A

Major Project

On

DDOS ATTACK DETECTION ALGORITHMS

Submitted to

Jawaharlal Nehru Technological University, Hyderabad

In Partial fulfillment of the requirements for the award of Degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

E.SRIKALA	(187R1A05P9)
B.MUKESH	(187R1A05J8)
N.NIKHILESH	(187R1A05L7)

Under the Guidance of

DR.M.MALYADRI.

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2018-22

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**DDOS Attack Detection Algorithms**" being submitted by **E.Srikala (187R1A05P9), B.Mukesh (187R1A05J8) & N.Nikhilesh (187R1A05L7)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr.M.Malyadri.
Associate Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HoD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Dr.M.Malyadri,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Mr. A. Uday Kiran, Mr. J. Narasimha Rao, Dr. T. S. Mastan Rao, Mrs. G. Latha, Mr. A. Kiran Kumar,** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

E.SRIKALA (187R1A05P9)

B.MUKESH (187R1A05J8)

N.NIKHILESH (187R1A05L7)

ABSTRACT

Distributed Denial of Service (DDoS) attack poses a severe threat to the Internet. It is difficult to find the exact signature of attacking. Moreover, it is hard to distinguish the difference between an unusually high volume of traffic which is caused by the attack or occurs when a huge number of users occasionally access the target machine at the same time. The entropy detection method is an effective method to detect the DDoS attack. It is mainly used to calculate the distribution randomness of some attributes in the network packets' headers. Here, we focus on the detection technology of DDoS attacks. We improve the previous entropy detection algorithm, and propose two enhanced detection methods based on cumulative entropy and time, respectively. Experiment results show that these methods could lead to more accurate and effective DDoS detection.

LIST OF FIGURES/TABLES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 3.1	Project Architecture	9
Figure 3.2	Data Flow Diagram	11
Figure 3.3	Use case diagram	13
Figure 3.4	Class diagram	14
Figure 3.5	Sequence diagram	15
Figure 3.6	Activity diagram	16

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
Screenshot 5.1.1	Homepage	23
Screenshot 5.1.2	Upload Dataset	24
Screenshot 5.1.3	PCAP Signature	25
Screenshot 5.1.4	Packet Analysis	26
Screenshot 5.1.5	CUSUM Array	27
Screenshot 5.1.6	CUSUM Based Attack Detection	28
Screenshot 5.1.7	Time Based Array	29
Screenshot 5.1.8	Time Based Attack Detection	30
Screenshot 5.2.1	Comparison Graph	31

TABLE OF CONTENTS

ABSTRACT		i
LIST OF FIGURES		ii
LIST OF SCREENSHOTS		iii
1.	**INTRODUCTION**	1
1.1	PROJECT SCOPE	1
1.2	PROJECT PURPOSE	1
1.3	PROJECT FEATURES	1
2.	**SYSTEM ANALYSIS**	2
2.1	PROBLEM DEFINITION	2
2.2	EXISTING SYSTEM	2
2.2.1	LIMITATIONS OF THE EXISTING SYSTEM	3
2.3	PROPOSED SYSTEM	3
2.3.1	ADVANTAGES OF PROPOSED SYSTEM	6
2.4	FEASIBILITY STUDY	6
2.4.1	ECONOMIC FEASIBILITY	6
2.4.2	TECHNICAL FEASIBILITY	6
2.4.3	SOCIAL FEASIBILITY	7
2.5	HARDWARE & SOFTWARE REQUIREMENTS	7
2.5.1	HARDWARE REQUIREMENTS	7
2.5.2	SOFTWARE REQUIREMENTS	8
3.	**ARCHITECTURE**	9
3.1	PROJECT ARCHITECTURE	9
3.2	DESCRIPTION	9
3.3	DATA FLOW DIAGRAM	11
3.4	USE CASE DIAGRAM	13
3.5	CLASS DIAGRAM	14
3.6	SEQUENCE DIAGRAM	15
3.7	ACTIVITY DIAGRAM	16
4.	**IMPLEMENTATION**	17
4.1	SAMPLE CODE	17

5. **RESULTS** 23
5.1 SCREENSHOTS 23
5.2 GRAPHS 31
6. **TESTING** 32
6.1 INTRODUCTION TO TESTING 32
6.2 TYPES OF TESTING 32
6.2.1 UNIT TESTING 32
6.2.2 INTEGRATION TESTING 32
6.2.3 FUNCTIONAL TESTING 33
6.3 TEST CASES 34
6.3.1 USER REQUIREMENTS 34
7. **CONCLUSION & FUTURE SCOPE** 35
7.1 PROJECT CONCLUSION 35
7.2 FUTURE SCOPE 35
8. **BIBLIOGRAPHY** 36
8.1 REFERENCES 36
8.2 WEBSITES 37

1.INTRODUCTION

INTRODUCTION

1.1 PROJECT SCOPE

The traditional Denial of Service (DoS) attack is usually a point-to-point attack. The attacker makes use of proper service requests to occupy excessive service resources to force the server crash, or to make other legal users unable to attain timely service responses.

1.2 PROJECT PURPOSE

DDoS attacks have brought tremendous threat to the security of the Internet, and also gain much research attention in the area of network security. Now, the DDoS attacks tend to become more distributed and automated, and the destruction is more serious. The attacks have some technical trends:(1)make use of clusters of controlled PCs to start intensive attacks;(2)produce randomly distributed source IP addresses to conceal the track;(3)change the structure of attack packets randomly;(4)explore the bugs and weaknesses of both network protocols and operating systems;(5)send packets faster with no apparent attack characteristics. Hybrid attacks make the defense even harder.

1.3 PROJECT FEATURES

Along with the development of computer and network technology, the impact of DoS attacks has been significantly mitigated. Distributed Denial of Service. (DDoS) attack is an extension of the traditional DoS attack. DDoS attack is a kind of distributed, cooperative large-scale attack. It has the same working principles as DoS, but compared with the traditional DoS whose attack originated from a single attacker point, the realization of DDoS comes from hundreds or even thousands of PC attackers which have been installed by Daemon, and it is a group-based attack behavior.

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is ‚"what must be done to solve the problem?" The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

The entropy detection method is an effective method to detect the DDoS attack. It is mainly used to calculate the distribution randomness of some attributes in the network packets' headers. In this paper, we focus on the detection technology of DDoS attacks. We improve the previous entropy detection algorithm, and propose two enhanced detection methods based on cumulative entropy and time, respectively. Experiment results show that these methods could lead to more accurate and effective DDoS detection.

2.2 EXISTING SYSTEM

In the past years, it was discovered that DDoS attack methods and tools are becoming more sophisticated, effective, and also more difficult to trace to the real attackers. On the defense side, current technologies are still unable to withstand large-scale attacks. Defending the DDoS attacks involves three phases: before the attack, during the attack and after the attack. The first one is precaution, which needs a process or long time to deploy the network to guard against the attack. The last one is the second line of defense. Therefore, a practical way to defend against a DDoS attack is to prevent the attack flow from reaching the target and to ensure its availability. Protection using history-based IP filtering is a method when facing the attack. But the premise of defense is to detect the attack timely and exactly. The main DDoS detection methods comprise two categories: signature-based detection and anomaly detection.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

- Current technologies are still unable to withstand large-scale attacks.
- Cannot completely eliminate the malicious traffic.
- Cannot distinguish the malicious and legitimate traffic associated with the same resources.
- Difficult to detect all malicious addresses.

2.3 PROPOSED SYSTEM

Here, we are using entropy based calculation to detect DDOS attack as all existing techniques are based on signature and anomaly and these techniques are not reliable to detect attacks. Signature based technique will match predefined attack signature with new request signature and if signature matched then it will detect attack and if new signature arrived then this signature technique will not work and in anomaly also all known attacks will be trained and if new attacks arrived then anomaly technique will also not work.

In DDOS attacks, attackers will send a continuous huge amount of requests to the server and server will keep busy on handling those requests and ignore genuine requests and by processing such huge requests the server may crash. To overcome this attack the author introduces cumulative entropy-based calculation (CUSUM) to detect attacks. This technique can be initialize with WINDOW and this window will be filled with all IP addresses and once this window filled up then application will calculate CUSUM and if CUSUM has less number of values then requests will be considered normal and when huge amount of request arrived then CUSUM value will increase and system will raise attack alarm.

CUSUM is a sliding window-based technique and the author says we can use time-based techniques also where monitoring time will be set for request, for example application will monitor first packet and then keeps on capturing for next 10 seconds and store all packets in vector. After 10 seconds the vector will be analyzed using CUSUM formula to check request quantity and if more requests arrive then the vector will have more high values then the system will raise an alarm for attack. Here, we analyzed DDOS attacks filled with all IP addresses and once this window fills up then the application will calculate CUSUM and if CUSUM has less number of values then requests will be considered normal.

Cumulative Entropy:

Cumulative Sum (CUSUM) is an algorithm from statistical process control that could detect the mean variation of a statistical process. CUSUM is based on the fact that if some change happens, the probability distribution of the random sequence will also be changed.

In our DDoS attack method, suppose X_n is the entropy value calculated by using sliding window[8] at each time interval of Δ_n, and the random sequence X_n is extracted as a network service random model. On normal occasions, this sequence is independent and distributed. Assume the variation parameter is the average value of sequence {Xn}. Before change E(Xn)=α is very small, and always positive.

Before attack, when the network is normal, the distribution of IP addresses should be stable, and have little randomness, thus the entropy value should be small. But when DDoS attacks happen, this average value will increase suddenly, E(Xn) will become far bigger than α, and become a constant or dynamic value. The value 1 shows that attack happens, while 0 shows the normal case. N is the detection threshold.

The advantage of this improved algorithm lies in that it implicitly a concept of process culminating. In the previous entropy detection algorithms, we always judge the network condition according to a threshold. For example, when the network entropy is bigger than a value, we say the network is abnormal or some attack happens. But this judgment may not be suitable on some occasions.

For example, the traffic flow in the network suddenly increases, but the flow is actually from some legal users. The function of the culminating process is to avoid the false alarm when the network has something abnormal just at a time point. We need to accumulate the total entropy during a time period, and judge this value whether it exceeds the limit or not, and the results in this way should be more accurate.

Time-Based Entropy

Based on the cumulative entropy detection described before, we make some improvement. Here, we give up the threshold value N and do not cumulate the entropy. Instead, we propose a time-based entropy detection method. The main concept is to use time to judge the network condition, not according to a threshold value to judge the attack condition.

The advantage of this algorithm is that we could control the total attack detection time by setting those two parameters: t and n, where t is the data collection interval, and n is the element number of vector X. For example, when $t = 2s$, $n = 15$, the system will give an alarm only when the network abnormal entropy persists over 30s.

A sudden traffic increase in a short time might still be a normal traffic, and we allow it. But if the network's anomaly lasts for more than 30s, or even longer, we could believe that the system might be abnormal, and some attacks might happen.

The threshold-based approach is widely applicable, and it may lead to a more real-time and timely attack detection. But for the time-based approach, we tend to emphasize the time tolerance. In some allowable range, we could ignore the network anomalies. But only when exceeding our tolerable limit, defined by a time period, we regard the network as abnormal or attack happens.

These two approaches may have their own advantages under different environments. In some cases, the DDoS detection that combines threshold-based and time-based approaches may be more efficient, and have fewer false alarms.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Easy to deploy and provision across the network.
- Does not affect the performance of the host.
- Blocks attackers if there are abnormal requests.
- Server/hosts do not crash.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and the business proposal is put forth with a very general plan for the project and some cost estimates.During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the user.
Three key considerations involved in the feasibility analysis are:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system.The following are some hardware requirements.

● Processor	:	Minimum intel i3
● Hard disk	:	512GB or above
● RAM	:	8GB and Above.

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software Components of the system. The following are some Software requirements.

• Operating System	:	Windows-8,10.
• Programming language	:	Python 3.7.4.
• Tools	:	Spyder 5.2.2.

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

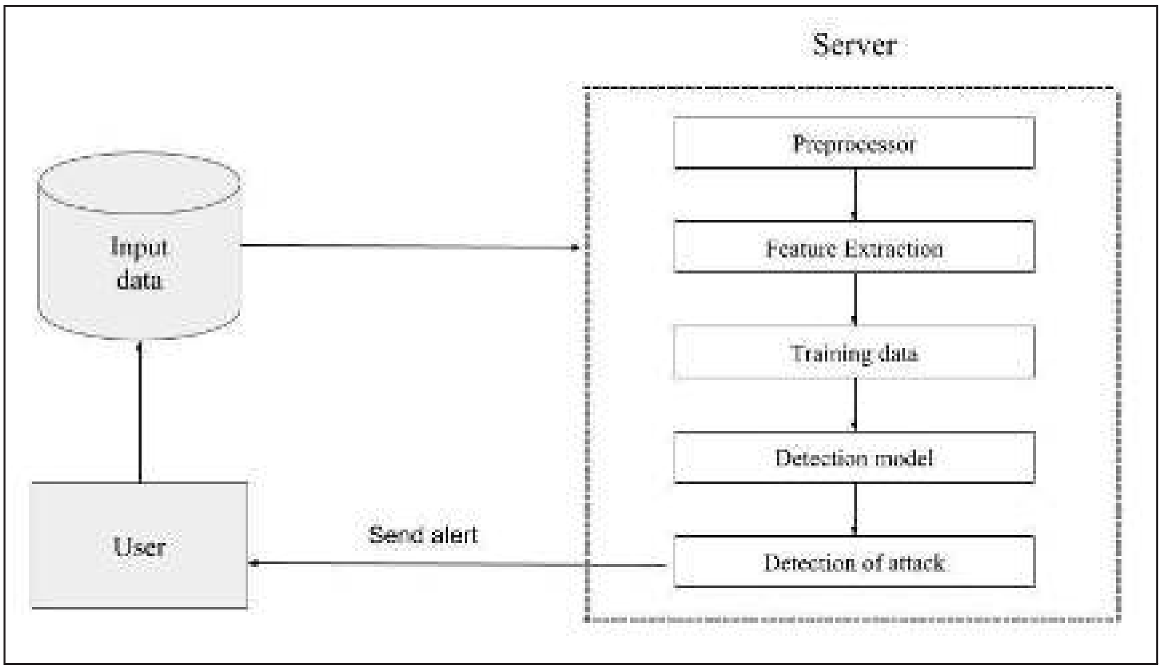


Figure 3.1: Project Architecture

3.2 DESCRIPTION

User: User will input PCAP dataset. PCAP dataset is a dataset which stores all information about attacks like from which IP address how many requests are sent.

Input Data: Input data has PCAP dataset (Packet CAPture).

Server: The dataset which is uploaded to server will undergo the following preprocessing, feature extraction, training data, detection model, detection of attack.

Send control: After detecting an attack a control is sent to the user.

Algorithm :

1. As its name implies, CUSUM involves the calculation of a Cumulative Sum (which is what makes it "sequential"). Samples from a process x_n are assigned weights w_n , and summed as follows :

   $S_0 = 0$

   $S_n+1 = \max(0,S_n + x_n - w_n)$
2. When the value of S exceeds a certain threshold value, a change in value has been found. The above formula only detects changes in the positive direction.
3. When negative changes need to be found as well, the minimum operation should be used instead of max operation, and this time a change as been found when the value of S is below the (negative) value of threshold value.

3.3 DATA FLOW DIAGRAM:

The DFD is also called a bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

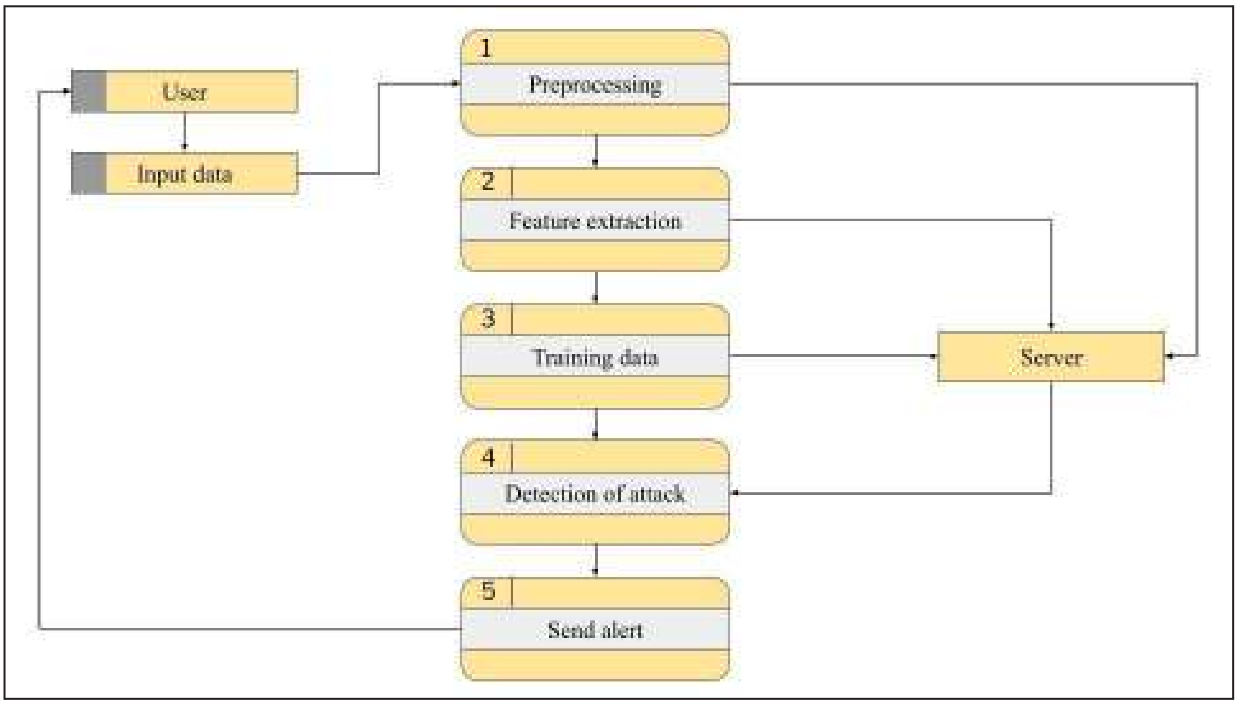


Figure 3.3: Data Flow Diagram.

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML comprises two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development processes.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of the OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

3.4 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

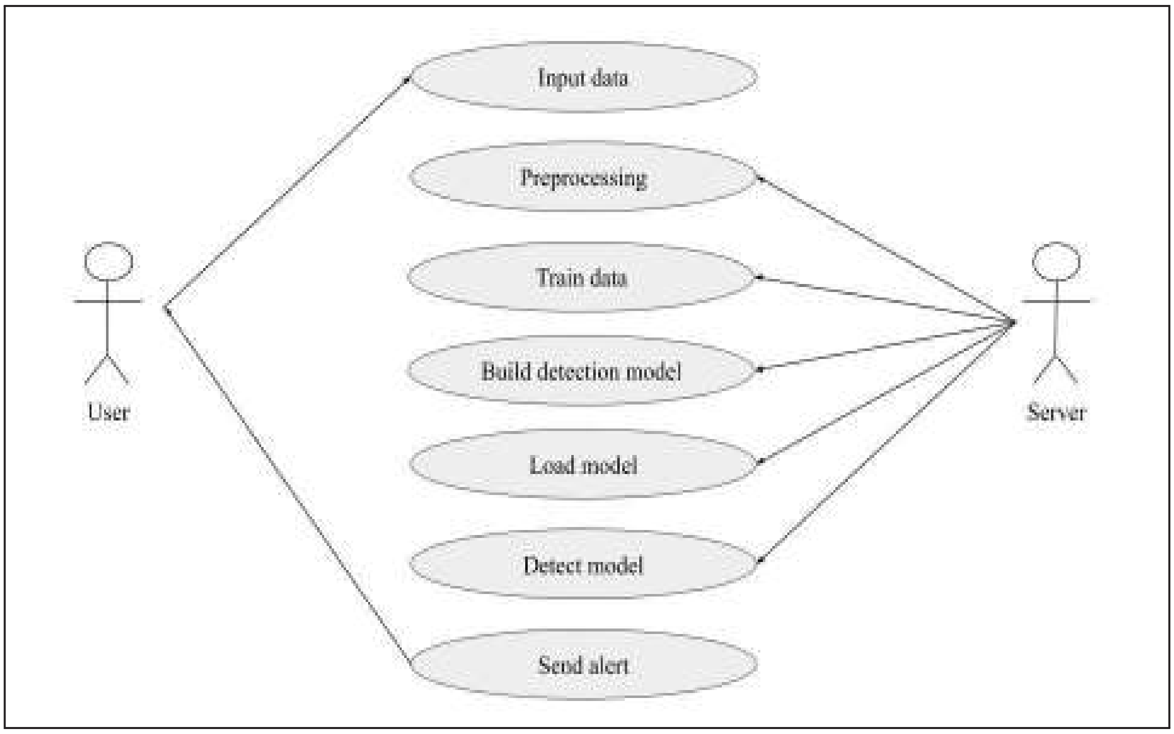


Figure 3.4: Use Case Diagram

3.5 CLASS DIAGRAM

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

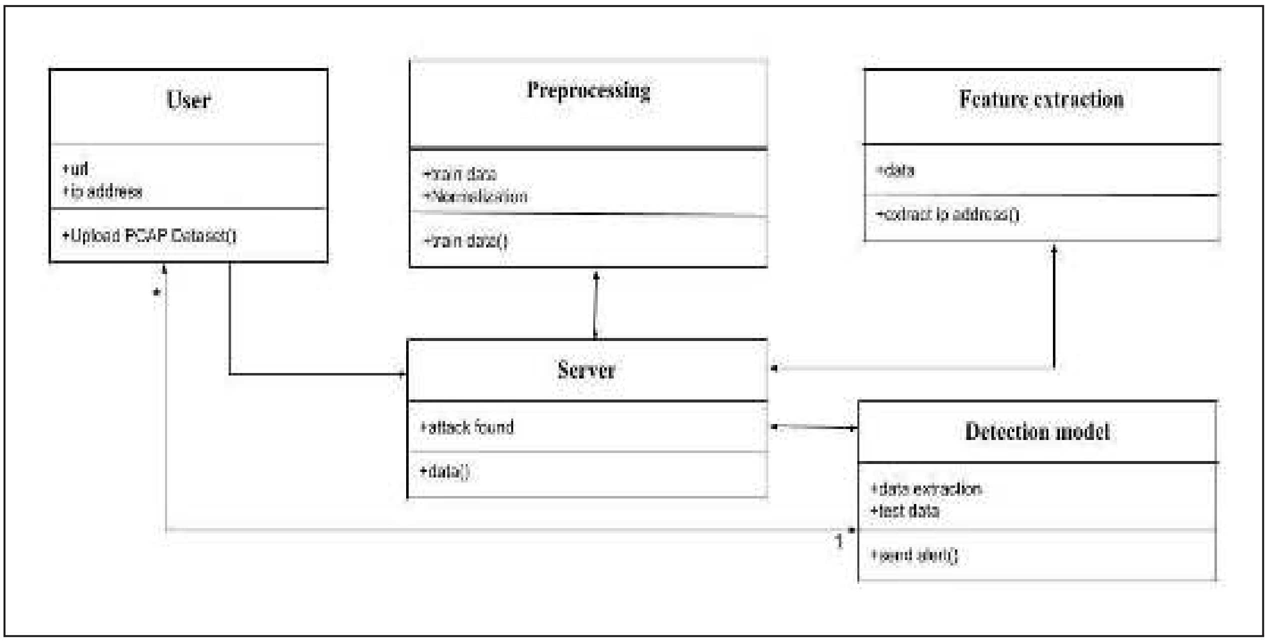


Figure 3.5: Class Diagram

3.6 SEQUENCE DIAGRAM

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

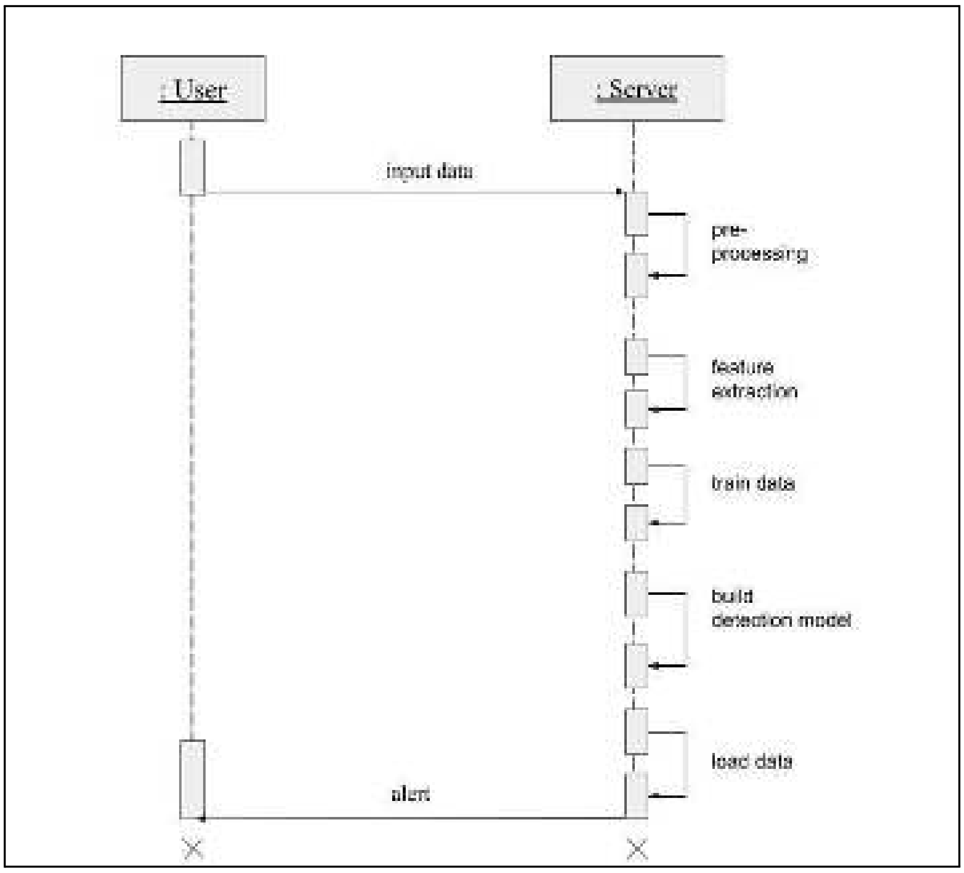


Figure 3.6: Sequence Diagram

3.7 ACTIVITY DIAGRAM

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.

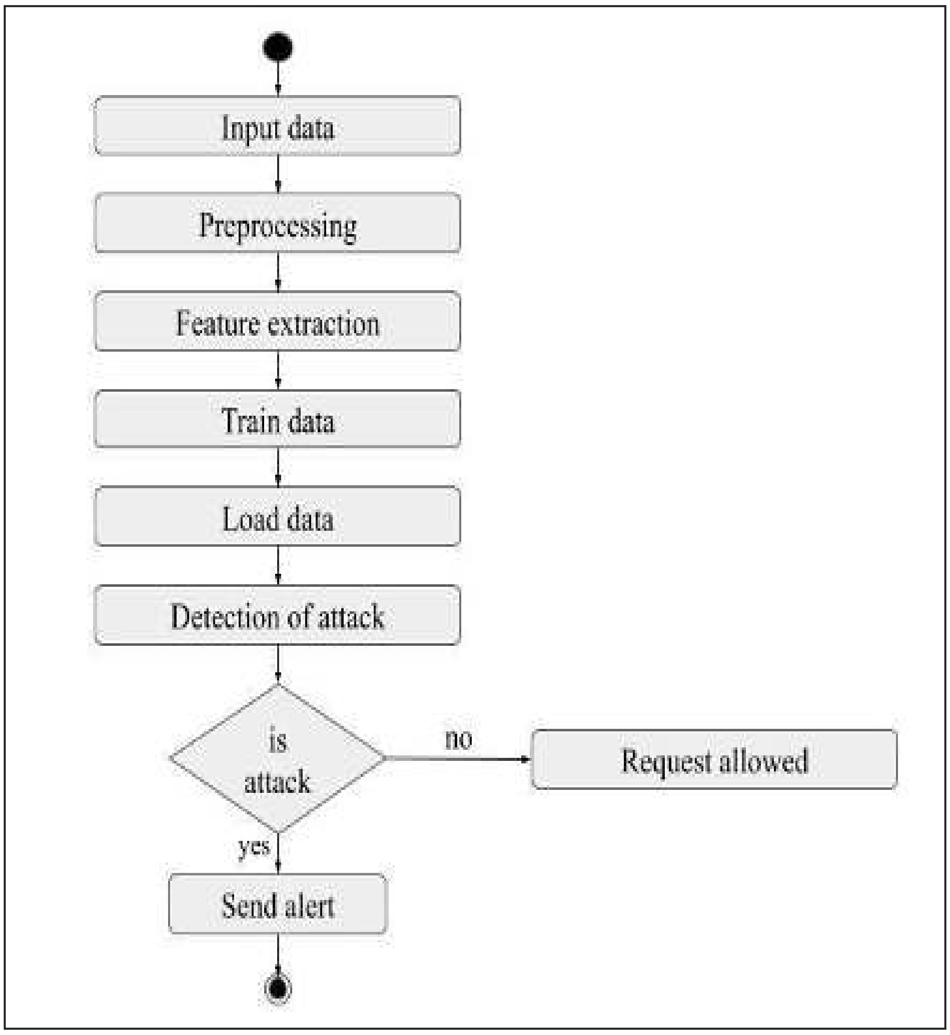


Figure 3.7: Activity Diagram

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
from tkinter import messagebox
from tkinter import *
from tkinter.filedialog import askopenfilename
from tkinter import simpledialog
import tkinter
import numpy as np
from tkinter import filedialog
import matplotlib.pyplot as plt
from scapy.all import *
from multiprocessing import Queue
from PcapEntropy import *
from TimeBased import *

main = tkinter.Tk()
main.title("DDoS Attack Detection Algorithms Based on Entropy Computing")
main.geometry("1300x1200")

global filename
global cu_sum
global timebased

 def uploadPCAP():
    global filename
    filename = filedialog.askopenfilename(initialdir = "Win_Pcap_Files")
    pathlabel.config(text=filename)
    text.delete('1.0', END)
    text.insert(END,'PCAP Signatures loaded\n')


    def runCUSUM():
    global cu_sum
    text.delete('1.0', END)
    queue = Queue()
    packets = rdpcap(filename)
    for pkt in packets:
        queue.put(pkt)
    total_packets = queue.qsize();
    text.insert(END,"Packets loaded to Queue\n");
    text.insert(END,"Total available packets in Queue are : "+str(queue.qsize()))
    cu_sum = CUSUM(queue,text)
    cu_sum.start()

    def graph():
    unique, count = np.unique(cu_sum.getMalicious(),return_counts=True)
    unique = np.sort(count)
    unique1, count1 = np.unique(timebased.getMalicious(),return_counts=True)
    unique1 = np.sort(count1)
    plt.figure(figsize=(10,6))
    plt.grid(True)
    plt.xlabel('Time')
    plt.ylabel('Entropy')
    plt.plot(unique, 'ro-', color = 'blue')
    plt.plot(unique1, 'ro-', color = 'orange')
    plt.legend(['CUSUM', 'Time Based'], loc='upper left')
    #plt.xticks(wordloss.index)
    plt.title('CUSUM & Time Based Attack Detection Graph')
    plt.show()

    def runTimeBased():
    global timebased
    text.delete('1.0', END)
    queue = Queue()
    packets = rdpcap(filename)
    for pkt in packets:
        queue.put(pkt)
    total_packets = queue.qsize();
    text.insert(END,"Packets loaded to Queue\n");
    text.insert(END,"Total available packets in Queue are : "+str(queue.qsize()))
    timebased = TimeBased(queue,text)
    timebased.start()


def close():
    main.destroy()

font = ('times', 16, 'bold')
title = Label(main, text='DDoS Attack Detection Algorithms Based on Entropy
Computing')
title.config(bg='Red', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)
```

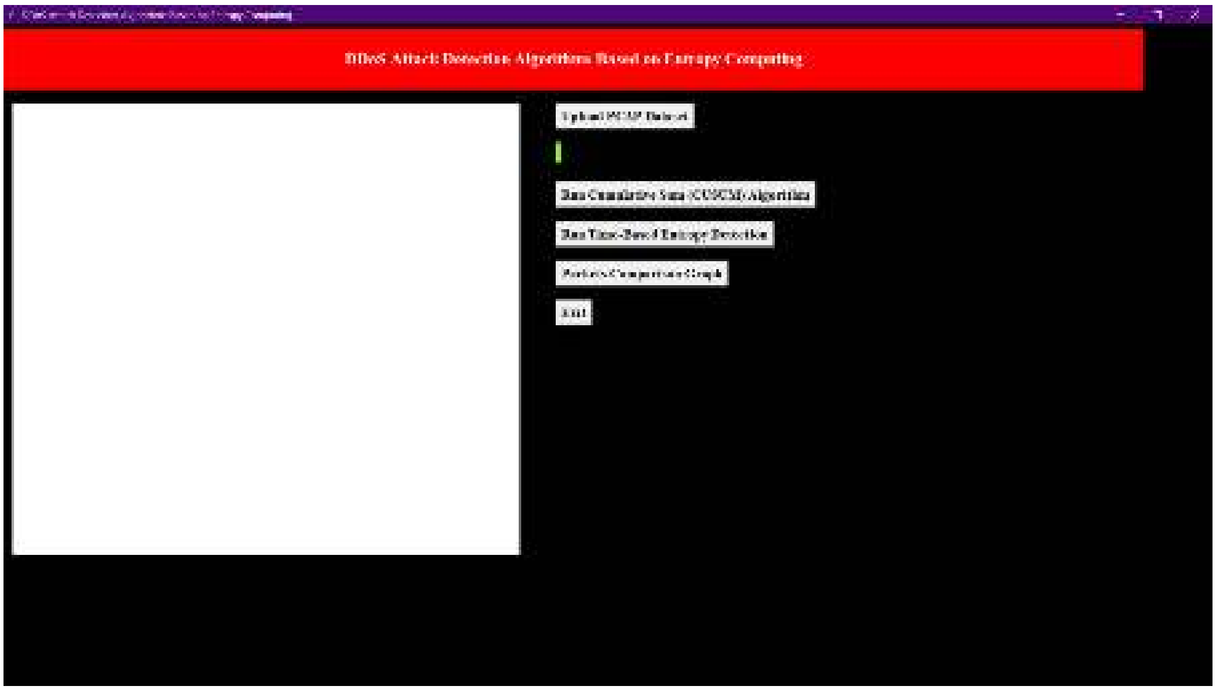
```
font1 = ('times', 13, 'bold')
upload = Button(main, text="Upload PCAP Dataset", command=uploadPCAP)
upload.place(x=700,y=100)
upload.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='lawn green', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=700,y=150)

cusumButton = Button(main, text="Run Cumulative Sum (CUSUM)
Algorithm", command=runCUSUM)
cusumButton.place(x=700,y=200)
cusumButton.config(font=font1)

timeButton = Button(main, text="Run Time-Based Entropy Detection",
command=runTimeBased)
timeButton.place(x=700,y=250)
timeButton.config(font=font1)

graphButton = Button(main, text="Packets Comparison Graph",
command=graph)
graphButton.place(x=700,y=300)
graphButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close)
exitButton.place(x=700,y=350)
exitButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=80)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=100)
text.config(font=font1)


main.config(bg='Black')
main.mainloop()
```

```
from threading import Thread
from scapy.all import *
from datetime import datetime
from tkinter import *
from tkinter import messagebox
import numpy as np

class CUSUM(Thread):
    __flagsTCP = {
        'F': 'FIN',
        'S': 'SYN',
        'R': 'RST',
        'P': 'PSH',
        'A': 'ACK',
        'U': 'URG',
        'E': 'ECE',
        'C': 'CWR',
        }
    __ip_cnt_TCP = {}

    malicious = []

    def __init__(self, queue, text):
        Thread.__init__(self)
        self.stopped = False
        self.queue = queue
        self.text = text
        self.malicious.clear()
        self. __ip_cnt_TCP.clear()


    def stop(self):
        self.stopped = True

    def getMalicious(self):
        return self.malicious

    def stopfilter(self, x):
        return self.stopped

    def detect_TCPflood(self, packet, window):
        if UDP in packet:
            print("========"+str(packet))
        if TCP in packet:
            pckt_src=packet[IP].src
            pckt_dst=packet[IP].dst


 stream = pckt_src + ':' + pckt_dst

        if stream in self.__ip_cnt_TCP:
            self.__ip_cnt_TCP[stream] += 1
        else:
            self.__ip_cnt_TCP[stream] = 1

        for stream in self.__ip_cnt_TCP:
            pckts_sent = self.__ip_cnt_TCP[stream]
            src = stream.split(':')[0]
            dst = stream.split(':')[1]
            if src in window.keys():
                window[src] = window.get(src) + 1
            else:
                window[src] = 1
            if len(window) >= 5:
                entropy = []
                for key,value in window.items():
                    entropy.append(value)
                value = np.cumsum(entropy)
                if value[0] > 5:
                    print("CUSUM with attack : "+str(value))
                    self.text.insert(END,"Possible TCP-SYN-Flood Attack from %s
--> %s --> %s\n"%(src,dst,str(pckts_sent)))
                    self.malicious.append(np.sum(value))
                else:
                    self.malicious.append(np.sum(value))
                    print("CUSUM without attack : "+str(value))
                    print(END,"Normal traffic from %s --> %s -->
%s\n"%(src,dst,packet.ttl))
                window.clear()


    def process(self, queue):
        self.malicious.clear()
        window = {}
        while not queue.empty():
            pkt = queue.get()
            if IP in pkt:
               pckt_src=pkt[IP].src
               pckt_dst=pkt[IP].dst
               #print("IP Packet: %s  ==>  %s  ,
%s"%(pckt_src,pckt_dst,str(datetime.now().strftime("%Y-%m-%d
%H:%M:%S"))), end=' ')



 if TCP in pkt:
               src_port=pkt.sport
               dst_port=pkt.dport
               #print(", Port: %s --> %s, "%(src_port,dst_port), end='')
               #print([__flagsTCP[x] for x in pkt.sprintf('%TCP.flags%')])
               self.detect_TCPflood(pkt,window)
         queue.empty()
         messagebox.showinfo("CUSUM Based Attack Detection","CUSUM
Based Attack Detection : "+str(len(self.getMalicious())))


   def run(self):
         print("Sniffing started. ")
         self.process(self.queue)
```

5.RESULTS

5.1.1 HOMEPAGE :

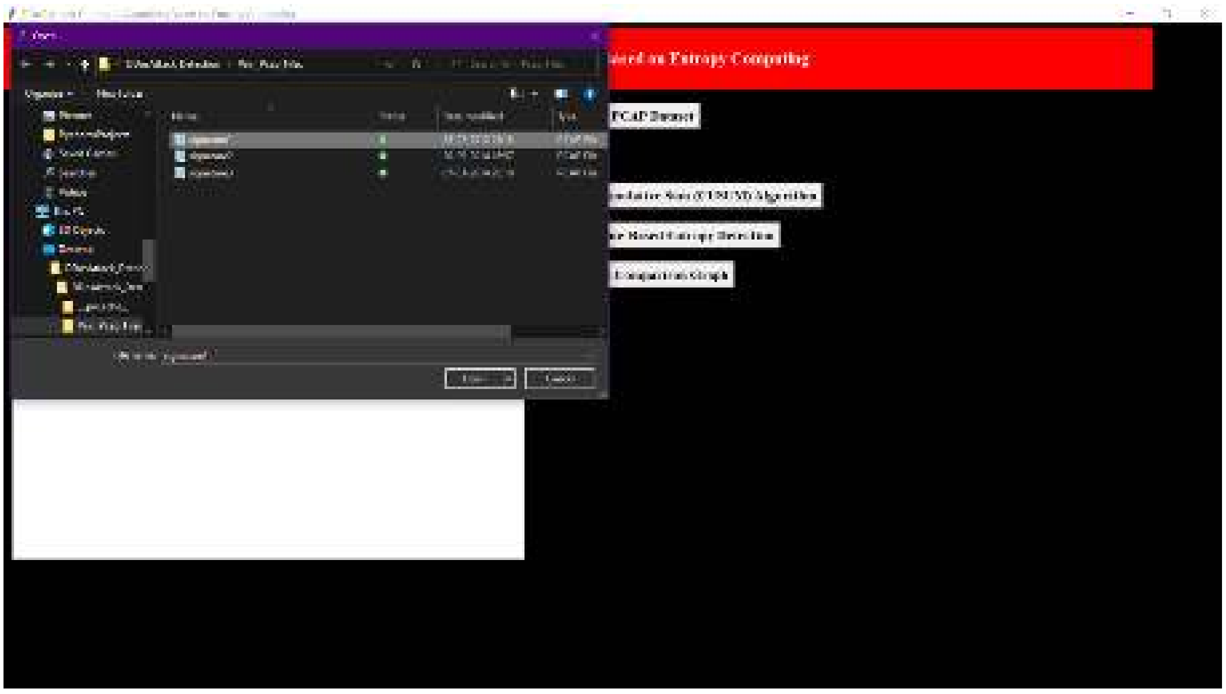
This is the result of the homepage. In the below screen click on ‘Upload PCAP Dataset’ button to upload PCAP file.


DDoS Attack Detection Algorithms Based on Entropy Computing
DDoS Attack Detection Algorithms Based on Entropy Computing
Upload PCAP Dataset
Run Cumulative Sum (CUSUM) Algorithm
Run Time-Based Entropy Detection
Packets Comparison Graph
Exit

Screenshot 5.1: Home Page .

5.1.2 UPLOAD DATASET:

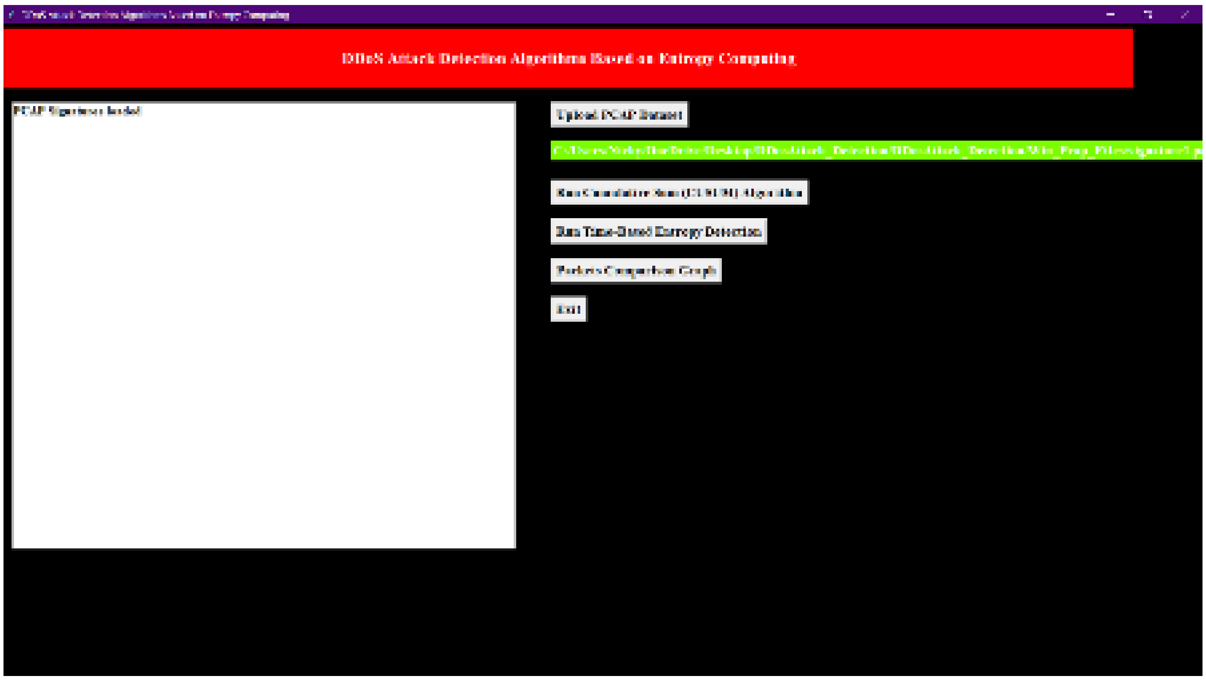
In the screen below, select and upload the ‘signature.pcap’ file and then click on ‘Open’ button.



Screenshot 5.1.2: Upload Dataset.

5.1.3 PCAP SIGNATURES:

In the below screen pcap file loaded and now click on ‘Run Improved Cumulative Sum (CUSUM) Algorithm’ button to analyze the pcap file with CUSUM technique.


DDoS Attack Detection Algorithms Based on Entropy Computing
Upload PCAP Dataset
Run Cumulative Sum (CUSUM) Algorithm
Run Time-Based Entropy Detection
Packets Comparison Graph
Exit

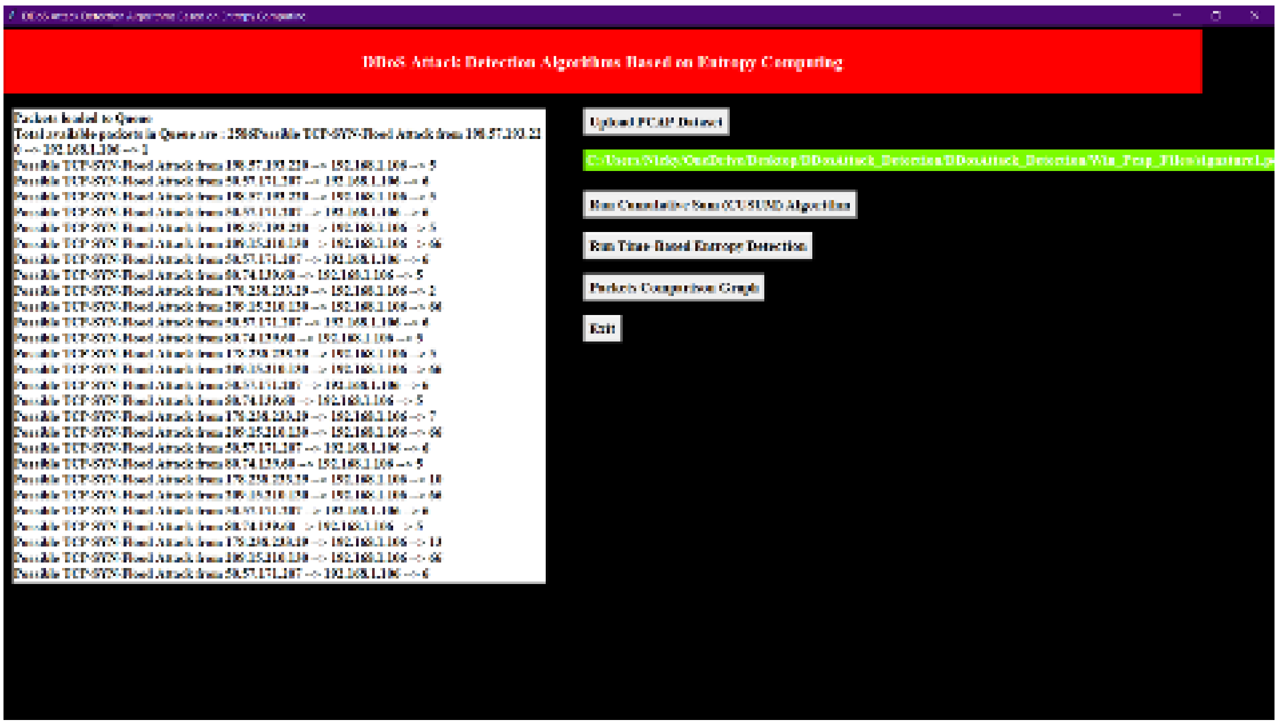
PCAP Signatures loaded

C:/Users/[illegible]/Desktop/DDos Attack_Detection/DDos Attack_Detection/Win_Pcap_Files/signatures[illegible]

Screenshot 5.1.3: PCAP Signatures

5.1.4 PACKET ANALYSIS:

In the below screen each packet is analyzed and we can see source and destination port number and we can see CUSUM values for each packet in the next screen.

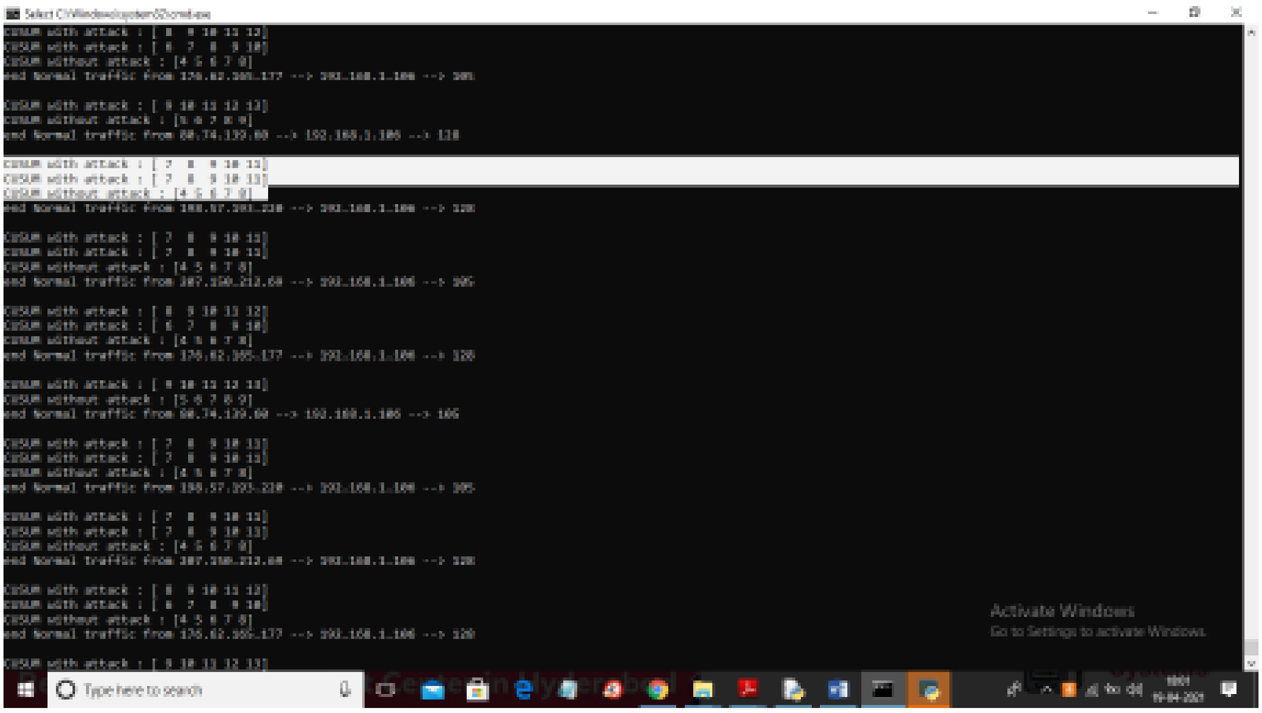

DDoS Attack Detection Algorithms Based on Entropy Computing
DDoS Attack Detection Algorithms Based on Entropy Computing
Upload PCAP Dataset
C:/Users/Vicky/OneDrive/Desktop/DDosAttack_Detection/DDosAttack_Detection/Win_Pcap_Files/signature1.p
Run Cumulative Sum (CUSUM) Algorithm
Run Time Based Entropy Detection
Packets Comparison Graph
Exit

```
Packets loaded to Queue
Total available packets in Queue are : 258Possible TCP-SYN-Flood Attack from 198.57.193.21
9 --> 192.168.1.106 --> 1
Possible TCP-SYN-Flood Attack from 198.57.193.219 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 198.57.193.219 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 198.57.193.219 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 88.74.129.68 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.233.19 --> 192.168.1.106 --> 2
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 88.74.129.68 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.233.19 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 88.74.129.68 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.233.19 --> 192.168.1.106 --> 7
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 88.74.129.68 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.233.19 --> 192.168.1.106 --> 10
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 88.74.129.68 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.233.19 --> 192.168.1.106 --> 13
Possible TCP-SYN-Flood Attack from 209.15.210.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 58.57.171.207 --> 192.168.1.106 --> 6
```

Screenshot 5.1.4: Packet Analysis

5.1.5 CUSUM ARRAY:

In above screen for each packet we got CUSUM array and we can see CUSUM array with less values has no attack and when attack occur then CUSUM array contains more values and in above screen without attack CUSUM value is [4 5 6 7 8] and when attack occur then CUSUM values changed to [7 8 9 10 11]. So by analyzing this CUSUM array we can detect whether a request is normal or attack.

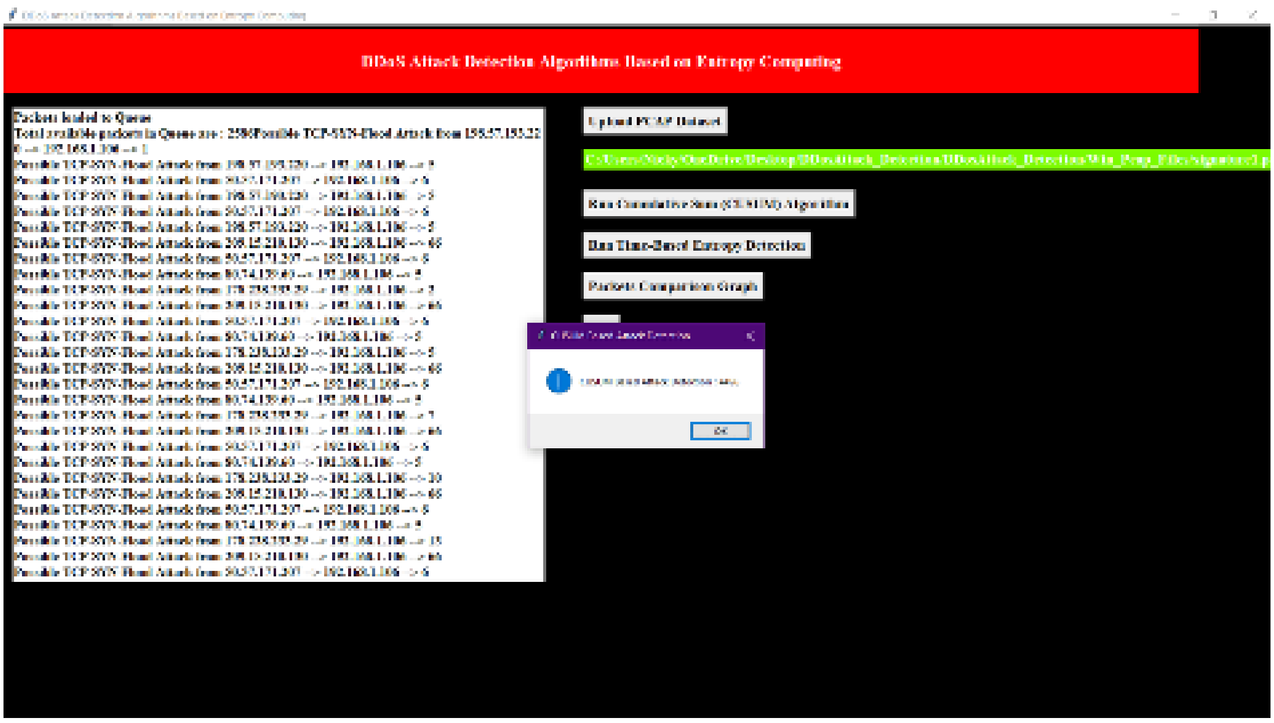


```
Select C:\Windows\system32\cmd.exe
CUSUM with attack : [ 8  9 10 11 12]
CUSUM with attack : [ 6  7  8  9 10]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 176.82.165.177 --> 192.168.1.106 --> 105

CUSUM with attack : [ 9 10 11 12 13]
CUSUM without attack : [5 6 7 8 9]
end Normal traffic from 80.74.139.60 --> 192.168.1.106 --> 120

CUSUM with attack : [ 7  8  9 10 11]
CUSUM with attack : [ 7  8  9 10 11]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 198.57.193.220 --> 192.168.1.106 --> 120

CUSUM with attack : [ 7  8  9 10 11]
CUSUM with attack : [ 7  8  9 10 11]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 207.150.212.60 --> 192.168.1.106 --> 105

CUSUM with attack : [ 8  9 10 11 12]
CUSUM with attack : [ 6  7  8  9 10]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 176.82.165.177 --> 192.168.1.106 --> 120

CUSUM with attack : [ 9 10 11 12 13]
CUSUM without attack : [5 6 7 8 9]
end Normal traffic from 80.74.139.60 --> 192.168.1.106 --> 105

CUSUM with attack : [ 7  8  9 10 11]
CUSUM with attack : [ 7  8  9 10 11]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 198.57.193.220 --> 192.168.1.106 --> 105

CUSUM with attack : [ 7  8  9 10 11]
CUSUM with attack : [ 7  8  9 10 11]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 207.150.212.60 --> 192.168.1.106 --> 120

CUSUM with attack : [ 8  9 10 11 12]
CUSUM with attack : [ 6  7  8  9 10]
CUSUM without attack : [4 5 6 7 8]
end Normal traffic from 176.82.165.177 --> 192.168.1.106 --> 120

CUSUM with attack : [ 9 10 11 12 13]
```

Screenshot 5.1.5 :CUSUM Array.

5.1.6 CUSUM BASED ATTACK DETECTION:

In the below screen we can see the PCAP file contains a total of 4496 attacks and now click on ‘Run Time-Based Entropy Detection’ button to run Time Based technique.


DDoS Attack Detection Algorithms Based on Entropy Computing
Upload PCAP Dataset
Run Cumulative Sum (CUSUM) Algorithm
Run Time-Based Entropy Detection
Packets Comparison Graph
OK

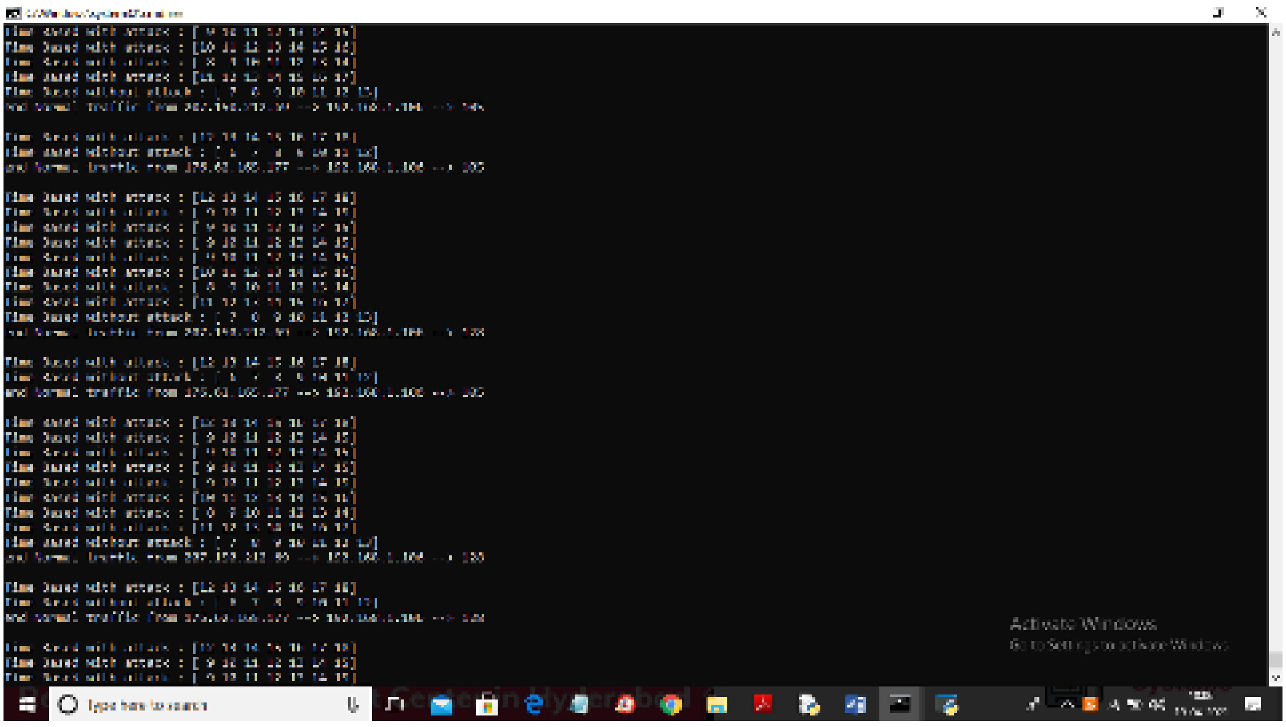
```
Packets loaded to Queue
Total available packets in Queue are : 2596Possible TCP-SYN-Flood Attack from 198.57.193.22
0 --> 192.168.1.106 --> 1
Possible TCP-SYN-Flood Attack from 198.57.193.220 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 198.57.193.220 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 198.57.193.220 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 80.74.139.60 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.133.29 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 80.74.139.60 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.133.29 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 80.74.139.60 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.133.29 --> 192.168.1.106 --> 7
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 80.74.139.60 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.133.29 --> 192.168.1.106 --> 10
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Possible TCP-SYN-Flood Attack from 80.74.139.60 --> 192.168.1.106 --> 5
Possible TCP-SYN-Flood Attack from 178.238.133.29 --> 192.168.1.106 --> 15
Possible TCP-SYN-Flood Attack from 208.15.218.130 --> 192.168.1.106 --> 66
Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
```

C:/Users/Vicky/OneDrive/Desktop/DDosAttack_Detection/DDosAttack_Detection/Win_Pcap_Files/signature1.p

Screenshot 5.1.6: CUSUM Based Attack Detection

5.1.7 TIME-BASED ARRAY:

In the below screen with Time Based also we can see more request variation in attack scenario compared to normal scenario.

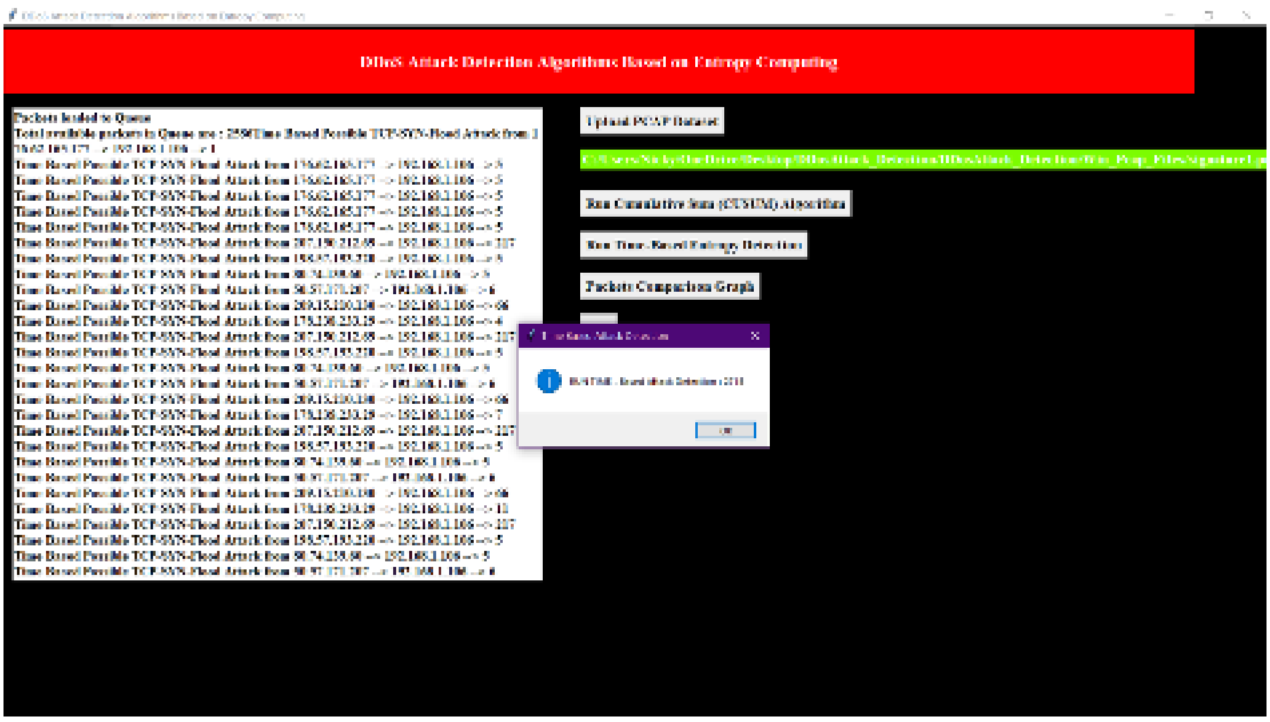


```
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [10 11 12 13 14 15 16]
Time Based with attack : [ 8  9 10 11 12 13 14]
Time Based with attack : [11 12 13 14 15 16 17]
Time Based without attack : [ 7  8  9 10 11 12 13]
and Normal traffic from 207.198.212.89 --> 192.168.1.108 --> 105

Time Based with attack : [12 13 14 15 16 17 18]
Time Based without attack : [ 6  7  8  9 10 11 12]
and Normal traffic from 175.62.165.177 --> 192.168.1.108 --> 105

Time Based with attack : [12 13 14 15 16 17 18]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [10 11 12 13 14 15 16]
Time Based with attack : [ 8  9 10 11 12 13 14]
Time Based with attack : [11 12 13 14 15 16 17]
Time Based without attack : [ 7  8  9 10 11 12 13]
and Normal traffic from 207.198.212.89 --> 192.168.1.108 --> 108

Time Based with attack : [12 13 14 15 16 17 18]
Time Based without attack : [ 6  7  8  9 10 11 12]
and Normal traffic from 175.62.165.177 --> 192.168.1.108 --> 105

Time Based with attack : [12 13 14 15 16 17 18]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [10 11 12 13 14 15 16]
Time Based with attack : [ 8  9 10 11 12 13 14]
Time Based with attack : [11 12 13 14 15 16 17]
Time Based without attack : [ 7  8  9 10 11 12 13]
and Normal traffic from 207.198.212.89 --> 192.168.1.108 --> 108

Time Based with attack : [12 13 14 15 16 17 18]
Time Based without attack : [ 6  7  8  9 10 11 12]
and Normal traffic from 175.62.165.177 --> 192.168.1.108 --> 108

Time Based with attack : [12 13 14 15 16 17 18]
Time Based with attack : [ 9 10 11 12 13 14 15]
Time Based with attack : [ 9 10 11 12 13 14 15]
```

Screenshot 5.1.7 :TIME-BASED ARRAY

5.1.8 TIME – BASED ATTACK DETECTION:

In the below screen with time based application detected 2715 attacks as this technique monitors once in 10 seconds so its detection rate will be less and in the black console we can see Time Based CUSUM values and now click on 'Packets Comparison Graph' button to get below graph.


DDoS Attack Detection Algorithms Based on Entropy Computing
Upload PCAP Dataset
Run Cumulative Sum (CUSUM) Algorithm
Run Time_Based Entropy Detection
Packets Comparison Graph
OK

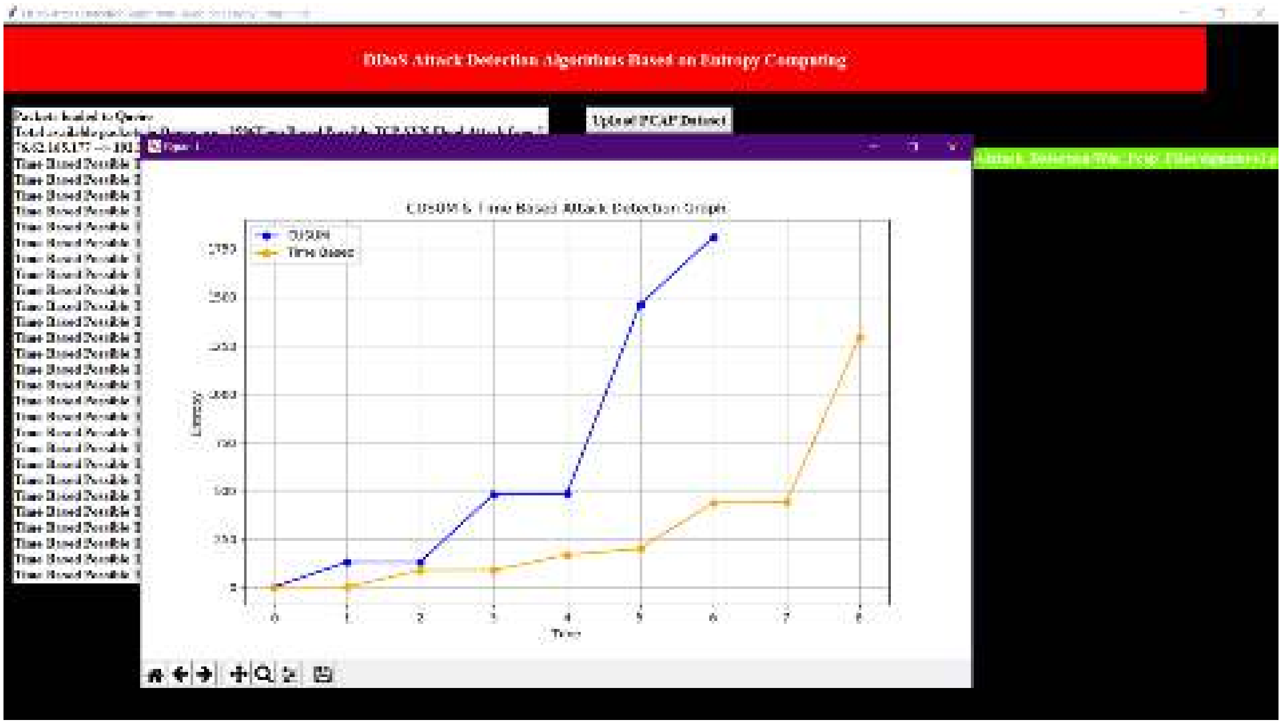
```
Packets loaded to Queue
Total available packets in Queue are : 2598Time Based Possible TCP-SYN-Flood Attack from 1
76.62.165.177 --> 192.168.1.106 --> 1
Time Based Possible TCP-SYN-Flood Attack from 176.62.165.177 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 176.62.165.177 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 176.62.165.177 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 176.62.165.177 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 176.62.165.177 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 207.190.212.65 --> 192.168.1.106 --> 127
Time Based Possible TCP-SYN-Flood Attack from 155.57.193.218 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.74.135.60 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Time Based Possible TCP-SYN-Flood Attack from 209.15.130.138 --> 192.168.1.106 --> 66
Time Based Possible TCP-SYN-Flood Attack from 178.138.233.19 --> 192.168.1.106 --> 4
Time Based Possible TCP-SYN-Flood Attack from 207.190.212.65 --> 192.168.1.106 --> 127
Time Based Possible TCP-SYN-Flood Attack from 155.57.193.218 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.74.135.60 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Time Based Possible TCP-SYN-Flood Attack from 209.15.130.138 --> 192.168.1.106 --> 66
Time Based Possible TCP-SYN-Flood Attack from 178.138.233.19 --> 192.168.1.106 --> 7
Time Based Possible TCP-SYN-Flood Attack from 207.190.212.65 --> 192.168.1.106 --> 127
Time Based Possible TCP-SYN-Flood Attack from 155.57.193.218 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.74.135.60 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
Time Based Possible TCP-SYN-Flood Attack from 209.15.130.138 --> 192.168.1.106 --> 66
Time Based Possible TCP-SYN-Flood Attack from 178.138.233.19 --> 192.168.1.106 --> 11
Time Based Possible TCP-SYN-Flood Attack from 207.190.212.65 --> 192.168.1.106 --> 127
Time Based Possible TCP-SYN-Flood Attack from 155.57.193.218 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.74.135.60 --> 192.168.1.106 --> 5
Time Based Possible TCP-SYN-Flood Attack from 50.57.171.207 --> 192.168.1.106 --> 6
```

[illegible] TIME - Based Attack Detection : 2715

Screenshot 5.1.8:Time-Based Attack Detection.

5.2.1 COMPARISON GRAPH:

In the below graph x-axis represents time and y-axis represents entropy values and in the below graph blue is for Window Based CUSUM and orange line is for Time Based CUSUM and in both techniques we can see entropy values increase heavily when attack occurred. In the above graph up to 5 seconds in x-axis both techniques have low entropy values and after application received more requests from the same IP which causes growth in entropy value .


DDoS Attack Detection Algorithms Based on Entropy Comparing
Upload PCAP Dataset
Packets loaded to Queue
Time Based Possible
CUSUM & Time Based Attack Detection Graph
CUSUM
Time Based
Entropy
Time
0
1
2
3
4
5
6
7
8

Screenshot 5.2.1: Comparison Graph.

6.TESTING

TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.In addition, systematic coverage pertaining to identifying Business process flows; data fields, predefined processes.

Test strategy and approach:

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.3 TEST CASES

6.3.1 USER REQUIREMENTS:

Test case ID	Test case Name	purpose	Test case	Output
1	DDOS Attack	DDOS Attack Detection	User gives PCAP dataset (Packet CAPture contains information about requests) as input	Detects Malicious requests
2	DDOS Attack	DDOS Attack Detection	User gives PCAP dataset (Packet CAPture contains information about requests) as input	Detects Malicious requests

7.CONCLUSION

CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

In this paper, we studied the DDoS detection algorithms based on entropy monitoring, and proposed two improved entropy detection approaches: cumulative entropy and time-based methods. We also conducted experiments for the traditional entropy detection method and the cumulative entropy detection method, respectively. From the test results, we could see that our cumulative entropy detection method has good detection capability. For different network environments, how to configure the threshold value is a key point, which influences the detection efficiency. In the time-based entropy detection method, we introduced a new concept of time cumulating. By setting a system's tolerable detection time, DDoS detection can be carried out without giving a typical threshold value.

7.2 FUTURE SCOPE

Unfortunately, the current trend shows that DDoS attacks will only become more frequent, more massive, and more effective. On top of that, the ongoing digitization of businesses took a giant leap because of the COVID-19 pandemic. As a result, companies of all sizes have a good portion of their assets online, making them perfect targets.

When it comes to defense against DDoS, the current state is not promising at all. Hackers are already effectively using cloud technologies in order to execute DDoS. Businesses need to quickly adapt and start using similar solutions in their security stacks. However, hackers always seem to be one step ahead when it comes to methods, technologies, and tactics.

8.BIBLIOGRAPHY

BIBLIOGRAPHY

8.1 REFERENCES

1. Shannon, C.E., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press (1963)

2. Siaterlis, C., Maglaris, V.: Detecting Incoming and Outgoing DDoS Attacks at the Edge using a Single set of Network Characteristics. In: 10th IEEE Symposium on Computers and Communications (ISCC 2005), pp. 469–475 (2005)

3. Chang, R.K.C.: Defending against Flooding-based Distributed Denial-of-Service Attacks: a Tutorial. IEEE Communications Magazine 40(10), 42–51 (2002)

4. Cao, Y., Li, H., Lv, D.: DDoS-based TCP SYN Flood and Defense. Electrical Technology (2004)

5. Dittrich, D.: The Stacheldraht' Distributed Denial of Service Attack Tool (1999), http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.

6. Risso, F., Delgioanni, L., Varenni, G., Viano, P., Pai, N.: WinPcap: The Windows Packet Capture Library, http://www.winpcap.org/

7. Yuan, J., Mills, K.: Monitoring the Macroscopic Effect of DDoS Flooding Attacks. IEEE Transactions on Dependable and Secure Computing 2(4) (2005)

8. Feinstein, L., Schnackenberg, D.: Statistical Approaches to DDoS Attack Detection and Response. In: 2003 DARPA Information Survivability Conference and Exposition (DISCEX 2003), pp. 303–314 (2003)

9. Limwiwatkul, L., Rungsawangr, A.: Distributed Denial of Service Detection using TCP/IP Header and Traffic Measurement Analysis. In: 2004 International Symposium on Communications and Information Technologies (ISCIT 2004), Sapporo, Japan (2004)

10.Lu,J,Yin,C.,Zhuang,X.,Lu,K.,Li,O.:DDOS Attack Detection based on non-parameter CUSUM.In:Computer and Network(2004)

11.Lin,B,Li,O,Liu,Q.:DDOS Attack Detection Based on Sequential Change Detection.Computer Engineering31(9)(2005)

12.Li,Q., Chang.,E,-C.,Chan,M.C.: On the Effectiveness of DDOS Attackon Statistical Filtering.IEEE INFOCOM 2005,1373-1383(2005).

13.Li.,L.,Lee,G.:DDOS Attack Detection and Wavelets.In:12th International Conference on Computer Communications and Networks(ICCCN 2003),pp.421-427(2003)

14.Porras,P.A.,Neumann,P.G.:EMERALD:Event Monitoring Enabling Responses to Anomalous Live Disturbances.In:1997 national Information Systems Security Conference(NISSC 1997).pp-353-365(1997).

15.Jin.,S.Y.,Yeung,G.S.:DDOS Detection Based on Feature Space Modeling.In:3rd International Conference on Machine Learning and Cybernetics,Shanghai,pp. 4210-4215(2004)

8.2 GITHUB LINK:

https://github.com/NikhileshNicky/DDOS-ATTACK-DETECTION-ALGORITHMS.git