JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Denial of Service Detection and Mitigation

Amey Todkar[1], Asawari Walkade[1], Nehal Gahlot[1], Dr. Jaiswal Rupesh[1]
[1]SCTR's Pune Institute of Computer Technology, Pune, 411043

Corresponding Author: Asawari Walkade (e-mail: asawariwalkade03@gmail.com)

ABSTRACT*:* Everything on the internet runs on systems connected together forming a network of nodes spanning billions of devices. With such networks some parts of networks are segregated for specific purpose or for specific companies, some run special services & apps, some are private and some are public, but with all of these different types of networks means that there will be different types of data. Some reliable and other not so much. In information security, availability plays an important role. It stands for providing specific data to specific authorised people whenever they need in whichever format, they need it. DoS or Denial of Service attacks this very concept. Though DoS has been a problem since 2000, the attacks have only evolved and are still a persistent threat. But what must be considered is that even though the attacks have become way more sophisticated than before, the main concept behind DoS remains the same. Overload a service, app, node or a network with excessive data or traffic and crash it thus rendering it unavailable to users. But this also means in the end it's all about packet signatures which will be at the core of these attacks. This implementation and research work concentrates on capturing & analysing traffic in a network to compare it with attack signatures to detect a DoS attack and try to block it before it can harm the system. A signature-based detection technique and a customizable IDS was employed.

INDEX TERMS: Denial of Service, DOS Attack, Detection, Mitigation, SNORT, IDS.

1 INTRODUCTION

DoS attacks have been prevalent since the year 2000 to induce temporary or permanent unavailability in a system or network, thus causing data loss, delays in service and even financial losses for huge companies. Although at start the attack was rather basic flooding of a service on a webserver, in 2021, the attack has numerous types and can attack anything from a router to a service. Especially after 2020 Covid pandemic, the implications of DoS attacks have sky-rocketed as everything moved online and single attack could mean heavy losses all around.

The approach that was adopted to this implementation was to divide the process to reach the end goal into different steps. For each step, researching, learning and implementation was required. The very first step was to set up a virtual lab to execute DoS attacks. This would be followed by setting up an IDS in the target system to capture packets. This IDS was then configured to detect malicious packets using rules and signatures.

2 LITERATURE SURVEY

'Review of detection techniques against DDoS attacks on an SDN' by Haider Dhia, Mohammed Anbar & Yw Chong talks about differentiation between the methods of detection in SDN. They talk about Time-based Detection, Entropy-based Detection and Low-Traffic Flows detection which are all Anomaly-based detection. It talks about how the reference lines are set in each technique and their pros and cons [3]. 'Methodologies for detecting DoS/DDoS attacks against Network servers' by Mohammed Alnezi and Martin Reed tells us about the different classifications of IDS and how they can be used for DoS and DDoS detection. The paper also covers the detection techniques categorized based on the different characteristics used somewhat similar to the previous paper [3] but with a more practical rather than theoretical approach, talking about using TTL and other IP headers, traffic arrival rates, tracking source IPs and so on. [4] "Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks." by Birkinshaw goes over the use of SDN to tackle the DoS and port scanning attacks. Although SDN was not used directly, the paper was very enlightening about the different ways an attack can be simulated, the network topologies used in SDN, different countermeasures used against network attacks and configuring IPS. [1] "Evaluating Machine Learning Algorithms for Detecting DDoS Attacks" by Manjula Suresh and R. Anitha brought up the various machine learning algorithm, to monitor and sort out the data traffic travelling in a network, useful in intrusion detection [1].

"A Design for Building an IPS Using Open-Source Products" by Mike Smith from SANS institution is about building an IPS, not only for DoS but for all kinds of attacks, using open-source tools. Mike has used Snort for its different features and ease of writing rules. He also uses the BASE console for Snort and talks about why BASE and not the other open-source console ACID. Because configuring Snort as an IPS is a very tedious and error-ridden task, he used SnortSAM interface to build Snort in an IPS. The paper helped a lot in selection of components for the implementation [8]. Along with the paper by Mike Smith the manual or documentation of Snort was also referred to understand the installation, dependencies and configuration process. It illustrates in detail how to write rules along with customizations and different output functions. It also introduced us to the Spade plugin to implement the anomaly-based detection along with Snort which had its own documentation [7][9]. "Mitigation of DoS and Port Scan Attacks Using Snort." by Alka Gupta covers the Snort implementation for different DoS attacks and port scan attempts as well as the attack tool implementations for the same [2].

DoS is basically flooding a target system with data to crash its resources. For example, say a router was target. I would simply spoof multiple IP addresses and connect to the router thus overloading the routing table and making genuine connections to the router impossible. Here the target was a network device, and the packets were basic routing protocol frames. With different targets the data being sent changes therefore giving rise to different DoS attacks. 'How you send data' is also a way to differentiate between DoS attacks. For example, was the data sent from one attacking system or multiple? Was it all sent in one go or was it sent with delays?

Some very basic types of DoS attacks are:

1. SYN flood: SYN packets are submitted by the client to request a connection to the server as a part of the TCP handshake. Attacker will flood the server with huge number of SYN packets from spoofed IPs.
2. LAND attack: A fabricated SYN packet with same destination and source IP is sent to the target server, thus causing a race condition on the server.
3. SYN-ACK Flood: Like SYN flood, this attack will open the communications channel on the server but instead of flooding the server with SYN packets it forces the server to create numerous SYN-ACK packets.
4. ACK&PUSH-ACK flood: After the TCP connection is established the packets sent are ACK and PUSH-ACK. The attacker fabricates these packets in such a way that the server can't differentiate between genuine and false traffic. Another specialised version of this same attack is using fragmented ACK packets of max size 1500 bits. This is used mainly on network devices like routers who end up exhausting their resources trying to assemble these packets again. These packets also are hard to detect in an IPS.
5. Different protocol floods:
    a) UDP flood: As UDP doesn't have a handshake mechanism the IP control methods are very scarce. This allows for huge volume of traffic to be generated and spoofed on services using UDP
    b) DNS flood: Attacks a DNS server by creating multiple fake DNS requests using UDP protocol.
    c) VOIP flood: UDP flood attack that targets VOIP
    d) NTP flood: Publicly accessible NTP servers are used to create large numbers of UDP packets on a certain network
    e) SNMP flood: Target network device is sent many small packets from spoofed IPs and as all the listening devices on the network try to reply to these packets, the network can't cope with such a load and crashes
    f) HTTP flood: This is an application layer attack which uses GET & POST methods in HTTP to flood a server and exhaust it's resources. This is mostly executed using a botnet.
    g) Recursive HTTP flood: Just like web-spider tool would, multiple IPs will send GET request for each resource on the server.
    h) ICMP flood: A flood of ICMP echo requests
    i) IP Null attack: When the IP headers containing invalid or null headers are sent some servers cannot process these headers and spend their resources trying to make sense of these packets

j) Smurf Attack: multiple devices are infected with malware which on the trigger will make the device ping a certain IP address.

k) Ping of Death: These attacks have ping packets way larger than their normal size causing the server to allocate more than usual resources and thus exhausting them

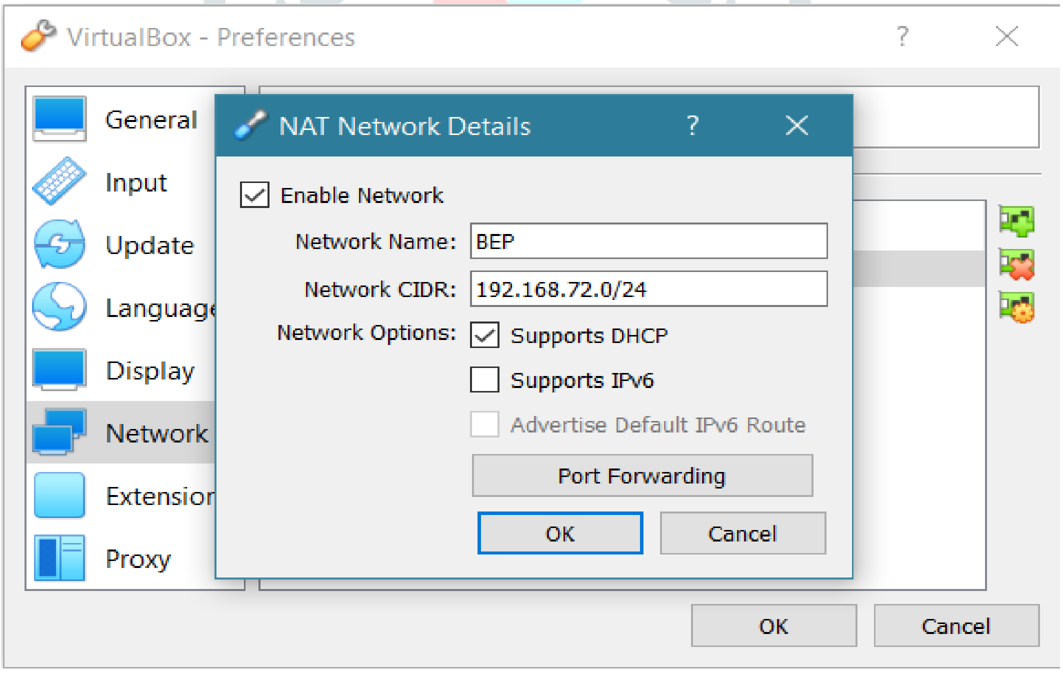
Though some of the attacks have very readable signatures like those in the ping of death, some like in DNS or HTTP Flood are very hard to catch. This job of detection and mitigation of DoS or any attack for that matter falls upon the IDS or IPS.

IDS (Intrusion Detection System) or IPS (Intrusion Prevention System) are the systems, either host-based or network-based, that are tasked with detecting, logging, and acting upon attacks known or unknown. The two methods that an IDS or IPS uses are Signature-based and Anomaly-based.

A signature-based IPS will read/capture the traffic on the network or host, compare the packets with a set of signatures and decide according to pre-set rules on the action to be taken. Whereas an anomaly-based IPS will understand what the 'normal' traffic for the network or the host is. Then it will use this 'normal' as a reference line to decide if the traffic is malicious or not. The signature-based systems will be very reliable and will stop most known attacks with very low chances of false positives. It will also be fast relative to anomaly-based systems which although unreliable and slow can stop unknown or zero-day attacks. Also, Role of ML is increasing in the area of attack traffic detection and mitigation [21-44].

3 THE IMPLEMENTATION PERFORMED

The implementation was divided into four steps. The first step, which was to build a virtual lab, was completed using VirtualBox Hypervisor. Two VMs were created, one with Parrot OS which would act as the attacker and the second with Ubuntu OS which would act as the target system. These VMs were then configured to connect to each other in a LAN network using the NAT Network option provided in VirtualBox.



NAT Network Details

☑ Enable Network

Network Name: BEP

Network CIDR: 192.168.72.0/24

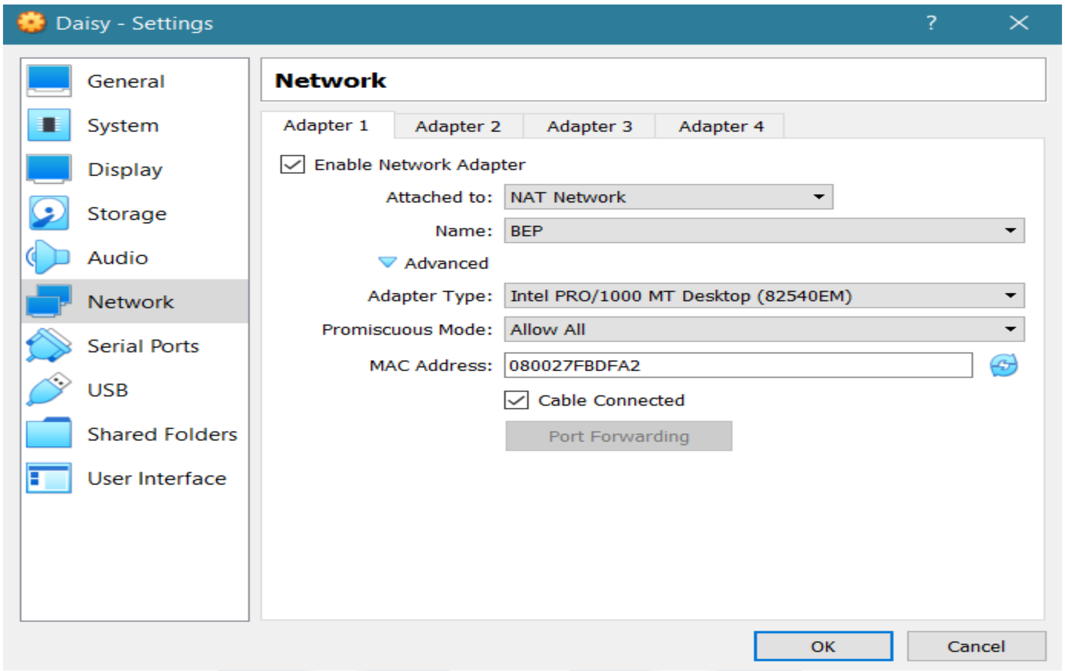
Network Options: ☑ Supports DHCP

□ Supports IPv6

□ Advertise Default IPv6 Route

Fig 1: NAT Network configuration

Thus, both the VMs acted as if they were part of the same LAN with the CIDR IP address of 192.168.72.0/24. Both the NICs were configured in promiscuous mode to allow capturing of all kinds of packets.



Network

☑ Enable Network Adapter

Attached to: NAT Network

Name: BEP

Advanced

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

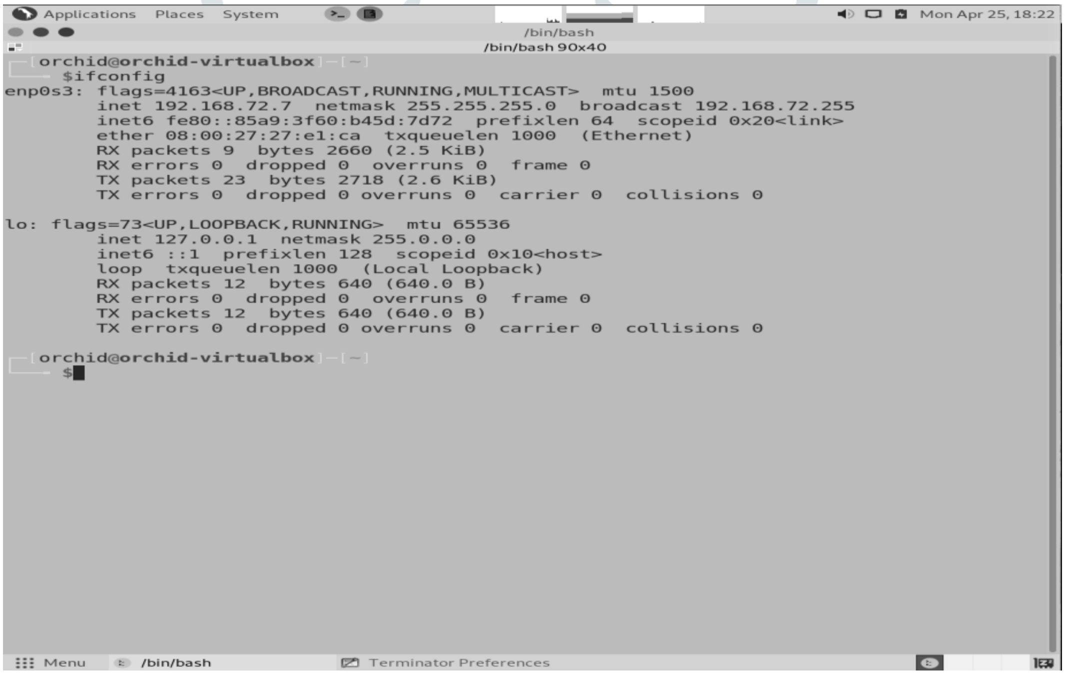
Promiscuous Mode: Allow All

MAC Address: 080027FBDFA2

☑ Cable Connected

Fig 2: NIC configuration for one of the VMs

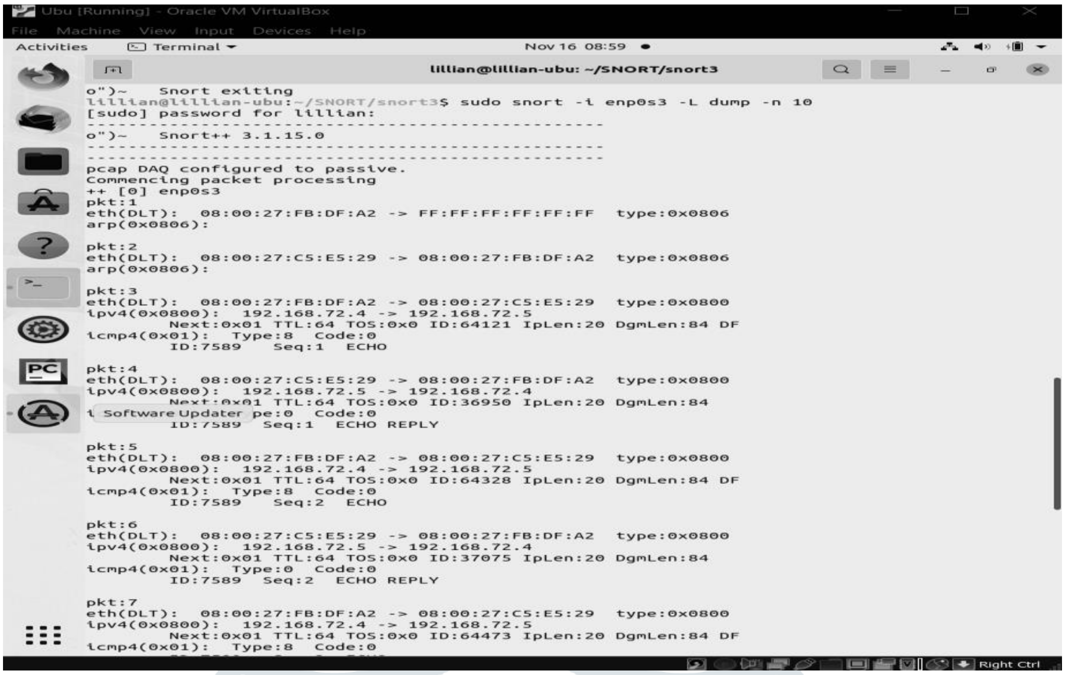
After this, Snort3 was installed. After the installation was done the first thing to be checked was the packet capture mode of Snort. The expected result was to capture each packet, and this was also accomplished.



```
┌─[orchid@orchid-virtualbox]─[~]
└──── $ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.72.7  netmask 255.255.255.0  broadcast 192.168.72.255
        inet6 fe80::85a9:3f60:b45d:7d72  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:27:e1:ca  txqueuelen 1000  (Ethernet)
        RX packets 9  bytes 2660 (2.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 2718 (2.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 12  bytes 640 (640.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 640 (640.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

┌─[orchid@orchid-virtualbox]─[~]
└──── $
```

Fig 3: The network config of the attacking system.

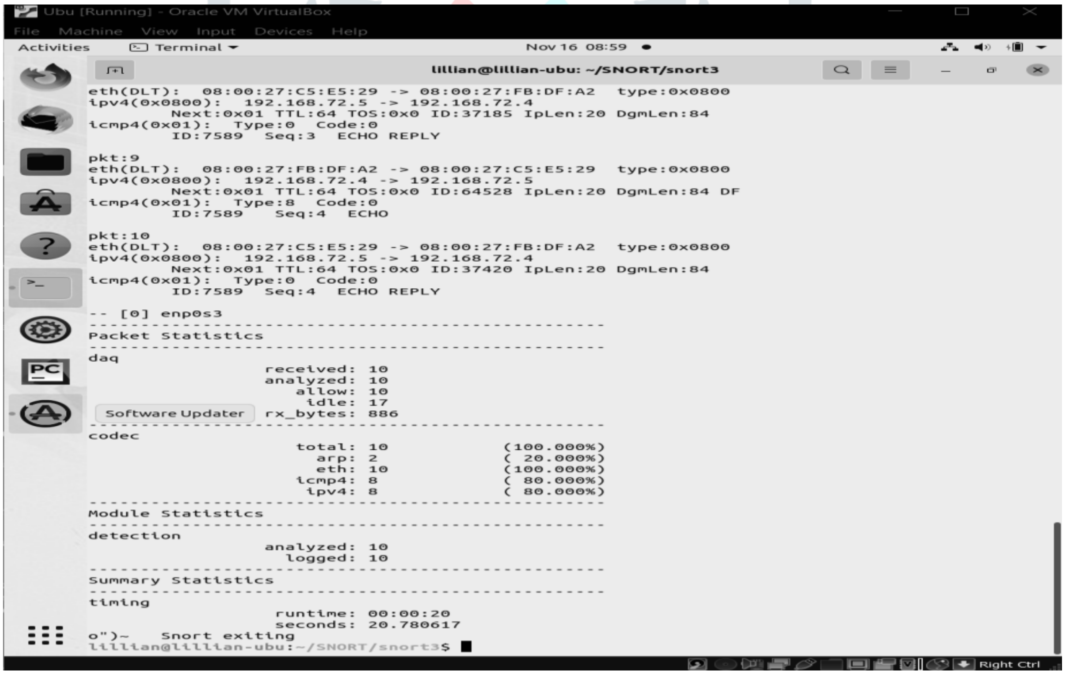
This allows us to do two things; identify the local network the attacking system is connected to and also know which network interface is connected to the network. This allows us to confirm the packets are from the attacking system when they arrive on the target system.


Ubu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Nov 16 08:59
lillian@lillian-ubu: ~/SNORT/snort3
Software Updater
Right Ctrl

```
o")~    Snort exiting
lillian@lillian-ubu:~/SNORT/snort3$ sudo snort -i enp0s3 -L dump -n 10
[sudo] password for lillian:
--------------------------------------------------
o")~    Snort++ 3.1.15.0
--------------------------------------------------
--------------------------------------------------
pcap DAQ configured to passive.
Commencing packet processing
++ [0] enp0s3
pkt:1
eth(DLT):   08:00:27:FB:DF:A2 -> FF:FF:FF:FF:FF:FF   type:0x0806
arp(0x0806):

pkt:2
eth(DLT):   08:00:27:C5:E5:29 -> 08:00:27:FB:DF:A2   type:0x0806
arp(0x0806):

pkt:3
eth(DLT):   08:00:27:FB:DF:A2 -> 08:00:27:C5:E5:29   type:0x0800
ipv4(0x0800):   192.168.72.4 -> 192.168.72.5
        Next:0x01 TTL:64 TOS:0x0 ID:64121 IpLen:20 DgmLen:84 DF
icmp4(0x01):   Type:8   Code:0
        ID:7589    Seq:1   ECHO

pkt:4
eth(DLT):   08:00:27:C5:E5:29 -> 08:00:27:FB:DF:A2   type:0x0800
ipv4(0x0800):   192.168.72.5 -> 192.168.72.4
        Next:0x01 TTL:64 TOS:0x0 ID:36950 IpLen:20 DgmLen:84
i                  pe:0   Code:0
        ID:7589   Seq:1   ECHO REPLY

pkt:5
eth(DLT):   08:00:27:FB:DF:A2 -> 08:00:27:C5:E5:29   type:0x0800
ipv4(0x0800):   192.168.72.4 -> 192.168.72.5
        Next:0x01 TTL:64 TOS:0x0 ID:64328 IpLen:20 DgmLen:84 DF
icmp4(0x01):   Type:8   Code:0
        ID:7589    Seq:2   ECHO

pkt:6
eth(DLT):   08:00:27:C5:E5:29 -> 08:00:27:FB:DF:A2   type:0x0800
ipv4(0x0800):   192.168.72.5 -> 192.168.72.4
        Next:0x01 TTL:64 TOS:0x0 ID:37075 IpLen:20 DgmLen:84
icmp4(0x01):   Type:0   Code:0
        ID:7589   Seq:2   ECHO REPLY

pkt:7
eth(DLT):   08:00:27:FB:DF:A2 -> 08:00:27:C5:E5:29   type:0x0800
ipv4(0x0800):   192.168.72.4 -> 192.168.72.5
        Next:0x01 TTL:64 TOS:0x0 ID:64473 IpLen:20 DgmLen:84 DF
icmp4(0x01):   Type:8   Code:0
```

Fig 4: Snort capturing ICMP Ping packets.

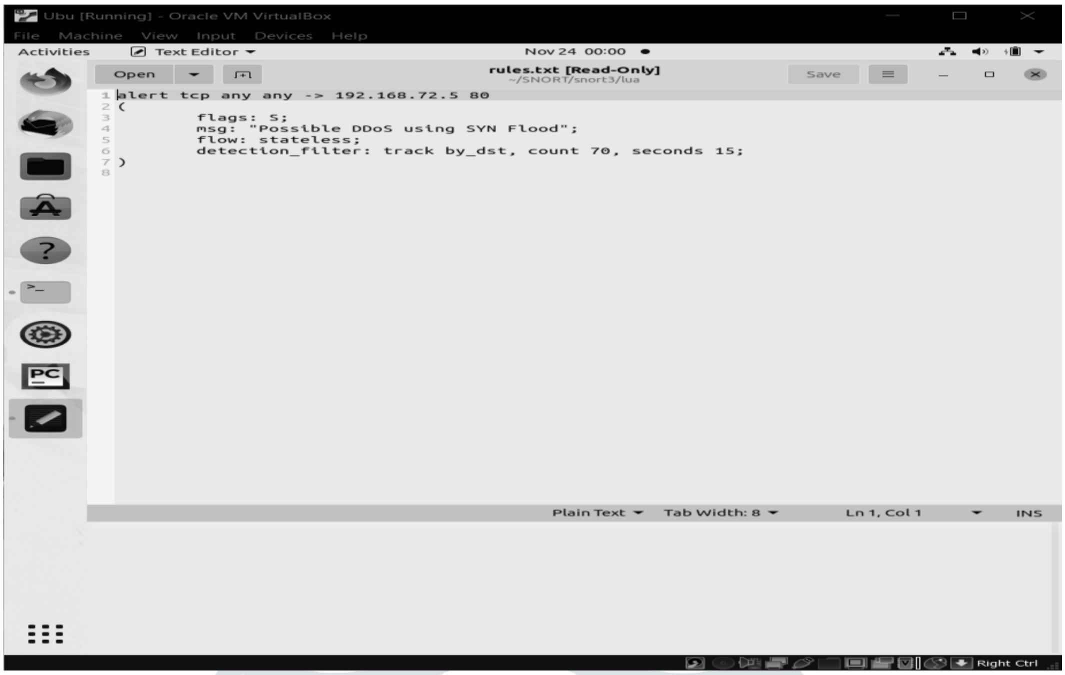
To test out the implementation the Snort implementation was run in capture mode. As the ICMP packets were sent over the network, they were captured by the Snort and the details of the packets were dumped on the terminal.


Ubu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Nov 16 08:59
lillian@lillian-ubu: ~/SNORT/snort3
Software Updater
Right Ctrl

```
eth(DLT):   08:00:27:C5:E5:29 -> 08:00:27:FB:DF:A2   type:0x0800
ipv4(0x0800):   192.168.72.5 -> 192.168.72.4
        Next:0x01 TTL:64 TOS:0x0 ID:37185 IpLen:20 DgmLen:84
icmp4(0x01):   Type:0   Code:0
        ID:7589   Seq:3   ECHO REPLY

pkt:9
eth(DLT):   08:00:27:FB:DF:A2 -> 08:00:27:C5:E5:29   type:0x0800
ipv4(0x0800):   192.168.72.4 -> 192.168.72.5
        Next:0x01 TTL:64 TOS:0x0 ID:64528 IpLen:20 DgmLen:84 DF
icmp4(0x01):   Type:8   Code:0
        ID:7589    Seq:4   ECHO

pkt:10
eth(DLT):   08:00:27:C5:E5:29 -> 08:00:27:FB:DF:A2   type:0x0800
ipv4(0x0800):   192.168.72.5 -> 192.168.72.4
        Next:0x01 TTL:64 TOS:0x0 ID:37420 IpLen:20 DgmLen:84
icmp4(0x01):   Type:0   Code:0
        ID:7589   Seq:4   ECHO REPLY

-- [0] enp0s3
--------------------------------------------------
Packet Statistics
--------------------------------------------------
daq
                 received: 10
                 analyzed: 10
                    allow: 10
                     idle: 17
                 rx_bytes: 886
--------------------------------------------------
codec
                    total: 10           (100.000%)
                      arp: 2            ( 20.000%)
                      eth: 10           (100.000%)
                    icmp4: 8            ( 80.000%)
                     ipv4: 8            ( 80.000%)
--------------------------------------------------
Module Statistics
--------------------------------------------------
detection
                 analyzed: 10
                   logged: 10
--------------------------------------------------
Summary Statistics
--------------------------------------------------
timing
                  runtime: 00:00:20
                  seconds: 20.780617
o")~    Snort exiting
lillian@lillian-ubu:~/SNORT/snort3$ █
```

Fig 5: Packet (quantitative) analysis of the captured network traffic.

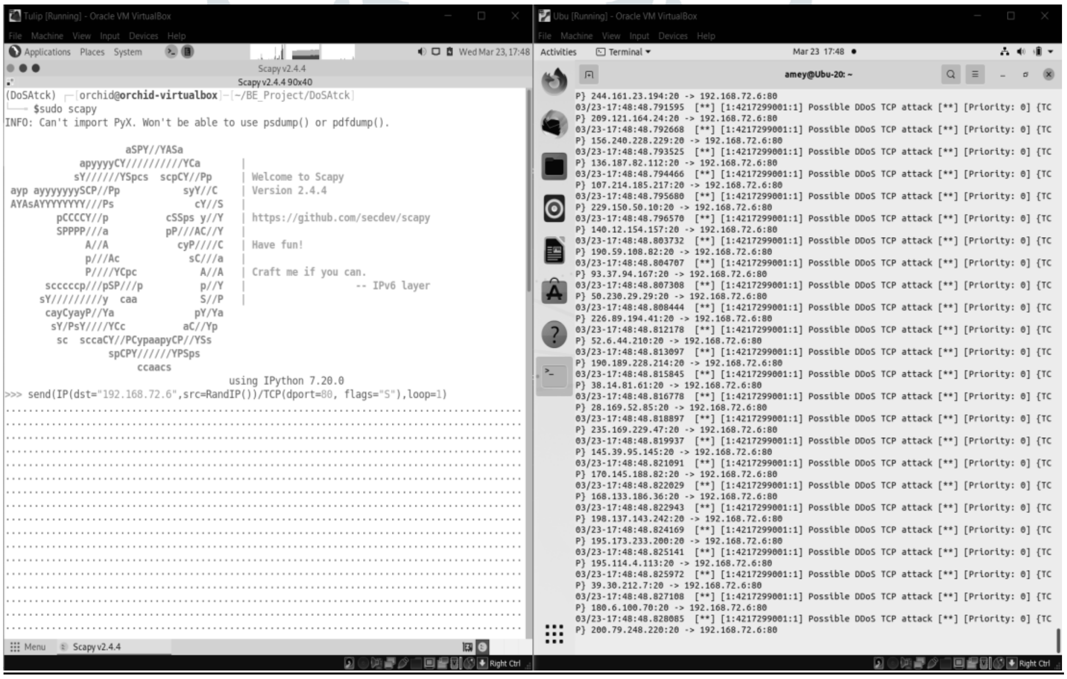
Snort also provides us with basic statistics about the packets received or captured before the service is terminated. This data is useful for knowing the efficiency and performance of the implementation.



```
alert tcp any any -> 192.168.72.5 80
(
        flags: S;
        msg: "Possible DDoS using SYN Flood";
        flow: stateless;
        detection_filter: track by_dst, count 70, seconds 15;
)
```

Fig 6: Custom Snort rule to detect and alert on SYN Flood attack.

A custom rule for snort3 was written, using a manually made txt file and including it in the default configuration file. This rule was meant to alert put up console alerts in case of a SYN Flood attack. What this rule does is flag all the SYN packets and track the number of SYN packets being received by the destination over a span of time.


Tulip [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Wed Mar 23, 17:48
Scapy v2.4.4
Scapy v2.4.4 90x40
Menu
Scapy v2.4.4
Right Ctrl
Ubu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Mar 23 17:48
amey@Ubu-20: ~
Right Ctrl

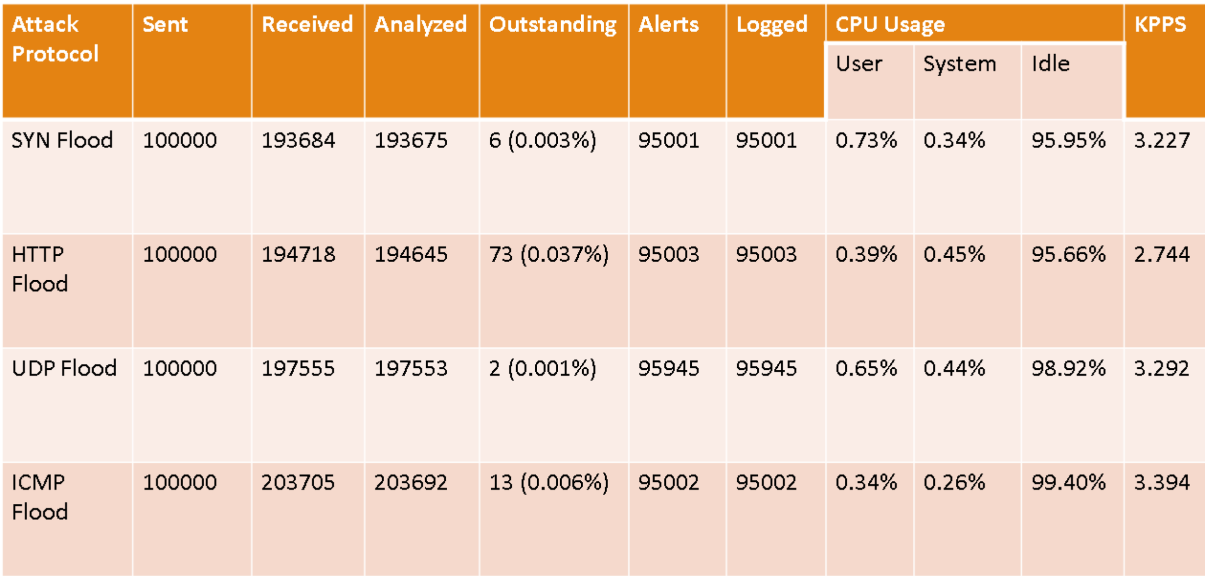
```
(DoSAtck) ┌─[orchid@orchid-virtualbox]─[~/BE_Project/DoSAtck]
└──╼ $sudo scapy
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().

                     aSPY//YASa
             apyyyyCY//////////YCa       |
            sY//////YSpcs  scpCY//Pp     | Welcome to Scapy
 ayp ayyyyyyySCP//Pp           syY//C    | Version 2.4.4
 AYAsAYYYYYYYY///Ps              cY//S   |
         pCCCCY//p          cSSps y//Y   | https://github.com/secdev/scapy
         SPPPP///a          pP///AC//Y   |
              A//A            cyP////C   | Have fun!
              p///Ac            sC///a   |
              P////YCpc           A//A   | Craft me if you can.
       scccccp///pSP///p          p//Y   |                -- IPv6 layer
      sY/////////y  caa           S//P   |
       cayCyayP//Ya              pY/Ya
        sY/PsY////YCc          aC//Yp
         sc  sccaCY//PCypaapyCP//YSs
                  spCPY//////YPSps
                       ccaacs
                                       using IPython 7.20.0
>>> send(IP(dst="192.168.72.6",src=RandIP())/TCP(dport=80, flags="S"),loop=1)
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
```

```
P} 244.161.23.194:20 -> 192.168.72.6:80
03/23-17:48:48.791595  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 209.121.164.24:20 -> 192.168.72.6:80
03/23-17:48:48.792668  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 156.240.228.229:20 -> 192.168.72.6:80
03/23-17:48:48.793525  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 136.187.82.112:20 -> 192.168.72.6:80
03/23-17:48:48.794466  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 107.214.185.217:20 -> 192.168.72.6:80
03/23-17:48:48.795680  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 229.150.50.10:20 -> 192.168.72.6:80
03/23-17:48:48.796570  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 140.12.154.157:20 -> 192.168.72.6:80
03/23-17:48:48.803732  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 190.59.108.82:20 -> 192.168.72.6:80
03/23-17:48:48.804707  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 93.37.94.167:20 -> 192.168.72.6:80
03/23-17:48:48.807308  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 50.230.29.29:20 -> 192.168.72.6:80
03/23-17:48:48.808444  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 226.89.194.41:20 -> 192.168.72.6:80
03/23-17:48:48.812178  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 52.6.44.210:20 -> 192.168.72.6:80
03/23-17:48:48.813097  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 190.189.228.214:20 -> 192.168.72.6:80
03/23-17:48:48.815845  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 38.14.81.61:20 -> 192.168.72.6:80
03/23-17:48:48.816778  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 28.169.52.85:20 -> 192.168.72.6:80
03/23-17:48:48.818897  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 235.169.229.47:20 -> 192.168.72.6:80
03/23-17:48:48.819937  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 145.39.95.145:20 -> 192.168.72.6:80
03/23-17:48:48.821091  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 170.145.188.82:20 -> 192.168.72.6:80
03/23-17:48:48.822029  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 168.133.186.36:20 -> 192.168.72.6:80
03/23-17:48:48.822943  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 198.137.143.242:20 -> 192.168.72.6:80
03/23-17:48:48.824169  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 195.173.233.200:20 -> 192.168.72.6:80
03/23-17:48:48.825141  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 195.114.4.113:20 -> 192.168.72.6:80
03/23-17:48:48.825972  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 39.30.212.7:20 -> 192.168.72.6:80
03/23-17:48:48.827108  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 180.6.100.70:20 -> 192.168.72.6:80
03/23-17:48:48.828085  [**] [1:4217299001:1] Possible DDoS TCP attack [**] [Priority: 0] {TC
P} 200.79.248.220:20 -> 192.168.72.6:80
```

Fig 7: Scapy being used to perform SYN flood attack on target system and the snort implementation putting forth alerts for the same.

Then the SYN packets were forged using the packet forging tool Scapy. This simulated the SYN Flood attack on the target system. The target system then clearly, detected and alerted us about the SYN Flood attack.

The following are the statistics obtained from the implementation performed:



Attack Protocol	Sent	Received	Analyzed	Outstanding	Alerts	Logged	CPU Usage			KPPS
							User	System	Idle	
SYN Flood	100000	193684	193675	6 (0.003%)	95001	95001	0.73%	0.34%	95.95%	3.227
HTTP Flood	100000	194718	194645	73 (0.037%)	95003	95003	0.39%	0.45%	95.66%	2.744
UDP Flood	100000	197555	197553	2 (0.001%)	95945	95945	0.65%	0.44%	98.92%	3.292
ICMP Flood	100000	203705	203692	13 (0.006%)	95002	95002	0.34%	0.26%	99.40%	3.394

Table: Compiled data of packet detection and resource usage

4 CONCLUSIONS

DoS attacks can be of various types, using all kinds of protocols and attacking different aspects of a system. The IPS/IDS options available, although very reliable lack the customization that an open-source custom-built IDS would. The implementation discussed above provides exactly that.

The Snort IPS allow for easy customization as the snort rule language is not that hard to learn plus the number of plugins available for the software makes it a very good choice compared to other open-source option. The community support for the software is also very good and up to date. This allows for a robust yet reliable implementation that could be configured further to do anything from logging and alerting to mitigation of attacks and everything in between.

For this phase of the implementation, two VMs were created, connected to a LAN network using the NAT network option available in VirtualBox. The NICs are configured in promiscuous mode. Then the network configuration of attacking system was done, which sent ping packets to the target system which was able to capture these packets. Following this, a custom rule for snort3 was written, which can detect and alert on SYN flood attacks. Scapy was used, which is a packet forging tool, to perform this SYN flood attack on the target system. The target system was able to detect and alert us about the SYN flood attack.

After implementing the proposed system, statistical data has been gathered to judge the efficiency and performance of the implementation. Four basic attacks have been simulated on the snort implementation to test the selected evaluation parameters. Packet statistics, alerts generated and logged, CPU usage and Network usage have been selected as the evaluation parameters for this research analysis.

References:

[1] Suresh M., Anitha R. (2011) Evaluating Machine Learning Algorithms for Detecting DDoS Attacks. In: Wyld D.C., Wozniak M., Chaki N., Meghanathan N., Nagamalai D. (eds) Advances in Network Security and Applications. CNSA 2011. Communications in Computer and Information Science, vol 196. Springer, Berlin, Heidelberg.

[2] Gupta, Alka, and Lalitsen Sharma. "Mitigation of DoS and Port Scan Attacks Using Snort." International Journal of Computer Sciences and Engineering 7 (2019): 248-258.

[3] Zubaydi, Haider Dhia, Mohammed Anbar, and Chong Yung Wey. "Review on detection techniques against DDoS attacks on a software-defined networking controller." 2017 Palestinian International Conference on Information and Communication Technology (PICICT). IEEE, 2017.

[4] Alenezi, Mohammed, and Martin J. Reed. "Methodologies for detecting DoS/DDoS attacks against network servers." The Seventh International Conference on Systems and Networks Communications ICSNC. 2012.

[5] Masdari M, Jalali M. A survey and taxonomy of DoS attacks in cloud computing. Security and Communication Networks. 2016 Nov 10;9(16):3724-51.

[6] Jhaveri RH, Patel SJ, Jinwala DC. DoS attacks in mobile ad hoc networks: A survey. In2012 second international conference on advanced computing & communication technologies 2012 Jan 7 (pp. 535-541). IEEE.

[7] Martin Roech "Writing Snort Rules: How to write snort rules and keep your sanity" Internet: https://paginas.fe.up.pt/~mgi98020/pgr/writing_snort_rules.htm

[8] Mike Smith "A Design for building an IPS using open-source products" Internet: https://www.sans.org/white-papers/1662/

[9] Kent Gruber "Build Your own Intrusion Detection System" Internet: https://medium.com/@KentGruber/build-your-own-intrusion-detection-system-e652f574037d

[10] OccupyTheWeb, Linux basics for hackers: Getting started with networking, scripting, and security in Kali, 1st Edition; Nos Starch Press, Inc.

[11] Manso, Pedro, José Moura, and Carlos Serrão. "SDN-based intrusion detection system for early detection and mitigation of DDoS attacks." Information 10.3 (2019): 106.

[12] Barki, Lohit, et al. "Detection of distributed denial of service attacks in software defined networks." 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2016.

[13] Zhijun, Wu, et al. "Low-rate DoS attacks, detection, defense, and challenges: a survey." IEEE Access 8 (2020): 43920-43943.

[14] Kamboj, P., Trivedi, M. C., Yadav, V. K., & Singh, V. K. (2017, October). Detection techniques of DDoS attacks: A survey. In 2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON) (pp. 675-679). IEEE.

[15] Zubaydi, Haider Dhia, Mohammed Anbar, and Chong Yung Wey. "Review on detection techniques against DDoS attacks on a software-defined networking controller." 2017 Palestinian International Conference on Information and Communication Technology (PICICT). IEEE, 2017.

[16] Alenezi, Mohammed, and Martin J. Reed. "Methodologies for detecting DoS/DDoS attacks against network servers." The Seventh International Conference on Systems and Networks Communications ICSNC. 2012.

[17] Suresh, Manjula, and R. Anitha. "Evaluating machine learning algorithms for detecting DDoS attacks." International Conference on Network Security and Applications. Springer, Berlin, Heidelberg, 2011.

[18] Birkinshaw, Celyn, Elpida Rouka, and Vassilios G. Vassilakis. "Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks." Journal of Network and Computer Applications 136 (2019): 71-85.

[19] Ponomarev, Stanislav. Intrusion Detection System of industrial control networks using network telemetry. Louisiana Tech University, 2015.

[20] Xiao, Yuelei, and Xing Xiao. "An intrusion detection system based on a simplified residual network." Information 10.11 (2019): 356.

[21] Jaiswal R.C. and Lokhande S.D., Gulavani Aditya "Implementation and Analysis of DoS Attack Detection Algorithms", International Journal of Science and Research (IJSR), volume-4, Issue-5, May-2015, pp. 2085-2089.(ISSN (Online): 2319-7064, Index Copernicus Value (2013): 6.14 | Impact Factor (2013):4.438

[22] Jaiswal R.C. and Lokhande S.D., A. Ahmed, P. Mahajan, "Performance Evaluation of Clustering Algorithms for IP Traffic Recognition", International Journal of Science and Research (IJSR), volume-4, Issue-5, May-2015, pp. 2786-2792.(ISSN (Online): 2319-7064, Index Copernicus Value (2013): 6.14|Impact Factor (2013):4.438

[23] Jaiswal R.C. and S.D.Lokhande, "Systematic Performance Analysis of Bit-Torrent Traffic", Helix SCI INDEXED E-ISSN: 2319-5592; P-ISSNs: 2277-3495, Helix Vol. 9 (2): pp. 4858- 4863, DOI 10.29042/2019-4858-4863, April 2019.

[24] Jaiswal R.C. and Nikita Kakade, "Skin disease detection and classification using Image Processing Techniques", Journal of Emerging Technologies and Innovative Research (JETIR), ISSN-2349-5162; UGC approved Journal:5.87, Volume 4, Issue 12, December 2017.

[25] Jaiswal R.C. and Aishwarya Gaikwad, "Experimental Analysis of Bit torrent Traffic based on Heavy-Tailed Probability Distributions", International Journal of Computer Applications, ISSN No. (0975 – 8887),Impact Factor .3.1579(2016),Volume 155 – No 2, December 2016.

[26] Jaiswal R.C. and Lokhande S.D., "Evaluation of Effect of Seeds and downloaders on the Performance of Bit Torrent Network using Markov Chain Modelling", Journal of Communication Engineering & Systems, Volume 6, Issue 1. (ISSN: 2321-5151 (print version), ISSN: 2249-8613 (electronic version) IF (2016): 0.709).

[27] Jaiswal R.C. and Lokhande S.D., "Performance Analysis for IPv4 and IPv6 Internet Traffic", ICTACT Journal on Communication Technology, September 2015, volume: 06, issue: 04, pp. 1208-1217.(Print: ISSN: 0976-0091, Online ISSN:2229-6948 (Impact Factor: 0.789 in 2015).

[28] Jaiswal R.C. and Lokhande S.D, "Performance Evaluation of Wireless Networks", Coimbatore Institute of Information Technology International Journal, volume-7, Issue-8, July-2015, pp. 1237-1242. (Print: ISSN 0974 – 9616 |Impact Factor: 0.572)

[29] Jaiswal R.C. and Lokhande S.D, "A Novel Approach for Real Time Internet Traffic Classification", ICTACT Journal on Communication Technology, September 2015, volume: 06, issue: 03, pp. 1160-1166.(Print: ISSN: 0976-0091, Online ISSN:2229-6948 (Impact Factor: 0.789 in 2015).

[30] "Measurement, Modeling and Analysis of HTTP Web Traffic", IMCIET-International Multi Conference on Innovations in Engineering and Technology-ICCC-International Conference on Communication and Computing -2014, PP-242-258, ISBN:9789351072690, VVIT, Bangalore.

[31] "Comparative Analysis using Bagging, LogitBoost and Rotation Forest Machine Learning Algorithms for Real Time Internet Traffic Classification", IMCIP-International Multi Conference on Information Processing –ICDMW- International Conference on Data Mining and Warehousing-2014, PP113-124, ISBN: 9789351072539, University Visvesvaraya College of Engg. Department of Computer Science and Engineering Bangalore University, Bangalore.

[32] "Statistical Features Processing Based Real Time Internet Traffic Recognition and Comparative Study of Six Machine Learning Techniques", IMCIP- International Multi Conference on Information Processing-(ICCN- International Conference on Communication Networks-2014, PP-120-129, ISBN: 9789351072515, University Visvesvaraya College of Engg. Department of Computer Science and Engineering Bangalore University, Bangalore.

[33] "Analysis of Early Traffic Processing and Comparison of Machine Learning Algorithms for Real Time Internet Traffic Identification Using Statistical Approach ", ICACNI-2014-International Conference on Advanced Computing, Networking, and Informatics), Kolkata, India,DOI: 10.1007/978-3-319-07350-7_64, Volume 28 of the book series Smart Innovation, Systems and Technologies (SIST),Page:577-587.

[34] "*Machine* Learning Based Internet Traffic Recognition with Statistical Approach*", I*ndicon-2013-IIT Bombay IEEE conference. Inspec Accession Number: 14062512, DOI: 10.1109/INDCON.2013.6726074

[35] Jaiswal R. C. and Sahil Nahar, "Recognition and Selection of Learning Styles to Personalize Courses for Students", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, Indexed in Google Scholar, Microsoft Academic, CiteSeerX, Thomson Reuters, Mendeley : reference manager, ISSN-2349-5162, Impact Factor:7.95, Volume 9, Issue 2 pp. b235-b252, February 2022.

[36] Jaiswal R. C. and Rushikesh Karwankar, " Demand Forecasting for Inventory Optimization ", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, Indexed in Google Scholar, Microsoft Academic, CiteSeerX, Thomson Reuters, Mendeley : reference manager, ISSN-2349-5162, Impact Factor:7.95, Volume 8, Issue 12 pp. 121-131, January 2022.

[37] Jaiswal R. C. and Prajwal Pitlehra, "Credit Analysis Using K-Nearest Neighbours' Model", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, ISSN-2349-5162, Impact Factor:7.95, Volume 8, Issue 5, pp. 504-511, May 2021.

[38] Jaiswal R. C. and Akshat Kaushik, "Automated Attendance Monitoring system using discriminative Local Binary Histograms and PostgreSQL", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, ISSN-2349-5162, Impact Factor:5.87, Volume 7, Issue 11, pp. 80-86, November 2020.

[39] Jaiswal R. C. and Danish khan, "Arduino based Weather Monitoring and Forecasting System using SARIMA Time-Series Forecasting", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, ISSN-2349-5162, Impact Factor:5.87, Volume 7, Issue 11, pp. 1149-1154, November 2020.

[40] Jaiswal R.C. and Aashay Pawar, "Stock Market Study Using Supervised Machine Learning", International Journal of Innovative Science and Research Technology (IJISRT), Open Access, Peer Reviewed and refereed Journal , ISSN: 2456-2165; IC Value: 45.98; SJ Impact Factor:6.253, Volume 5 Issue I, pp. 190-193, Jan 2020.

[41] Jaiswal R.C. and Onkar Gagare, "Head Mounted Display", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Open Access, Peer Reviewed and refereed Journal, ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:7.177, Volume 7 Issue XI, pp. 535-541, Nov 2019.

[42] Jaiswal R.C. and Nehal Borole, "Autonomous Vehicle Prototype Development and Navigation using ROS", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Open Access, Peer Reviewed and refereed Journal, ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:7.177, Volume 7 Issue XI, pp. 510-514, Nov 2019.

[43] Jaiswal R.C. and Shreya Mondhe, "Stock Market Prediction Using Machine Learning & Robotic Process Automation", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, ISSN-2349-5162, Volume 6, Issue 6, pp. 926-929, February 2019.

[44] Jaiswal R. C. and Sahil Nahar, "Recognition and Selection of Learning Styles to Personalize Courses for Students", Journal of Emerging Technologies and Innovative Research (JETIR), Open Access, Peer Reviewed and refereed Journal, Indexed in Google Scholar, Microsoft Academic, CiteSeerX, Thomson Reuters, Mendeley : reference manager, ISSN-2349-5162, Impact Factor:7.95, Volume 9, Issue 2 pp. b235-b252, February 2022