DoS and DDoS Attack Prevention Firewall Algorithm

A Project Report

Submitted in Partial fulfillment of Major Project for the award of Degree

Master of computer application

**MAJOR PROJECT REPORT**

**MCA**

Submitted by:

***Mehabub Hossain***

Dept. of Computer Science & Applications

**EXECUTIVE SUMMARY**

In this project DoS and DDoS attack prevention firewall algorithm is developed for the purposes on DoS and DDoS attack. The security problems of network resources in computer networks. Presently the main problem of computer networks are Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks which can block them. The methods mostly base on using firewall and IDS/IPS mechanisms to fight the attacks are not sufficient enough. In this project presents a new method for counteracting DoS and DDoS attacks - firewall rule with token bucket, packets implementation from Quality of Services method. This project gives possibility for user to finish they work which was started before the DoS and DDoS attack occurs and they do not suffer from Dos and DDoS attacks.

**Introduction**

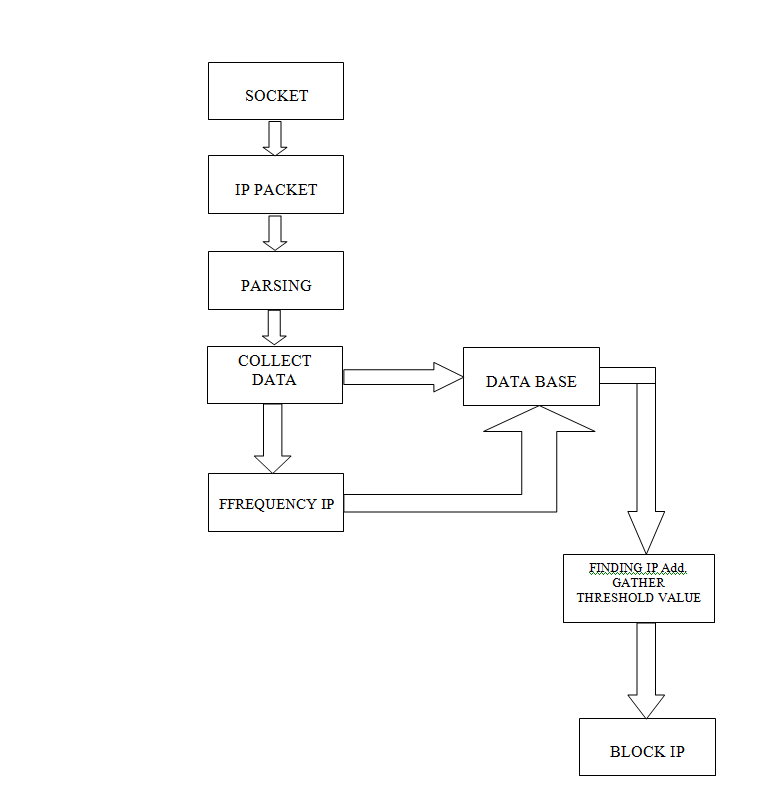
**1.1 Problem statement**

Dos and DDoS Attack Prevention Firewall Algorithm is developed for the purposes of protect on DoS and DDoS attack. The security problems of network resources in computer networks. Presently the main problem of computer networks are Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks which can block them. The methods mostly base on using firewall and IDS/IPS mechanisms to fight the attacks are not sufficient enough. In this project presents a new method for counteracting DoS and DDoS attacks - firewall rule with token bucket, packets implementation from Quality of Services method. This project gives possibility for user to finish they work which was started before the DoS and DDoS attack occurs and they do not suffer from Dos and DDoS attacks.

**1.2 Objective**

My project Dos and DDoS Attack Prevention Firewall Algorithm presents the potential use of implementation fuzzy observance for discovering and protecting network from suffering of Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks. DoS and DDoS attack able to block Web servers. Such attacks could be started from anywhere in the network. This project could be implemented on routers and predict the moment of the attack. Such prediction gives a possibility for the network administrators to protect server resources.

**Fig 1: Working process**



**1.2.1 Proposed System:**

* Firewall System is developed for the purposes of protect on DoS and DDoS attack.
* In this project, a regular performance will help protect server and websites.
* DoS and DDoS attack able to block Web servers. Such attacks could be started from anywhere in the network.

**1.2.2 Identification of Needs:**

* For Improve Performance of Firewall (DoS and DDoS tool).
* For Incoming packets details by which database can regularly check new source Ip.

**1.3 Risk Analysis:**

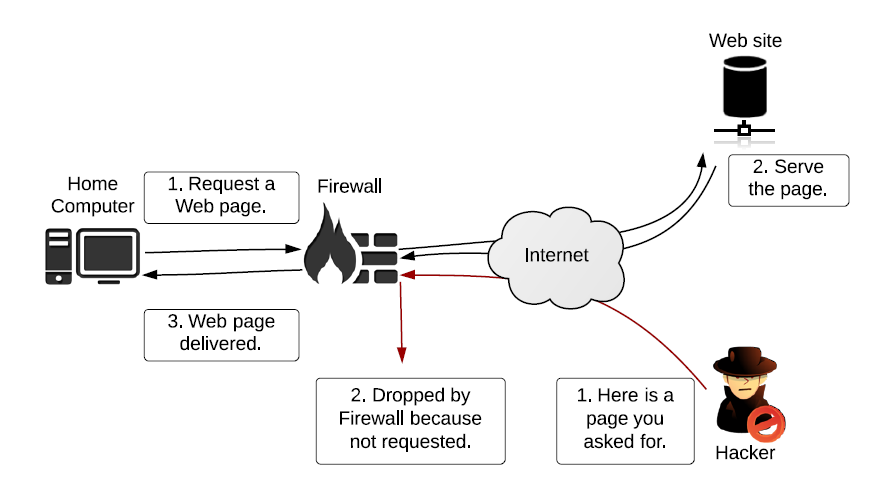
For every access request granted, a potential gap in security is created. The best way to combat unwarranted access is to preemptively identify and analysis area of vulnerability. However, the complex nature of firewall configurations combined with the time-consuming burden of patching tens of thousands of vulnerabilities makes threats difficult so see and access.

**1.4 Motivation**

Distributed denial of service (DDoS) and Denial of Service (DoS) is one of the most diffused types of Cyber attacks that represent a great concern for governments and institutions today. These attacks are an insidious foe to online service providers as their businesses depend on the availability of their web sites for critical business functions and productivity. My project Dos and DDoS Attack Prevention Firewall Algorithm protect this kinds of attack.

Computer networking has arguably been one of the most important advancements in modern computing. Allowing disparate applications to trade information, conduct business, exchange financial transactions, and even the routine act of sending an email are some of the most common things we do with computers today. Even with the advancement of ever faster computer chips, the trend continues to connect devices at an astounding rate. In addition, there is also a thriving mobile device market, thus increasing the amount of traffic owing between systems. An important aspect of this interconnected system is security. Without security, the convenience and speed of networked transactions would present more risk than the majority of applications could handle. In order to mitigate the risk and provide a much more secure communication channel, the firewall device was designed and deployed. It is one of the most widely used and important networking tools, existing in virtually every organization connected to a network. In fact, over the past two decades the landscape of network security has come to rely heavily on this single type of device. The primary purpose of a firewall is to act as the first line of defense against malicious and unauthorized traffic, keeping the information that the organization does not want out, while allowing approved access to flow illustrates how a firewall is commonly used in a network environment.

Fig2: Common Firewall placement in a network

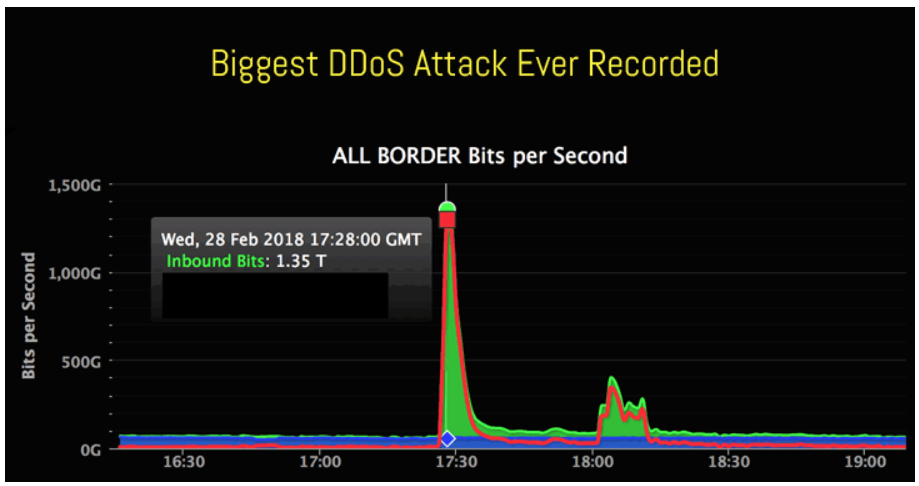


**1.5 Background & History:**

**1.5.1 Github Website:**

On Wednesday, February 28, 2018, GitHub's code hosting website hit with the largest-ever distributed denial of service (DDoS) attack that peaked at record 1.35 Tbps.

Fig 3: DDoS attack record

****

**1.5.2 Melbourne IT:**

Domain name registrar Melbourne IT, as well as two of its subsidiaries Net registry and TPP Wholesale, suffered a DDoS attack on April 13. The assault began at 10:00 local time, forcing the victimized organizations to inform customers that their cloud hosting and mailing platforms, among other services, were at the time unavailable.

By 11:30, the companies had returned normal service by implementing “our DDoS mitigation services as standard operating procedure and… international traffic management measures.” It took them another hour to tell customers that they had resolved the issues and that they would continue to monitor the situation.

**1.5.3 Dream Host:**

At 09:20 PDT on August 24, a DDoS attack deluged web hosting provider and domain name registrar Dream Host, knocking its systems –particularly its DNS infrastructure – offline. The Register’s Iain Thomson believes the attack originated from those who opposed the company’s decision to take on as Punished Stormer, a reincarnation of the neo-Nazi Daly Stormer website for which Cloud Flare Terminated Services following the Charlottesville protests, as a customer that same day. Dream Host mitigated the attack a few hours later.

**1.5.4 Electroneum:**

Electroneum cryptocurrency startup had crowdfunded $40 million worth of Bitcoin and Ether following an initial coin offering (ICO). Just before it launched its mobile mining app on November 2, the company’s website suffered a DDoS attack.

The campaign led Electroneum to lock investors out of their accounts while it worked to restore its network access. In the meantime, the Financial Conduct Authority took a moment to remind investors that ICOs offer no protection, which means investors should “be prepared to lose [their] entire stake.”

**1.6 System architecture:**

* Processor: Intel Core i3 Processor 1.8 GHz or higher.
* RAM: 2 GB or higher.
* Hard Disk: 80 GB or higher
* Internet Speed: 2g speed or higher

**1.6.1 Software Requirements:**

* Operating System : Linux
* PYTHON : Python 2.7
* Internet

**1.6.2 Language Used:**

* Python , Shell Script

**1.6.3 Database Used:**

* MariaDB 10.2 (MySql)

**1.7 Data Dictionary:**

**Incoming\_sourece Table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Field name | Data Type | Data Size | Description | Null | Key | Default | Extra |
| Ip | Varchar | 30 | Incoming packets | YES | - | NULL | - |
| Frequency | Decimal | 40 | IF DUPLICATE IP NOT FOUND INSERT THE NEW IP WITH FREQUENCY 1 | YES | - | NULL | - |

**1.8 Why Python:**

Python started as a scripting language for Linux like Perl but less cryptic. Now it is used for both web and desktop applications and is available on Windows too. Desktop GUI APIs like GTK have their [Python implementations](http://www.pygtk.org/) and Python based web frameworks like [Django](http://www.djangoproject.com/) are preferred by many over PHP et al. for web applications.

**1.9 Why Shell Script:**

A shell script is a computer program designed to be run by the Unix shell, a command-line interpreter. The various dialects of shell scripts are considered to decrypting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text.

**1.10 Organization of Chapter:**

In this Organization of the project is as follows. The whole project is divided into five chapters. First chapter is the Introduction. This chapter is fully dedicated to the problem statement, Objective, Motivation, Background etc. my second chapter is fully dedicated to the previous research made by the various researchers in the past and they are somehow related to this work. The different work along with the time line the author and their combination is mentioned n this reference. The third chapter of this project deals with the existing systems working on this topic. The fourth chapter of this project about result and execution testing. Finally the Fifth chapter of this project conclusion and future work.

**Literature Survey**

**2.1 Related Work:**

My project DoS and DDoS Attack Prevention Firewall algorithm presents the potential use of implementation fuzzy observance for discovering and protecting network from suffering of Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks. DoS and DDoS attack able to block Web servers. Such attacks could be started from anywhere in the network. This project could be implemented on routers and predict the moment of the attack. Such prediction gives a possibility for the network administrators to protect server resources.

The Internet has played an important role in society in many ways such as in economics, government, business and our daily personal life. Among various Internet based attacks, Denial of Service (DoS) attack is the most critical and provides continuous threat in cyber security. They are characterized as attempts to flood a network, disrupt connections between two computers, prevent an individual from accessing a service or disrupt service to a specific system. DoS attacks either forces a victim computer to reset, or consume its resources. Due to which, the targeted computer can no longer provide its intended services to its legitimate users. Early DoS attacks used to generate packets from a single source which was then aimed at a single destination. The evolution of the DDoS and DoS attack describes a single source attacks against multiple targets, multiple source attacks against single targets, and multiple source attacks against multiple targets. Around 2001, a new type of DoS attack became rampant, called a Distributed Denial of Service attack, or DDoS. In this case ,multiple systems are used to attack a single target. The flood of incoming traffic to the target will force to shut down the system. Due to which, the legitimate requests to the affected system are denied. As DDoS attack is launched from multiple sources, it very difficult to detect and block than a DoS attack. It, leads to revenue losses and increase the costs of mitigating the attacks to restore the services.

The denial of service attack is the most powerful damaging attacks used by hackers to harm a business or organization. This attack is one of most dangerous cyber-attacks. It causes service outages and the loss of millions, depending on the time of attack. In past few years, the use of the attack has enlarged due to the accessibility of free tools. This tool can be blocked simply by having a good firewall, but an extensive and clever DoS attack can avoid most of the restrictions. A Denial of Service attacks against Web sites occur when a hacker attempts to make the Web

server, or servers, unavailable for legitimate users and finally, to take the service slowing them down. This is attained by flooding the server's request queue with fake requests. After this, server will not be capable to handle the requests of genuine users. For some time, it was thought that these types of attacks were generally used against large companies, government sites, and activist sites as a form of protest to interrupt their Web presence. In general, there are two forms of the DoS attack. The first form is on that can crash a server. The second form of DoS attack only floods a service. Online Vulnerability Scanner is a tool which is capable to detect DoS Attack in Web application and compare its performance. We proposed an aegis algorithm which can be used to moderate DoS attack in Web application Vulnerability.

The increasing rate of Cyber Attacks based on the DDoS principle has created various new areas of concern for information security. It has also raised a pertinent question - Are we protected against such simplified attacks? With significant rise in the number of attacks and resulting reports of high vulnerability to Distributed Denial-of-Service (DDoS) attacks, perhaps we need to reconsider and revisit the pros and cons of the various Protocol such as ICMP, TCP/IP etc. This research is aimed at giving readers a brief outline of DDoS Attacks and its constituents, primarily the ICMP Protocol. We also propose an algorithm to carry out a DDoS Attack based on ICMP Flooding technique.

In this research work that is taken up, two specific attacks are being concentrated on; the UDP flood attack and Ping of Death attack. UDP Flood attack, user datagram protocol is a session less networking protocol. In this type of attack, attacker floods port on a remote host with numerous UDP packets and when the host checks for the packet, which is to be received at that port and when there is no legitimate packet found, it replies with an ICMP destination unreachable packet. This will ultimately lead to inaccessibility. The other attack is Ping of Death attack which involves the attacker sending numerous malformed packets or malicious pings to a computer which results in the overflow of memory buffers allocated for the packets causing the denial of service.

A “denial-of-service” attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. Examples include attempts to “flood” a network, thereby preventing legitimate network traffic attempts to disrupt connections between two machines, thereby preventing access to a service attempts to prevent a particular individual from accessing a service attempts to disrupt service to a specific system or person Denial-of-service attacks come in a variety of forms and aim at a variety of services. There are three basic types of attack: consumption of scarce, limited, or nonrenewable resources by sending illegitimate traffic there by denying service to the legitimate users.

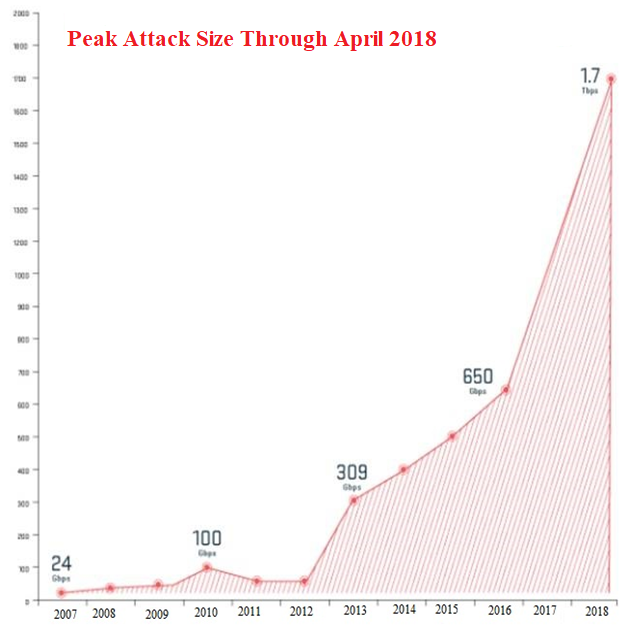
DoS attacks can generally be classified as either a Flood Attack or a Malformed (or crafted) Packet Attack and that where attacks originate simultaneously from several compromised

sources that these can be classified as Distributed DoS attacks. Fundamental to the IP protocol every packet has a source and destination address field that is used to determine the originating and destination end points. The process of forwarding these packets by intermediate routers partly relies on the destination field; the source address will only be used when a response to the packet is required. This makes the implementation of DDoS flooding attacks easy to accomplish because fake or “spoofed” source addresses can be used, and packets will generally be forwarded unchallenged to the specified destination. This allows a DoS or DDoS attack to be carried out from any location and with total anonymity.

If an attack is underway from a single address then it is possible to arrange for a “block” of the offending source IP address at the ISP or the border router. However, when a DDoS attack occurs the problem is not as easy to resolve because packets appear to be coming from hundreds or even thousands of different hosts, there is absolutely no point trying to implement temporary Access Control Lists on routing devices or modify the border Firewall rulebase, it is too late – you are left at the mercy of the attack under way.

Most recently since 28 Feb 2018 at 17:28:00 the world's largest distributed denial of service attack to date – measuring 1.35Tbps – knocked GitHub offline for a few minutes. The code repository GitHub was taken off air in a 1.3Tbps denial of service attack.

**Fig 4: Attack Size through April 2018**

****

* 1. **Types of DDoS Attacks:**

DDoS attacks can be generated in two different ways:

Direct attack and Reflector attack.

* **Direct attack:**

In a direct attack, a large number of attack packets are sent to the victim machine directly. In this attack, the attacker spoofs the source IP address so that the response is misdirected and goes elsewhere

* **Reflector attack:**

In case of an reflector attack, many innocent intermediate nodes known as reflectors (Botnets or Zombies) are used to generate an attack. An attacker sends packets that require responses to the reflectors with the packets’ inscribed source address set to the victim’s address. The attack packets can be constructed using TCP, UDP, ICMP or IGMP protocols.

* 1. **DDoS Attacks at Various Levels:**

**Ddos attack on application layer:**

An application layer distributed denial of service attack is usually initiated by hiring machines, bots, or taking control of remote systems. These components are used to ping multiple fake requests to server making the services of an application or server unavailable to its intended users. Such an attack targets everything that can eat huge chunks of the bandwidth, processing speed, and memory to slow down or disrupt services.

Examples of application layer attack are:

1. **HTTP Flood:**

In HTTP flood DDoS attack 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.

1. **Slowloris:**

Slowloris is a highly-targeted attack, enabling one web server to take down another server, without affecting other services or ports on the target network. Slowloris does this by holding as many connections to the target web server open for as long as possible. It accomplishes this by creating connections to the target server, but sending only a partial request. Slowloris constantly sends more HTTP headers, but never completes a request. The targeted server keeps each of these false connections open. This eventually overflows the maximum concurrent connection pool, and leads to denial of additional connections from legitimate clients.

1. **DDoS attack on network and transport layer:**

The main target of this type of attacks is to overwhelm the network infrastructure consisting of servers, routers and switches by sending a large volume of attack traffic. These attacks can be generated by exploiting protocol weaknesses. Network/Transport layer attacks can be further characterized according to degree of automation, exploited vulnerabilities, types of attack networks used, attacks rates generated, victim types and impacts of the attack.

Examples of network/transport layer protocol are:

**3.1 SYN Flood:**

A SYN flood DDoS attack exploits a known weakness in the TCP connection sequence (the ―three-way handshake‖), wherein a SYN request to initiate a TCP connection with a host must be answered by a SYN-ACK response from that host, and then confirmed by an ACK response from the requester. In a SYN flood scenario, the requester sends multiple SYN requests, but either does not respond to the host’s SYN-ACK response, or sends the SYN requests from a spoofed IP address. Either way, the host system continues to wait for acknowledgement for each of the requests, binding resources until no new connections can be made, and unlimitedly resulting in denial of services.

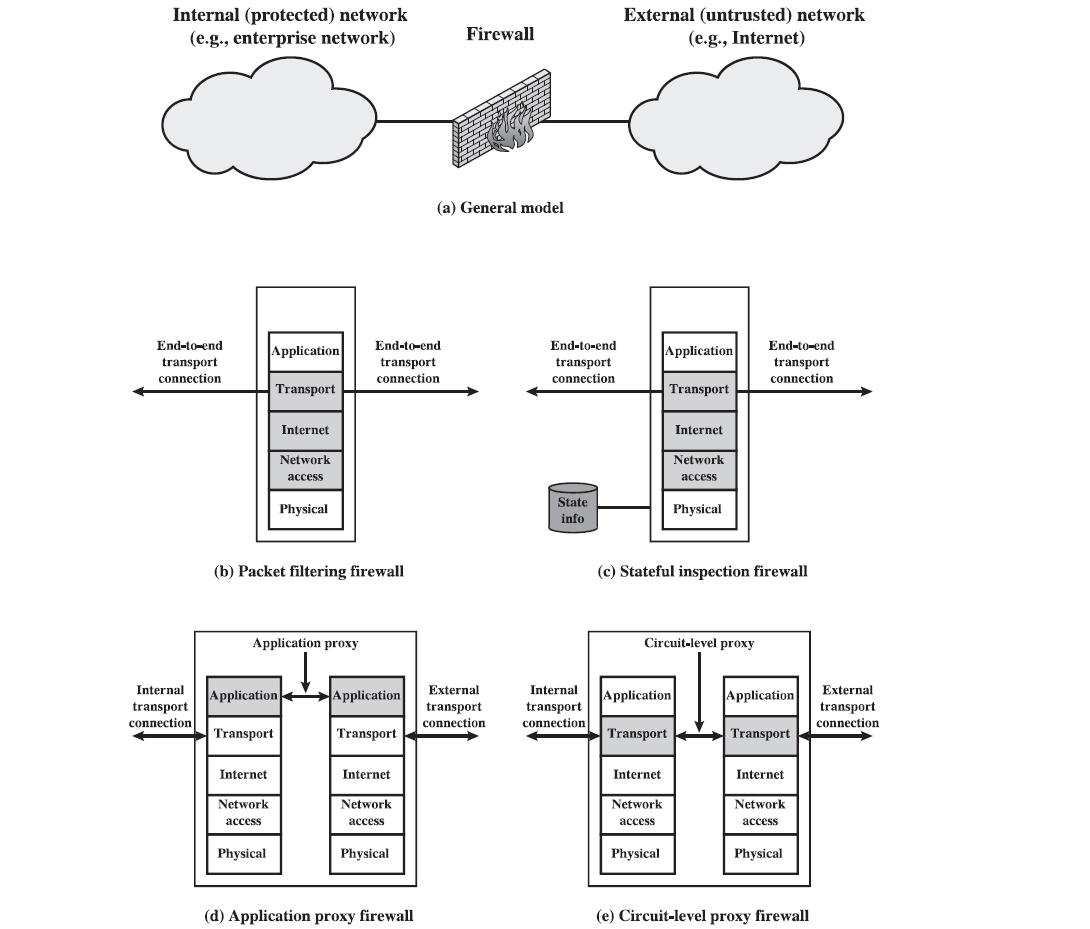
**3.2 UDP Flood:**

This DDoS attack leverages the User Datagram Protocol (UDP), a sessionless networking protocol. This type of attack floods random ports on a remote host with numerous UDP packets, causing the host to repeatedly check for the application listening at that port, and (when no application is found) reply with an ICMP Destination Unreachable packet. This process saps host resources, and can ultimately lead to inaccessibility.

**3.2ICMP (Ping) Flood:**

Similar in principle to the UDP flood attack, an ICMP flood overwhelms the target resource with ICMP Echo Request (ping) packets, generally sending packets as fast as possible without waiting for replies. This type of attack can consume both outgoing and incoming bandwidth, since the victim’s servers will often attempt to respond with ICMP Echo Reply packets, resulting a significant overall system slowdown.

**Process in Firewall:**



**Proposed Mythology**

**3.1** **Introduction:**

My project Dos and DDoS Attack Prevention Firewall Algorithm policy management is a challenging task due to the complexity and interdependency of policy rules. This is further exacerbated by the continuous evolution of network and system environments.

The process of configuring a firewall is tedious and error prone. Therefore, effective mechanisms and tools for policy management are crucial to the success of firewalls.

Existing policy analysis tools, such as Firewall Policy Advisor and f0ireman, with the goal of detecting policy anomalies have been introduced. Firewall Policy Advisor only has the capability of detecting pair wise anomalies in firewall rules. fireman can detect anomalies among multiple rules by analyzing the relationships between one rule and the collections of packet spaces derived from all preceding rules.

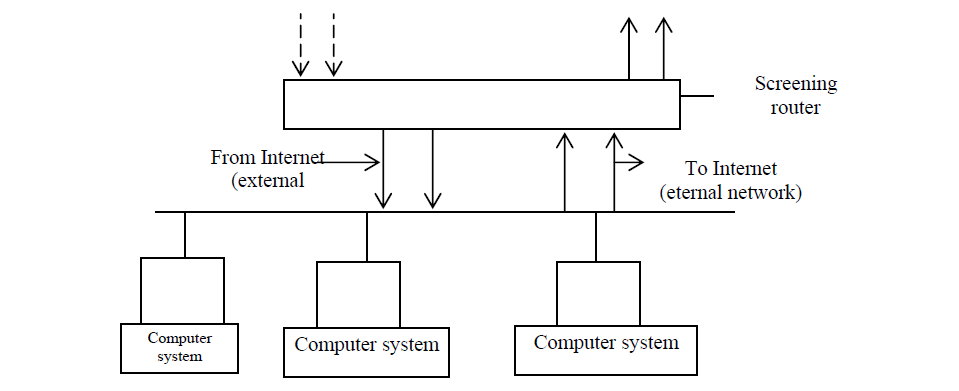
However, fireman also has limitations in detecting anomalies. For each firewall rule, fireman only examines all preceding rules but ignores all subsequent rules when performing anomaly analysis. In addition, each analysis result from fireman can only show that there is a misconfiguration between one rule and its preceding rules, but cannot accurately indicate all rules involved in an anomaly.

**3.2 Preprocessing:**

**3.2.1 Creating socket:**

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The *socket* library provides specific classes for handling the common transports as well as a generic interface for handling the rest.



**Using firewalls to do packet filtering**

**3.2.2 Receiving data:**

When I created socket then we are receive incoming IP packets data, for attackers send data.

**3.2.3 Parshing data:**

Parshing is a common programming task that splits the given sequence of characters of values into smaller parts based on same rules.

**3.2.4 Data Base connectivity:**

Before connecting to a MySQL database, make sure of the followings

#Database connecting

created a database FIREWALL

create table PACKET

This table has fields IP and FREQUENCY

User ID "ROOT" and password "mehabub" are set to access FIREWALL

**3.2.5 Data Base security:**

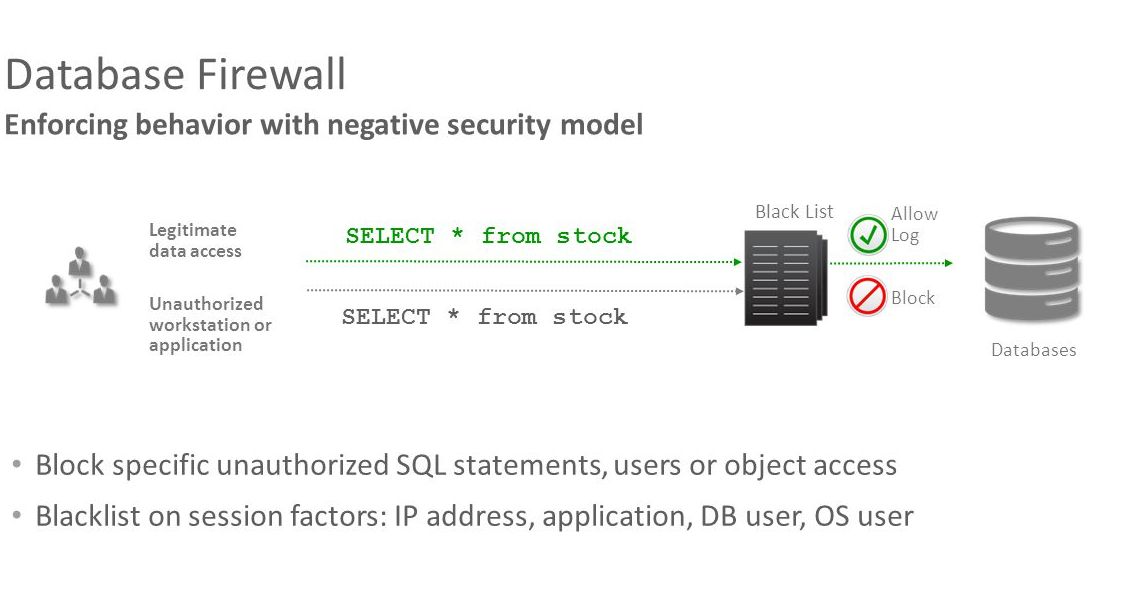


Fig 5: **Behavior of Firewall**

**3.2.6 Storing data in data base:**

All incoming packets stored in database on MariaDb.

#unpack data

iph = unpack('!BBHHHBBH4s4s' , ip\_header)

version\_ihl = iph[0]

version = version\_ihl >> 4

ihl = version\_ihl & 0xF

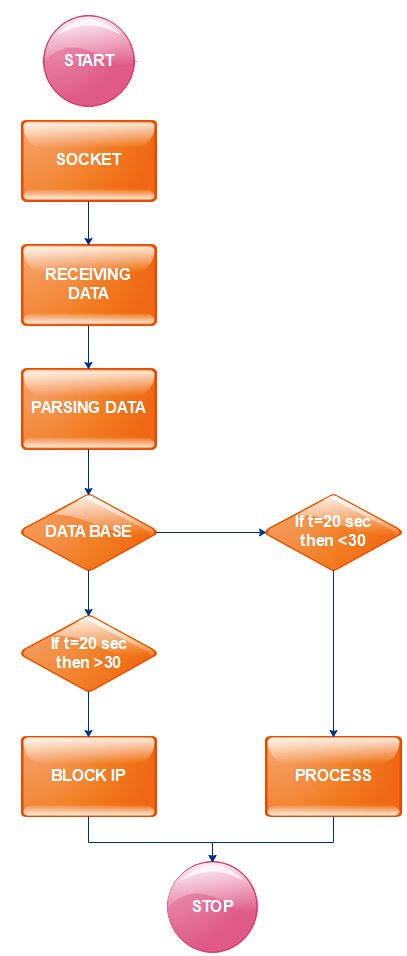
iph\_length = ihl \* 4

ttl = iph[5]

protocol = iph[6]

s\_addr = socket.inet\_ntoa(iph[8]);

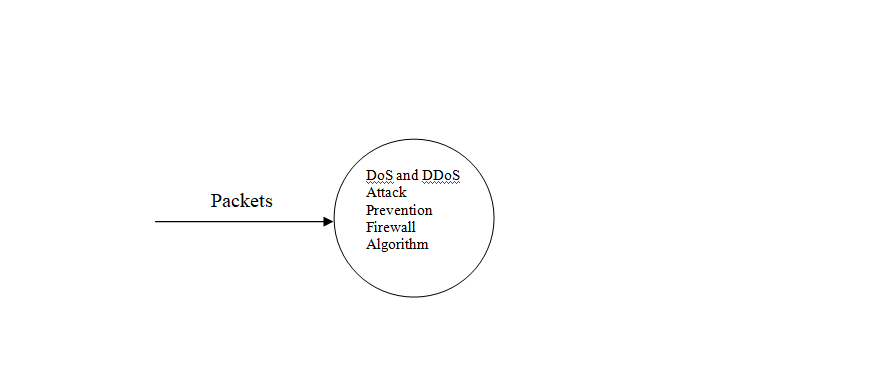
d\_addr = socket.inet\_ntoa(iph[9]);



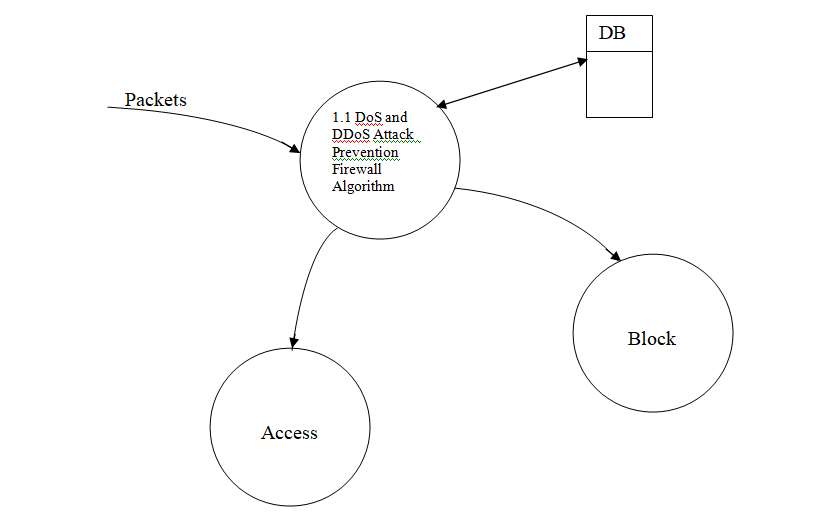
**3.3 FIREWALL System Flow chart**

**DFD**

**0 level DFD**

****

**1level DFD**

****

**3.4 0-level and 1-level DFD**

**3.5 CODE**

**Create a socket:**

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The *socket* library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

Sockets have their own vocabulary.

import socket

import sys

try:

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

except socket.error, msg:

print 'Socket could not be created. Error Code : ' + str(msg[0]) + ' Message ' + msg[1]

sys.exit();

print'Socket Created'

**Start MariaDB:**

root@DESKTOP-A9I9B2P:~ service mysql start

root@DESKTOP-A9I9B2P:~ mysql –u root –p

Enter password:\*\*\*\*\*\*\*

**Create database:**

CREATE database firewall

Use firewall:

Sql(firewall): create table packet(

Ip varchar(20),

Frequency number(20);

**Receive a packet:**

while True:

packet = s.recvfrom(65565)

#packet string from tuple

packet = packet[0]

#take first 20 characters for the ip header

ip\_header = packet[0:20]

**Unpack the packet:**

iph = unpack('!BBHHHBBH4s4s' , ip\_header)

version\_ihl = iph[0]

version = version\_ihl >> 4

ihl = version\_ihl & 0xF

iph\_length = ihl \* 4

ttl = iph[5]

protocol = iph[6]

s\_addr = socket.inet\_ntoa(iph[8]);

d\_addr = socket.inet\_ntoa(iph[9]);

tcph = unpack('!HHLLBBHHH' , tcp\_header)

source\_port = tcph[0]

dest\_port = tcph[1]

sequence = tcph[2]

acknowledgement = tcph[3]

doff\_reserved = tcph[4]

tcph\_length = doff\_reserved >> 4

h\_size = iph\_length + tcph\_length \* 4

data\_size = len(packet) - h\_size

**FETCHING DUPLICATE IP, IF EXIEST IP:**

sql1="select \* from PACKET where IP='"+sip+"'"

cursor.execute(sql1)

results = cursor.fetchone()

if results==None

**DUPLICATE IP NOT FOUND INSERT THE NEW IP WITH FREQUENCY 1:**

sql = "INSERT INTO PACKET VALUES ('"+sip+"','1')"

cursor.execute(sql)

print "RECORD Inserted Successfully"

db.commit()

**DUPLICATE IP FOUND THE INCREASE THE FREQUNCCY BY 1:**

tempcount=int(results[1])+1

try:

cursor.execute("update PACKET set FREQUENCY='"+str(tempcount)+"' where ip='"+sip+"'")

print "update Successfully"

db.commit()

**Data store in Txt file and show execution time:**

f = open("ip.txt", "a")

f.write(sip + '\n')

t2=datetime.now()

t=t2-t1

print t

**When threshold value greater than 20 then block the source IP:**

#sql2="select IP from PACKET where IP>20"

print cursor.execute("select IP from PACKET where IP>20")

results = cursor.fetchone()

print results

print "Find Successfully"

t1=datetime.now()

except :

print "Error"

**3.6 Proposed Algorithm:**

The research paper have found an algorithm prevention algorithm to cracking ddos and DoS attack by limiting the no of access to user. This helps to determine whether user is ddos attacker user. When an attackers using genuine address, the proxy server uses the Deficit Round Robin algorithm to collect the address of the client request. When source IP send request >20 sec, then block this particular IP.

Step 1 Maintain the database for the list of users,X

Step 2 Analyse the User

Get the packets of the incoming ip.

Packet=name of the incoming ip

Match it with the user list in the database

For i=0 to X.count

If ip=X(i).Name then

X. ip\_count++

Status=―Registered‖

Else

X. ip\_Count++

Status=―Unregistered‖

End if

Next

Step 3 Response to the Request

If Status=―Registered‖ then

Process the Request and send the Response

End if

If Status=―Unregistered‖ then

Add name to the alert list, A

A.Name=ip

A.Alert\_count++

If A.Alert\_count >Threshold\_Value

If Server\_peak\_period=True

Add ip to Temp\_Blocked List

Temp\_Block=ip

End if

Else

Block the user permanently

P\_Block=ip

End if

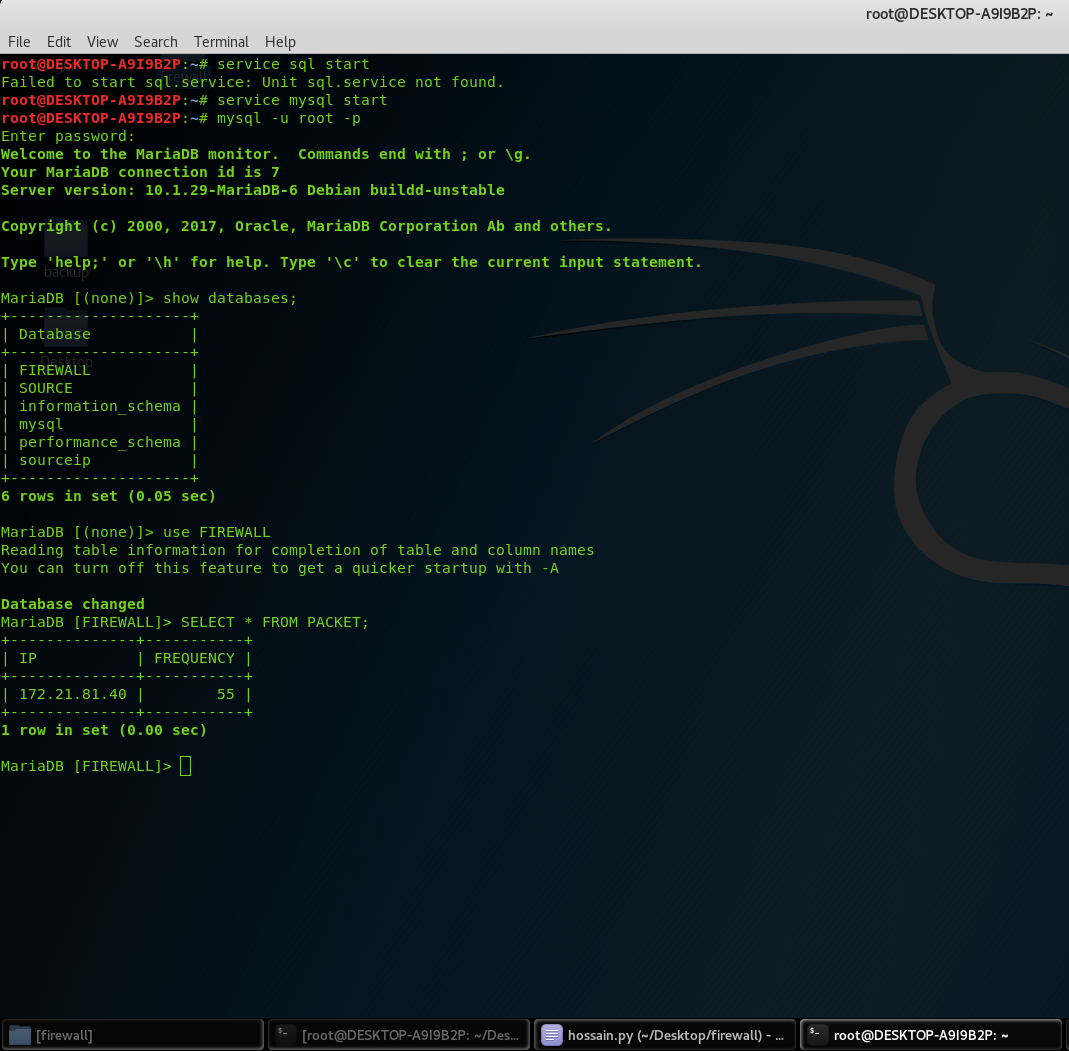
End if

End if

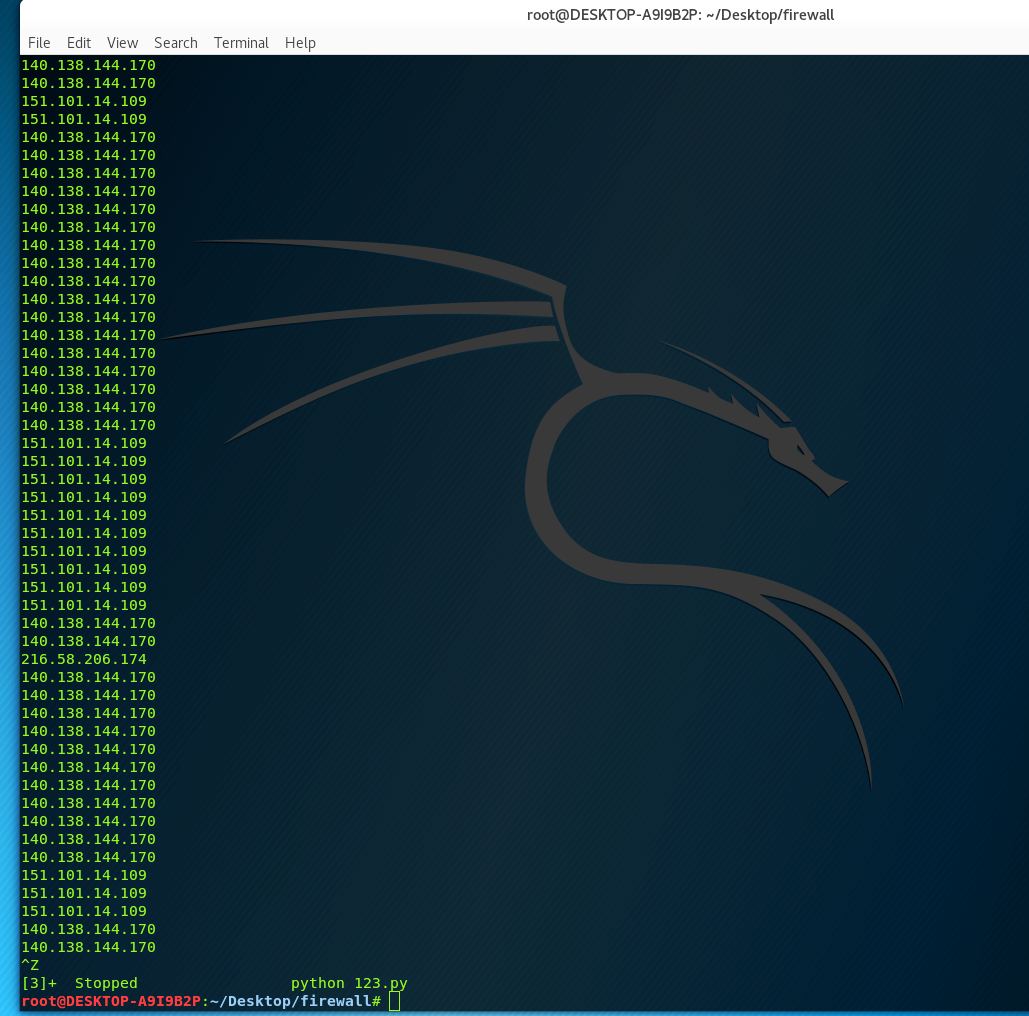
Result and testing

Snapshot

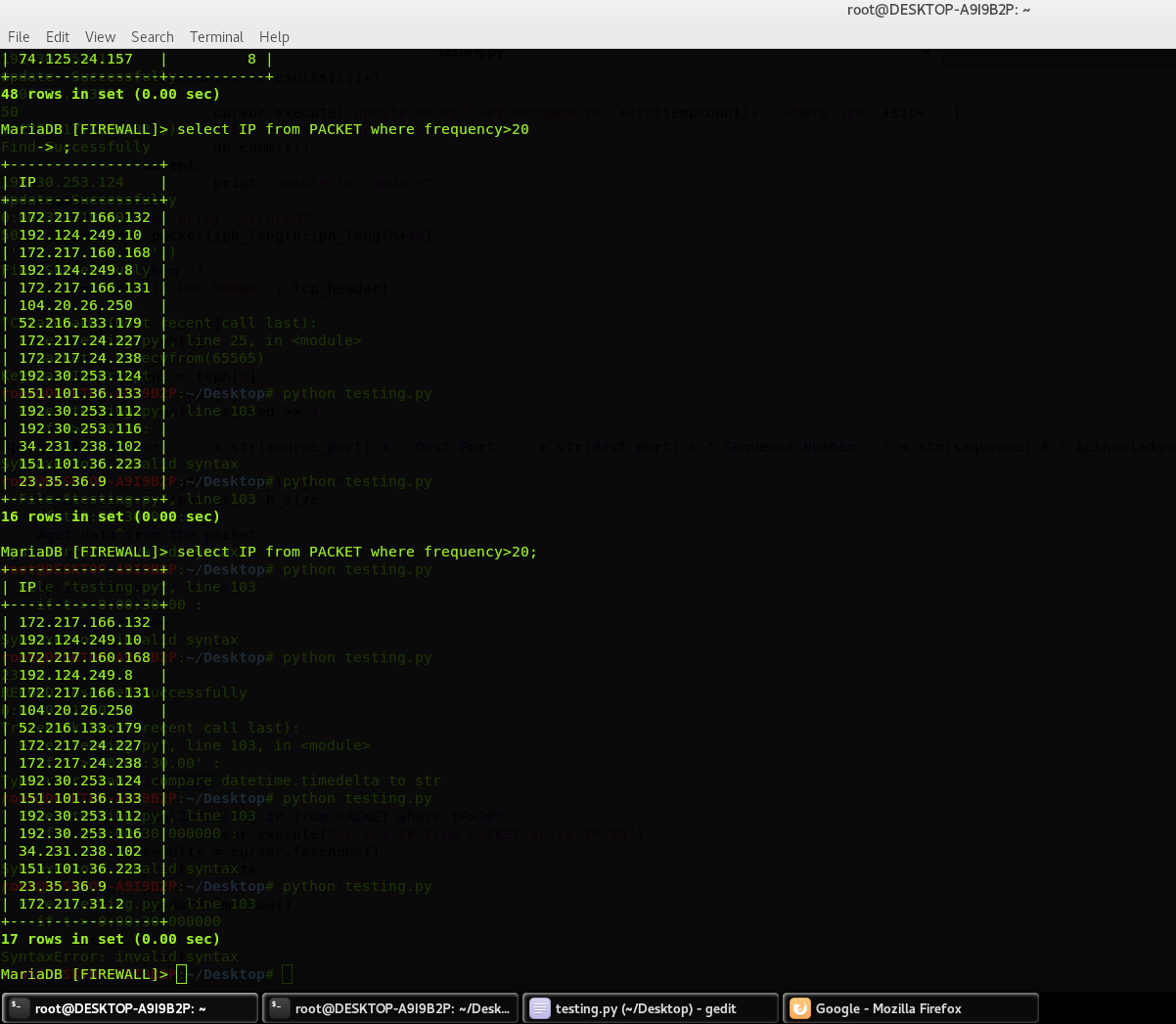
**Connect MariaDB database**



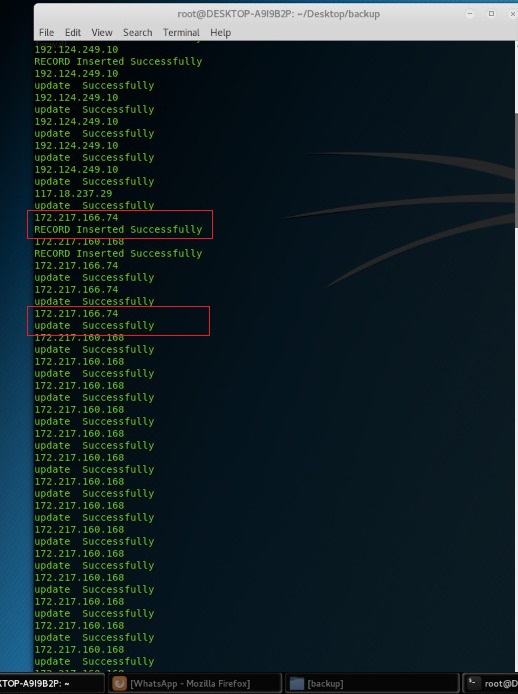
**Incoming packets**



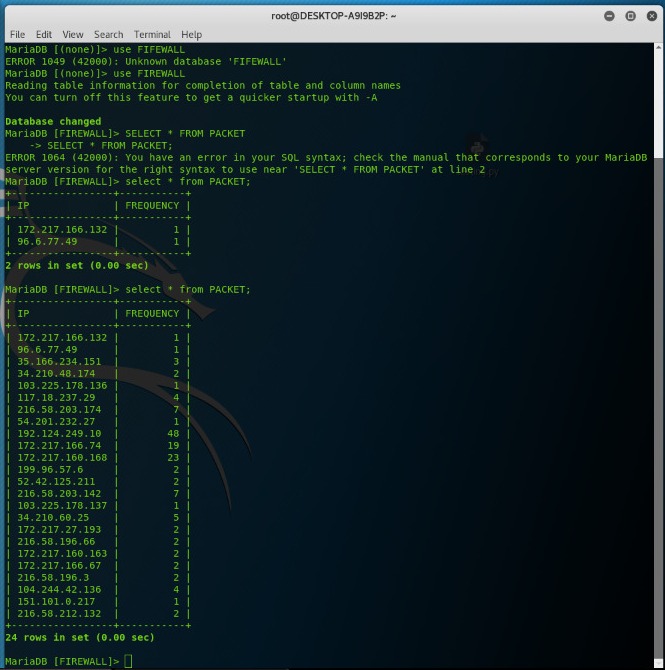
Store in Data base



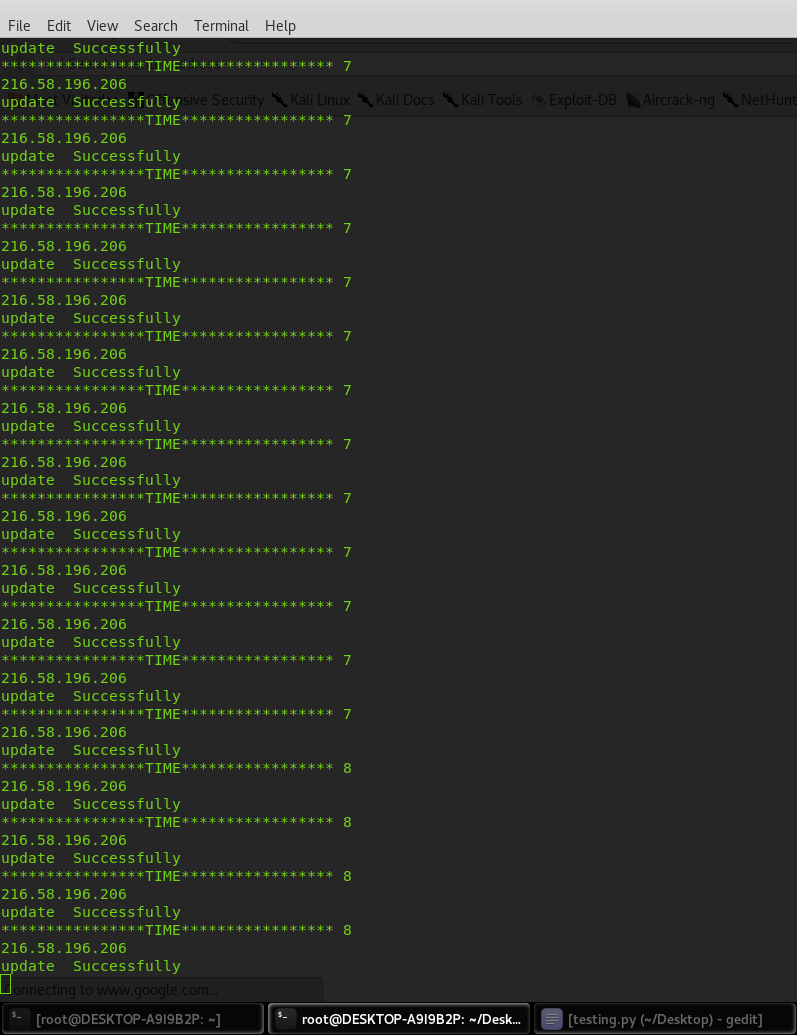
**If Duplicate IP not found insert the new IP with frequency 1. And Duplicate IP found then increase the frequency by 1:**



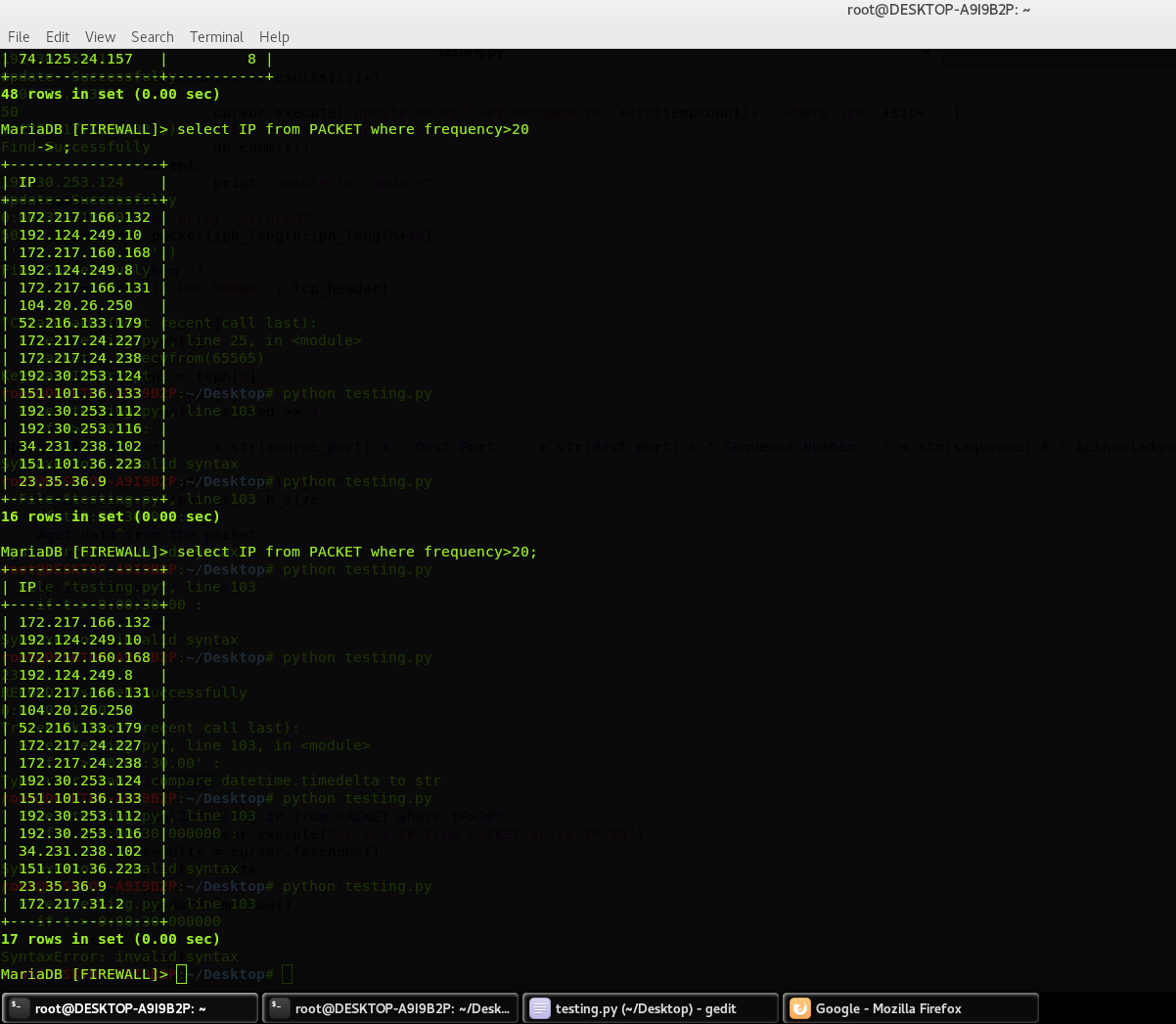
**Insert data in database with increase frequency**



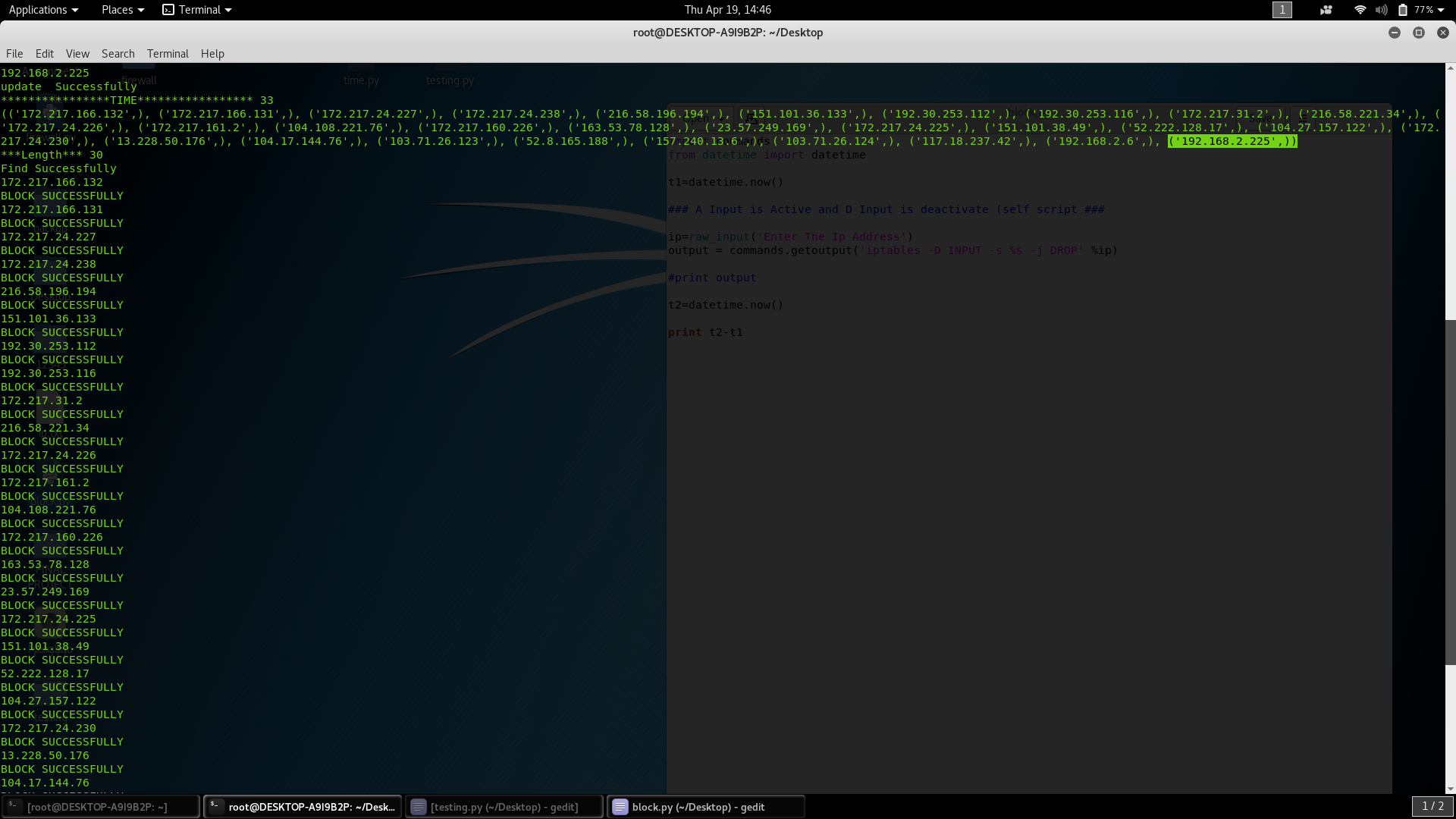
**Sniffing the Incoming packets in Time limit 30sec**

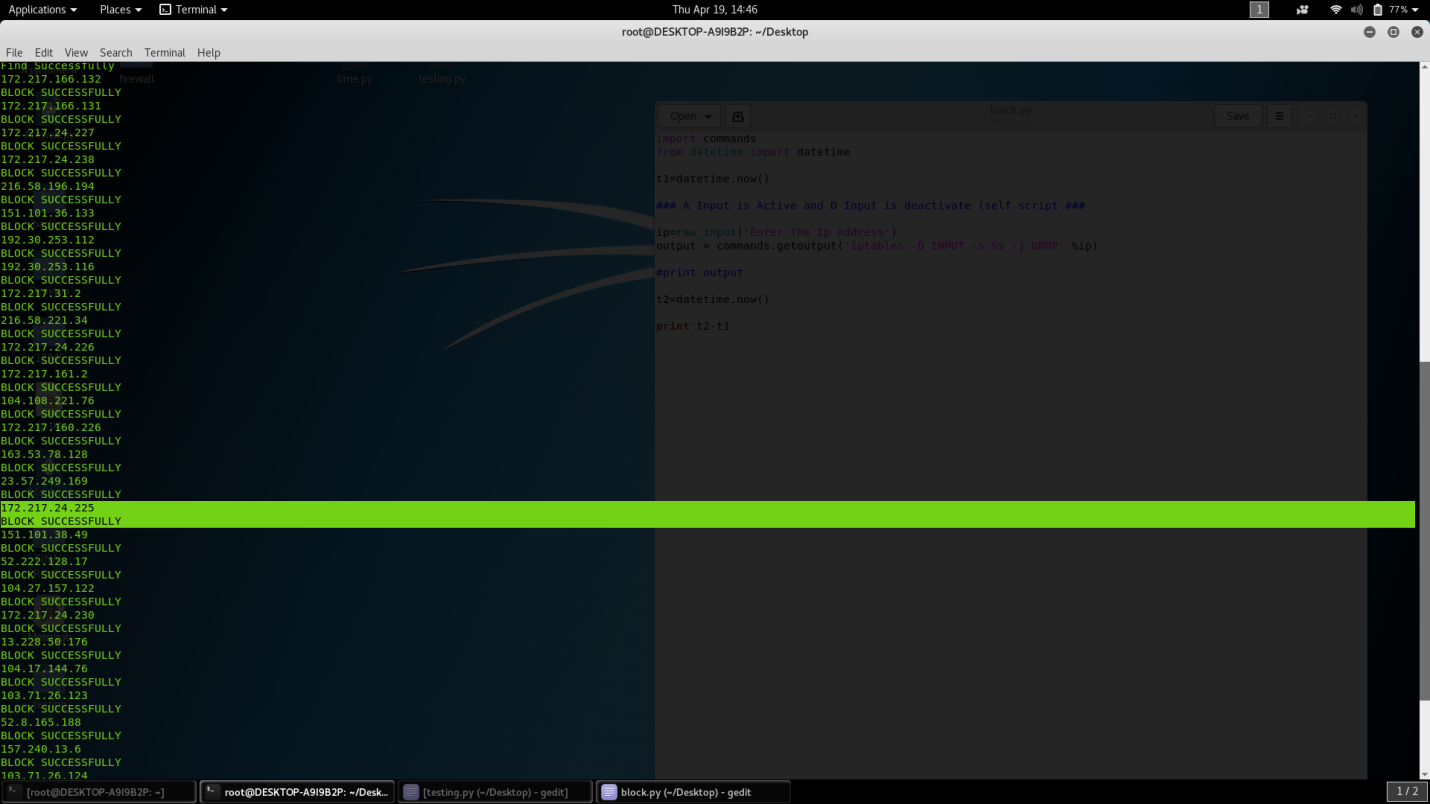
****

**Show ip packets in database >20 in 30 sec**

****

**When cross the threshold values >200 then block this packets**

****

****

**4.1 Test case design**

**4.1.2 Test Case 1**

1. Validation error messages should be displayed properly at correct position.
2. All error messages should be displayed in Terminal.
3. All numeric values should be formatted properly.
4. Algorithm tips should be meaningful.
5. All inputs packets show in binary format.
6. Check all timers.
7. Checked for threshold value.
8. Input data with first to last position is correctly inserted.
9. Less than 30 sec crossed the threshold values than block this IP.
10. Check all values.
11. Check repeated value an algorithm.
12. When repeated value insert then show the message. (insert repeated value)
13. When new value insert then show the message. (New IP insert)
14. All numeric values should be formatted properly.
15. Block DoS DDoS attack IP then sow output.(Block this IP)
16. Check for all incoming packets.
17. Check for all incoming frequency.
18. Check for all incoming time.
19. Check for all single IP value.
20. Check incoming source downloadable in .txt file.

**4.1.3 Test Case 2**

**4.1.3 Database Testing Test Scenarios:**

1. Check if correct data is getting saved in database upon successful data submit.
2. Check values for columns which are not accepting null values.
3. Check for data integrity. Data should be stored in single or multiple tables based on design.
4. Store data names should be given as per the standards e.g. FIREWALL\_<Tablename>\_<ColumnName>
5. Tables should have primary key column.
6. For every database add/update operation log should be added.
7. Required table index should be created.
8. Data should be rolled back in case of failed transaction.
9. Check the field in minimum and maximum values.
10. MariaDB database use for Allgorithm
11. Check the threshold value.
12. Check if database fields are designed with correct data type and data length.
13. Check if input data is not truncated while saving.
14. Test stored procedures and triggers with sample input data.
15. Check if data is committed to database only when the operation is successfully completed.
16. Check if all tables constraints.
17. Check all store input packets and frequency.
18. Check all thresholds values.
19. Database logical names should be given according to database name.

**4.2 Testing:**

Testing is the process of evaluating a system or its component(s) with the intent to find

whether it satisfies the specified requirements or not.

Testing is executing a system in order to identify any gaps, errors, or missing requirements

in contrary to the actual requirements.

We have done Unit Testing and Alpha Testing.

**4.3 Black Box Testing:**

We test the software from a user’s point of view and also perform testing without seeing the internal system code.

Steps to Perform Black Box testing which we follow in our project:

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario). Also some invalid inputs

(Negative test scenario).

* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test case are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any fixed and re-tested.

**4.4 White Box Testing:**

As we know, White Box Testing is a testing technique which evaluates the code and the internal structure of a program. So we do white box testing on our project. And it is done by both the developers as well as testers.

It helps us to understand which line of code is actually executed and which is not. This may indicate that there is either a missing logic or a typo, which eventually can lead to some negative consequences.

Steps to Perform White Box Testing which we follow in our project:

* Understand the functionality of an application through its source code. It means that a
* Tester must be well versed with the programming language and the other tools as well
* Techniques used to develop the software.
* Create the tests and execute them.

**4.5. Test Execution:**

**4.5.1 Unit Testing:**

A testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules.

The main aim is to isolate each unit of the system to identify, analyze and fix the defects.

**4.5.2 Integration Testing:**

Integration testing focuses on unit tested modules and build the program structure that

is dictated by the design phase.

**4.5.3 Validation Testing:**

It ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate Environment.

We validate our project with captcha. Also validate the password. And validate the format of all fields like email format. Etc.

**4.5.4 System Testing:**

System testing tests the integration of each module in the system. It also tests to find discrepancies between the system and its original objective, current specification and system documentation. The primary concern is the compatibility of individual modules. Entire system is working properly or not will be tested here, and specified path MariaDB connection will correct or not, and giving to the system and by comparing with expected outputs.

**5.1 Conclusion**

My Project Dos and DDoS attack prevention algorithm will help Dos and DDoS attack. In this dissertation many elements of firewalls and networks were covered with the goal of providing a framework for simulating and comprehending the large solution space represented by the addressable network space.

**5.2 Future Scope**

* In future, we will try fully use AI system.
* In future, we will try to capture the attacker’s location.
* In future, we will try to protect all types of attacks.
* Firewalls combine the capabilities of traditional firewalls , including packet filtering, network address translation (NAT), URL blocking and virtual network (VPNs) with Quality of service(QoS) in future

**5.3 Bibliography**

The following books were referred during the analysis and execution place of the project

PYTHON:

Python PDF, Python Tutorial point

MYSQL:

MYSQL DOCS and tutorialspoint.org

Self Script

Spilling the Script: A Concise Guide to Self-Knowing

**5.4 Reference**

[www.google.com](http://www.google.com)

<https://www.akamai.com/us/en/resources/ddos-attacks.jsp>

[www.w3schools.com](http://www.w3schools.com)

<https://en.wikipedia.org/wiki/Firewall>

Cisco (2017) Cisco systems packet tracer

Firewall policy modeling, analysis and simulation: a survey. Sourceforge, Tech. Rep.

S. A. Joshi and Varsha S. Pimprale, “ Network Intrusion Detection system (NIDS) based on Data Mining”, International Journal ofEngineering Science and Innovative Technology (IJESIT), ISSN: 2319-5967, Volume 2, Issue 1, January 2015

A Countermeasure to Resource-Inflated Denial-of-Service Attacks, IEEE transactions on information forensics and security, vol. 10, no. 1, january 2016.