*International Journal of*
*Emerging Research in Management &Technology*
*ISSN: 2278-9359 (Volume-5, Issue-5)*

**Research Article** | **May 2016**

# Analysis of Dos and DDos Attacks

**[1]Mustafa Aijaz[*], [2]Suraiya Parveen**
[1]Department of Computer Science, Jamia Hamdard, New Delhi, India
[2]Assistant professor, Department of Computer Science, Jamia Hamdard, New Delhi, India

***Abstract—***

***Availability is one of the three main components of computer security, along with confidentiality and integrity. Denial of service (DoS) and Distributed Denial of Service (DDoS) are threats that exhaust the resources of a system thereby making them unavailable for legitimate users and thus violating the security component-Availability. Dos and DDoS attack pose an immense threat to entire internet world today mainly because of the ease through which they can be carried out. These attacks can easily exhaust the computing and communication resources of its victim within a short period of time. It has become necessary for researchers and developers to understand the behaviour of these DDoS attacks because It affects the target network with little or no advance warning. This paper presents a classification of DoS/DDoS attacks and analysis of some of the major Dos/DDos attacks. The analysis of these attacks is performed using a traffic analyzer. We further discuss an efficient packet filtering technique using firewall to defend against DoS/DDoS attacks. Firewall scripts are written using command-line tool iptables in Linux to deny the suspicious traffic.***

***Keywords— botnet, dos, ddos, firewall, iptables, wireshark***

## I. INTRODUCTION

Denial of Service (DOS) and distributed denial of service (DDos) attacks are a major threat to the Internet today. This Threat in the form of Internet availability is a big issue that is hampering the growth of online organizations as they rely on having their websites available to users 24x7. It leads to the disruption of services by attempting to limit access to these service or a machine. The aims of these attacks is to render a network incapable of providing normal service by targeting the network's bandwidth or its connectivity. These attacks are achieved by sending at a victim a stream of packets that slows or even stops his network or his processing capabilities. The use of single computer to carry out these attacks is termed as DOS attack whereas the use of multiple computers, called zombies to carry out a coordinated attack one or more system or service is termed as Distributed Denial of Service (DDoS) attack. DoS and DDos attacks to networks are numerous and potentially devastating. In DDos attack first, different computers are compromised by using Trojans, worms, etc. and then used by the attackers. These compromised machines are named as zombies while the main controller machine is termed as master. This master zombie relationship works similar to that of a client-server architecture. It can be very difficult to detect the DDoS attacks because the zombies may be situated across the globe. As a result, they cannot be differentiated from the legitimate traffic. Analysis of these attacks is thus a crucial task in determining their impact and implementing the appropriate countermeasure.

In this paper we analyse some of the major Dos/DDos attacks using the traffic analyser, wireshark and discuss the efficient packet filtering technique using firewall to defend against these attacks. Firewall scripts are written using iptables that is a command-line tool in Linux to deny the suspicious traffic. Packet analyzer tool is used to show the effectiveness of the scripts in mitigating the various kinds of DoS/DDoS attacks.

The paper is organized as follows: Section II gives a brief understanding of DoS Attacks, to form the basis for subsequent sections. Section III describes the classification of the attacks on the basis of TCP/IP protocol suite. Section IV describes iptables. Section V presents the experimental setup and measurements and finally, section VI presents the conclusions of this work.

## II. BACKGROUND

Gligor et al. [1] defined DoS as: "a group of otherwise authorized users of a specific service is said to deny service to another group of authorized users if the former group makes the specified service unavailable to the latter group for a period of time which exceeds the intended (and advertised) waiting time." This definition takes into aspect the timeliness aspect of availability, and we use it as the standard definition of denial of service. According to the survey, nearly 92% of the attacks are DoS attacks. This type of attack doesn't cause any

damage to the data but it does not provide the required resource [2].A DoS attack floods a network and system with useless traffic making their resources unavailable for the legitimate users so that no one can access it. The main targets of these attacks are servers, default gateways and even end user systems. Attackers can create a situation in which the organizations can come to a grinding halt. The main aim of these attackers is target the availability of these organisations because doing so does not require any administrative privileges on the target system. Most DoS attacks depend upon the weaknesses in TCP/IP stack protocols. Some of the classical examples of DoS attacks are TCP Syn Flood, UDP Flood, ICMP flood, Smurf and Incomplete HTTP Requests, etc. Attackers either make use of single computer or multiple computers to launch these attacks. The usage of multiple computers to perform the attack is known as DDoS attack. In

 

DDos, Firstly the attacker takes control of multiple hosts over the internet instructing them to contact the target web server[3]. These controller machines are known as master while the targeted hosts are called zombies. The zombies are mainly compromised using Trojans, worms, etc. This master zombie relationship works similar to that of a client-server architecture. It can be very difficult to detect the DDoS attacks because the zombies may be situated across the globe.

Denial of service attacks come in a variety of forms and aim at a variety of services [4]. CERT Coordination Center defines three basic types of attacks[5]: 1) consumption of scarce, limited, or non-renewable resources, 2) destruction or alteration of configuration information, 3) physical destruction or alteration of network components. The targeted resources can be network bandwidth, CPU, memory, I/O bandwidth, disk space, or any combination of them.

DDOS detection and prevention techniques are categorised into three approaches that are usually practiced today. These approaches are based on DDOS detection at the Victim of the attack, Source of the attack and the Intermediate network (Network level). Victim-end DDOS solutions, the most widely used approach, try to defuse DDOS attack at its final stage of the life cycle, at the victim of the attack [6]. These solutions provide direct benefits to the host as it is the only DDOS solution that can stop an incoming DDOS attack. An individual cannot use Source-end or network level DDOS solutions to protect a set of hosts from being attacked because they require large scale cooperation and drastic changes to the routing infrastructure to work effectively. The use of iptables in linux to stop a DDos attack is one of the methods used for DDos prevention in Victim-end DDos solutions.

## III. TYPES OF DOS/DDOS ATTACKS

In DOS/DDOS attacks the attacker sends malformed packets from one or more sources to the victim spoofing the source address of the packets. Some of the common DDoS attacks are discussed below.

### *A. TCP SYN Flood Attack*

In a SYN Flood attack, the victim is flooded with half open connections. The attack uses the three-way handshake mechanism. In a three-way handshake mechanism first The client system begins by sending a SYN message to the server. The server then acknowledges the SYN message by sending SYN-ACK message to the client. The client then finishes establishing the connection by responding with an ACK message. This opens the connection between the client and the server and then the data can be exchanged. However in a TCP SYN flood attack the client does not send an ACK packet to the server. In this case the server is waiting for an ACK packet from the client This status is called a "half open" connection . This kind of incomplete connection is stored in Backlog Queue. After 75 seconds the incomplete connection is removed from Backlog Queue and it is disconnected.

However, if the client sends lots of SYN packets before the server removes incomplete connections from Backlog Queue, then Backlog Queue in the server is overflowed. In this case the server cannot accept TCP connection at all. This type of attack is TCP SYN Flood Attack.

### *B. UDP Flood Attack*

This DDoS attack uses the User Datagram Protocol (UDP), a sessionless networking protocol. In UDP Flood attack attacker sends large number of UDP packets to a victim system, due to which there is saturation of the network and the depletion of available bandwidth for legitimate service requests to the victim system [7]. The victim system on receiving a UDP packet will try to determine the waiting application on the destination port. When there is no application waiting on the port, it will generate an ICMP packet of "destination unreachable" to the forged source address. If UDP packets being delivered to ports of the victim are large the host resources will be sapped which will lead to inaccessibility. In a UDP flood attack, the attacker can also spoof the IP address of the packets .As a result, the return ICMP packets will not reach their host, thereby anonymizing the attack. There are a number of commercially-available software packages that can be used to perform a UDP flood attack (e.g., UDP Unicorn).

### *C. ICMP(ping) Flood Attack*

It is similar to the UDP flood attack. This attack simply exploits the Internet Control Message Protocol (ICMP), which enables users to send an echo packet to a remote host to check whether it's alive. An ICMP flood attack involves flooding the victim's network with request packets. These packets request for a reply from the victim. By generally sending packets as fast as possible without waiting for replies results in the saturation of the bandwidth of the victim's network. This type of attack can consume both outgoing and incoming bandwidth, since the victim's servers will often attempt to respond with ICMP Echo Reply packets, resulting a significant overall system slowdown. Carrying out such an attack is dependent on attackers knowing the IP address of their target. As a result a ping flood attack can be broken down into three categories[8]:

1) A targeted local disclosed ping flood targets a single computer on a local network.
2) A router disclosed ping flood targets routers in order to disrupt communications between computers on a network.
3) A blind ping flood involves using an external program to uncover the IP address of the target computer or router before executing an attack.

### *D. Smurf Attack*

This attack is similar to the ICMP (ping) flood attack as it also uses the echo response mechanism of ICMP. In a smurf attack, the victim is flooded with Internet Control Message Protocol (ICMP) echo-reply packets. This attack uses

 

IP broadcasting in which when a packet is sent to an IP broadcast address from a machine on the local network, that packet is delivered to all machines on that network. As such, in this attack the attacker broadcasts packets with the spoofed source IP address targeted to the victim. Since the packets are sent at broadcast address, it is received by all the nodes within the network [9]. Each node responds back to the victim machine since the source IP address is spoofed as that of the victim's address. This creates a large amount of echo response packets thereby making the network unstable and causing a network congestion to the victim. However smurf attacks are not effective under IPv6 as when a node receives a packet in IPv6 with a link layer broadcast address it doesn't generate a response.

### *E. Ping of Death(PoD)*

It is a type of attack in which the attacker floods the victim system with malformed or malicious pings. The aim of the attacker is to destabilize or halt the victim's system or service using the malformed and oversized packets using just a simple ping command. The maximum packet length of an IP packet is 65,535 bytes. However, the Data Link Layer limits the maximum frame size to - for example 1500 bytes over an Ethernet network. As such, a large IP packet is split across multiple IP packets or fragments and the recipient host reassembles the IP fragments into the complete packet. In a Ping of Death the attacker manipulates the fragment content and the victim ends up with an IP packet which is larger than 65,535 bytes when reassembled. This causes memory buffers overflow allocated for the packet, leading to denial of service for legitimate packets. Although vulnerabilities leading to PoD are being patched in several systems, unpatched systems are still vulnerable to these attacks. Ping of death attacks are particularly effective because the attacker's identity can be easily spoofed and no detailed knowledge of the victim's machine is required.

### *F. HTTP Flood Attacks*

HTTP flood is a type of Distributed Denial of Service (DDoS) attack in which the attacker exploits the HTTP GET or POST requests to attack a web server or application. HTTP floods do not use malformed packets, spoofing or reflection techniques, and require less bandwidth than other attacks to bring down the targeted site or server. These attacks are also significantly harder to detect and block. An HTTP client like a web browser "talks" to an application or server by sending an HTTP request either of GET or POST type. A GET method is used to request a document from the server while a POST method is used to send some information from the client to the server or to access dynamically generated resources.

In incomplete HTTP Flood attack using the GET method, the client sends HTTP requests to the web server but in a different way. Client sends just a part of the HTTP header and never sends the complete header. Client continues to send subsequent headers at regular intervals to keep socket alive. By sending multiple incomplete requests the server's resources get exhausted. These requests consume all the available resources on the server, thereby denying the legitimate users' requests. HTTP GET-based attacks are simpler to create, and can be more effective in case of a large number of botnets.

Incomplete HTTP Flood attack using the POST method is the same as the GET method. The only difference is that in this case, client sends incomplete HTTP requests with the help of POST method instead of GET method. It forces the server or application to allocate the maximum resources possible in response to each single request. As such it is the most resource consuming.

## IV. IPTABLES

Iptables is a Linux kernel based packet filter firewall. The iptables modules are present in the kernel itself, there is no separate daemon for it. It is a rule based firewall system and it is normally pre-installed on a Unix operating system for controlling the incoming and outgoing packets. By-default the iptables runs without any rules and we can create, add, edit rules into it. Iptables is a very fast and effective firewall. The iptables rules control the incoming and outgoing traffic on a network device. It filters packets by the fields in IP, TCP, UDP, and ICMP packet headers. Iptables is very powerful and its features include:

1. Filtering - (rejecting unwanted traffic). You can filter incoming and outgoing traffic by user, group, time/date, or service (application).
2. NAT (Routing). If your computer has two or more network cards or in the case of virtualization you can use a spare computer as a router, one network card connected to the Internet and the other to your LAN with iptables monitoring and filtering traffic.
3. Logging (monitoring) network traffic.
4. Block brute force or DOS attacks

Iptables has 3 filtering points for the default table: INPUT, OUTPUT and FORWARD. These are called chains in iptables. As their names suggest, they specify whether a packets is destined for the system (INPUT), originating from it (OUTPUT) or is routed to another node in the network (FORWARD).

INPUT Chain:

INPUT Chain is for managing packets input to the server. Here we can add Rules to control INPUT connections from remote to the server.

FORWARD Chain:

To add Rules to manage packet connections from one network interface (NIC) to another on the same machine.

OUTPUT Chain:

 

The OUTPUT Chain control packets from the server to outside. Here we can add different rules to manage outbound connection from the server.

To insert a rule –i option is, and to append, -A option is used. We need to specify the chain, for which we wish to write the rule. The -j option specifies the target, i.e. what we want to do with the packet if a rule is matched. Some of the values are ACCEPT, DROP (or REJECT), RETURN etc. The default structure of iptables is :Tables —> Chains —> Rules. The rules are defined to control the packets for Input/Output. It is very important to understand the ordering of the rules. Iptables starts at the top of a chain, with the first rule, and proceeds down the chain until the FIRST instance of Drop, Reject, or Accept. The basic syntax of an Iptable rule is:

iptables -option [Chain] [Rule] -j [Target]

A Target is the action to be taken if there is a match to the rule, for example Drop, Accept, Log, Reject, or send the packet to another, possibly user defined chain.

## V. EXPERIMENTAL SETUP AND OBSERVATIONS

A suitable test environment was created for carrying out the attack and testing its attributes. Ubuntu 15.10 was used as the testing OS. An Apache server httpd version 2.4.20 was installed in one of the systems with a webpage hosted in the same. Any host connected to this network would be able to request and view this webpage. We further installed wireshark on the victim system for monitoring the network attack traffic. Different tools were used for carrying out the different Dos/DDos attack.

### *A. TCP SYN Flood Attack*

The attack was made by using the tool Hping. The victim's machine was flooded using the tool by running the following Hping command from attacker's system:

**# hping3 --flood –S –p 80 192.168.0.5**

**--flood** flag sends the packet at a fast rate

**-S** flag sets the SYN flag on in TCP mode

**-p** 80 sends the packet to port 80 on victim's machine

On victim machine, the following traffic was captured and analysed using Wireshark.

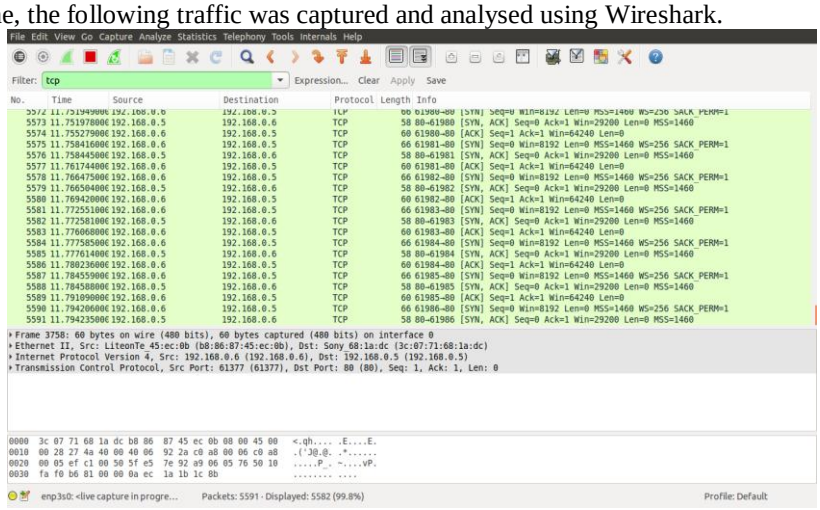


Figure 1.Wireshark Window showing captured packets

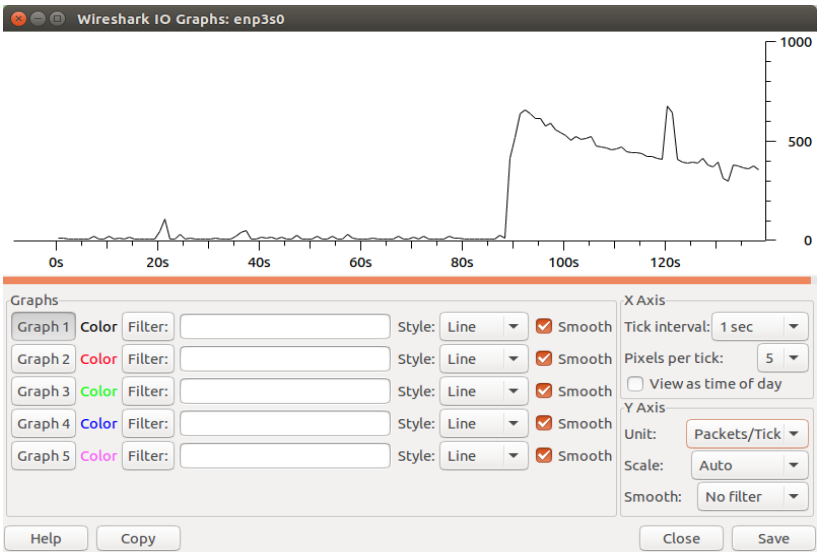


Figure 2. packets in SYN FLOOD

 

Fig 1 and Fig 2 show the number of packets in SYN Flood attack. Having a large number of open connections will exhaust the victim's resources in a short period of time. Fig 2 shows the rapid increase in the number of tcp packets being received by the victim system.

To mitigate the TCP SYN flood attack we use the following iptables script:

**# iptables -A INPUT -p tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 -j DROP**
**# iptables -A INPUT -p tcp -m state --state NEW -m recent --set -j ACCEPT**

These rules limit the rate of SYN requests from one IP to 20 per minute. Using the above rule we reduce the number of packets flooding the victim and obtain the following:

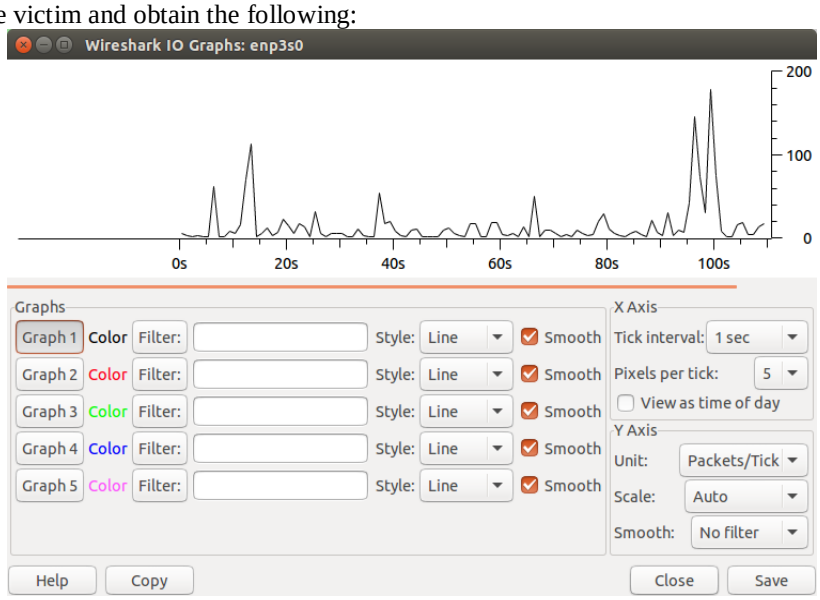


Figure 3. reduced packets in SYN FLOOD

As can be seen from figure 4 the number of packets have been reduced from 800 packets to as low as 50 packet/sec.

### *B. UDP Flood Attack*

The attack was made by flooding the victim's machine with udp packets using Hping tool with the following command:

# **hping3 –p 80 –i u1000 --udp 192.168.0.5**

**-p** 80 sends the packet to port 80 on victim's machine (192.168.0.5)
**-i** u1000 sets the interval between packets as 100 packets per second.
**--udp** flag sets the udp mode

On victim machine, the following traffic was captured and analysed using Wireshark.

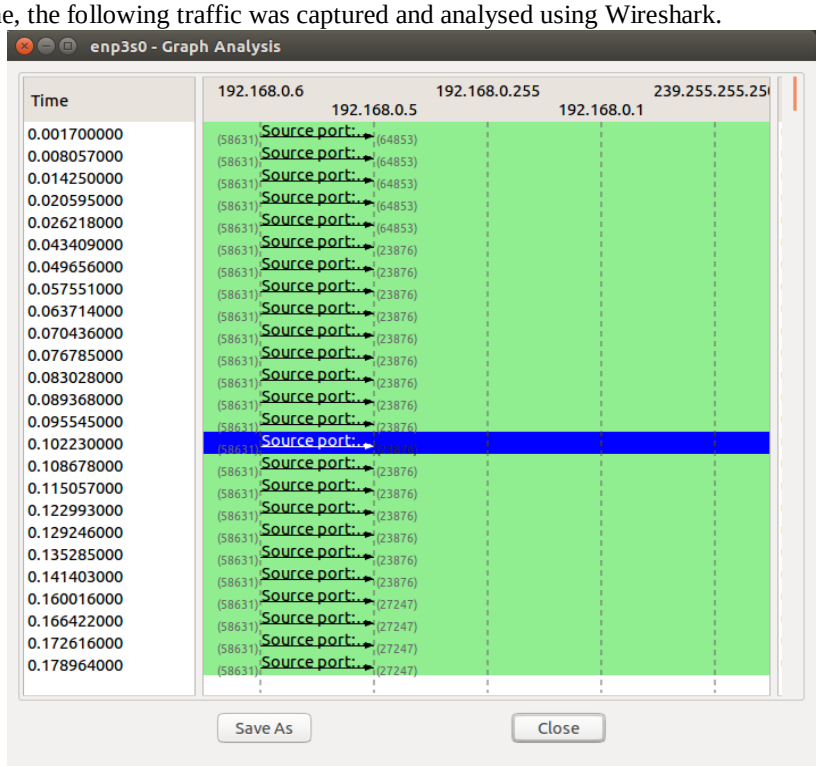


Figure 4. UDP Flood Attack

 

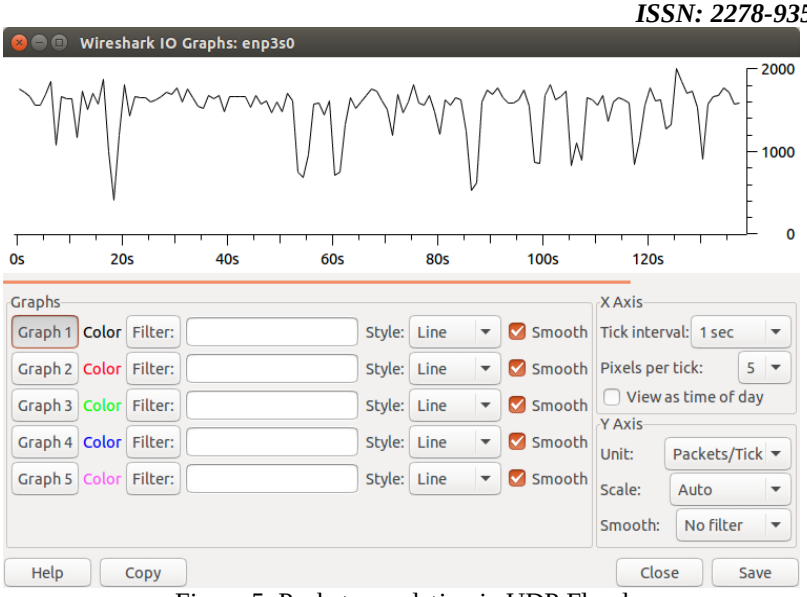


Figure 5. Packets escalation in UDP Flood

Figure 4 shows UDP flood attack where the source 192.168.0.6 is sending udp packets at a fast rate to the victim 192.168.0.5. Figure 5 shows the large number of udp packets received on the victim's machine.

To mitigate the UDP flood attack we use the following iptables script:

**# iptables -A INPUT -p udp -m udp --dport 20100:20500 -m state --state NEW -m recent --update --seconds 30 --hitcount 10 --name DEFAULT --rsource -j DROP**

**# iptables --A INPUT -p udp -m udp --dport 20100:20500 -m state --state NEW -m recent --set --name DEFAULT rsource**

Using the above rule we reduce the number of packets flooding the victim and obtain the following:

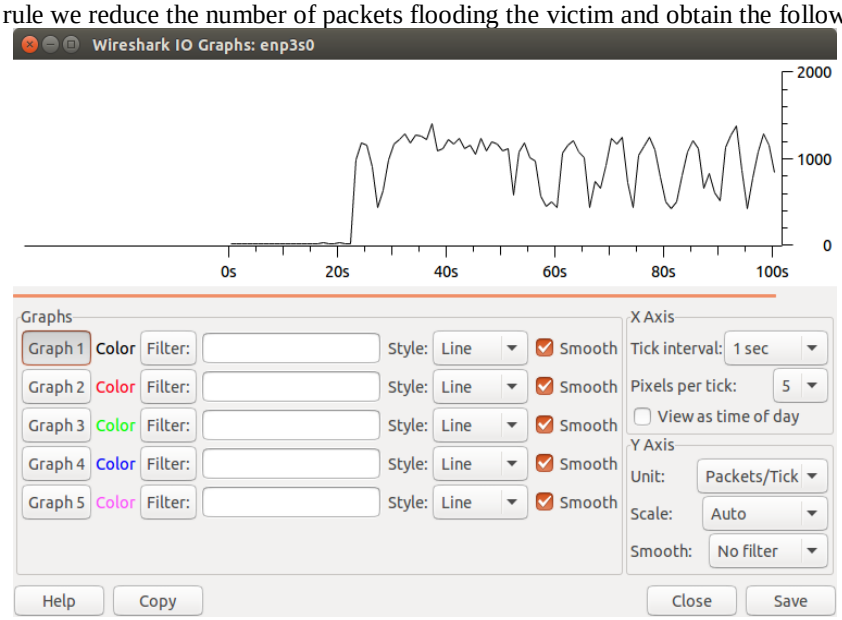


Figure 6. Reduced number of Packets

As can be seen in Figure 6, the number of udp flood packets have been reduced from 1700 to 900 packets/second.

### *C. ICMP(ping) Flood Attack*

The attack was made by flooding the victim's machine with ICMP echo packets. We use the tool hyenae. Running hyenae from command line on the attacker's system we use the following command:

# hyenae -I 1 -A 4 -a icmp-echo -s %-% -d %-192.168.0.5 -t 128

Description:

**-d %-192.168.0.5** is used to specify the destination ip address that is 192.168.0.5. % is used in the case where the mac address of victim machine is not known.

**-s %-%** is used for spoofing the source ip address and mac address

**-t 128** the time to live content of the packet which in this case is set to 128 ms.

**-I 1-A 4** indicates that we are attacking from the local machine.

**-a icmp-echo** indicates that we are sending icmp echo packets

Figure 7 and Figure 8 show the victim machine (192.168.0.5) under ICMP flood attack.

 

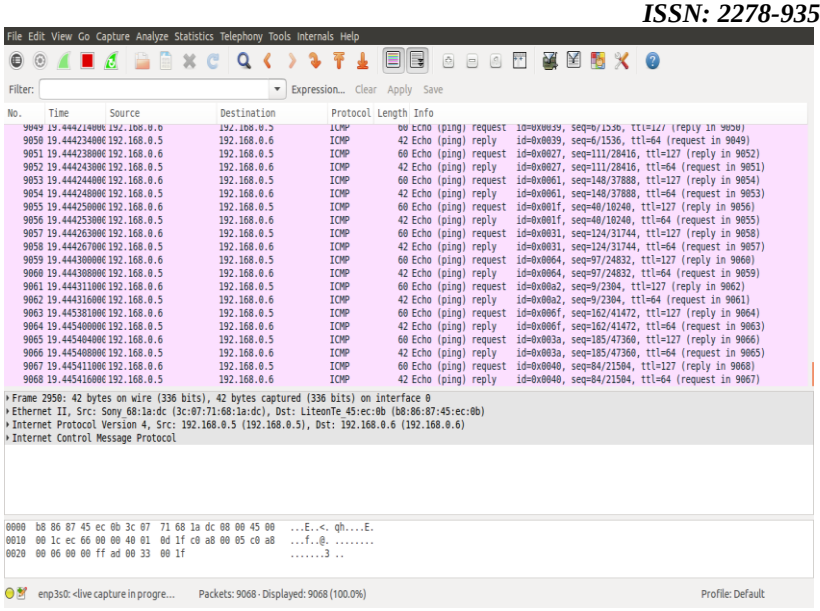


Figure 7. ICMP echo packets received on victim machine

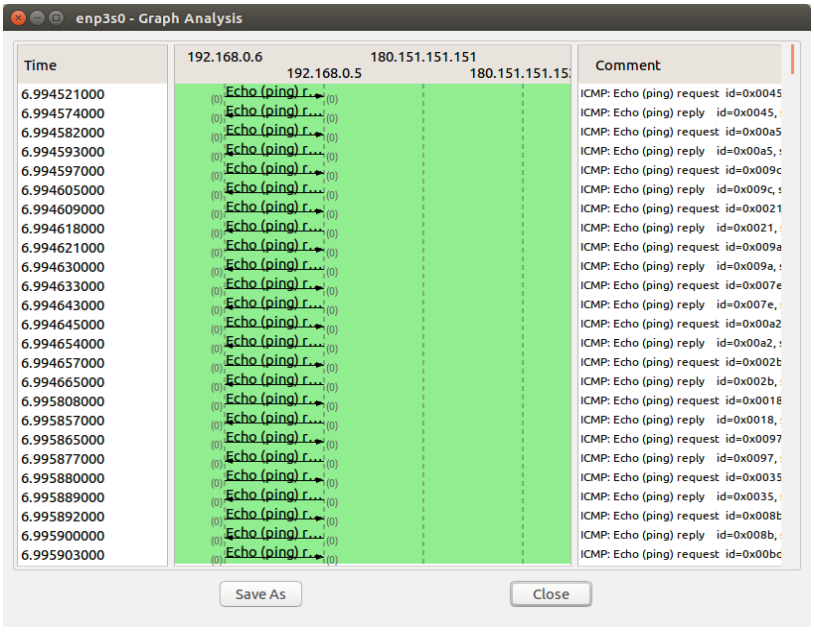


Figure 8. Echo packets from source to victim

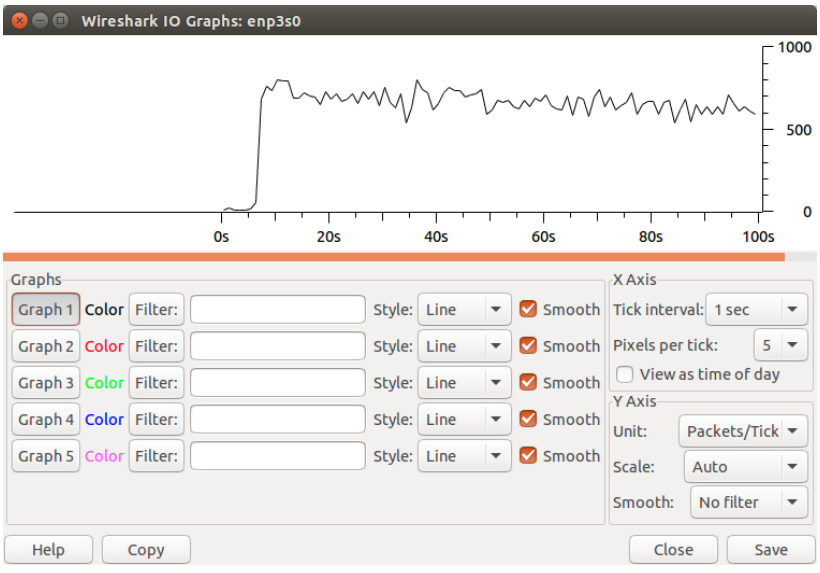


Figure 9. Number of packets received on victim machine

 

To mitigate the ICMP echo flood attack we use the following iptables script on the victim machine:

**# iptables -N icmp_flood**
**# iptables -A INPUT -p icmp -j icmp_flood**
**# iptables -A icmp_flood -m limit --limit 1/s --limit-burst 3 -j RETURN**
**# iptables -A icmp_flood -j DROP**

Using the above script we obtain the following result:

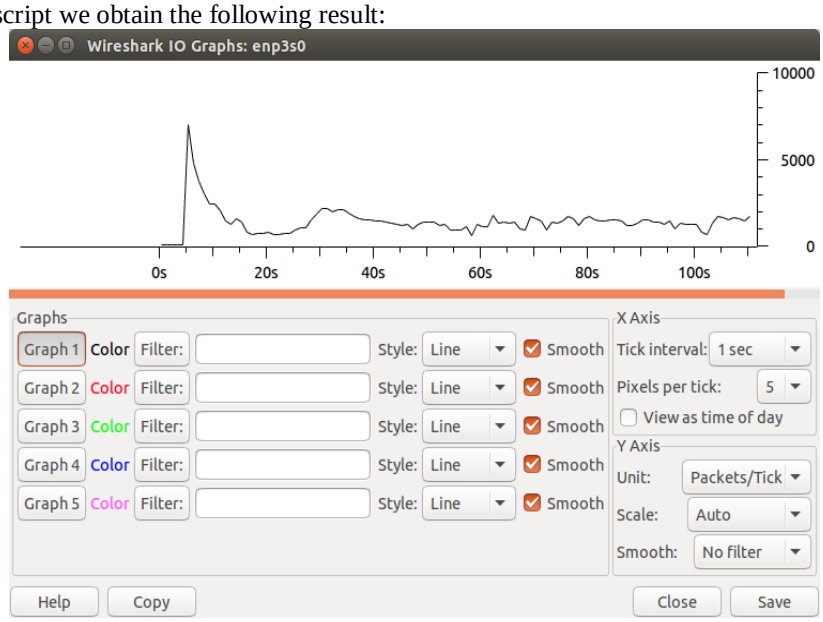


Figure 10. Reduced number of packets on victim machine

As can be seen from figure 9 and figure 10 the number of packets flooding the system have been reduced from over 600 packets/sec to less than 150 packets/seconds.

### *D. Smurf Attack*

The attack is carried out using the tool hyenae. The victim's MAC address and IP address is set as the source MAC address and source IP address for sending the requests. The destination MAC and IP addresses are set to the broadcast address so that the request is received by every host within the subnet. As a result, every host will respond back to victim which causes the unusual consumption of the victim's resources. Running hyenae from command line on the attacker's system we use the following command:

# hyenae -I 1 -A 4 -a icmp-echo -s 3c:07:71:68:1a:dc-192.168.0.8 -d ff:ff:ff:ff:ff:ff-255.255.255.255 -t 128

Description:

**-d ff:ff:ff:ff:ff:ff-255.255.255.255** is used to set the destination ip address to the broadcast address
**--s 3c:07:71:68:1a:dc-192.168.0.8** is used for specifying the victim's mac address and ip address as the source.
**-t 128** the time to live content of the packet which in this case is set to 128 ms.
**-I 1-A 4** indicates that we are attacking from the local machine.
**-a icmp-echo** indicates that we are sending icmp echo packets

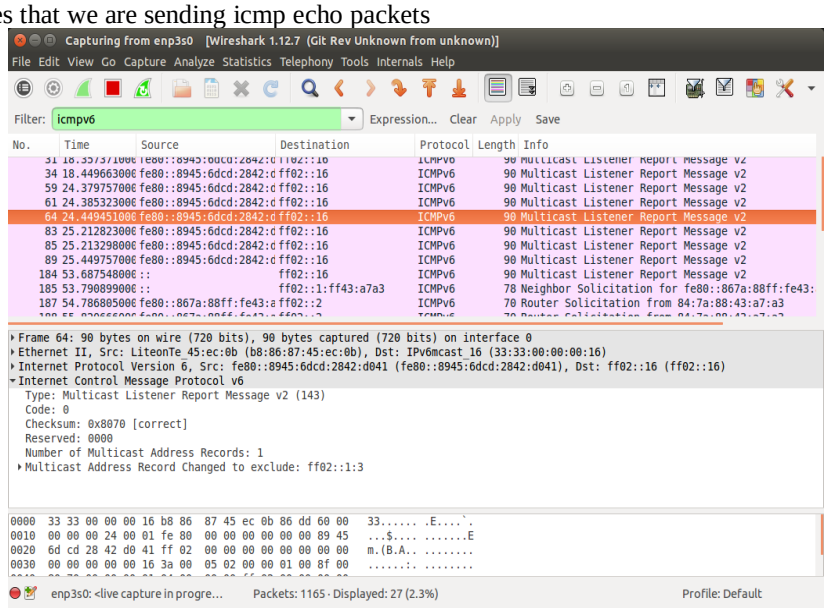


Figure 11. ICMP packets receive on victim machine

 

Figure 11 shows the flood of packets received from multiple hosts by the victim system.

The following iptable script is used to mitigate the attack:

**#iptables -A INPUT -p icmp -m icmp --icmp-type address-mask-request -j DROP**
**#iptables -A INPUT -p icmp -m icmp --icmp-type timestamp-request -j DROP**
**#iptables -A INPUT -p icmp --icmp-type echo-request -j DROP**
**#iptables -A OUTPUT -p icmp --icmp-type echo-reply -j DROP**

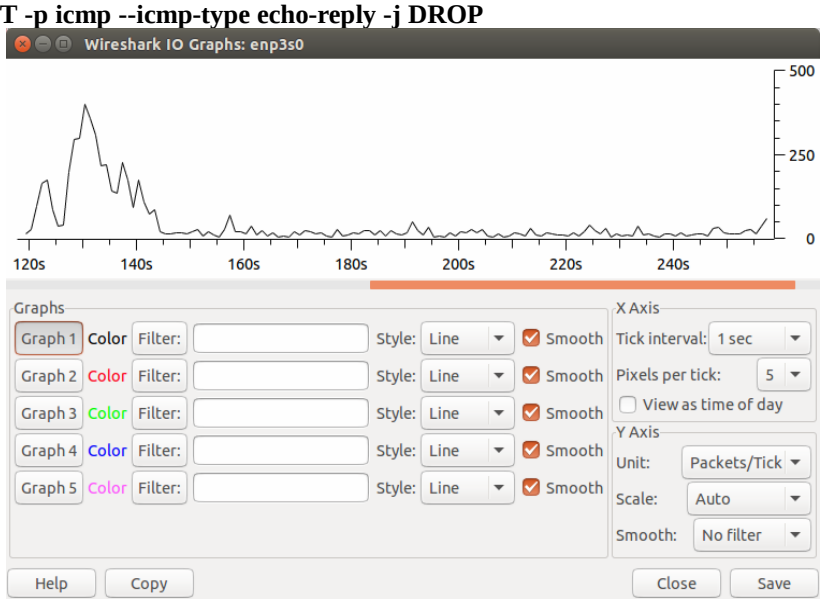


Figure 12. Reduced number of packets on victim machine

Figure 12 shows the decline in the number of packets received after applying the iptable script.

### *E. Incomplete HTTP requests using GET method*

The attack was carried out using the tool slowhttptest. By sending fragments of HTTP header using the GET method we drain the resources of the victim server (192.168.0.8). Running slowhttptest from command line on the attacker's system we use the following command:

# slowhttptest –C 1000 –B –I 110 –r 200 –S 8192 –t GET –u http://192.168.0.8 –x 10 –p 3

Description:

-C specifies the number of connections;
-B specifies the change rate in header;
-r is used for specifying the number of connections per second;
-S specifies the byte value of content length header;
-t specifies the HTTP flood method(POST or GET);
-u is the url of victim server
-x is the number of bytes sent
-p specifies the time to live of packet in seconds

Using the above command we capture the following traffic on the victim system:

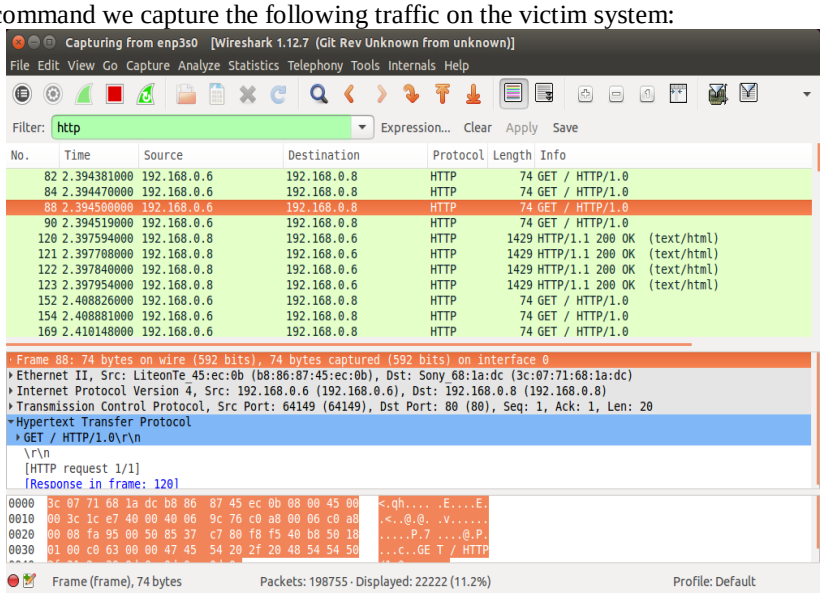


Figure 13. HTTP GET packet header captured

 

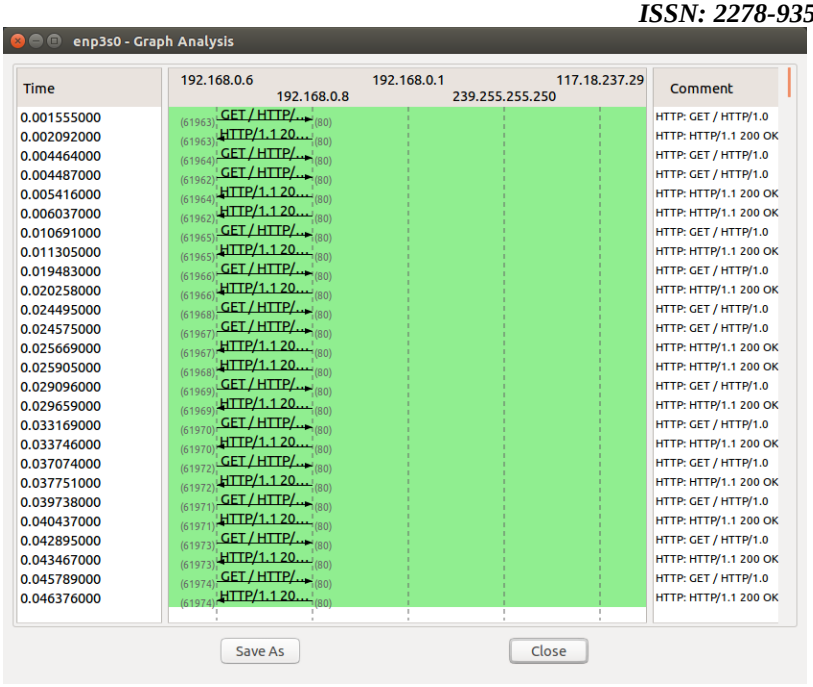


Figure 14. HTTP GET requests

As can be seen from figure 13 and figure 14 the attacker(192.168.0.6) is sending part of HTTP headers using GET method to the victim(192.168.0.8).

We use the following iptable script to stop the HTTP Flood attack:

**# iptables -A INPUT -p tcp --dport 80 -m limit --limit 100/sec --limit-burst 100 –j DROP**
**# iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m limit --limit 50/minute --limit-burst 200 -j ACCEPT**
**# iptables -A INPUT -m state --state RELATED,ESTABLISHED -m limit --limit 50/second --limit-burst 50 -j ACCEPT**

The above script makes the firewall limit the number of connections on the port 80(the default http port).

### *F. Incomplete HTTP requests using POST method*

This attack is similar to HTTP Flood attack using the GET method but instead of sending incomplete HTTP requests using the GET method we use the POST method. We using the same tool as in GET method. The following command was executed on the attacking system.

# slowhttptest –C 1000 –B –I 110 –r 200 –S 8192 –t POST –u http://192.168.0.8 –x 10 –p 3

The command description is the same as in HTTP flood GET method. The only difference is that in –t field we use POST keyword.

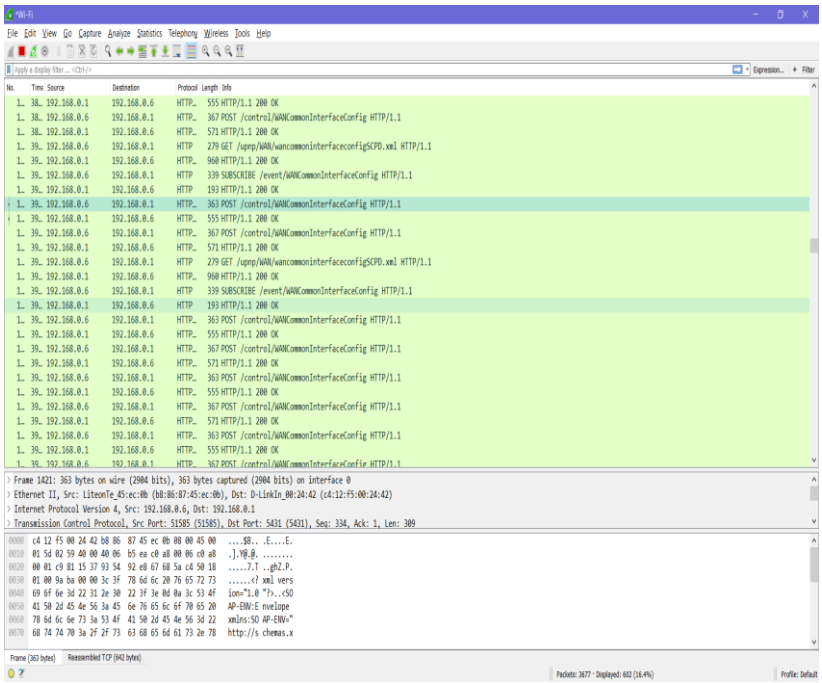

Figure 15. HTTP POST Flood requests

 

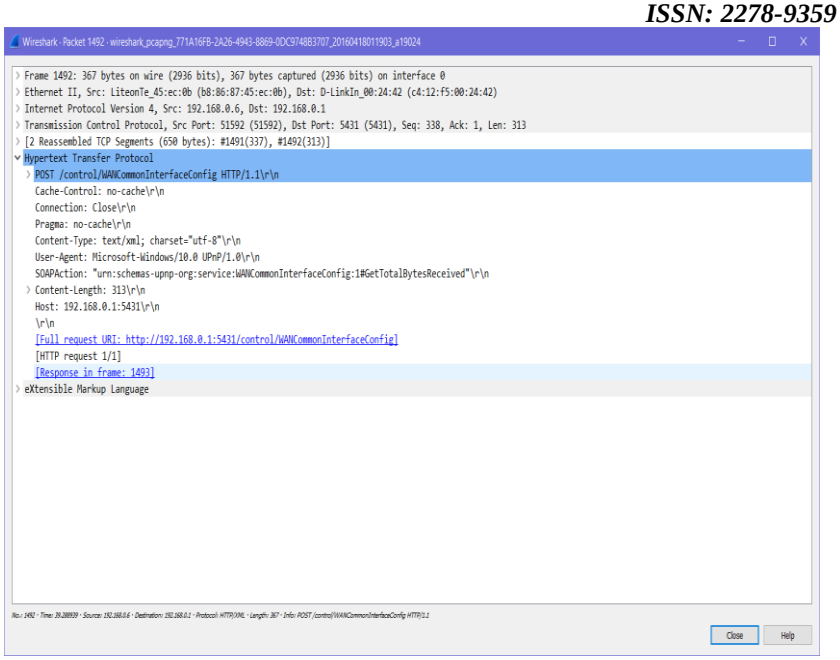

Figure 16. HTTP POST header content

Figure 15 and figure 16 show the incomplete HTTP POST header request being sent to the victim server thereby flooding it. The iptables script given below is used to mitigate the HTTP flood attack using POST method. The script limits the number of connections on destination port 80.

**# iptables -A INPUT -p tcp --dport 80 -m limit --limit 100/sec --limit-burst 100 –j DROP**
**# iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m limit --limit 50/minute --limit-burst 200 -j ACCEPT**
**# iptables -A INPUT -m state --state RELATED,ESTABLISHED -m limit --limit 50/second --limit-burst 50 -j ACCEPT**

## VI. CONCLUSIONS

Dos/DDos attacks are a major threat to today's internet. With little or no warning these attacks render a service or resource unavailable. As a result Mitigation of DoS/DDoS attacks is a crucial task and a part of an overall risk management strategy for an organization. To ensure business continuity it is necessary to identify these attacks and develop an effective strategy.

In this paper, we provided classification of DoS and DDoS attacks under TCP/IP. Along with this, we experimentally analyzed these attacks capturing their traffic generation with the help of the network analyser, wireshark. We provided information on some of the major Dos/DDos attack tools being used. Further we gave a countermeasure for these attacks and that is using iptables, and explored the capability of iptables rules in defending against these attacks. As can be seen in the paper that using iptables is simple and cost effective. However Dos/DDos flood attacks on a large scale cannot be fully mitigated using iptables. In a nutshell The basic aim of this paper has been to provide a foundation for possible DoS/DDos attacks in different scenarios along with an effective countermeasure.

## REFERENCES

[1] Yu, C.-F.; Gligor, V.D ,"A specification and verification method for preventing denial of service", IEEE Trans. Software Eng., Volume 16, Issue 6, pp. 581–592, June 1990.
[2] Dhwani Garg," DDOS Mitigation Techniques-A Survey", International Journal of Advances in Computer Networks and its Security.
[3] William Stallings', "Network Security Essentials".
[4] Mehmud Abliz,"Internet Denial of Service Attacks and Defense Mechanisms", University of Pittsburgh Technical Report, No. TR-11-178, March 2011.
[5] CERT/CC. 1997. Denial of service attacks.
[6] Hasantha Alahakoon," Distributed Denial of Service (DDOS) Defence Solutions".
[7] F. Lau, S. H. Rubin, M. H. Smith and L. Trajkovic, "Distributed Denial of Service Attacks" IEEE International Conference on Systems, Man, and Cybernetics, Nashville, 8-11 October 2000, pp. 2275-2280.
[8] https://www.incapsula.com/ddos/attack-glossary/ping-icmp-flood.html.
[9] Nikhil Tripathi, B.M. Mehtre,"DoS and DDoS Attacks: Impact, Analysis and Countermeasures".

 