Smart Detection and Handling of DDoS Attacks

Aakriti Suman [1], Abhigya Parashar[2], Annada Shukla[3], Diya Bhaumick[4], Dr. K. R. Shobha[5], Dr. Satish Tunga[6]
Dept. of Electronics and Telecommunication Engg
Ramaiah Institute of Technology
Bangalore, India
aakritisuman0@gmail.com[1], abhigyaparashar22@gmail.com[2], annadashukla98@gmail.com[3], dbhau22@gmail.com[4], shobha_shankar@msrit.edu[5], satish.tunga@msrit.edu[6]

***Abstract*—The present era depends entirely on the Internet which serves as a global source of knowledge for all users. Distributed Denial-of-Service (DDoS) attack is a malicious attempt which sabotages a targeted website, device or network's usual traffic by flooding the target or its surrounding infrastructure with Internet traffic causing damage to network stability. This work focuses on the development of an algorithm to detect a DDoS attack with all forms of TCP flag floods used to overload the server. Since DDoS attacks can cripple the internet infrastructure and resource availability to a great extent, this work also includes designing an algorithm to counteract all the malicious IP addresses that flood the internet. It ensures smooth communication and no denial of services to clients in a network. It blocks the unwanted IP addresses, keeping the system safe from service request overflow due to DDOS attacks. Road accidents are a major problem in today's scenario. Rash driving, drunk driving, and unwillingness to follow the traffic rules are its major causes.**

Keywords—Denial of Service, DDoS, IP addresses, TCP flooding, flags, cybersecurity

I. INTRODUCTION

The recent growth in the IT sector and corporations' DDoS attacks has taken on a toll on the company's tech infrastructure. The growing number of vulnerable Internet of Things (IoT) apps that are being compromised and recruited into botnets such as Reaper is one explanation for their increased prevalence. DDoS attacks have increased significantly in a year or two by a range of 160-180 percent. The amount of data fired at victims of the DDoS attack has also improved dramatically, due mainly to amplification attacks like the strategy of the Memcached amplification attack. Earlier this year, about 15,000 Memcached assaults were initiated by cybercriminals, including a GitHub attack that maxed at an incredible 1.35 Tbps. Although the number of attacks across all size categories rose substantially, small-scale attacks (5 Gbps and below) again saw the highest growth in 2019, continuing the pattern from the previous year. The combination of DDoS-for-hire and botnet rental services has made DDoS attacks much easier to carry out, but the fact that the perpetrators appear to want to participate in small-scale attacks in many cases suggests that their goal can often be anything but to take a site offline altogether. It's almost impossible to prevent a DDoS attack when malicious actors can launch over 1 Tbps on your servers, and that means it's more important than ever to understand how to stop a DDoS attack after it has started to affect your high development and integration of computer technologies based on TCP / IP operations, such as a quick microprocessor, colossal memory, high-speed network, and reliable system design [1]. Distributed Denial of Service (DDoS) attack is one of the main problems and impediments that impact service quality. In a DDoS attack, the attackers use a group of computers, often loaded with a trojan, to overload their target server 's resources or bandwidth, in an effort to drain target resources, interrupt contact, and deprive legitimate users the services. TCP SYN flood attacks are the most prevalent DDoS attacks, as recorded in Kaspersky lab[2], and in Q3 2016, over 81 percent of DDoS attacks are TCP SYN flooding. The attackers exploit the vulnerability of the TCP protocol (three handshakes) to give the victim server a series of SYN requests to pull down and block the services. To address these types of attacks, in particular, TCP SYN flood attack the detection algorithm is a good choice. The detection algorithm analyzes network traffic to detect an event or a set of events that match a specific pattern, such as a sequence of bytes in network traffic, or known malicious instruction sequences used by malware while the detector algorithm begins to work by monitoring network traffic, and it classifies traffic as either normal or abnormal based on heuristics or regulations. Most of the previous detection approaches to detect the SYN flood attack are based on the SYN arrival time, the discrepancy between the SYN and Non-ACK packets, or the Non and FIN packets. It tracks different network activities to assess whether or not an intruder operation has occurred, and will send alerts to the system or network administrator as soon as an attack is detected. This work focuses on detecting SYN flood attacks in the detector-based virtual world that used the statistical features defining the characteristics of the TCP / IP headers. Using Wireshark Network tool, several features are removed, after which the most important features were picked based on various filter algorithms, eventually, the final data set was analyzed using different algorithms and mitigated the attacks.

II. RELATED WORK

In [3], the authors implemented an adaptive mitigation strategy to guard against SYN flood attacks; used advanced rules in windows firewalls to track TCP traffic; the threat is observed and thwarted by the firewall if the gap between the IP address counter that passes the network more than once and the IP address counter that completed its three handshakes exceeds the predefined threshold. The experiment 's findings have shown that the proposed one can predict and minimize a flood attack by SYN. However, detecting the SYN flood attack and preventing it based on the firewall, especially the firewall easily melts under a load of a trivial attack, is not successful. Detection of irregularities was proposed by[4], based on the TCP / IP header. Our work detected the SYN flood attack by filtering the packets based on key rules; sorting the protocol, TCP flags, and IP addresses whether spoofed or not. The abnormal packet was then analyzed using the characteristics of the IP header, such as detection, time to live,

and total IP length, plus TCP headers such as source ports and header length. The proposed system has shown good work in detecting SYN flood attacks, but detecting SYN floods based on such features is not enough, particularly the attacker may use the normal packets as a legitimate user today. Throughout the wide network, SYN per second is usual, while the change can be an attack in the network. For detecting a SYN flood attack, the authors in [5] used a difference between incoming SYN packets and outgoing SYN-ACK packets. Their work relied on linear prediction analysis to estimate over time the SYN flood attack. The method proposed has obtained good results with a small delay in detection. On the other hand, it proposed a model for detecting SYN flood attacks [6].

The authors presented a framework that integrates the system based on thresholds and misuse detection systems, their implementation framework included CPU load analysis before, during, and after TCP SYN flood attacks. The findings of the proposed system revealed that the CPU load was increased during the attack, and the CPU load was effectively reduced after the detection process. A mechanism was presented to detect SYN flood attacks[7], their work depends on the variation in the arrival time of packets, they divide the arriving packets into five groups according to their flow flags, such as a packet traffic group that completes the three handshakes, and a traffic group that ends with Reset flag. The findings revealed that this approach can easily detect the high-rate attack but can not detect the low-rate attack where the traffic meets the same distribution as legal traffic. Much of the previous research relied on the specified interval on the counter of SYN packets to detect SYN flood attacks, these methods showed good performance. The identification mechanism for each flow relies on new mathematical features that are derived from the TCP / IP header. The traffic from the virtual world was obtained to evaluate and remove the features, after which the optimal features were selected using the process of intersection between three common filter methods.

III. Algorithm Design

TCP is a connection-oriented protocol, which means a connection is established and maintained until the application programs at each end have finished exchanging messages. The server must be listening (passive open) for connection requests from clients before a connection is established. The Transmission Control Protocol (TCP) is one of the main protocols of the Internet protocol suite. It determines how to break application data into packets that networks can deliver, sends packets to and accepts packets from the network layer, manages flow control and -- because it is meant to provide error-free data transmission -- handles retransmission of dropped or garbled packets and acknowledges all packets that arrive. In the Open Systems Interconnection (OSI) communication model, TCP covers parts of Layer 4, the transport layer, and parts of Layer 5, the session layer.

TCP is important because it establishes the rules and standard procedures for the way information is communicated over the internet. It is the foundation for the internet as it exists today and ensures that data transmission is carried out uniformly, regardless of the location, hardware or software involved. For this reason, it is flexible and highly scalable, meaning new protocols can be introduced to it and it will accommodate them. It is also nonproprietary, meaning no one person or company owns it.

The algorithm works on various realizations and studies. The detection and mitigation procedure of the algorithm depends on various studies and concepts of network trafficking and network protocols. The algorithm is designed in such a way that it satisfies all the technicalities of a secure networking system.

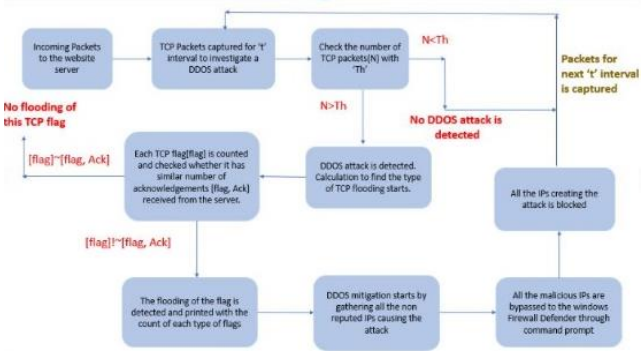


Fig. 1. Design of proposed algorithm

Fig 1 shows a detailed block diagram of the working of the algorithm. The algorithm is an iterative process which helps the system to keep checking for DDOS attack at regular intervals. A detailed description of the algorithm is explained below :

1. The first step is to analyse the incoming packets to a web server. No matter what is being done — chatting, video streaming, gaming, surfing, buying stuff — it's essentially an exchange of data packets between two computers (networks). A 'packet' is the smallest unit of information flowing in a network (or between networks).

The packets carry the data in the protocols that the internet uses i.e. Transmission Control Protocol/Internet Protocol (TCP/IP), Hyper-Text Transfer Protocol(Http), User-Datagram Protocol(UDP). These packets can be analysed using various network packet analyzers such as WireShark, tcpDump, NetworkMiner, Fiddler, EtherApe, CommView etc. In this project, Wireshark has been used to analyse the packets received on a domestic machine.

2. The second step is to capture TCP packets from the analyzed packets in step 1 for the 't' interval of time. This captured packet is then used to detect whether it is creating a DDOS or not. As discussed in section 5.3, the calculation of interval 't' depends on the traffic of a particular web server.

3. The TCP packets captured from the web server for the 't' interval are counted and checked with the Threshold(Th). If the number of packets received in this interval is greater than the determined threshold, it is considered as occurence of DDOS attack. The determination of threshold totally depends on the network traffic and sudden abrupt increase in the number of packets clearly indicates that it is unusual and a DDOS attack is declared. If the number of packets are less than or equal to threshold, then a clear indication of 'No DDOS ATTACK' in the command prompt is visible else 'A DDOS ATTACK IS DETECTED' is printed and the procedure to find the type of the flooding starts.

4. Each TCP flag sent by the client is acknowledged by the server.

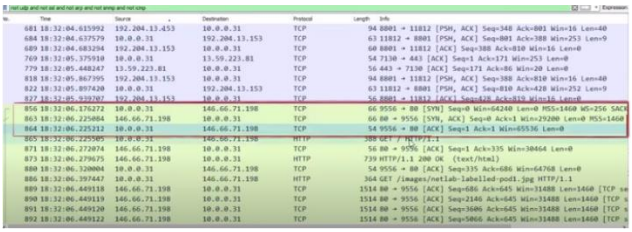


Fig. 2. Acknowledgement of SYN flags during three way handshake process

In Fig 2, the area marked as brown shows how a three-way handshake takes place. A [SYN] packet is sent by the client to request for connection establishment from the server. The server sends a [SYN, ACK] packet to acknowledge the request and finally an [ACK] packet is sent by the client to the server to confirm the connection establishment. This is one example where a TCP flag is acknowledged by the server. Similarly other TCP flags such as [FIN], [URG], [PSH] etc are acknowledged by sending [FIN, ACK], [URG, ACK], [PSH, ACK] respectively from the server to the client. Hence, for a given 't' interval, the number of flags and its respective acknowledgements should be similar in number. During DDOS attack, the difference between these two is quite large which confuses the server and the network which ultimately floods the network with unwanted packets. The intensity of the flooding of a particular type of flag depends on the difference between the number of [flag] and [flag, ACK] where the flag stands for any type of TCP flags. The algorithm prints the type of flooding by comparing the number of flags and its respective acknowledgements and tells what type of TCP flooding is happening. The algorithm also prints the count of each type of TCP flag. Sometimes, the flooding can also be of type -Acknowledgement. Fo eg: flooding of a server with [PSH, ACK], [SYN, ACK] etc.If the number of [PSH,ACK] as compared with [PSH], then it is [PSH, ACK] type of attack as shown in Fig 3 below

Fig. 3. [PSH, ACK] flooding

Hence, the algorithm also prints the acknowledgements of all types of flags as well.

5. After detecting a DDOS attack, the algorithm starts the mitigation process automatically. During a DDOS attack, the packets from reputed IPs cannot access the web server hence no genuine IP will be detected while checking for a DDOS attack. Each web server has its own capacity to withstand the number of incoming packets. During a DDOS attack, the server is overwhelmed with a huge number of TCP packets and hence does not have the capacity to respond to any IP. Hence, a genuine IP does not have the capacity to reach the server and finally an 'Access Denied' message pops up on the screen. This is the reason why a genuine IP cannot access the website during a DDOS attack. Therefore, when the algorithm detects a DDOS attack for the particular time interval 't', it extracts all the IPs which are responsible for flooding of the server.

6. As explained in step 5, all the malicious IPs are extracted and are bypassed to the Windows Firewall Defender through command prompt. The command prompt keeps running in Administrator mode in the background.

7. The algorithm goes back to step 2 and then again captures the packet for the next 't' seconds from the incoming packets of the web server.

Whole algorithm is iterative in nature. It is a loop which will keep detecting DDOS attacks and will keep intimidating the user in the background about the nature of the incoming packets.

IV. Implementation of Algorithm

This section deals with how the algorithm, explained in sec III, can be executed using python. A dataset has been acquired using wireshark. To execute this program, all commands will be run using command prompt in administrator mode.
The program flow of the python script is shown in fig 4.

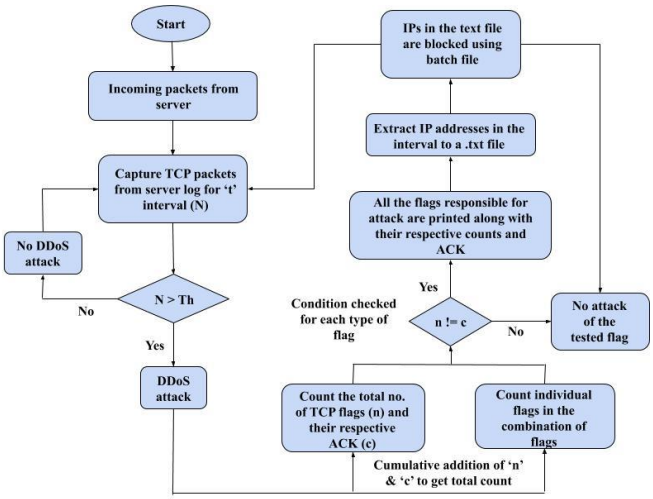


Fig. 4. Program flow of algorithm

The flowchart has been explained in this section. To run the code, open the command prompt in administrator mode and direct it to the location where the code is saved for execution. The following steps are then executed:

- The system is connected to the internet. The continuous network traffic to and fro from the system is captured using wireshark. This traffic is stored in a csv file. The python script reads this csv file which consists of all kinds of incoming and outgoing traffic of the system.
- Since this project deals with DDoS attacks using TCP packets, the code filters out the packets that follow the TCP protocol from the rest of the protocols such as HTTP, HTTPS, UDP, ARP etc. The code also preprocesses the data and removes junk value to prepare the data for further operation.

- The code works on a ‘t’ second interval time. The code considers the traffic on the system for ‘t’ second intervals and captures them. Once the packets dealing in TCP protocol have been extracted from the captured set, the total number of such packets is counted. ‘Th’ is a threshold value that is specific to each system. The threshold is a limiting value of the number of packets in a system beyond which a DDoS attack is declared. This threshold value is determined by a systematic study of the pattern of traffic in a system in a network as this value can vary greatly between big systems like servers, websites and small systems like home PCs.
- If the number of packets is within the threshold value, there is no DDoS attack. The code goes on to capture packets for the next ‘t’ seconds interval and repeats the procedure.
- If the number of packets is beyond threshold value, a DDoS attack is declared. However, it is not enough to just decide on a DDoS attack. The nature of the attack must be studied too. To eliminate the chances of just high traffic being tagged as DDoS attack, the flags raised in each of these packets must be studied.
- After determining a high traffic on the system, count the flags in each packet. Flags are associated with every packet in a network. For every flag incoming, the system sends an acknowledgement flag to confirm the reception of the packet. In a DDoS attack, the resources of the system are overwhelmed and an acknowledgement flag for every packet cannot be sent. The code takes advantage of this phenomena. It counts every flag as ‘n’ and its respective acknowledgement flags as ‘c’. If ‘n’ is equal to ‘c’, then attack is not of that particular flag. If ‘n’ is not equal to ‘c’, it means that system is flooded with a DDoS attack of that particular flag. This step is repeated for all flags as DDoS attack does not necessarily deal with flooding of one TCP flag.
- On the occasion when attackers raise multiple flags in one flag to increase flooding but to pass undetected, the code counts individual flags in a combination and adds them cumulatively to the original count.
- To provide data for analysis for field experts, the count of each flag and their acknowledgements are printed in a tabular column for further study.
- It is known that when traffic is captured on a network, the source and destination IPs are also captured. If a DDoS attack is declared for a particular ‘t’ interval, the IPs which were responsible for the attack are also known. The only way to stop a DDoS attack is to block these IPs. To block them, IPs are extracted from the dataset to a .txt file. A batch file reads this text file to block the IP addresses.
- Manually blocking the attacker IP addresses is a big ordeal. To automate this process, a batch file is created to block them. The batch file uses ‘netsh commands’ to add new rules to Windows defender firewall and advanced security. Netsh commands require administrative privileges to function. Hence, batch file is provided with administrative privileges as soon as it is called. This quickly and efficiently blocks all IPs causing the attack.
- Once, all IPs are extracted to a text file in the interval, the code does a subprocess call to trigger the batch file. The batch file reads through the text file and blocks all IPs line by line.
- After the IPs are blocked, the code proceeds to capture TCP packets for the next ‘t’ intervals and the process repeats.

The code can successfully and efficiently block DDoS attacks.

V. Results and Discussions

This section shows a detailed description of results obtained from the algorithm. The discussion contains a detailed analysis of the outputs generated by the attacker's side using the LOIC tool as explained in earlier sections of this report and the outputs generated by the DDoS algorithm to mitigate the attack. Hence, to simplify the discussion, this section is divided into two parts- Attacker’s Side and Victim’s Side.

A. Attacer’s side:

1. Generation of DDoS attack:
   To generate a DDoS attack on a particular host, the IP address of that host is needed to bombard a huge number of packets from the attacker tool. Fig 5 is a representation of the command window when the ‘ipconfig’ command is typed in order to use the ‘arp -a’ command which scans all the neighboring IPs under the same network. This is shown in Fig 8.

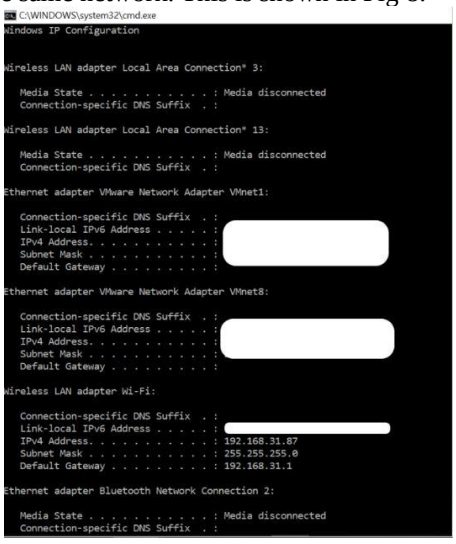


Fig. 5. Finding active local IP addresses

The column ‘Type’ shows whether the IP is ‘dynamic’ i.e. whether the IP changes with the change in network or is ‘static’. The Internet Addresses listed are the IPs that are in the same network as of attacker for eg. the same router is being used by both the attacker and the victim. Either one or all of them can be attacked. In this project, only one IP address is chosen as a victim IP.

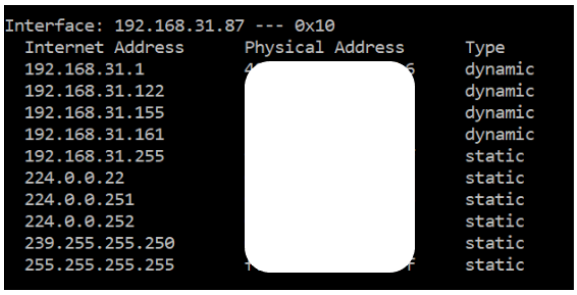


Fig. 6. Finding active local IP addresses

Besides the IP, an attacker also needs the port number of that particular IP. IP address, protocol and the port number are the three inputs to the LOIC tool.

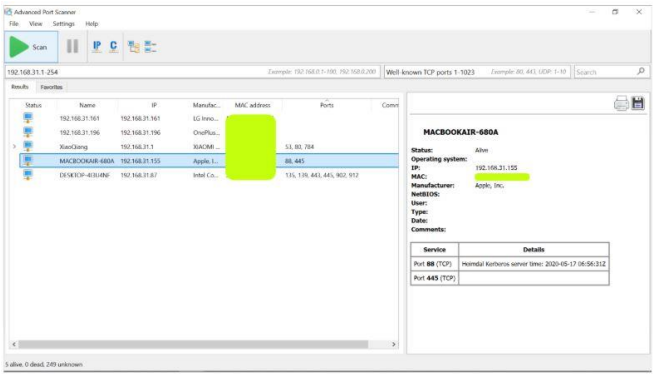


Fig. 7. Advanced port scanner output

An advanced port scanner is used to detect the ports available for the chosen IP. In Fig 7, the highlighted row shows the victim IP address, the machine type of the victim IP as well as the available ports which can be attacked.

Fig 8 shows the LOIC attacking interface. From this tool, either a website's URL or an IP address can be locked on and attacked. The image shows the highlighted IP address that is going to be attacked after getting locked on. The port dialogue box shows the port selected for attacking (chosen from fig 7). Method shows the protocol that has been chosen. LOIC can generate TCP, UDP or HTTP kind of attacks. Since, this project exclusively detects and mitigates TCP flooding, TCP option is chosen. The threads option is filled which determines the number of packets that will be sent to the Victim IP per second. A fake TCP message is set. The rate of generation is also selected using the speed slider After filling all these options, the attack is finally generated. The generated TCP DDoS attack is shown in Fig 9.

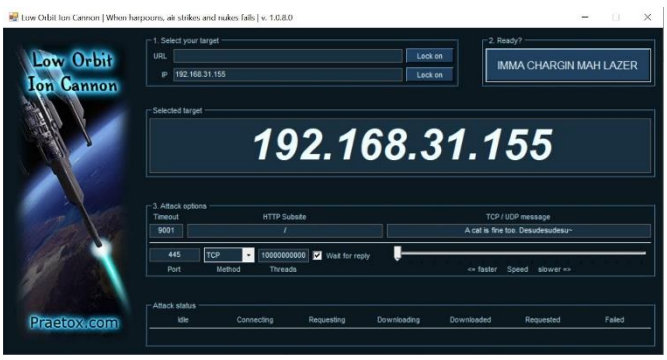


Fig. 8. LOIC tool interface

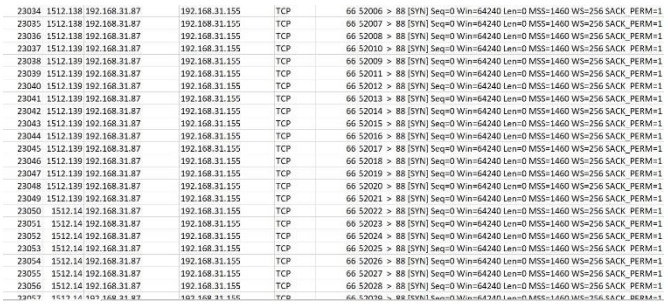
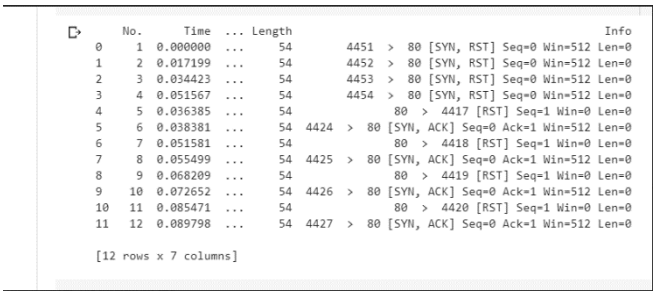


Fig. 9. DDoS attack generated

B. *Victim's side:*

One of the most important advantages of this algorithm is that it needs administrator rights to run the algorithm on the user's PC. Hence inappropriate use of the algorithm is restricted by the algorithm itself. The packets captured after the attack is generated is stored in a csv file. A preview of the captured packets is shown in Fig 10

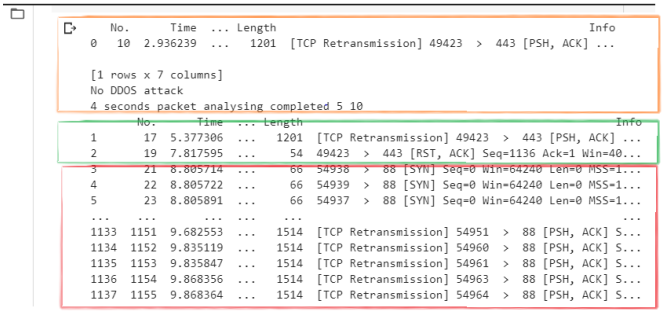


	No.	Time	...	Length				Info
0	1	0.000000	...	54		4451	> 80	[SYN, RST] Seq=0 Win=512 Len=0
1	2	0.017199	...	54		4452	> 80	[SYN, RST] Seq=0 Win=512 Len=0
2	3	0.034423	...	54		4453	> 80	[SYN, RST] Seq=0 Win=512 Len=0
3	4	0.051567	...	54		4454	> 80	[SYN, RST] Seq=0 Win=512 Len=0
4	5	0.036385	...	54		80	> 4417	[RST] Seq=1 Win=0 Len=0
5	6	0.038381	...	54	4424	> 80		[SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
6	7	0.051581	...	54		80	> 4418	[RST] Seq=1 Win=0 Len=0
7	8	0.055499	...	54	4425	> 80		[SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
8	9	0.068209	...	54		80	> 4419	[RST] Seq=1 Win=0 Len=0
9	10	0.072652	...	54	4426	> 80		[SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
10	11	0.085471	...	54		80	> 4420	[RST] Seq=1 Win=0 Len=0
11	12	0.089798	...	54	4427	> 80		[SYN, ACK] Seq=0 Ack=1 Win=512 Len=0

[12 rows x 7 columns]

Fig. 10. Captured packets on the victim's side.

The packets are captured from the incoming server for every 't' interval. Here, the interval is 4 seconds. The captured packet for four second intervals is shown in Fig 11. The packets that lie under the orange rectangular region are the number of packets captured for the first four seconds interval. Since, only one TCP flag is captured, which surely is not a flooding, no DDoS attack is detected. Similarly, in the green rectangular region, there are only two packets captured. This region is formed between 4-8 seconds. This can be easily proved by analysing the 'Time' column. The red rectangular region contains packets from 8-12 seconds. The number of packets captured is high and might lead to DDoS attack. It will be decided in the further part of the algorithm.



	No.	Time	...	Length	Info
0	10	2.936239	...	1201	[TCP Retransmission] 49423 > 443 [PSH, ACK] ...

[1 rows x 7 columns]
No DDOS attack
4 seconds packet analysing completed 5 10

	No.	Time	...	Length	Info
1	17	5.377306	...	1201	[TCP Retransmission] 49423 > 443 [PSH, ACK] ...
2	19	7.817595	...	54	49423 > 443 [RST, ACK] Seq=1136 Ack=1 Win=40...
3	21	8.805714	...	66	54938 > 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1...
4	22	8.805722	...	66	54939 > 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1...
5	23	8.805891	...	66	54937 > 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1...
...	...	...	...	...	...
1133	1151	9.682553	...	1514	[TCP Retransmission] 54951 > 88 [PSH, ACK] S...
1134	1152	9.835119	...	1514	[TCP Retransmission] 54960 > 88 [PSH, ACK] S...
1135	1153	9.835847	...	1514	[TCP Retransmission] 54961 > 88 [PSH, ACK] S...
1136	1154	9.868356	...	1514	[TCP Retransmission] 54963 > 88 [PSH, ACK] S...
1137	1155	9.868364	...	1514	[TCP Retransmission] 54964 > 88 [PSH, ACK] S...

Fig. 11. Packets captured in 4sec interval.

As mentioned in the earlier part of this section, the initial step is to open the command prompt of the PC in administrator mode as shown in Fig 12. Then the path where the algorithm

resides is added and the file is run. Here the algorithm's file name is 'final.py'.

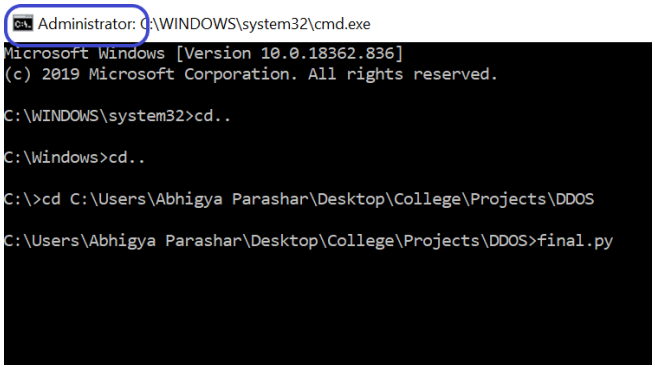


Fig. 12. Cmd in administrative mode

As explained in Fig 11, the orange and rectangular regions do not show DDoS attack. The same has been detected by the algorithm as well. This can be analysed in Fig 13. It is a snap of the cmd window, which shows that from t0=0 to t0=4 (not inclusive), there is no DDoS attack as the length of packets is 1 i.e. number of TCP packets captured in only 1. Similarly, from t0=4 to t0=8, the number of packets captured is 2 and hence no DDoS is detected again.

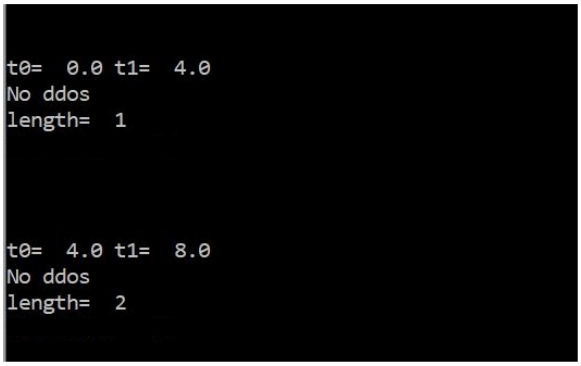


Fig. 13. No DDoS detected in the first two intervals

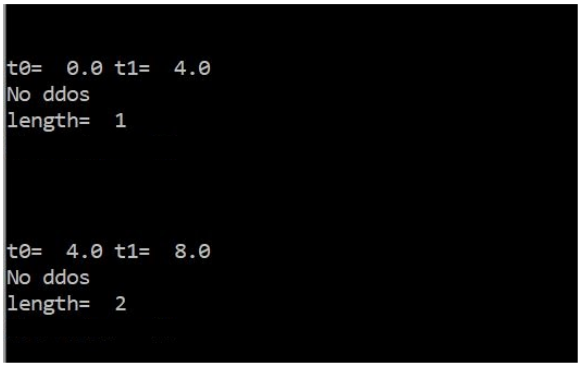


Fig. 14. No DDoS detected in the first two intervals

Fig 14 is a very important part of the algorithm. In the interval of 8-12 seconds, the number of packets captured is very high and hence a DDoS attack is detected. This is indicated as a rectangular region in Fig 7.8. The IPs in the curly braces show the IPs which are malicious and responsible for generating the attack. The attacker packets are coming from these IPs and must be blocked to mitigate the attack. The third and fourth row tells the type of attack the PC is experiencing. In this case, two types of TCP floods are happening - RST/RST-ACK and PSH/PSH-ACK. This will help the technical person to analyse the type of attack at the initial stage only. This will help in taking necessary actions from the victim's side such as restricting packets containing RST and PSH flags temporarily. The algorithm also shows the count of each type of flag. As explained earlier, each flag should have an equivalent number of respective Acknowledgments. In Fig 14, the number of packets with RST flag is 526 but with [RST, ACK] is only 36 which is clearly not equivalent. Similarly, the number of packets with [PSH] flag is 0 but with [PSH, ACK] is 1321. Hence, to be precise the user's PC is experiencing RST-ACK and PSH-ACK type TCP flooding.
As discussed earlier, in Fig 14, the list of IP addresses which are visible on the second row of the output screen are responsible for the flooding of TCP packets on the victim's IP. This leads to the beginning of the mitigation process of the attack. These IPs are collected in a text file and each line which consists of one IP, is read by the batch file. This can be understood from Fig 15. After reading every IP, the batch file returns OK. This means that the IP read is now accepted by the system and the blocking procedure can start. When each IP is read and accepted by the batch file, a rule book is created for that IP as shown in Fig 16. In the rule book, Rule name is set to 'Block', Enabled is set to 'Yes', Direction is set to 'In', Profiles is set to 'Domain, Private, Public', and the Remote IP is set with the IP that needs to be blocked. This rule book means that all the incoming IPs with this IP value will be blocked. The IP can come from Domain, private or any public profile. In all these conditions, this IP will be blocked by the system and hence, never the same IP can be used to generate the DDoS attack to this system again.

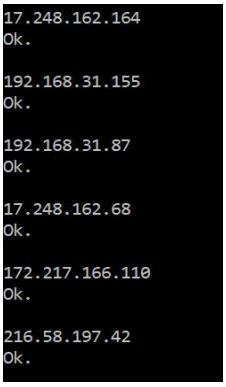


Fig. 15. List of Ips to be blocked

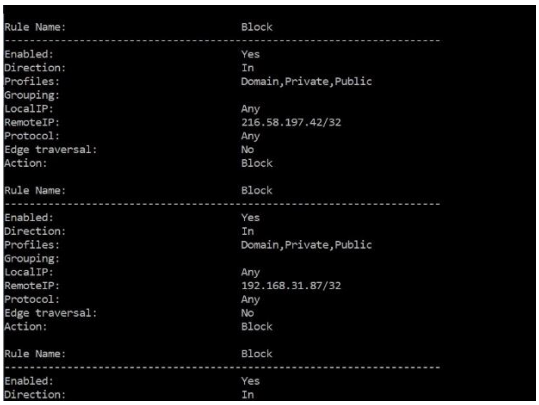


Fig. 16. IP addresses blocked

In the last interval, i.e. when t0=28 and t1=32, no DDoS attack is detected as the length of the packets is only 6 which surely says that it is not an attack. This is shown in Fig 17.

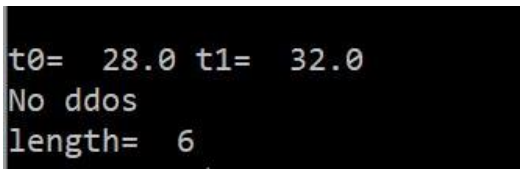


Fig. 17. Last interval

To verify that the IPs are blocked, the user can run the Windows Firewall Defender application and check in the Inbound Rules tab. In Fig 18, the blue rectangular region contains a series of rows that shows that some characteristics are being blocked to access the PC. To check these characteristics, right click on that characteristic and select block properties. This is shown in Fig 19.

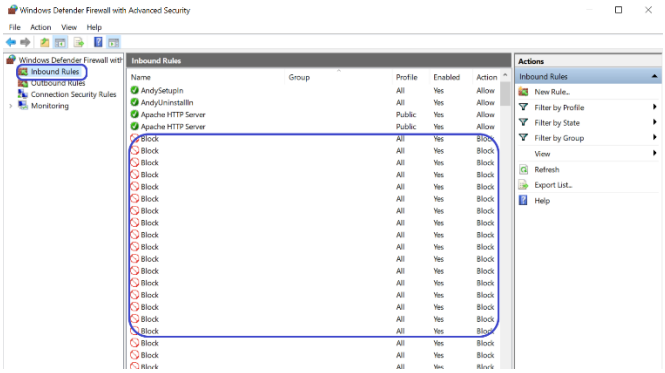


Fig. 18. Windows Defender Firewall and advanced security

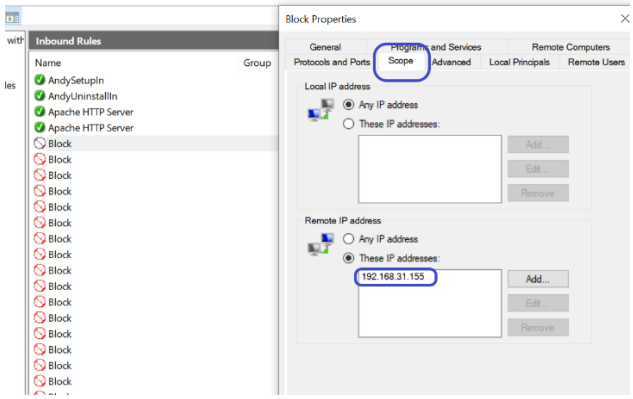


Fig. 19. Rule Block Properties

Select the ‘Scope’ pane. The IP address which is blocked will be visible. It is marked as a rectangular encircle in Fig 19.

This concludes the generation, detection and mitigation of a TCP DDoS attack on a system.

VI. Conclusion

The growing number of DDOS incidents as well sophistication in DDOS attacks of present age poses undefeated challenges before many internet services. The main issues defending against DDOS attacks are:

- To detect different types of DDoS attack reliably and accurately.
- To segregate attack traffic from legitimate traffic
- To react against the huge volume generated by DDOS attack in an automated manner with minimum collateral damage.

In this work we have tried to address in defending web services against DDOS attacks. The dataset is successfully generated using the LOIC tool which consists of different types of DDOS attack in TCP protocol. The dataset is preprocessed according to the algorithm requirement. The algorithm detects the DDOS attacks in ‘t’ interval of time. The DDOS attack detected is of TCP protocol hence different TCP flag flooding is detected every defined interval. Once the flag flooding is detected, the algorithm works to mitigate these malicious Bots or IPs. These source IPs are taken from the original dataset and the whole algorithm works through command Prompt. The vast improvements in performance under proposed defense clearly manifest supremacy of approach.

References

[1] C.Gong, J. Liu, Q. Zhang, H. Chen, and Z. Gong, “ The characteristics of cloud computing”. In Parallel Processing Workshops (ICPPW), 2010 39th International Conference on, 2010, pp. 275-279. IEEE.

[2] Kaspersky Lab, https://securelist.com/analysis/quarterly-malware-reports/74550/kaspersky-ddos-intelligence-report-for-q1-2016/.Mr.Sethuram Rao, Vishnupriya.S.M, Mirnalini.Y, Padmapriya.

[3] K. Hussain,, H. Syed Jawad, D. Veena, N. Muhammad, and A. Muhammad Awais. "An Adaptive SYN Flooding attack Mitigation in DDOS Environment." International Journal of Computer Science and Network Security (IJCSNS) 16, 2016, PP.27-33.

[4] S. H. C. Haris, R. B. Ahmad, and M. A. H. A. Ghani. "Detecting TCP SYN flood attack based on anomaly detection." In-Network Applications Protocols and Services (NETAPPS), 2010 Second International Conference on. IEEE, 2010, , pp. 240-244.

[5] D. Divakaran, H. Murthy, and T.Gonsalves. "Detection of SYN flooding attacks using linear prediction analysis." In Networks, 2006. ICon'06. 14th IEEE International Conference on, vol. 1, IEEE, 2006, pp. 1-6.

[6] D. Kshirsagar, S. Sawant, A. Rathod, and S. Wathore. "CPU Load Analysis & Minimization for TCP SYN Flood Detection." Procedia Computer Science 85,2016, PP. 626-633.

[7] Y. Oshita, S. Ata, and M. Murata, “Detecting Distributed Denial-of-Service Attacks by Analyzing TCP SYN Packets Statistically,” Proceeding of the IEEE Communications Society Globecom, pp.2043-2049, 2004.