centroids, training\_data)$
    while True **do**
        $old\_membership \leftarrow membership$
        $old\_membership\_df \leftarrow membership\_df$
        $membership \leftarrow calculate\_membership(cluster\_count, centroids, training\_data)$
        if $old\_membership == membership$ **then**
            break
        end if
    end while
    Identify **attack clusters**.
    $min\_membership \leftarrow$ minimum membership in attack clutser.
    while True **do**
        Capture packets during runtime.
        $testing\_data \leftarrow dataframe\ for\ testing\ data$
        $membership \leftarrow calculate\_membership(cluster\_count, centroids, testing\_data)$
        $attack\_packets \leftarrow$ Total packets having membership value above $min\_membership$ value.
        if $attack\_packets \geq threshold * total\_packets$ **then**
            return DDoS
        end if
    end while
end function

9.

Software Testing

9.1 Type of Testing Used

- Unit Testing
- Integration Testing
- System Testing
- Positive Testing
- Negative Testing

9.2 Test Cases and Test Results

1. Unit Testing

Unit	Description	Scenario	Expected Output	Generated Output
SVM	Attack Traffic	Attack has occured	Alert generation	Alert Generated
SVM	Normal Network Traffic	SDN functioning in normal mode	No alert generation	Alert is not Generated
EBD	Attack Traffic	Attack has occured	Alert generation	Alert Generated
EBD	Normal Network Traffic	SDN functioning in normal mode	No alert generation	Alert is not Generated
FCM	Attack Traffic	Attack has occured	Alert generation	Alert Generated
FCM	Normal Network Traffic	SDN functioning in normal mode	No alert generation	Alert is not Generated

2. Integration Testing

Description	Scenario	Expected Output	Generated Output
All three algorithms are running simultaneously and attack traffic in the network	Attack has occured	Alert generation	Alert Generated
All three algorithms are running simultaneously and Normal traffic in the network	SDN functioning in normal mode	no alert generation	Alert is not Generated

3. System Testing

Description	**Scenario**	**Expected Output**	**Generated Output**
Time span between attack detection and alert generation	Attack has occured	Instantaneous alert generation	Alert generated within 1 sec.
Normal Network Traffic	SDN functioning in normal mode	No alert generation	Alert is not Generated

4. Positive Testing

Description	**Scenario**	**Expected Output**	**Generated Output**
Attack Traffic	Attack has occured	Alert generation	Alert Generated

5. Negative Testing

Description	**Scenario**	**Expected Output**	**Generated Output**
Normal Network Traffic	SDN functioning in normal mode	No alert generation	Alert is not Generated

10.

Results

10.1 Screenshots

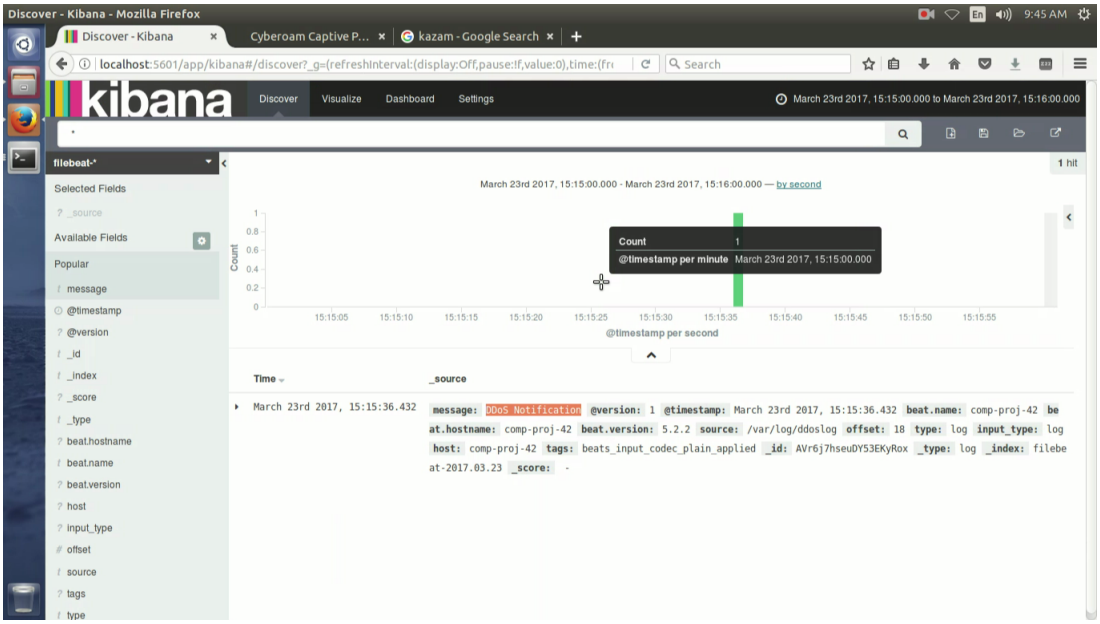


Figure 10.1: Kibana Dashboard

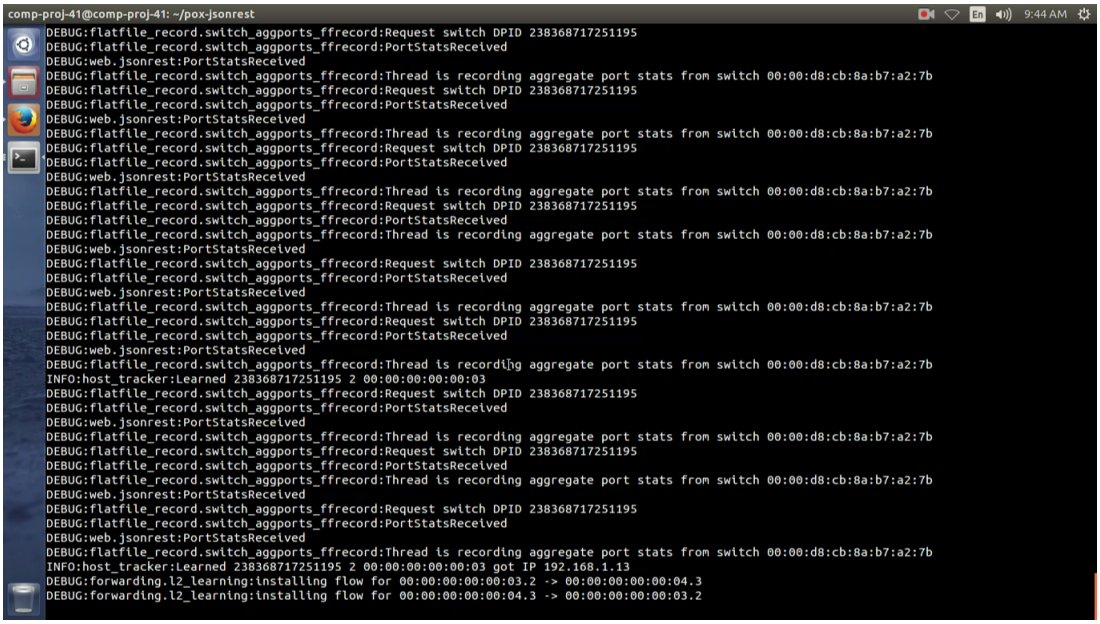


Figure 10.2: POX Controller Output

10.2 Output

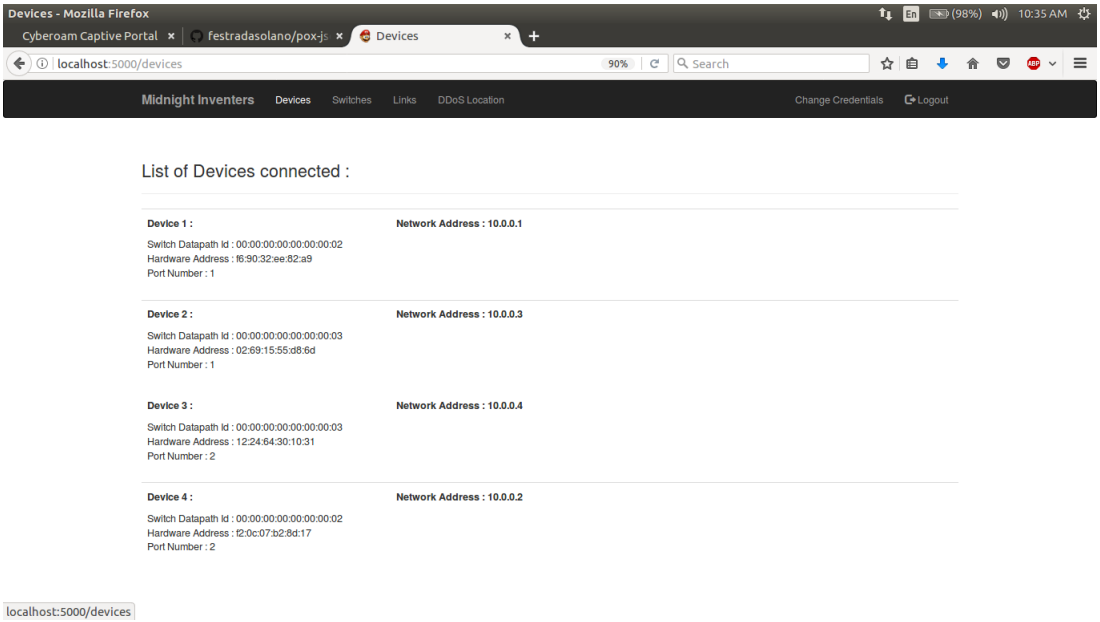


Figure 10.3: Flask - List of Devices

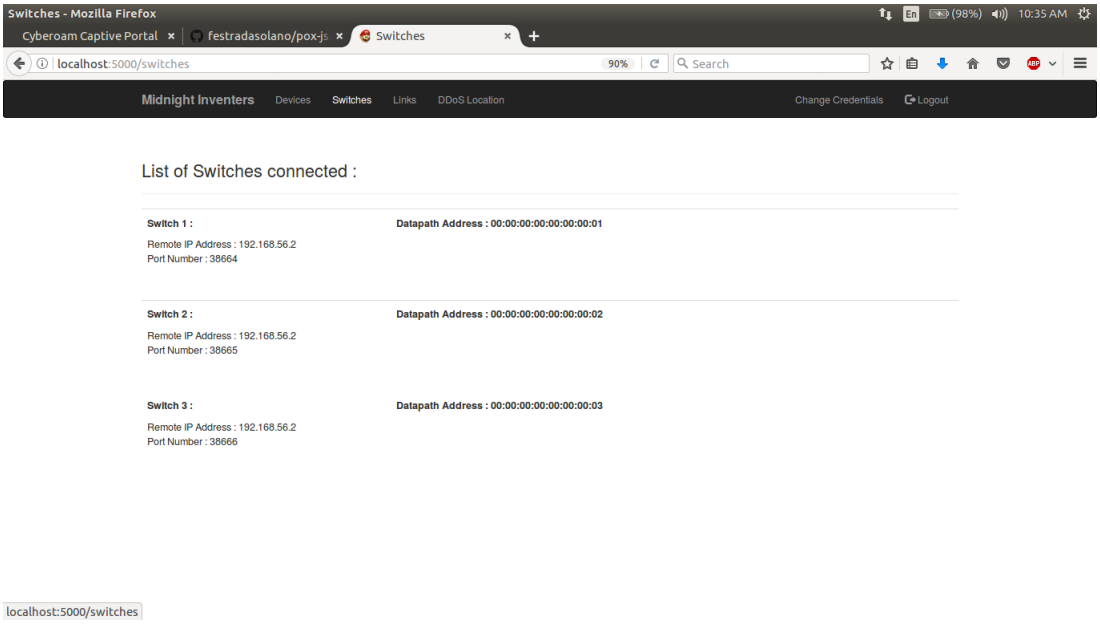


Figure 10.4: Flask - List of Switches

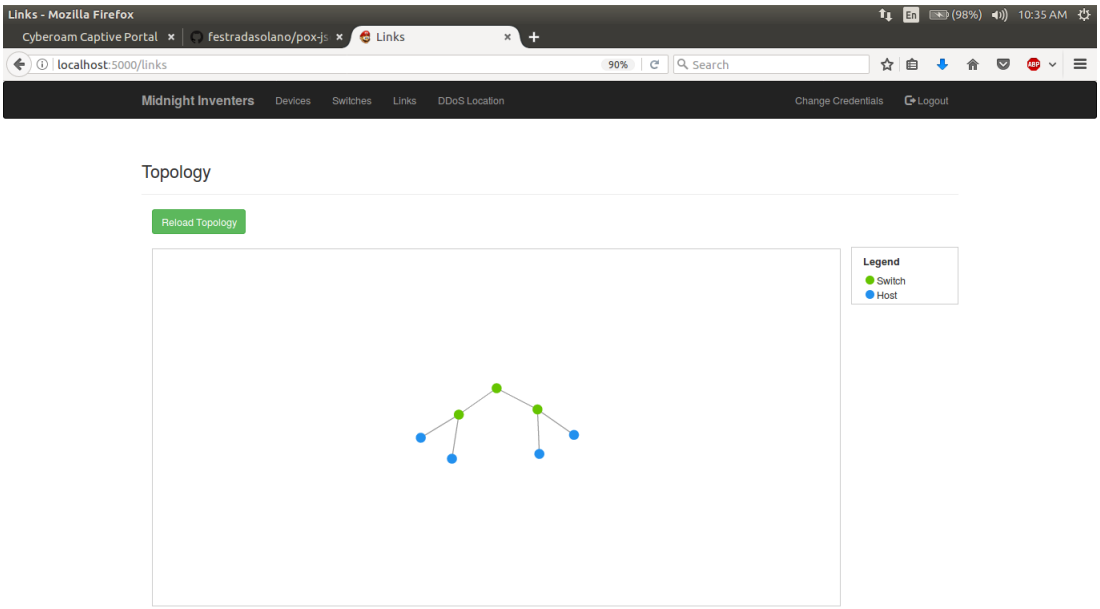


Figure 10.5: Flask - SDN Topology

```
achyuth@achyuth-Lenovo-Y50-70: ~/Desktop/.chintu/Untitled Folder/Dataset
The ratio is:  0.499338624339
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'

The ratio is:  0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'

The ratio is:  0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
160913
The ratio is:  0.000105647150945
----------------------DDoS Detected-------------------------
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
88289
The ratio is:  0.000113264393073
----------------------DDoS Detected-------------------------
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
86153
The ratio is:  0.00011607256857
----------------------DDoS Detected-------------------------
```

Figure 10.6: SVM Algorithm Output

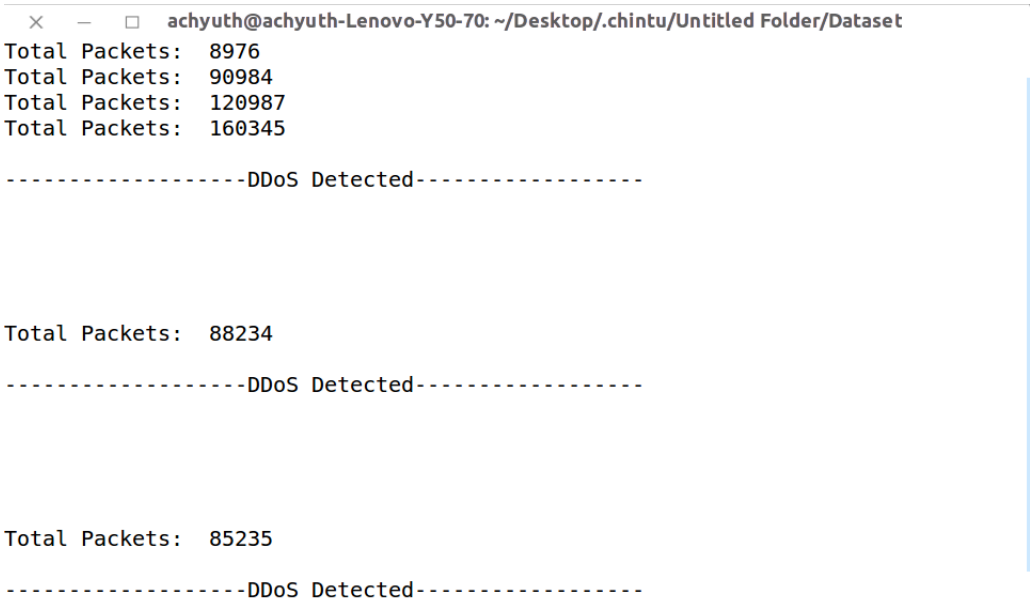


Figure 10.7: Entropy Based Discretization Algorithm Output

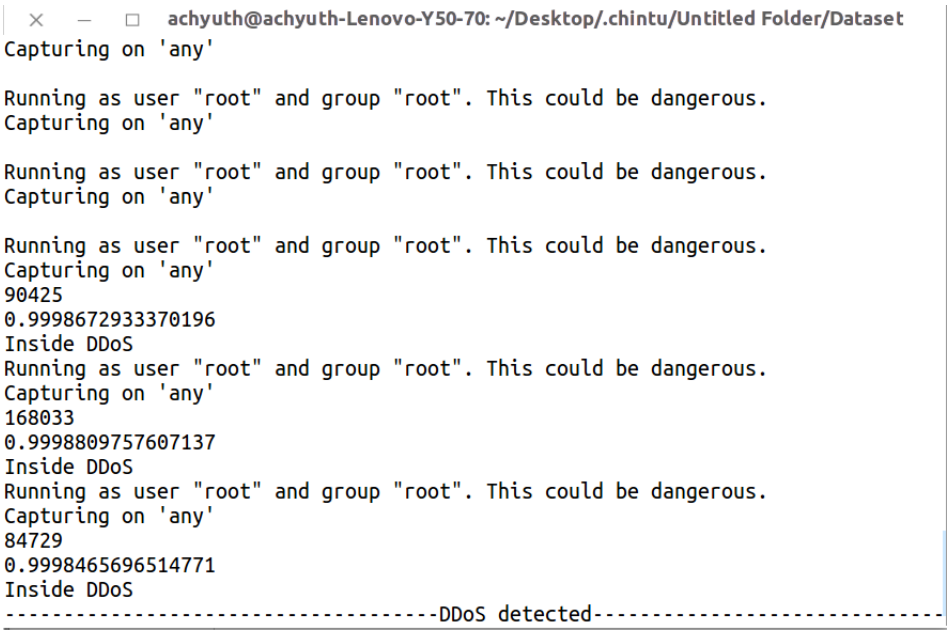


Figure 10.8: Fuzzy C-Means Algorithm Output

11.

Deployment and Maintenance

11.1 Installation and un-installation

11.1.1 Installation

Given the required hardware is present along with root priveleges (sudo) the following softwares can be installed. We illustrate the installation procedure considering Ubuntu 14.04 is installed, however any Linux Operating System will work well with little changes in the procedure.

According to the Software Requirements the installation procedure is as follows:

1. Oracle VirtualBox
   It can either be installed from the software center of Ubuntu or more advanced users can install it from the command line.
   Step 1: Go to the Ubuntu Software Center and search for VirtualBox
   Step 2: Click on Install and provide the password

2. Open vSwitch
   You can install it from the command line for Ubuntu 14.04. However if you prefer the latest version which is not updated in its repository, you can install from tar file

   $ sudo apt-get install openvswitch-switch

3. tshark
   If you simply need tshark then it can be installed using the following command. If you need Wireshark then tshark will be installed along with it by default.

   $ sudo apt-get install tshark

4. Ubuntu VM's
   After VirtualBox or any hypervisor is available you need to install 4 virtual machines ech having at least 1 logical core, 1.5 GB RAM and 10 GB minimum hard disk.

   For VirtualBox select the downloaed ISO of Ubuntu 14.04 along with the suitable VM specifications to install them.

5. Apache Web Server
   After the VM's have been installed install the Apache web server using simple command.

   $ sudo apt-get install apache2

6. Ostinato
   This is an open source packet generation tool which can be installed using

   $ sudo apt-get install ostinato

7. Elasticsearch

   $ sudo apt-get install elasticsearch

8. Logstash

   $ sudo apt-get install logstash

9. Kibana

   $ sudo apt-get install kibana

10. hsflowd
    Download the debian file for hsflowd from the official site. For any further queries read the documentation. This need to be installed on both the physical machines.

    $ sudo dpkg -i hsflowd.deb

11. sFlow-RT
    Download the sflow-rt application from the official site. You just need to run the main program of sflow-rt.

12. POX
    Download the POX controller from the GitHub repo which also includes the json-rest part integrated with it. You just need to enter the directory and run the controller using the pox.py program.

13. Python Flask
    You will need python pip for installing Flask. Install pip using

    $ sudo apt-get install python-pip
    $ sudo pip install Flask

14. Filebeat Agent
    Install the filebeat agent for tailing the log file and sending to the elasticsearch server

    $ sudo apt-get install filebeat

15. Watcher
    This is available as a plugin for elasticsearch. Refer elasticsearch documentation for installing watcher plugin.

11.1.2 Uninstallation

Remove the VM's from VirtualBox and free the used space For all other softwares a simple purge command will work well.

$ sudo apt-get purge –remove <name_of_software>

12.

Conclusion and Future Scope

The report introduces the problems faced in the Software Defined Networking Environment such as DDoS, application layer attack, brute force and phishing. As the SDN Controller seperates the Application layer and the Data flow plane in the network, it becomes a point of vulnerability for the attackers to target by using such attacks.

DDoS attack when targetted on the controller can pose a huge impact on the whole network compromizing the system responsible for the Business logic. Thus it is very important to use a mechanism which can detect and mitigate such anomaly based attacks, we focus on the prior and not the later part of the mechanism.

There are several methods which can be used for the detection of DDoS such as use of neural networks, chaotic systems, filtering algorithms etc. Considering the advantages of Support Vector Machines, Entropy Based Discretization and Fuzzy C Means Clustering algorithms, we will be using them.

We then procede by stating the product features,user characteristics, system features, interfaces and quality attributes like correctness,usability etc. We also provide necessary diagramatic representations like the sequence diagram, use case diagram etc.

We then give a description of the system components, system architecture as well as the functioning of the whole network environment. We also specify two cases of the DDoS attack scenarios namely, internal and external attacks and the method in which we detect the attack using the proposed algorithmic strategies. After the detection of the attack, the network administrator will be notified by the mechanism and a graphical interface would be provided for the administrator for finding the point of attack.

The current implementation of our project does not include the mitigation of DDoS attacks due to the chances of false positives being generated, this would result in blocking out legitimate users which will seriously impact the organization. So, the future scope of our project includes developing a module for mitigation, increasing the accuracy of the system to remove false positives.

13.

Bibliography

[1] Diego Kreutz, Fernando M. V. Ramos, Paulo Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig, "Software-Defined Networking: A Comprehensive Survey", Version 2.01, 8 Oct 2014

[2] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks", ONF white paper, April 13, 2012

[3] Shiva Rowshanrad, Sahar Namvarasl, Vajihe Abdi, Maryam Hajizadeh, Manijeh Keshtgary, "A survey on SDN, the future of networking", in Journal of Advanced Computer Science and Technology, pg 232-248, 2014

[4] Rishikesh Sahay, Gregory Blanc, Zonghua Zhang, Herve Debar, "Towards Autonomic DDoS Mitigation using Software Defined Networking", Feb 2015, http://www.internetsociety.org/doc/towards-autonomic-DDoS-mitigation-using-software-defined-networking

[5] Seyed Mohammad Mousavi, "Early Detection of DDoS Attacks in Software Defined Networks Controller", 2014

[6] Shunsuke Oshima, Takuo Nakashima, Toshinori Sueyoshi, "Early DoS/DDoS Detection Method using Short-term Statistics", Presented at International Conference on Complex, Intelligent and Software Intensive Systems, 2010

[7] Kokila RT, S.Thamarai Selvi, Kannan Govindarajan, "DDoS Detection and Analysis in SDN-based Environment Using Support Vector Machine Classifier", presented at Sixth International Conference on Advanced Computing(ICoAC), 2014,

[8] Xue Li, Dongming Yuan, Hefei Hu, Jing Ran, Shulan Li, "DDoS detection in SDN switches using support vector machine classifier", in Joint International Mechanical, Electronic and Information Technology Conference, 2015

[9] Rui Wang, Zhiping Jia, Lei Ju, "An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking", in IEEE Trustcom/BigDataSE/ISPA, DOI 10.1109/Trustcom.2015.389, 2015

[10] T.Subbulakshmi, Dr. S. Mercy Shalinie, V.GanapathiSubramanian, K.BalaKrishnan, D. Anand-Kumar, K.Kannathal, "Detection of DDoS Attacks using Enhanced Support Vector Machines with Real Time Generated Dataset", in IEEE-ICoAC, 2011

[11] I Gde Dharma N., M. Fiqri Muthohar, Alvin Prayuda J. D., Priagung K., Deokjai Choi, "Time-based DDoS Detection and Mitigation for SDN Controller", Presented at APNOMS, 2015

[12] Shibo Luo, Jun Wu, Jianhua Li, "A Defense Mechanism for Distributed Denial of Service Attack in Software-Defined Networks", Presented at Ninth International Conference on Frontier of Computer Science and Technology, 2015

[13] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments", 2013, http://dx.doi.org/10.1016/j.bjp.2013.10.014

[14] Seyed Mohammad Mousavi, Marc St-Hilaire, "Early Detection of DDoS Attacks against SDN Controllers", Presented at International Conference on Computing, Networking and Communications, Communications and Information Security Symposium, 2015

[15] P. K. Agrawal, B. B. Gupta, Satbir Jain, "SVM Based scheme for Predicting Number of Zombies in a DDoS Attack", Presented at European Intelligence and Security Informatics Conference, 2011

[16] A. Ramamoorthi, T. Subbulakshmi, Dr. S. Mercy Shalinie, "Real Time Detection and Classification of DDoS Attacks using Enhanced SVM with String Kernels", in IEEE-International Conference on Recent Trends in Information Technology, ICRTIT 2011, June 2011

[17] Ingo Steinwart, Andreas Christmann, "Support Vector Machines", Springer, 2008

[18] Alexander Gelbukh, Flix Castro Espinoza, Sofa N. Galicia-Haro, "Nature-Inspired Computation and Machine Learning", Springer, 2014

[19] Vapnik, Vladimir, "The Nature of Statistical Learning Theory", Springer, 2000

[20] Jose Valente de Oliveira, Witold Pedrycz, "Advances in Fuzzy Clustering an its Applcations", Wiley, 2007

[21] Sindhu Arumugam, Dr. V Vanitha, Ms. V.P. Sumathi, "Detection of BotNet using Fuzzy C-Means Clustering by Analysing the network traffic", International Journal of Scientific & Engineering Research, Vol. 6 Issue 4, April 2015

[22] Disha Sharma, "Fuzzy Clustering as an Intrusion Detection Technique", International Journal of Computer Science & Communication Network, Vol. 1(1), September-October 2011

[23] Jerzy W. Grzymala Busse, "Discretization Based on Entropy & Multiple Scanning", Entropy, 2013.

[24] Claudio Rebelo De Sa, Carlos Soares, Arno Knobbe , "Entropy Based Discretization methods for ranking data", Elsevier Vol. 329.

[25] Jiawei Han, Micheline Kamber, Jian Pei, "Data Mining: Conepts and Techniques 3rd ed.", Morgan Kaufmann, July 2011.

[26] Kevin Meurer, "A Simple Guide to Entropy-Based Discretization", URL: http://kevinmeurer.com/a-simple-guide-to-entropy-based-discretization/

14.

Annexure A

14.1 IDEA Matrix

Table 14.1: IDEA Matrix

I	D	E	A
Innovative	Detect	Evaluate	Associate
Increase	Decrease	Effective	Acquire
Investment	Depict	Enhance	Analyse

Innovative	Up till now, DDoS detection systems uses a single mechanism for its working. Here, the system will be using two mechanisms i.e. support vector machine, fuzzy c means clustering and entropy based discretization simultaneously which is **innovative**.
Increase	As the system will be using support vector machine, fuzzy c means clustering and entropy based discretization, efficiency of the system will **increase**.
Investment	As the system contains Software components only. So, **zero investment** is required to implement the system in already deployed software defined network.

Detect	The system will **detect** DDoS attack and raise notification to the network administrator team.
Decrease	The DDoS detection algorithm will **decrease** the possibility of false positive attack detection.
Depict	The system will **depict** appropriately whether the network traffic is normal or attack traffic.

Evaluate	An important part of the implementation is the **evaluation** of the rate of flow of packets per unit time.
Effective	The system analyses the traffic and detects the attack at runtime thus, increasing the **effectiveness**.
Enhance	The accuracy of the system will be **enhanced** by using an environment - specific training dataset for SVM and Fuzzy C Means.

Associate	The project is a step towards **association** and integration of next generation networking with advanced computing concepts like Machine Learning and Data Mining.
Acquire	The flow table containing the flow entries along with the information of the count of the packets is **acquired** from the SDN switches.
Analyse	The system will **analyse** the data acquired from the switches to perform the intended task of detecting the DDoS attack.

14.2 Mathematical Model

14.2.1 Mathematical Model

$$S = \{\{I\},\{P\},\{O\}\} \tag{14.1}$$

where,
$I =$ Input set,
$P = \{P_{EBD}, P_{SVM}, P_{FCM}\}$ Processing set,
$O =$ Output set

14.2.2 Input:

$$I = \{N\} \tag{14.2}$$

where,
$N =$ { Network Statistics },
$F = \{F_i \mid F_i \in T, \forall i\ F_i =$ Individual entry $\}$,
$T =$ { Flow Table },
$F \subseteq N$

14.2.3 Input for EBD:

$$I_{EBD} = \{U_i \mid U_i = (\text{ Source Address, Destination Address, Port no., Count}),\ U_i \subset F_i\} \tag{14.3}$$

14.2.4 Input for SVM:

$$I_{SVM} = \{V_i \mid V_i = (\text{ Source Address, Destination Address, Port no., Time, Protocol, Count}),\ V_i \subset F_i\} \tag{14.4}$$

14.2.5 Input for FCM:

$$I_{FCM} = \{V_i \mid V_i = (\text{Time}),\ V_i \subset F_i\} \tag{14.5}$$

14.2.6 Processing:

P_{EBD} (I_{EBD}, O_{EBD})
{

1. $$P_i = \frac{C_i}{N} \tag{14.6}$$

   where,
   $c =$ counter,
   $N = \sum_{i=0}^{n} C_i$,
   $n =$ no. of entries.

2. $$\varepsilon = \sum_{i=0}^{n} -P_i log P_i \tag{14.7}$$

3.

$$(\lambda < \varepsilon) \rightarrow (\beta = 0) \tag{14.8}$$

$$(\lambda > \varepsilon) \rightarrow (\beta = 1) \tag{14.9}$$

4.

$$O_{EBD} = \{\beta \mid \beta \in (0,1)\} \tag{14.10}$$

}

P_{SVM} (I_{SVM}, O_{SVM})
{ Here,
γ , c, x, x_1, x_2, b and $\overline{\omega}$ are constants.
These values are calculated from Training Phase.
Training of SVM is based on dataset used.

1.

$$RBF = e^{-\gamma(|x_1 - x_2|)+c} \tag{14.11}$$

where,
$\gamma =$ width of boundries of hyperplane,
$c =$ soft margin parameter,
$x_1, x_2 =$ support vectors.

2.

$$y = \overline{\omega} * x + b \tag{14.12}$$

where,
$x =$ support vector,
$b =$ boundary width,
$\overline{\omega} =$ libsvm.w().

3.

$$(y \leq -1) \rightarrow (\alpha = 1) \tag{14.13}$$

$$(y \geq 1) \rightarrow (\alpha = 0) \tag{14.14}$$

4.

$$O_{SVM} = \{\alpha \mid \alpha \in (0,1)\} \tag{14.15}$$

}

$P_{FCM}\ (I_{FCM}, O_{FCM})$
{ Here,
m is a constant.
This value is used for the calculation of the membership values of the data points.

1.

$$u_{ij} = \frac{\frac{-2}{d_{ij}^{m-1}}}{\sum_{i=1}^{c} d_{ij}^{m-1}} \tag{14.16}$$

where,
d_{ij}^{m-1} = distance of the i^{th} point from the j^{th} centroid
c = centroid.

2.

$$C_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} X_{ij}}{\sum_{n}^{j=1} u_{ij}^{m}} \tag{14.17}$$

where,
u_{ij}^{m} = membership value of the data point.
X_{ij} = data point.

3.

$$(X \geq \lambda) \rightarrow (w = 1) \tag{14.18}$$

$$(X < \lambda) \rightarrow (w = 0) \tag{14.19}$$

4.

$$O_{FCM} = \{w \mid w \in (0,1)\} \tag{14.20}$$

}

14.2.7 Output:

$$O_{EBD} = \{z \mid \exists z, z \in \beta\} \tag{14.21}$$

$$O_{SVM} = \{x \mid \exists x, x \in \alpha\} \tag{14.22}$$

$$O_{FCM} = \{w \mid \exists w, w \in \gamma\} \tag{14.23}$$

$$O_{system} = \{O_{EBD} \cup O_{SVM} \cup O_{FCM}\} \tag{14.24}$$

14.2.8 Feasibility Study

To check the feasibility of Entropy Based Discretization, Support Vector Machine abd Fuzzy C-Means Clustering:
Since, the problem we solve is essentially a decision problem, it is very essential to analyse the running time of the system. The strategies used in the system are SVM, Entropy Based Discretization and Fuzzy C-Means clustering. It is well known that EBD takes O(n) running time thus is in P.

Now, we will find the computational complexity for Support Vector Machines:
For a training dataset D=$\{x_i,y_i\}$ where, $x_i \in \Re$ and $y_i \in \{+1,-1\}$.
The Support Vector Machine classifies with seperating hyperplane given by:

$$D(x) = w^T x + b \tag{14.25}$$

The above equation is obtained by:

$$L = \frac{1}{2}w^2 - \sum \zeta_i \tag{14.26}$$

s.t. $y_i(w^T\ x_i + b) \geq 1 - \zeta_i$, $\zeta_i > 0$ for i=1,2,...,m

The problem is equivalent to the problem:

$$MaxL(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j k(x_i, y_j) \tag{14.27}$$

where k(x_1,y_1) is the Mercer's Kernel which is used for the Dimensional reduction.

We will use the following four steps to prove the complexity of the SVM problem:

1. Denote the problem as π and justify that it is in NP.
2. Selecting a known problem π' which is in NP.
3. Construct a tranformation function from π' to π.
4. Prove that the transformation function is polynomial.

- **The SVM Problem (π)**
  For a set of kernels A=$\{k_1, k_2, ..., k_n\}$ the objective function can be written as:

$$C(A') = \Big[\sum_{k \in A'} \Big(\frac{1}{q}\Big[\sum_{i=1}^{m} \alpha_i - \frac{1}{2q^2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k(x_i, x_j)\alpha_i \alpha_j\Big]\Big)^2 - B\Big]^2 \tag{14.28}$$

  Where, we have to find a subset A' $\subseteq$ A of size q to minimize the cost. The cost function is used for reaching a bound B using set of Kernels, dataset and the Support Vectors.

- **SVM is in NP**
  To justify that SVM is in NP, we have to show that any instance of the problem can be guessed and verified to solve or not in polynomial time.
  Let there be a function, that accepts q which returns random indices of q from the set of kernels and computes the most optimum solution A'. As the set of Lagrange Multipliers, dataset, q and Bound is known beforehand, the kernel evaluations are the only states. These kernel evaluations are merely dot products among the directional vectors. The evaluation is polynomial and thus the instances A' can easily be generated and verified. Hence, we can say that SVM is in NP.

- **Selecting a known NP-Complete Problem (π')**
  The known problem selected is the subset problem which is as defined,
  Given a set A, a size s(a) for each a $\in$ A and an integer B. Is there an A' $\subseteq$ A such that the sum of the sizes of the elements in is exactly B?

- **Transformation function $\mathscr{F}$ from π' to π**
  The transformation function $\mathscr{F}$ should map every instance (I) such that every instance of "yes" in π should map to the corresponding instance (I') in π'.
  First, A stands for set of kernels: A=$\{A_1, A_2, ..., A_n\}$
  To compute B the following procedure should be followed:

  1. Solve SVM for all k $\in$ A with dataset D.
  2. Use the kernel k^* and the dataset D to find the most optimal solution for SVM denoted by $M^* = (V^*, k^*)$
  3. Now, B can be calculated by:-

$$B = \Big[\sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k^*(x_i, x_j)\alpha_i, \alpha_j\Big]^2 \tag{14.29}$$

  4. Finally we have to calculate the size function s(a)=s(k $\in$ A) which can be done by the formula:-

$$s(k) = \Big[\big(\frac{1}{q}\sum_{i=1}^{m} \alpha_i - \frac{1}{2q^2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k(x_i, x_j)\alpha_i\alpha_j\big)\Big]^2 \tag{14.30}$$

     Where, s(k) is the value of the objective function.

  Now, we know that $\sum_{k\in A} s(k) = B$ when $k^*(x_i, x_j) = \frac{1}{q^2}\sum_{k\in A'}, (k_i, k_j)$
  This can be written in the form:

$$\Big[\sum_{k\in A'}\big(\frac{1}{q}\sum_{i=1}^{m} \alpha_i - \frac{1}{2q^2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k(x_i, x_j)\alpha_i\alpha_j\big)\Big]^2 = \Big[\sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k^*(x_i, x_j)\alpha_i\alpha_j\Big]^2 \tag{14.31}$$

  Taking squared-root and using summation property,

$$\sum_{k\in A'}\big(\frac{1}{q}\sum_{i=1}^{m} \alpha_i\big) + \sum_{k\in A'}\Big[-\frac{1}{2q^2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k(x_i, x_j)\alpha_i\alpha_j\Big] = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j k^*(x_i, x_j)\alpha_i\alpha_j \tag{14.32}$$

Simplifying the equation,

$$\sum_{k\in A'}\Big[\frac{1}{q^2}\sum_{i=1}^{m}\sum_{j=1}^{m}y_i y_j k(x_i,x_j)\alpha_i\alpha_j\Big]=\sum_{i=1}^{m}\sum_{j=1}^{m}y_i y_j k^*(x_i,x_j)\alpha_i\alpha_j \tag{14.33}$$

Substituting $\Delta_{i,j} = y_i y_j \alpha_i \alpha_j$ for simplification purpose

$$\sum_{k\in A'}\Big[\frac{1}{q^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\Delta_{i,j}k(x_i,x_j)\Big]=\sum_{i=1}^{m}\sum_{j=1}^{m}\Delta_{i,j}k^*(x_i,x_j) \tag{14.34}$$

Cancelling common terms,

$$\frac{1}{q^2}\sum_{k\in A'}k(x_i,x_j)=k^*(x_i,x_j) \tag{14.35}$$

Thus,

$$\sum_{k\in A'}s(k)=B\leftrightarrow k^*(x_i,x_j)=\frac{1}{q^2}\sum_{k\in A'}k(x_i,x_j) \tag{14.36}$$

This implies that the transformed subset problem is equivalent to SVM classification using an optimum kernel. Thus, for every instance of *(I')* if the answer is "yes" the answer can be mapped for the instance of *(I)*.

- **$\mathscr{F}$ is a polynomial Transformation**
The last part is to prove that the $\mathscr{F}$ is polynomial. The most expensive step is to find the Boundary (B) in the training phase.
This problem of training is a quadratic problem(QP). This problem can be seen as a convex problem. There are methods like Ellipsoids to solve it in polynomial time. It is also known that QP can be solved in O(m^3), for SVM "m" is the number of input vectors. Therefore, $\mathscr{F}$ is a polynomial transformation.

Now, we have to analyze the Fuzzy C-Means clustering.
the following are: the steps of the running of the FCM algorithm:

1. Initializing the centroids for the cluster.
2. Calculate the membership values of the data points for the initialized clusters.
3. recalculate the centroids of the cluster.
4. recalculate the membership values for every data point until the centroid values and the membership values do not change.

The calculation of the membership values take place using the following equation:-

$$u_{ij}=\frac{\frac{-2}{d_{ij}^{m-1}}}{\sum_{i=1}^{c}d_{ij}^{m-1}} \tag{14.37}$$

And, the calculation of the centroid is done by the following equation:-

$$C_i=\frac{\sum_{j=1}^{n}u_{ij}^{m}X_{ij}}{\sum_{n}^{j=1}u_{ij}^{m}} \tag{14.38}$$

1. Fuzzy C-Means is a generalization of Hard C-Means or K-Means clustering algorithm. The clustering based algorithms are well known to fall under the NP-Hard type of problems.
2. This is because, the clustering algorithms provide a global solution which causes them not to converge even for a large number of iterations. Thus, we have to limit the number of iterations that can take place.

3. Also, there is no linear relationship between the point of convergence and the number of iterations. Therefore, we can not be certain that for a particular number of iterations the clustering will converge. This is analogous to the Halting probelm in the Turing machine which is well known to be a NP-Hard problem.
4. The time complexity of FCM is $O(ndc^2i)$ where,
   n = Number of data points.
   c = Number of clusters.
   d = Number of dimensions.
   i = Number of iterations.

Thus,
SVM is NP-Complete, Entropy based mechanism is Polynomial and Fuzzy C-Means clustering is NP-Hard.
By Cook's Theorem, P $\in$ NP therefore, the proposed solution is NP-Hard.

15.

Annexure B

15.1 Distributed Computation

The controller in an Software Defined Network is the centralized entity, which is responsible for the interfacing between Application layer and the underlying network hardware. Since, the controller imposes the application logic onto the datapath of the SDN. This requires a lot of computation.

Furthermore, during the scenario of a DDoS attack there is an excruciating load on the network and especially the SDN controller. Thus, is unwise to run an application on top of the controller as it can cause an overhead on the system. We exploit the dormant processing power of the SDN switches to our advantage by using them as computational nodes. Our DDoS detection mechanism will work on the switches distributed throughout the network.

Each switch will scan its underlying traffic. The advantages of this are twofold firstly, there will not be an overhead on the remaining network and secondly, it will be easier for the network administrator for finding the origin of the DDoS attack vastly reducing the cost and effort for the recovery.

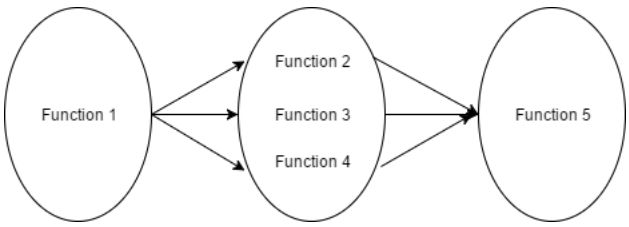
Thus, we divide the whole network traffic statistics into several concurrent switches for the computation making it a divide and conquer strategy.

15.2 Functional Dependancies

There are mainly Five Functions in the system namely,

1. A function which will continuously send the formatted traffic statistics which can be accepted by the Support Vector Machine, Entropy Based Descretization mechanism and Fuzzy C-Means Clustering.
2. A Function for the Support Vector Machine classifier which will classify the traffic into normal or attack traffic and update the log files if attack traffic is detected.
3. A Function for the Entropy Based Discretization Mechanism which will compare the flow of traffic with the threshold value and update the log files if the rate of flow of packets exceeds the threshold.
4. A Function for the Fuzzy C-Means Clustering algorithm which will cluster the incomming packets time difference betwwen two consecutive packets.
5. A Function which will monitor the log files to check whether there is an attack scenario or not.

The algorithms depend on the real-time input of the Network statistics, the function responsible for monitoring is dependent on the SVM, Entropy mechanism and Fuzzy Clustering. Thus, there is a one-many-one mapping between this function.



15.3 UML Diagrams

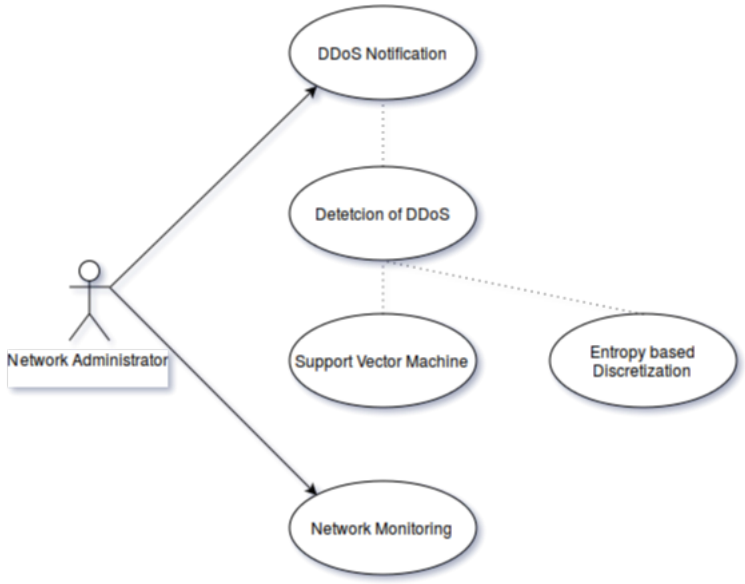


Figure 15.1: Use Case Diagram

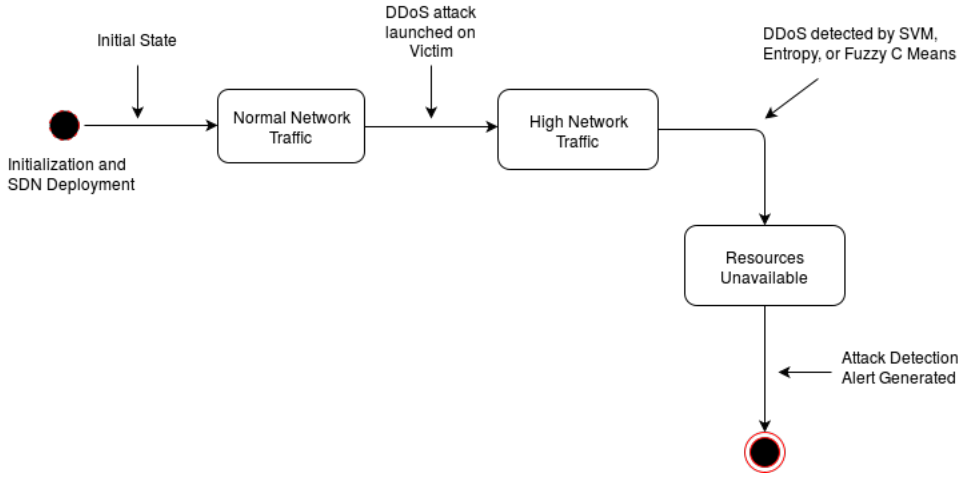


Figure 15.2: State Diagram

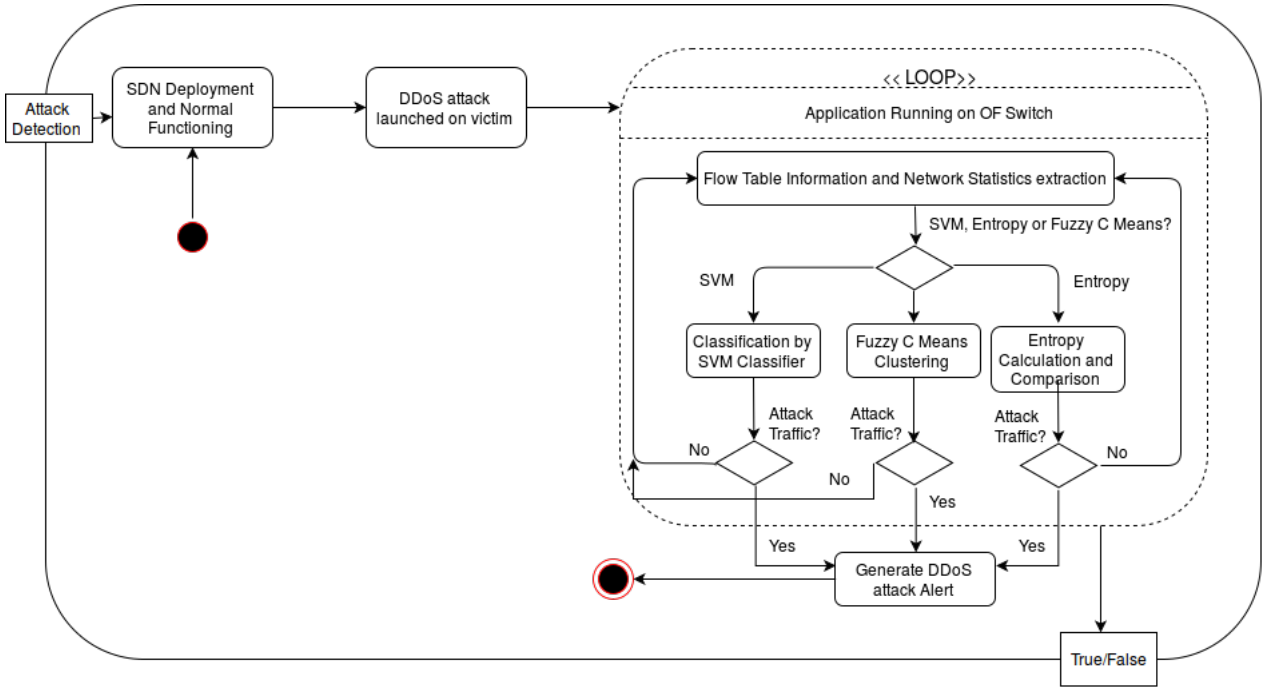


Figure 15.3: Activity Diagram

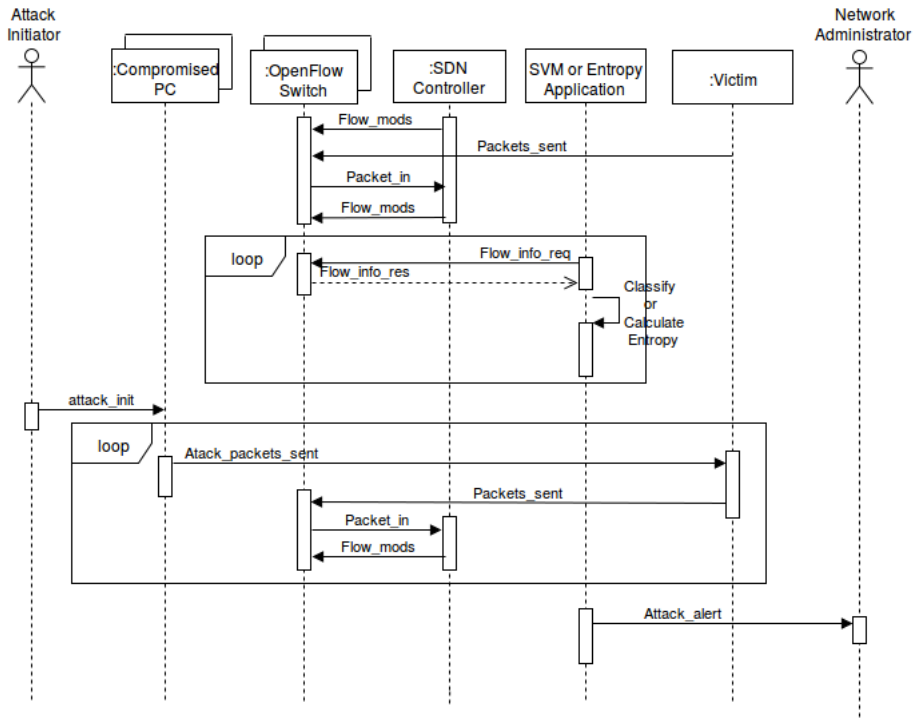


Figure 15.4: Sequence Diagram

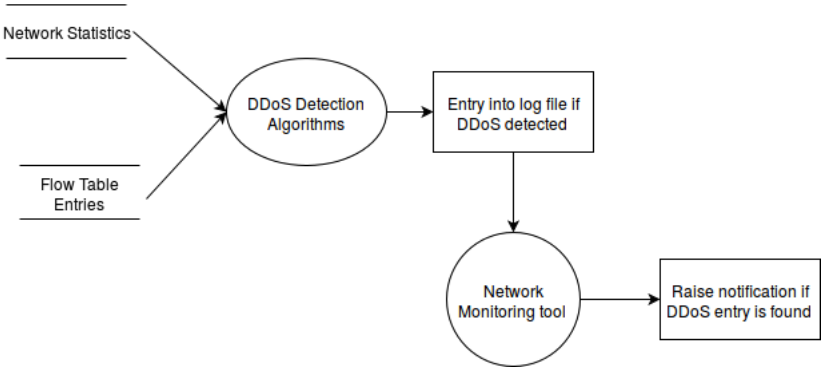


Figure 15.5: Dataflow Level 0 Diagram

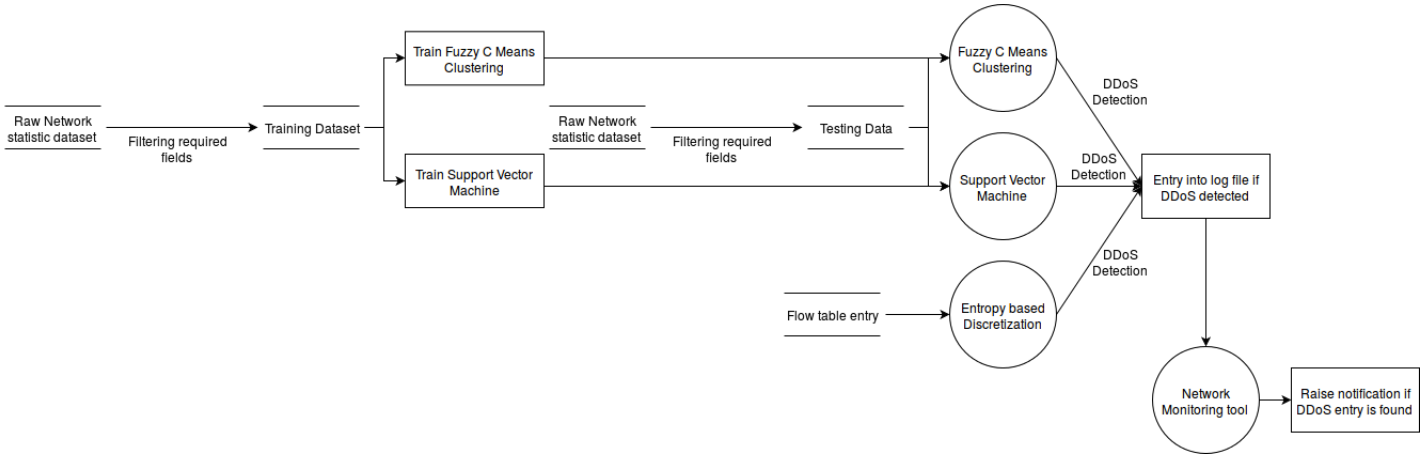


Figure 15.6: Dataflow Level 1 Diagram

16.

Annexure C

16.1 Project Planner - Figure

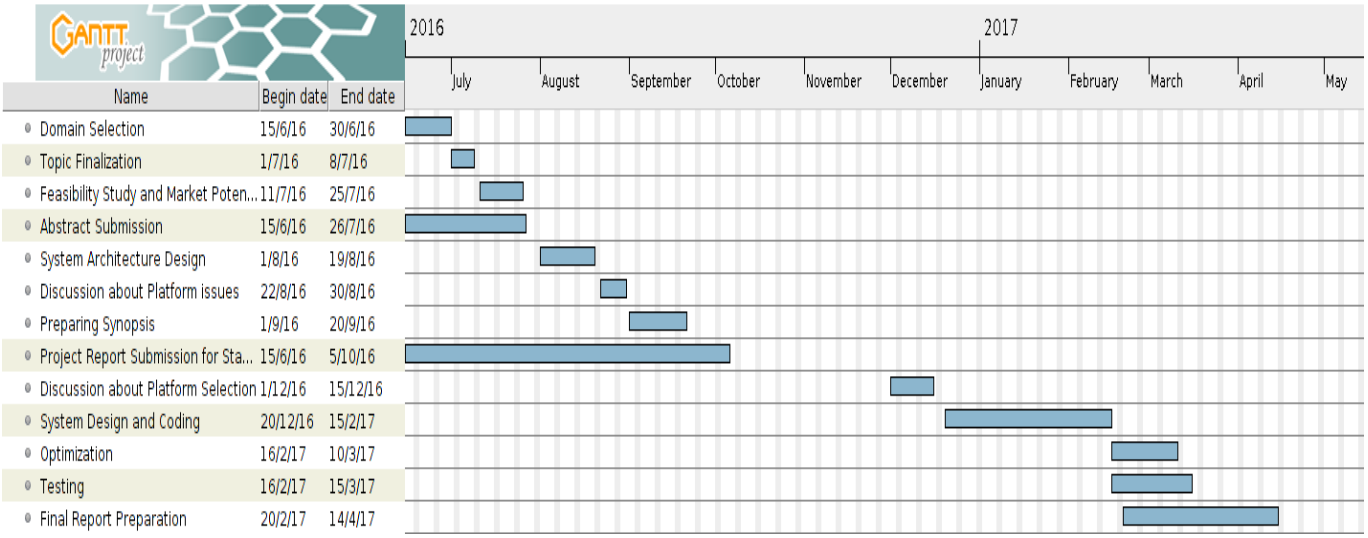


Figure 16.1: Gantt Chart

16.2 Project Planner - Table

Name	Start Date	Due Date	Priority
Domain Selection	15^{th} June 2016	30^{th} June 2016	High
Topic Finalization	1^{st} July 2016	10^{th} July 2016	High
Feasibility Study and Market Potential Analysis	11^{th} July 2016	25^{th} July 2016	High
Abstract Submission	-	26^{th} July 2016	Medium
System Architecture Design	1^{st} Aug 2016	20^{th} Aug 2016	High
Discussion about Platform issues	21^{st} Aug 2016	30^{th} Aug 2016	High
Preparing Synopsis	1^{st} Sept 2016	20^{th} sept 2016	High
Project Report Submission for Stage I	-	5^{th} Oct 2016	Medium
Discussion about Platform Selection	1^{st} Dec. 2016	15^{th} Dec. 2016	High
System Design and Coding	20^{th} Dec. 2016	15^{th} Feb. 2017	High
Testing	16^{th} Feb. 2016	15^{th} Macrh 2016	High
Final Report Preparation	-	15^{th} April 2016	High

Table 16.1: Project Planner

17.

Annexure D

17.1 Reviewers Comments of Paper Submitted

1. **Paper Title:**
2. **Name of the Conference:**
3. **Paper accepted/rejected:**
4. **Review comments by reviewer:**
5. **Corrective actions if any:**

18.

Annexure E

18.1 Plagiarism Report

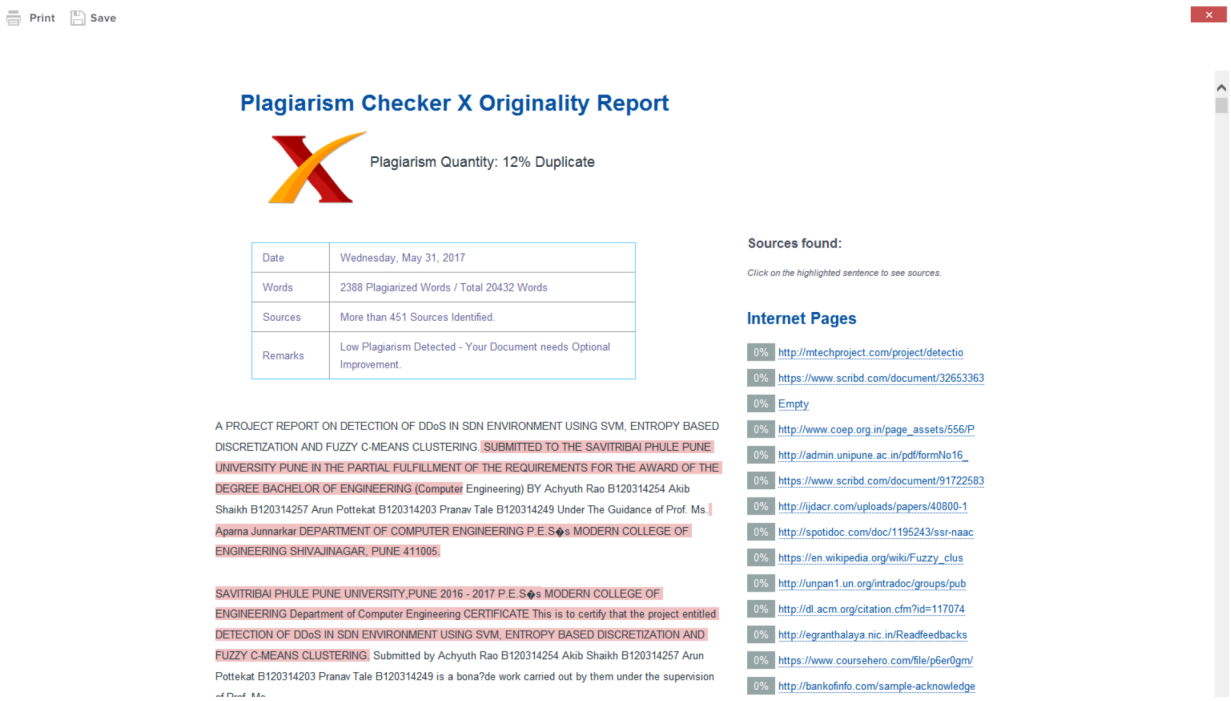


Figure 18.1: Plagiarism Report

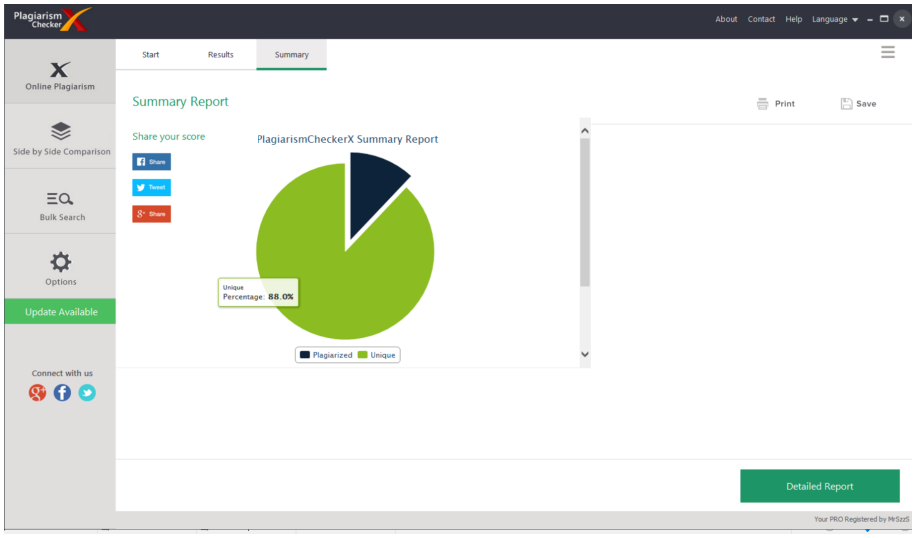


Figure 18.2: Plagiarism Report

19.

Annexure F

19.1 Term II Project Laboratory Assignments

19.1.1 Review of Design

At the end of Term 1 assessment i.e. after completing the design phase of our project the reviews received were :

1. To make sure high accuracy will be achieved.

Taking into consideration this review, instead of using libraries for developing the algorithms, the algorithms have been developed right from scratch to mould them into models very specific to the use case of our project i.e. DDoS Detection.

Along with the feedback report of Term 1 assessment, we had also participated in the annual project competition PICT-Impetus and Concepts 2017, gaining valuable insights and reviews for our project. Industrial personnel belonging to companies like VmWare, Calsoft, Synerzip, ParallelMinds, Veritas, Cybage and many more. Having vast expertise in the domains of our project and were well versed with the latest technologies like SDN, ELK Stack, sFlow. Some of the criticism/feedbacks we received were :

1. What kind of DDoS attacks are being detected?
2. Why mitigation is not being included in the scope of the project?
3. Change the controller used, from POX to OpenDayLight.
4. Include adaptive learning module to adapt to the changing network scenarios.

Some crictics were encouraging and required our attention to be drawn to the issue, one such issue was including the adaptive module to varying network environments. Hence in entropy based discretization we have included the adaptive module which modifies the normal network entropy based on average weighted mean of previous 5 entries of normal entropy.

Positive feedback was also given stating the innovativeness of the project, inclusion of well known production based tools like ELK Stack, sFlow and replicating a real world environment.
These feedbacks highlighted the strong and weak areas in our project which helped us a great deal in tackling the cross questions on the effectiveness of our project.

19.1.2 Project Installation and Setup

19.1.3 Programming Interfaces

The programming interfaces for SVM are:-

1. **TShark** - It is network capturing and analysis tool which is used for capturing packet data from a live source, or from a previously captured network stream. It's native capturing file format is pcap format.
2. **Numpy** - It is a python library which is used to perform mathematical operations over arrays.
3. **Pandas** - It is a python library which is used for statistical data analysis, by performing batch operations rather than line by line operation.

4. **Flask** - Python micro web development framework.
   Additional components :-
   (a) Flask_bootstrap
   (b) Flask_wtf
   (c) Flask_blueprint
   (d) JQuery-3.0
   (e) Flask_sqlalchemy

20.

Annexure G

20.1 Information of Project Group Members

1. Name : S. Achyuth Rao
2. Date of Birth : 19^{th} November, 1995
3. Gender : Male
4. Permanent Address : S-4, Mangalam apts., Tingre Nagar, Pune.
5. E-Mail : aachyutha2@gmail.com
6. Mobile/Contact No. : 07798845233
7. Placement Details : None
8. Paper Published : None

1. Name : Akib A. Shaikh
2. Date of Birth : 26^{th} April, 1996
3. Gender : Male
4. Permanent Address : 519, Centre Street, Camp, Pune.
5. E-Mail : akibshaikh117@gmail.com
6. Mobile/Contact No. : 09763851944
7. Placement Details : TCS Ltd.
8. Paper Published : None

1. Name : Arun P. Pottekat
2. Date of Birth : 22nd June, 1995
3. Gender : Male
4. Permanent Address : Citadel E/18, Ghorpadi, Pune.
5. E-Mail : apottekat@gmail.com
6. Mobile/Contact No. : 09561120935
7. Placement Details : None
8. Paper Published : None

1. Name : Pranav B. Tale
2. Date of Birth : 16^{th} June, 1995
3. Gender : Male
4. Permanent Address : 107-B, Rathi Nagar, Amravati, Maharashtra.
5. E-Mail : pranav.tale@gmail.com
6. Mobile/Contact No. : 09730692749
7. Placement Details : TCS Ltd.
8. Paper Published : None