DOSPREVENT: A PROACTIVE DENIAL OF SERVICE (DOS) ATTACK PREVENTION TOOL AGAINST CRITICAL INFORMATION INFRASTRUCTURE

[1]Adejimi, A. O., [1]Aborisade, D. O., [1]Alabi, O. A. and [2]Mahmood, Z. A.

1. Dept. of Computer Science, Federal University of Agriculture, Abeokuta, Ogun State.
2. Gateway ICT Polytechnic, Sapade, Ogun State.

Corresponding author: adejimiao@funaab.edu.ng

Abstract

Denial of service (DoS) attack is generally a malware attack to overwhelm a computer system, websites or network with unwarranted excessive traffic hence making it inaccessible to genuine users. Denial of service (DoS) attacks pose a significant threat to critical information infrastructure (CII) networks as they can disrupt essential services and potentially cause widespread damage to the infrastructure. This attack aimed at overcoming the availability of the information infrastructure's network with a huge number of traffic hence making it unavailable for business activities. This work proposes a preventive approach to tackle the issue of DoS attacks on critical information infrastructures using a packet filtering approach. The algorithm attempts to filter incoming packets and get their time-to-life value which was then used to determine the hop-count computation detecting DoS packets from legitimate packets. The hop-count gives accurate detection with 0.05% false positive with an accuracy of 97%. The system monitors the packets coming into the information infrastructure's network and proactively detect DoS attack before damaging the system. The proposed system is a preventive measure for CII against DoS attack.

Keywords: DoSPrevent, time to life, up-count, denial of service (DoS) attacks, packet.

1.0 Introduction

In this era of digital economy, a significant amount of information is stored online in various databases. From bank account details to ATM card password, even medical record history. Some of the systems that control essential services of the nation now deploy their services through computer networks to allow a wider coverage of their services. As a result of this, more than ever before, efforts are being made towards securing information infrastructure from theft, unauthorized access, information corruption and misdirection of information. A control system is considered to have been hacked when unauthorized access is obtained to one of its files. Cybersecurity of critical infrastructure is the safeguarding of data stored on these essential computer systems against illegal access, data corruption, and data misdirection (Jamali et al., 2023). Hackers can try to obtain data from a website in two different ways: distributed denial of service (DDOS) or denial of service (DOS).

DOS attacks are cyber-attacks carried out by making the website unavailable to its intended users. This attack is carried out by bombarding the website with too many requests at the same time such that the requests become too much for the system to handle. In a nutshell, the aim of a DOS attack is to slow down or crash the site. DOS attacks are carried out in two ways; through the application level or via a network bandwidth. DOS attack is done by sending too many requests at a time so that other clients are underserved or not served at all. It is done by dominating the network bandwidth, thereby causing heavy traffic on the site which eventually leads to a crash. DDOS attacks on the other hand is similar to DOS attack, but with a difference. Unlike DOS attacks that are carried out from a single computer, a distributed denial of service (DDOS) attack is carried out by distributing computers assembled to perform the hack. In other words, different computers are sending multiple and many requests at the same time.

Critical Information infrastructure refers to basic facilities, systems, and services in a country that if disrupted or destroyed would have devastating effects on the country’s economy. They are assets or systems that are essential for the maintenance of vital society function. CII could be in the form of classified data, critical databases, control systems (ICS/DCS, SCADA), or cyber-physical systems (Adejimi et al., 2023).

Denial of service (DoS) attack on Critical Information Infrastructure (CII) is a cyber-attack that is launched to cause traffic on a sector’s network by sending too many requests to their websites. The disruptions caused by a successful DoS attack will cause tremendous injury for a CII. Once systems go down, organizations might lose access to essential information and applications, they stand to lose revenue, miss out on opportunities and lose their integrity (Hurst et al., 2015). DoS attack on Information Infrastructure can cause loss of assets and delay in service which would cause catastrophic effects on a country’s economy and lives of people in the country. There is need to safeguard the information infrastructure against DoS attacks.

The rest of this paper is organized as follows; Section 2 gives the literature review. Section 3 describes the proposed technique with algorithms. Section 4 discussed the implementation of the proposed system; Section 5 gives the performance evaluation; Section 6 discussed the conclusion and the future work.

2.0 Literature Review

Denial of service (DoS) attacks have become a major threat to current critical information infrastructure computer networks. Early DoS attacks were technical games played among underground attackers. For instance, an attacker might want to get control of an Internet Relay Chat (IRC) channel via performing DoS attacks against the channel owner. By taking down popular sites an attack can gain recognition from other underground attackers. Easy-to-use DoS tools can be easily downloaded from the Internet, any computer user can then become DoS attacker as well. Companies might use DoS attacks to knock off their competitors within the market. Extortion via DoS attacks were on rise within the past years (Gu and Liu, 2007). Attackers threatened online businesses with DoS attacks and requested payments for protection. (Sivakalai and Jayapa, 2014).

Denial or degradation of service may result from malicious or benign actions. These actions may originate locally or remotely from the service, or user, experiencing denial or degradation of service. The communications bandwidth, memory buffers, computational resources, or the network protocol or application processing logic of the victim, or any systems on which the victim depends for delivering service (the domain name system or credit card payment service for example), maybe targeted. Network denial of service presents significant challenges to the continued use of the Internet for critical communications (Ortega-Fernandez & Liberati, 2023). DoS and DDoS attacks can be divided into three types, viz: Volume Based Attacks which includes User Datagram Protocol (UDP) floods, Internet Control Message Protocol (ICMP) floods, and other spoofed-packet floods. The attack's goal is to saturate the bandwidth of the attacked site, and magnitude is measured in bits per second (Bps). Protocol Attacks includes SYN (DoS attack) floods, fragmented packet attacks, Ping of Death, Smurf DDoS and more. This type of attack consumes actual server resources, or those of intermediate communication equipment, such as firewalls and load balancers, and is measured in packets per second (Pps). Application Layer Attacks comprised of seemingly legitimate and innocent requests. It **i**ncludes low-and-slow attacks, GET/POST floods. The goal of these attacks is to crash the web server, and the magnitude is measured in Requests per second (Rps).

Francisco et al. (2019) proposed an approach called Smart Detection. The approach was designed to combat DDoS attacks on the Internet in a modern collaborative way. In this approach, the system collected network traffic samples and classifies them. Attack notification messages were shared using a cloud platform for convenient use by traffic control protection systems. Core of the detection consists of a signature dataset system (SDS) and a machine learning algorithm (MLA). The experiments were performed using four modern benchmark datasets. The results showed an online detection rate (DR) of attacks above 96%, with high precision (PREC) and low "false alarm rate (FAR) using a sampling rate (SR) of 20% of network traffic.

Booth and Anderson (2020) proposed Critical Infrastructure Network DDoS Defense, via Cognitive Learning. The work presented how cognitive learning can be used to significantly mitigate any effects of DDoS network attacks on critical infrastructure. They followed the design science research (DSR) methodology. They used their high-level IT artifact as their proposed cognitive based design guidelines and algorithms, which greatly mitigate all OSI layer 3 (network), 4 (transport), and 7 (application) network-based DDoS attacks (L347). Through DSR, an IT artifact was created, then evaluated, and then redesigned with improvements based on the feedback from the evaluation. This cycle was then repeated several times. These cycles then continued, until an adequate level of new knowledge is acquired.

A hybrid detection system for DoS attacks was proposed by Cepheli et al. (2016). The proposed detection system used both anomaly-based and signature-based detection methods separately but in an integrated fashion and combines the outcomes of both detectors to enhance the overall detection accuracy. They applied two distinct datasets to their proposed system in order to test the detection performance of H-IDS and concluded that the proposed hybrid system gives better results than the systems based on non-hybrid detection.

3.0 The Proposed Technique

The proposed system is named 'DoSPrevent'. It is a preventive approach to tackle the issue of DoS attacks on critical information infrastructures using a packet filtering approach. The algorithm attempts to filter incoming packets and get their time-to-life value which was then used to determine the hop-count computation detecting DoS packets from legitimate packets.

3.1 Hop-count Computation

Hop count is a measurement that denotes the quantity of intermediate devices, such as routers, through which a data packet needs to pass in order to arrive at its intended destination within a network. It is the device through which a piece of data pass through. When a DOS attack is launched it comes with IP spoofing. IP spoofing is the creation of Internet Protocol (IP) packets which have a modified source address in order to either hide the identity of the sender, to impersonate another computer system, or both. Sending and receiving IP packets is a primary way in which networked computers and other devices communicate, and constitutes the basis of the modern internet. All IP packets contain a header which precedes the body of the packet and contains important routing information, including the source address. In a normal packet, the source IP address is the address of the sender of the packet. If the packet has been spoofed, the source address will be forged.

Each packet that enters the website will be filtered through the hop-count computation techniques. This is used to distinguish between a legitimate and an attacker packet. Each time that a packet of data moves from one router to another, let's say from the router of your home network to the one just outside your county line that is considered as one hop. The hop-count is the total number of hops that a packet of data travels. Hop-count information is not directly stored in the IP address, so it has to be computed using Time to Live (TTL) field. TTL is an 8-bit field in the IP header, it specifies the amount of time a packet is set to exist in the internet. When a packet of information is sent to the internet, for every router the packet go through, the TTL count is decreased by one before it is moved to the next hop. When a packet has reached its destination the TTL value extracted from it is known as the Final TTL. The problem in hop-count computation is that there is no consensus on initial TTL value, the destination only sees the final TTL value. Most modern Operating Systems use only a few selected initial TTL values, 30, 32, 60, 64, 128, and 255. Only a few Internet hosts are apart by more than 30 Hops, so one can get the initial TTL by selecting the smallest value in the initial value set that is larger than the final TTL value. The algorithm written in Python programming language is as follows:

Algorithm 1: Hop-count Computation Technique

```
Begin
 # This Program is called DoSPrevent
 Initial TTL=32 if final TTL <=32
 Initial TTL =64 if 32< final TTL<=64
 Initial TTL =128 if 64<final TTL<=128
 Initial TTL =255 if 128<final TTL <=255
 Set TTL= Initial TTL
 If TTL > 128
    Set Hop = 255 – TTL
 else if TTL >64
    hop = 128 -TTL
 else if TTL >32
    hop = 64 – TTL
 else hop = 32 – TTL
 Set Hop count = hop
 If Hop count >= 15
  Send email to server "DOS attack detected"
End.
```

The flowchart for the DoS Detection process is shown in figure 1.

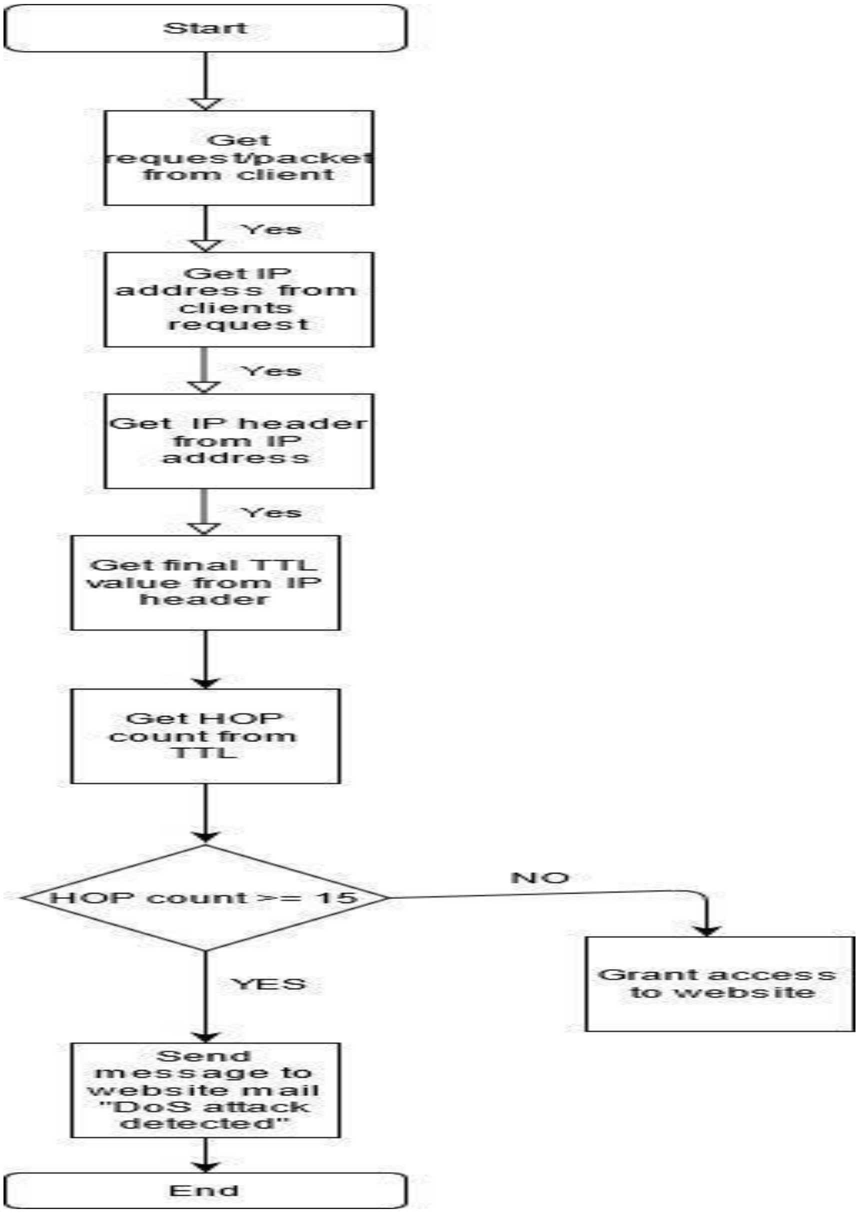

Start
Get request/packet from client
Yes
Get IP address from clients request
Yes
Get IP header from IP address
Yes
Get final TTL value from IP header
Get HOP count from TTL
HOP count >= 15
NO
Grant access to website
YES
Send message to website mail "DoS attack detected"
End

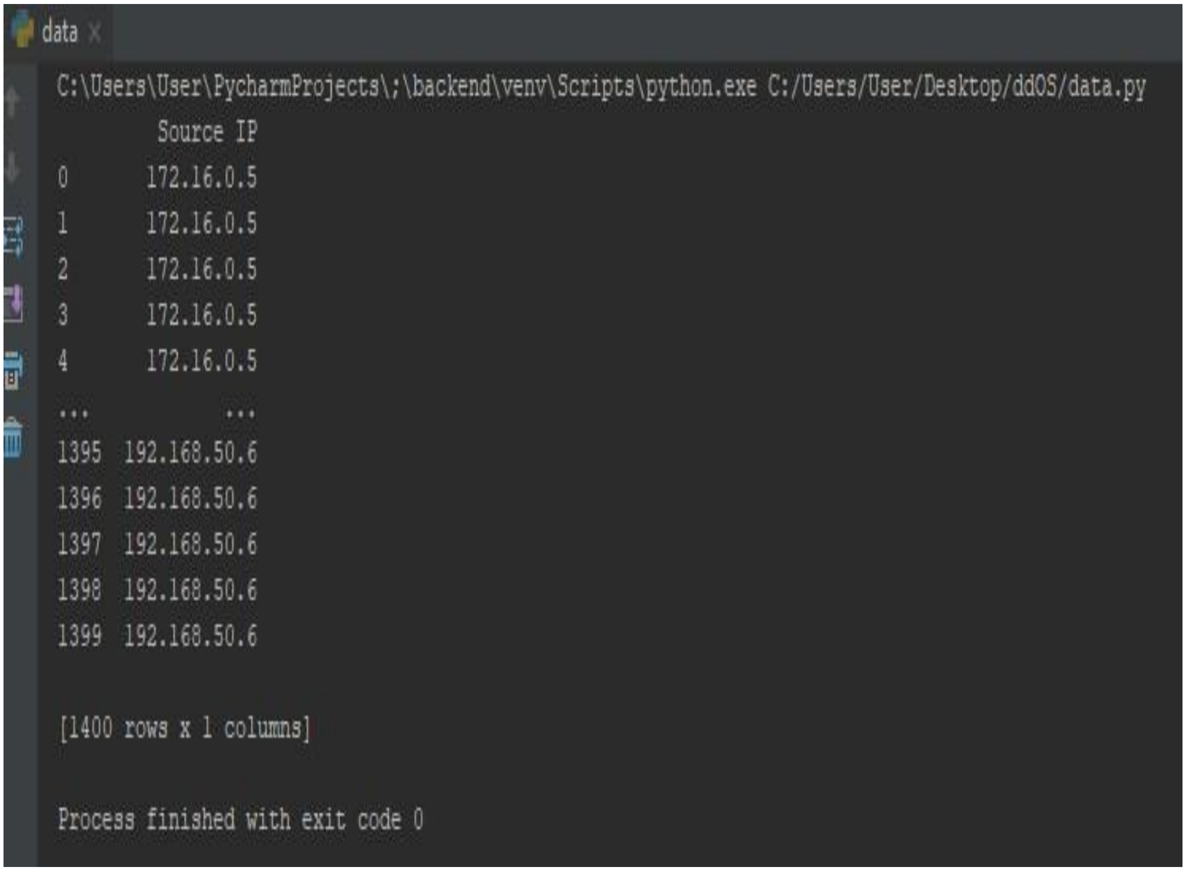
Figure 1: Flowchart for the DoS Detection

4.0 Implementation

The implementation of the proposed Packet Filtering DoS detection system was demonstrated using Python programming language due to its flexibility.

(a) Fetching and filtering the Packet IP addresses.

The packet sniffer was created with socket module in python. The raw socket type was used to get the packets. A raw socket provides access to the underlying protocols, which support socket abstractions since raw sockets are part of the internet socket Application Programming Interface (API), they can only be used to generate and receive IP packets. Both reading and writing a raw socket require creating a raw socket first. Here the INET family raw socket was used. The socket uses three parameters, the family parameter that describes the address family of the socket, the type of socket and the protocol of the packet. The result is shown in figure 2 and 3.

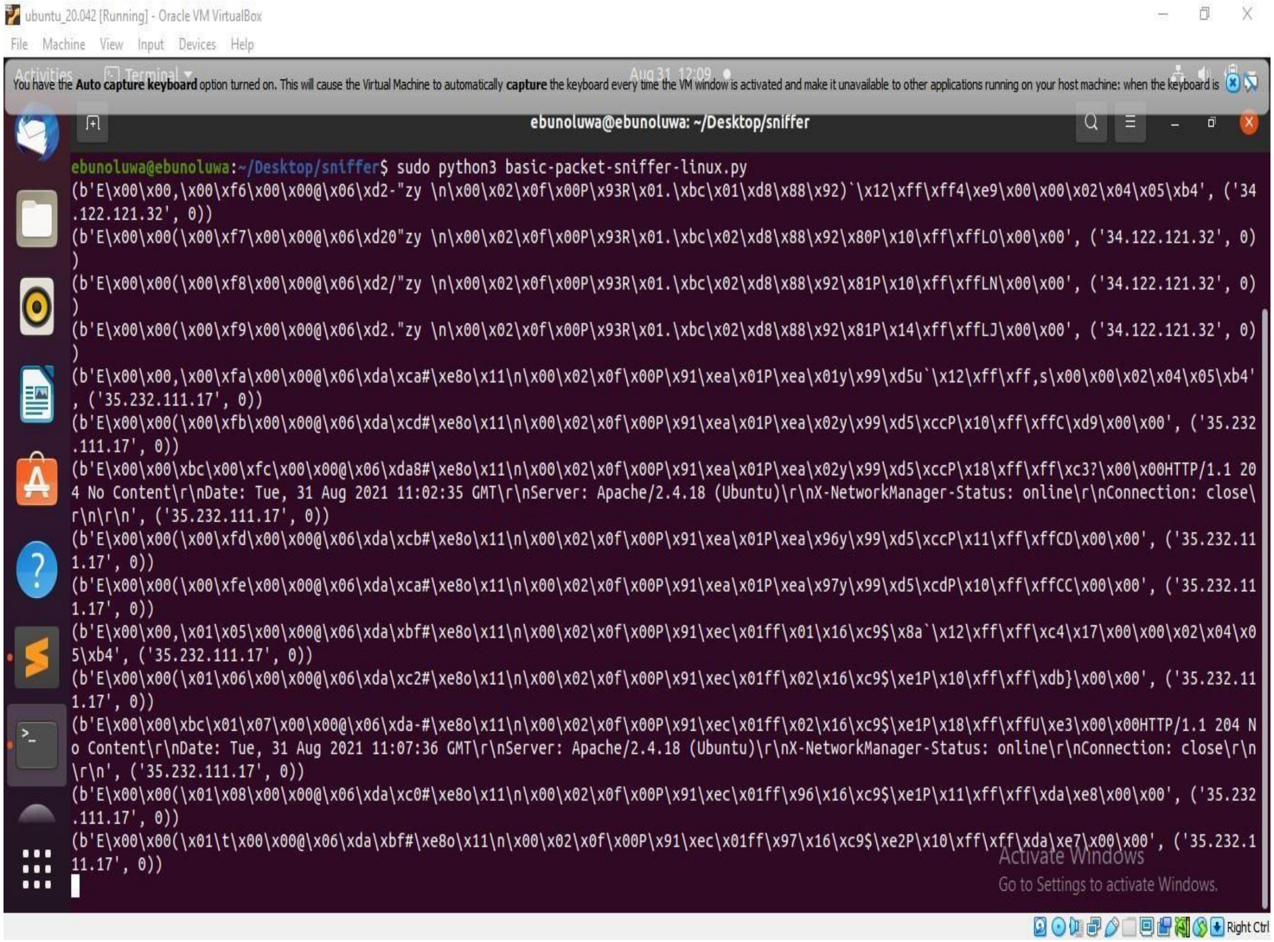


```
C:\Users\User\PycharmProjects\;\backend\venv\Scripts\python.exe C:/Users/User/Desktop/ddOS/data.py
        Source IP
0      172.16.0.5
1      172.16.0.5
2      172.16.0.5
3      172.16.0.5
4      172.16.0.5
...           ...
1395  192.168.50.6
1396  192.168.50.6
1397  192.168.50.6
1398  192.168.50.6
1399  192.168.50.6

[1400 rows x 1 columns]

Process finished with exit code 0
```

Figure 2: IP fetching


ubuntu_20.042 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Aug 31 12:09
ebunoluwa@ebunoluwa: ~/Desktop/sniffer
Activate Windows
Go to Settings to activate Windows.
Right Ctrl

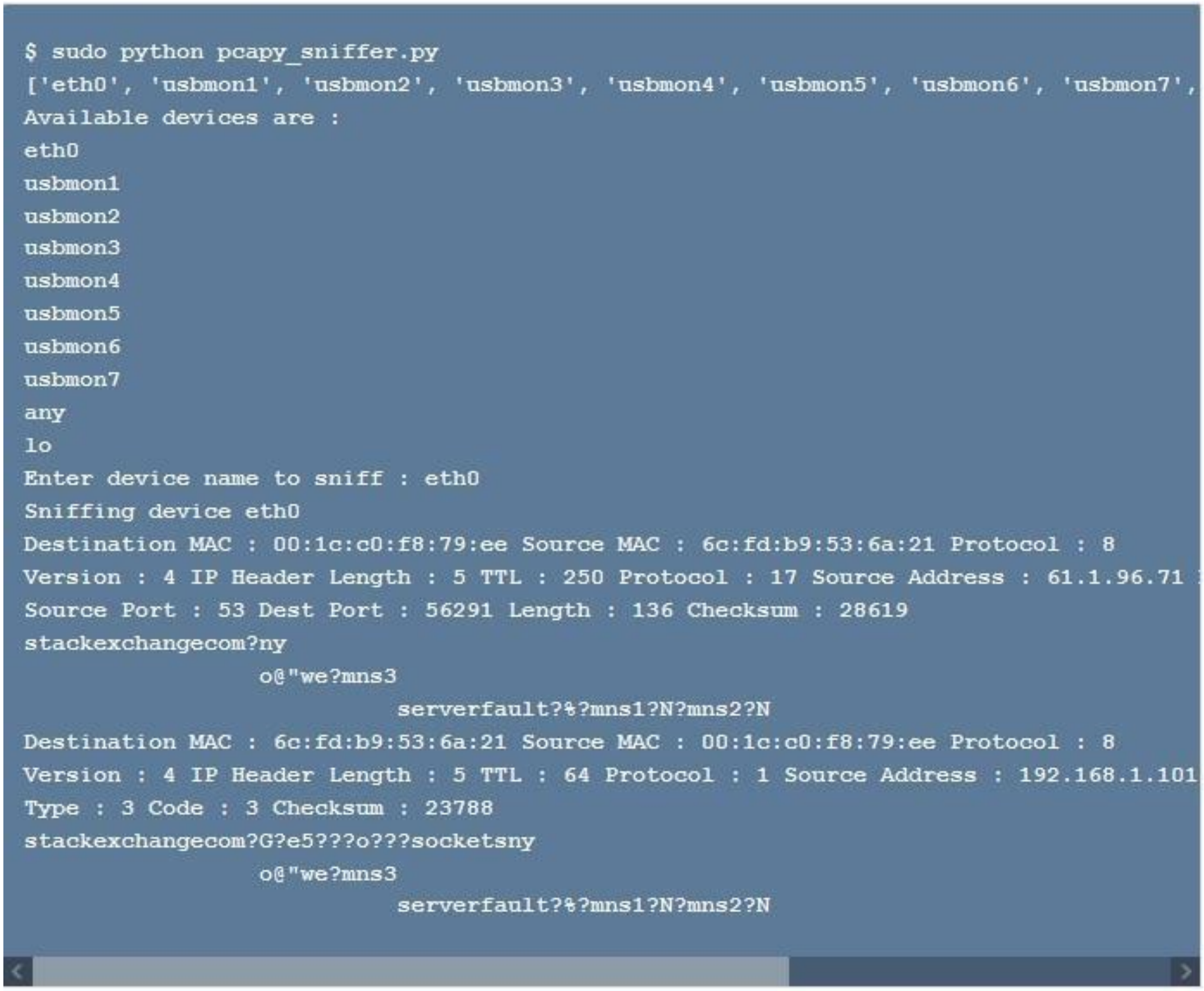
You have the **Auto capture keyboard** option turned on. This will cause the Virtual Machine to automatically **capture** the keyboard every time the VM window is activated and make it unavailable to other applications running on your host machine: when the keyboard is

```
ebunoluwa@ebunoluwa:~/Desktop/sniffer$ sudo python3 basic-packet-sniffer-linux.py
(b'E\x00\x00,\x00\xf6\x00\x00@\x06\xd2-"zy \n\x00\x02\x0f\x00P\x93R\x01.\xbc\x01\xd8\x88\x92)`\x12\xff\xff4\xe9\x00\x00\x02\x04\x05\xb4', ('34
.122.121.32', 0))
(b'E\x00\x00(\x00\xf7\x00\x00@\x06\xd20"zy \n\x00\x02\x0f\x00P\x93R\x01.\xbc\x02\xd8\x88\x92\x80P\x10\xff\xffLO\x00\x00', ('34.122.121.32', 0)
)
(b'E\x00\x00(\x00\xf8\x00\x00@\x06\xd2/"zy \n\x00\x02\x0f\x00P\x93R\x01.\xbc\x02\xd8\x88\x92\x81P\x10\xff\xffLN\x00\x00', ('34.122.121.32', 0)
)
(b'E\x00\x00(\x00\xf9\x00\x00@\x06\xd2."zy \n\x00\x02\x0f\x00P\x93R\x01.\xbc\x02\xd8\x88\x92\x81P\x14\xff\xffLJ\x00\x00', ('34.122.121.32', 0)
)
(b'E\x00\x00,\x00\xfa\x00\x00@\x06\xda\xca#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xea\x01P\xea\x01y\x99\xd5u`\x12\xff\xff,s\x00\x00\x02\x04\x05\xb4'
, ('35.232.111.17', 0))
(b'E\x00\x00(\x00\xfb\x00\x00@\x06\xda\xcd#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xea\x01P\xea\x02y\x99\xd5\xccP\x10\xff\xffC\xd9\x00\x00', ('35.232
.111.17', 0))
(b'E\x00\x00\xbc\x00\xfc\x00\x00@\x06\xda8#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xea\x01P\xea\x02y\x99\xd5\xccP\x18\xff\xff\xc3?\x00\x00HTTP/1.1 20
4 No Content\r\nDate: Tue, 31 Aug 2021 11:02:35 GMT\r\nServer: Apache/2.4.18 (Ubuntu)\r\nX-NetworkManager-Status: online\r\nConnection: close\
r\n\r\n', ('35.232.111.17', 0))
(b'E\x00\x00(\x00\xfd\x00\x00@\x06\xda\xcb#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xea\x01P\xea\x96y\x99\xd5\xccP\x11\xff\xffCD\x00\x00', ('35.232.11
1.17', 0))
(b'E\x00\x00(\x00\xfe\x00\x00@\x06\xda\xca#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xea\x01P\xea\x97y\x99\xd5\xcdP\x10\xff\xffCC\x00\x00', ('35.232.11
1.17', 0))
(b'E\x00\x00,\x01\x05\x00\x00@\x06\xda\xbf#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xec\x01ff\x01\x16\xc9$\x8a`\x12\xff\xff\xc4\x17\x00\x00\x02\x04\x0
5\xb4', ('35.232.111.17', 0))
(b'E\x00\x00(\x01\x06\x00\x00@\x06\xda\xc2#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xec\x01ff\x02\x16\xc9$\xe1P\x10\xff\xff\xdb}\x00\x00', ('35.232.11
1.17', 0))
(b'E\x00\x00\xbc\x01\x07\x00\x00@\x06\xda-#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xec\x01ff\x02\x16\xc9$\xe1P\x18\xff\xffU\xe3\x00\x00HTTP/1.1 204 N
o Content\r\nDate: Tue, 31 Aug 2021 11:07:36 GMT\r\nServer: Apache/2.4.18 (Ubuntu)\r\nX-NetworkManager-Status: online\r\nConnection: close\r\n
\r\n', ('35.232.111.17', 0))
(b'E\x00\x00(\x01\x08\x00\x00@\x06\xda\xc0#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xec\x01ff\x96\x16\xc9$\xe1P\x11\xff\xff\xda\xe8\x00\x00', ('35.232
.111.17', 0))
(b'E\x00\x00(\x01\t\x00\x00@\x06\xda\xbf#\xe8o\x11\n\x00\x02\x0f\x00P\x91\xec\x01ff\x97\x16\xc9$\xe2P\x10\xff\xff\xda\xe7\x00\x00', ('35.232.1
11.17', 0))
```

Figure 3: IP Filtering

(b) Parsing the Packet

Now the data will be sniffed and the headers will be unpacked. The unpack method was used to unpack the IP Header gotten. The Time to Live (TTL) value which is 8-bit field in the IP Header was derived. The TTL value is used to get Hop-count computation value. Figure 4 shows the output.

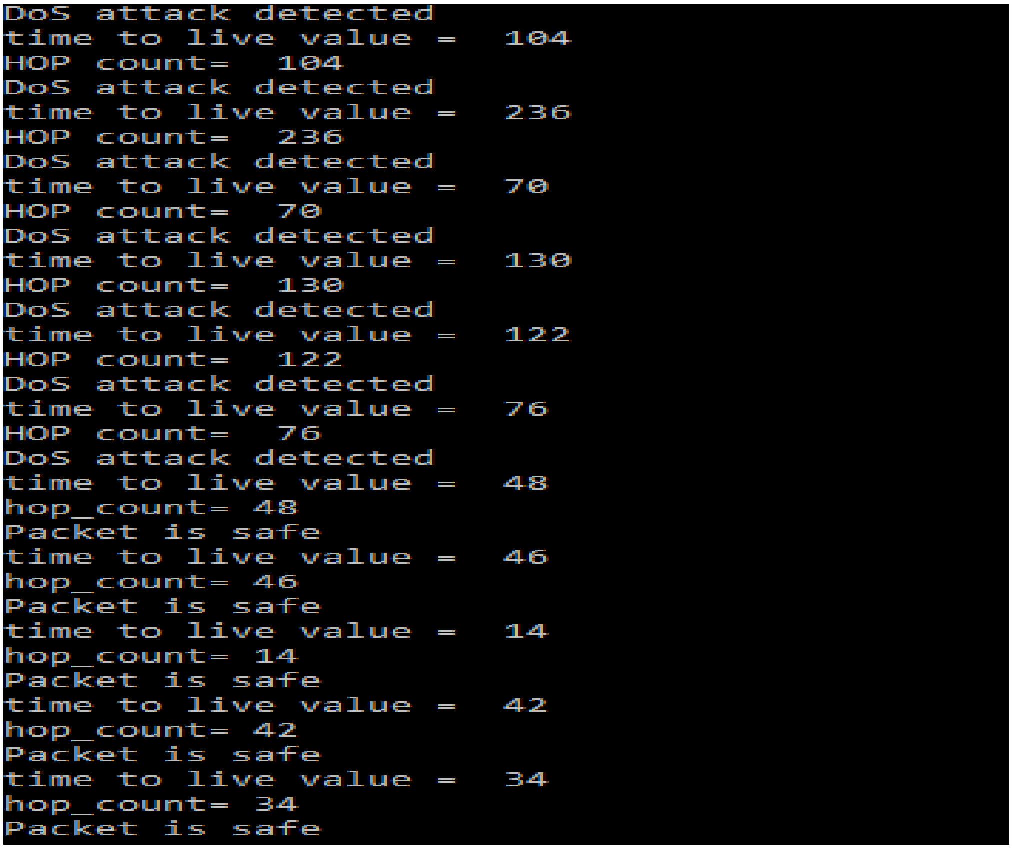


```
$ sudo python pcapy_sniffer.py
['eth0', 'usbmon1', 'usbmon2', 'usbmon3', 'usbmon4', 'usbmon5', 'usbmon6', 'usbmon7',
Available devices are :
eth0
usbmon1
usbmon2
usbmon3
usbmon4
usbmon5
usbmon6
usbmon7
any
lo
Enter device name to sniff : eth0
Sniffing device eth0
Destination MAC : 00:1c:c0:f8:79:ee Source MAC : 6c:fd:b9:53:6a:21 Protocol : 8
Version : 4 IP Header Length : 5 TTL : 250 Protocol : 17 Source Address : 61.1.96.71
Source Port : 53 Dest Port : 56291 Length : 136 Checksum : 28619
stackexchangecom?ny
                 o@"we?mns3
                           serverfault?%?mns1?N?mns2?N
Destination MAC : 6c:fd:b9:53:6a:21 Source MAC : 00:1c:c0:f8:79:ee Protocol : 8
Version : 4 IP Header Length : 5 TTL : 64 Protocol : 1 Source Address : 192.168.1.101
Type : 3 Code : 3 Checksum : 23788
stackexchangecom?G?e5???o???socketsny
                 o@"we?mns3
                           serverfault?%?mns1?N?mns2?N
```

Figure 4: Packet Parsing Output

(c) Detection Alert

After the Hop-count computation has been gotten, the limit will be checked. Once the limit exceeds 50, notification will be displayed showing that a DoS attack has been detected. Figure 5 shows the parsing results.



```
DoS attack detected
time to live value =   104
HOP count=   104
DoS attack detected
time to live value =   236
HOP count=   236
DoS attack detected
time to live value =   70
HOP count=   70
DoS attack detected
time to live value =   130
HOP count=   130
DoS attack detected
time to live value =   122
HOP count=   122
DoS attack detected
time to live value =   76
HOP count=   76
DoS attack detected
time to live value =   48
hop_count= 48
Packet is safe
time to live value =   46
hop_count= 46
Packet is safe
time to live value =   14
hop_count= 14
Packet is safe
time to live value =   42
hop_count= 42
Packet is safe
time to live value =   34
hop_count= 34
Packet is safe
```

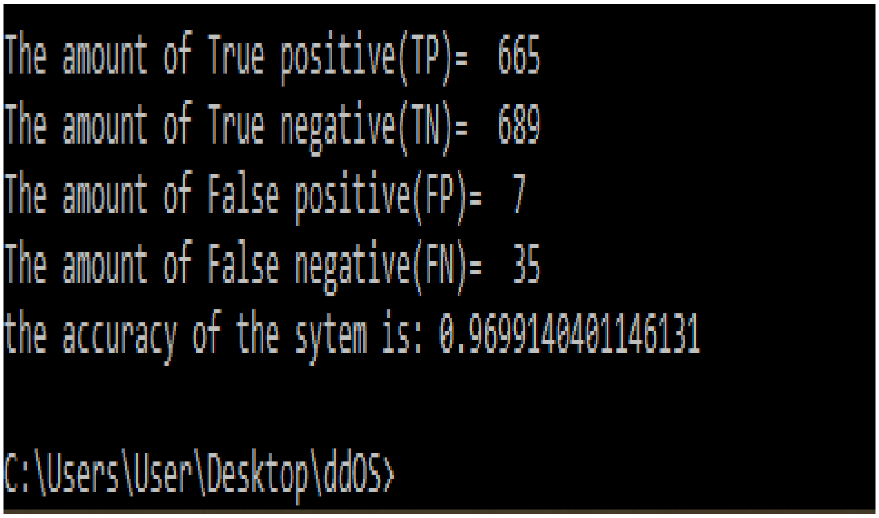
Figure 5: Detection Results

5.0 Performance Evaluation

The performance of the proposed packet filtering system ‘DoSPrevent’ is measured the use of confusion metrics. This metrics helps us to evaluate the performance of the packet filtering technique for detecting denial of service attack in terms of True Positive, True negative and accuracy. A True Positive (TP) is the number of correctly detected Denial of service attacks. The False Positive (FP) can be described as the number of incorrectly detected attacks. A True Negative (TN) is the number of correctly detected safe packets. The False Negative (FN) is the number of incorrectly detected safe packets. The Accuracy is how perfected the detection system is and can be calculated as:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

The system results for the evaluation metrics is shown in figure 6.



```
The amount of True positive(TP)=  665
The amount of True negative(TN)=  689
The amount of False positive(FP)=  7
The amount of False negative(FN)=  35
the accuracy of the sytem is: 0.9699140401146131

C:\Users\User\Desktop\ddOS>
```

Figure 6: System Results for the Evaluation Metrics

From the system result, TP = 665, TN = 689, FP = 7, FN = 35. Accuracy of the detection = 97% A bar chart displaying the evaluation measurements is shown in figure 7.

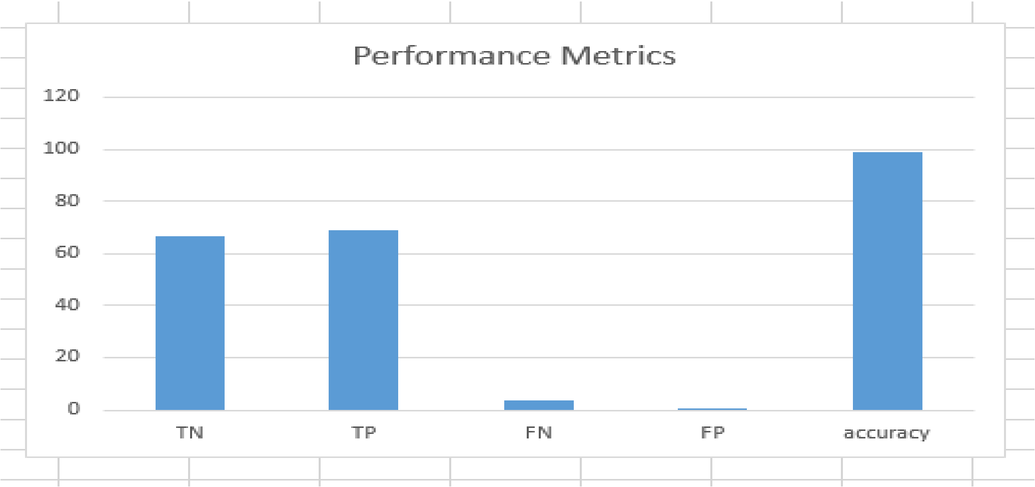


Figure 7: Bar chart showing the Evaluation Measurements

6.0 Conclusion

In this work, the packet filtering technique to compute hop-count and detect denial of service (DoS) attack is proposed and named ‘DoSPrevent’. The proposed system serves as a preventing mechanism for information infrastructure against denial of service attack (DoS). The system can only prevent the flow of DoS attack on the network so as to avoid any form of damage caused by DoS on the infrastructure. The system gives accurate detections and did not give a false positive detection. The system monitors the packets coming into the information infrastructure’s network and proactively detecst DoS attack before damage ng the system.

The proposed system, ‘DoSPrevent’ is able to conquer the problem of false positive detection and slow rate of detection observed in the past systems by filtering packets as they are coming into the network. Hence, it is recommended that the DoSPrevent be used to prevent DoS. Further research work can improve the DoSPrevent to be able to detect more attacks on CII.

References

Adejimi, Alaba O., Sodiya, Adesina Simon, Ojesanmi, Olusegun A. and Adeniran, Olusola J. (2024). "A structured model for identification and classification of critical information infrastructure," International Journal of Critical Infrastructures, Inderscience Enterprises Ltd, vol. 20(2), pages 139-162.

Booth T. and Andersson K. (2020). Critical Infrastructure Network DDoS Defense, via Cognitive Learning”. In 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC) pp. 1-6. IEEE

Cepheli Özge , Buyukcorak Saliha and Karabulut Kurt Gunes (2016). Hybrid Intrusion Detection System for DDoS Attacks. Journal of Electrical and Computer Engineering. Pg. 1-8. OI: 10.1155/2016/1075648.

Fransiscio, S. D. Lima Filho, F. A. F. Silveira, A. de Medeiros Brito Junior, G. Vargas-Solar, L. F. Silveira (2019). Smart detection: An online approach for DoS/DDoS attack detection using machine learning, Secur. Commun. Netw., vol. 2019, no. december, pp. 1–15, Oct. 2019. doi: 10.1155/2019/1574749.

Gu Qijum, Liu Peng (2007). Denial of Service Attacks. In book: Handbook of Computer Networks: Distributed Networks, Network Planning, Control, Management, and New Trends and Applications, Volume 3. DOI: 10.1002/9781118256107.ch29

Hurst, W. Shone, N. and Monnet, Q. (2015). Predicting the Effects of DDoS Attacks on a Network of Critical Infrastructures, *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pp. 1697-1702, doi:10.1109/CIT/IUCC/DASC/PICOM.2015.256.

Jagdeep Singh and Sunny Behal (2020). Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions. *Computer Science Review Volume 37, 100279.*

Jamali, M., Baghaee, H.R., Sadabadi, M.S., Gharehpetian, G.B. and Anvari-Moghaddam, A. (2023). Distributed Cooperative Event-Triggered Control of Cyber-Physical AC Microgrids Subject to Denial-of-Service Attacks. *IEEE Transactions on Smart Grid*.

Ortega-Fernandez, I. and Liberati, F. (2023). A Review of Denial of Service Attack and Mitigation in the Smart Grid Using Reinforcement Learning. *Energies*, *16*(2), p.635.

Özge Cepheli, Saliha Büyükçorak, Güneş Karabulut Kurt (2016). Hybrid Intrusion Detection System for DDoS Attacks. *Journal of Electrical and Computer Engineering, vol. 2016, Article ID 1075648, 8 pages*.

Sivakalai & Jayapriya Jayapal (2014). Identification & Avoidance of DDoS Attack for Secured Data Communication in Cloud. International Journal of Research in Computer Applications and Robotics. Vol.2 Issue.11, Pg.: 155-160 (2014).

Tan Z., Jamdagni A., He X., Nanda P., Liu R.P. (2011). Denial-of-Service Attack Detection Based on Multivariate Correlation Analysis. In: Lu BL., Zhang L., Kwok J. (eds) Neural Information Processing. ICONIP 2011. Lecture Notes in Computer Science, vol 7064. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24965-5_85.

Vitaly Klyuev (2014). An Intelligent DDoS Attack Detection System Using Packet Analysis and Support Vector Machine. *International Journal of Intelligent Computing Research (IJICR), Volume 5, Issue 3.*

Zhao, N., Zhao, X., Chen, M., Zong, G. and Zhang, H., (2023). Resilient distributed event-triggered platooning control of connected vehicles under denial-of-service attacks. *IEEE Transactions on Intelligent Transportation Systems*.