School of Computer Science and Electronic Engineering, University of Essex

Denial of Service attacks and mitigation techniques: Real time implementation with detailed analysis

Name: Subramani rao Sridhar rao

subramanirao@yahoo.com

Supervisor: Dr. Martin Reed

ABSTRACT

Amongst various online attacks hampering IT security, Denial of Service (DoS) has the most devastating effects. It has also put tremendous pressure over the security experts lately, in bringing out effective defense solutions. These attacks could be implemented diversely with a variety of tools and codes. Since there is not a single solution for DoS, this attack has managed to prevail on internet for nearly a decade. Hence, it becomes indispensable to carry out these attacks in small test bed environments in order to understand them better. Unlike other theoretical studies, this project lays down the steps involved in implementing these attacks in real time networks. These real time attacks are measured and analyzed using network traffic monitors. In addition to that, this project also details various defense strategies that could be enabled on Cisco routers in order to mitigate these attacks. The detection and mitigation mechanisms designed here are effective for small network topologies and can also be extended to analogous large domains.

subramanirao@yahoo.com

TABLE OF CONTENTS

subramanirao@yahoo.com

subramanirao@yahoo.com

Acknowledgements and Foreword

I dedicate this work to my father who supported and encouraged me for doing this post graduate study.

Many thanks to Dr. Martin Reed for his extended support and valuable suggestions in bringing out this project work.

I would also like to thank my University for giving me such an environment to learn new technology and practice them with live implementation.

subramanirao@yahoo.com

List of Figures

subramanirao@yahoo.com

List of Tables

subramanirao@yahoo.com

Terminologies Used:

BOTNET	Group or network of compromised systems used for attacks
DNS	Domain Name Server (Described below)
MITM	Abbreviation of Man in the middle attack
ROOTKIT	A special piece of code that provides admin or addition privileges illegally
SPOOFING	Camouflaging one's real IP address with a fake one (Usually Victim's IP)
TCP	Transmission Control Protocol, Used for reliable connection oriented data transfers.
ZOMBIE	A compromised system which can be controlled by a remote attacker

Keywords:

Denial of Service, Distributed Denial of Service, Spoofing, Netflow, Access Control Lists, Rate Limiting, Hyenae packet generator, Netstat utility.

subramanirao@yahoo.com

CHAPTER 1 INTRODUCTION

Amongst various security threats that have evolved lately, Denial of service (DoS) attack is the most destructive according to the security experts. A Denial of Service attack is a method of blocking service from its intended users. The severity of this attack varies with the magnitude of loss and the duration of attack. DoS attacks could be extended to Distributed Denial of Service (DDoS) attacks which does damage in a massive scale. DDoS consists of many systems that work together to launch a much powerful attack [4].

1.1 Background Study

DDoS attacks are referred to as cat-and-mouse game according an IEEE paper published by Xianjun Geng and Andrew B. Whinston. This paper also emphasizes on the fact of having global exposure about this topic which is mandatory in stopping these attacks. According to the author, DoS attacks with single host are seldom successful in casting a massive damage. Mostly, attackers scan for vulnerable loop holes to add more hosts to their attacking army. These innocent hosts join the attacker and aid in strengthening the attack unwillingly and unknowingly [6]. These host computers are called the **'zombies'**. Thus when individual system owners become aware of this scenario and tighten their security from falling prey to the attacker's instructions, DoS attacks could be greatly controlled.

DoS against Domain Name Servers (DNS) could be even more disastrous as the entire Internet infrastructure is build on it. This topic is well highlighted in an IEEE paper published by Steven Cheung [7]. DNS servers are responsible for translating the website addresses into respective IP formats which is then directed to its destination. When corrupt packets are sent or when the DNS server is flooded, IP translations will not be successful thus stopping legitimate requests. Every DNS server in the internet backbone is fed with the IP addresses of root server. When a particular webpage is requested by a web page, the corresponding IP address is fetched by the DNS server and directs it to the corresponding root server. The root server then forwards this request to the specific server to which the IP belongs. Thus the effect of DoS attacks could be dreadful when it is targeted towards DNS servers.

Another IEEE paper presented by three authors brings out a mathematical expression that measures the DoS attack effects and the performance of servers under such attacks. This paper measures attacks with low traffic rate that are much easier to manipulate and deduce interesting calculations. The parameters chosen for calculation are the availability, client's success probability and the overhead. Overhead could be calculated from the below formula,

i.e., *Overhead = ratio of attack traffic and the maximum server's capacity* [8].

This paper also deduces an expression for the service time for each client which is denoted as (T_s). Service time rendered by a server is usually constant when clients pose identical requests. However, during the time of attack, traffic flow is not constant and thus the service time differs for each source. These variations are denoted by the expression Var. After applying suitable normal distribution theorems and rules, the final deduction is obtained as

$$\mathbf{T_s = N\ (\overline{Ts}\ , Var)}$$

subramanirao@yahoo.com

From the above equation, it could be noted that, by reducing the 'service queue time', chances of DoS could be minimized. Thus the forged or attack packets will not be able to acquire the service queue. Experts and router designers are working towards this process of standardizing the jitter or delay variation which could help in stopping these attacks.

1.2 Motivation

The increasing number of attacks and the effects of these happenings invoked my interest in this subject. This unresolved issue is actively present in the IT world for nearly a decade and there has never been an ultimate solution for this. The magnitude of damage caused by DDoS pushed me to learn more about this topic and urged me to do my contribution. These attacks have got businesses down, crippled the economy of a nation and even changed government. Experts also predict that the future wars are going to be with IP packets as missiles since they are capable of bringing down a nation.

The attack which was carried out in Burma had kept the nation out of internet for several months. The traffic sent were unstoppable ranging from 10 to 15 Gbps which was several folds more than what the nation's network could withstand [20]. The whole nation was devoid of internet and the ecommerce industry came to a standstill condition. Nevertheless, there are effective solutions that are suggested in order to survive these attacks even though complete removal is not possible. Allocating extra bandwidth, tracing back the attacker, identifying and stopping the fake packets are few of the general suggestions widespread amongst experts. But the exact solution varies with the severity of the attack and the value of data that the company is trying to protect. Thus the ultimate motivation arose with a desire of stopping these attacks that could lead to safe and secure IT world. This dissertation report gives a baseline to DDoS and helps a novice technician to understand the subject better.

1.3 Dissertation Structure

This project has carried out real time live attacks on an Apache web server making it unavailable for its intended users. The following report would detail various flavors of attacks used on the server. A suitable lan environment was setup for this purpose using Cisco 2800 series routers and switches. This report also brings in some interesting details like legitimate traffic, attack traffic, server response and recovery time etc. In the later part, the attack was stopped by configuring Access control lists and Rate limiting features in the Cisco devices. The possible methods of mitigation and providing service even under attack are discussed under the Chapter 6. The graphs and readings in this report obtained by testing network under different topologies could form a very good base for future investigations.

1.4 Objectives of the project

The objectives of this project are broken down into smaller categories to make it effective and achievable. The first goal would be to understand the subject in detail by taking into consideration the previous incidents and attacks that happened in the past. IEEE papers and related white papers were carefully chosen to understand this subject matter in depth. The second objective would be set up a test bed environment and choose tools that are required to carry out live implementation. Since carrying out an attack in a real time environment needs extra care when compared to simulation, a dedicated LAN environment was chosen. There are various tools and codes that are capable of causing Denial of Service. Such an attacking tool is selected

subramanirao@yahoo.com

here for this project work. This project also demands proper ways to measure the attack when it is actually occurring. Apart from these, this project also aims at suggesting possible solutions in mitigating this attack. These attacks should be able to save the victim and also provide access to legitimate clients who would require service during an attack. Summing up all, this project's objectives would be to investigate and learn subject in depth, implementing the attacks, identifying and measuring the attack and finally adopting counter measures to defend Denial of Service attack.

subramanirao@yahoo.com

CHAPTER 2 LITERATURE REVIEW

2.1 Security threats and its impacts

There are many security threats that pose serious challenge towards the progress of IT economy. Amongst many attacks like *Man in the Middle Attack, Session Hijacking, Cross site scripting, Spamming* etc, Denial of Service is considered to be the most deadly weapon. In the year 2009, there were several series of Distributed Denial of Service attacks that were carried out against the US information systems and South Korea IT databases. The attacks originated from several countries like Canada, Japan, Australia and China which made the attack so powerful. In other attacks, many government websites were brought down including the Federal trade commission and Department of Transport [1].

2.2 What is a Denial of Service?

Denial of Service is an attack which makes an information or data unavailable to its intended hosts. There are various methods to carry out this attack and the strategies are explained in the following section. The underlying aspect would be to choke victim's network and thus make it inaccessible by other client. However, there are also other ways of making service unavailable rather than just dumping it with abundant IP packets. The victim could also be attacked at various loopholes making it unstable which depends on the nature of the attack.

There are many types of attacks crafted specially for [1]

- Congesting network resources,
- Draining CPU memory,
- Reducing computing power,
- Exploiting timers,
- Poisoning domain name translations etc.

There are also attacks that could be carried out at application level, hindering the normal functioning of a service. There are attacks that are designed to crash a web browser, email application or even a media player. When a specific application is disrupted and when normal functioning is hindered, it is called the **Application level Denial of Service.**

As a worst case scenario, there are attacks that can cause permanent damage to a system. These kinds of attacks are called the **Permanent Denial of Service** or **Phlashing** [27]. Permanent Denial of Service attacks are mostly firmware based that aims at completely destroying the hardware. **Firmwares** are the inbuilt code or program that is embedded on every electronic system for its proper functioning. When an attacker is able to change the firmware and replace it with a defective or corrupt one, the hardware could no longer be used. These attacks could be directed towards networking components like routers, switches or bridges and thus bringing an entire routing table to collapse. A fault in a single router might lead to a huge outage if it does not have enough backups and rerouting. Often devices, who try to upgrade their firmware online without checking for the signature of a trusted source, fall prey for this attack.

subramanirao@yahoo.com

2.3 Denial of Service attack mechanisms:

Denial of service attacks are further classified into many categories according to the style with which it is implemented. The following paragraphs discuss few of the most well known categories.

2.3.1 Distributed Denial of Service:

Distributed Denial of service has the cohesive strength of many compromised systems working towards a single cause. The first stage of this attack is to build its platform with many host systems that can work under remote commands. The attacker group would first scan networks to hunt for vulnerable systems that are weak in security features. According to researchers there are millions of host machines that are vulnerable without secure patches and proper updates that often fall victims to these attackers. Once the scanning procedure is completed, attackers would bring these hosts into control using software exploitations like *buffer overflow, dangling pointers, code injection* etc [25]. Special root kits are also used in many cases that are installed in a host system to incur these software exploitations. After having sufficient hosts under control, attackers also create backdoors that allows special access that is used for future entry. The attackers also update the hosts and tighten its security so that another attacker does not use the same host. Any future entry would be done using the back entry that has been specially crafted.

2.3.2 Low-rate TCP targeted Denial of Service:

Unlike the Distributed Denial of Service, low-rate TCP targeted attacks does not employ numerous packets to flood the network. Instead, it exploits the working mechanism of TCP timers thus bringing the throughput of a system to almost zero. These low-rate attacks are crafted to generate packets only periodically in very minimal quantity. Thus the attacking packets can easily disguise with the legitimate packets and escape from the Anti-Dos traffic monitoring systems. The attacks carried out this way exploiting the TCP timers are coined with the term **'shrew attacks'**. It is also indispensible to understand the TCP working procedure before discussing this attack.

During congestion in TCP, the congestion window is gradually reduced until the network is clear. Thus during congestion the sender's rate is reduced which apparently reduces the potential throughput. The TCP waits for the *Retransmission Time out* (RTO) to expire after which the data is sent again. When the congestion is more, the RTO timer is doubled after which the packets are retransmitted. Thus during a low rate attack, when packets are lost, TCP enters RTO. When an attacker is able to calculate this RTO time and sends attacking packets to create packet collision and loss, the attacker can push the TCP into waiting state. Hence, there is no need for flooding the network with packets, but only send packets when the timer is about to expire and push it again into the RTO waiting time. This type of attack can effortlessly escape the traffic monitors due to its low traffic rate and is a serious challenge for the security experts [24].

2.3.3. Reflective Denial of Service:

Reflective attacks are those which employ intermediate hosts for their attacks. These reflector attacks are hard to trace back and thus a powerful weapon in the attacker's community.

subramanirao@yahoo.com

From the below figure, the difference between **Direct and the Reflective attacks** could be well understood. In the direct attack, the attacking system sends out packets directly to the victim but hides its original IP address. It adopts the IP address of some other host which is R in this below diagram. The victim would have its further correspondence with host R assuming it was the source. But in the reflective attacks, the attacker floods millions of reflectors with its source address spoofed with the victim's. Thus all the reflectors would reply victim flooding its bandwidth. [26] In a direct attack, mitigation is easier as just the attacker's network needs to block. Using proper trace back mechanisms, the attacker's network could be identified and stopped. But in the reflective attack as the reflectors are spread across various networks makes it is harder to mitigate. A classic example for reflective attack would be **'Smurf attack'** that sends ICMP ping packets towards the victim. The Smurf attack is almost similar to the ICMP ping of death attack which is detailed in the later sections of this report. The only difference between ping of death and Smurf is that the latter uses reflectors in its operation.

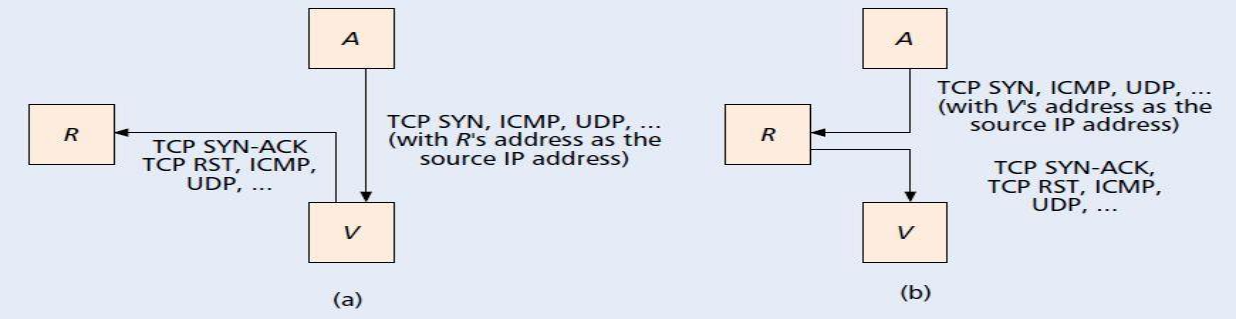

Figure 1: (a) Direct Denial of Service attack (b) Reflective Denial of Service attack [26]

2.4 Commercial importance of this attack

This project has got huge commercial importance in the market. Since it has been an unresolved issue in the market for nearly a decade, there are various researchers and security experts trying hard to find an end to it. There are many government agencies and organizations who recruit specialists who can save the nation's database from being attacked. Also due to the damage that could be caused using this attack, companies and e-commerce giants have special teams to mitigate this attack. There are also various small agencies and companies who provide anti-dos business on yearly contracts. Companies like VeriSign, Arbor networks and many others help companies in hosting their information securely without being crashed by the attackers [28] [29]. These companies promise in providing 99% uptime with immediate update message being sent to the network administrator whenever there is an anomaly. They also provide protection from DNS, HTTP and other web based attacks that leads to DDoS. Some companies also offer backup services for critical data so that the alternate server could be used when the original is compromised.

2.5 What does the attacker want?

There are several reasons why an attacker would like to cause DDoS. It could be a group of people who would like to bring down a specific webpage or website in order to keep it isolated from the business. Thus the company might lose all its online transactions and thus end up failing. Rivalry in business could be one main factor for these attacks. There have also been incidents where protestors show their dislike with an attack. This is often done when attackers

subramanirao@yahoo.com

target a government website and bring it down. Examples would definitely include the attacks carried out against the Georgian national, finance and the president's websites [21]. There have also been similar attacks on Iranian websites as a part of election protest [22]. Rioters chose these attacks against government and public websites as there is minimal chances of being caught. Another reason which aids to these attacks is the simplicity involved. Any beginner could perform this attack effortlessly without having much technical expertise. Attackers often post their attacking tools and scripts online to aid others who like to carry out similar operation. There are websites and forums that give out tools along with instruction manual that makes easier for anyone to carry out such attacks. People who carry out attacks without having actual knowledge about it are called the **'Script Kiddies'.**

There could also be other reasons where a group might not like the content published on a specific website and would like to bring it down. **Wiki leaks** are one such non-profit organization that brings out news from anonymous sources and whistle blowers. **Whistle blower** is a person who brings out illegal or unethical activities that occur in an organization, often in governmental bodies. This above mentioned website had oppositions all over the country and the access is totally banned. The Australian government has added this website to blacklisted sites and has stopped its access. Other countries like Iceland, Germany, China, Thailand, USA and many other countries have also banned this website. Mass Distributed Denial of Service attacks were carried out against this website and also organizations that supported Wiki leaks [23]. Apart from these possible reasons, there are also web criminals who conduct attacks for money. There are also many attackers who perform these attacks just to show that they are capable and boast about it in their hacker's community.

2.6 A Study of past DoS occurrences

There are many occurrences that have been happening since the last ten years and there is still no effective control for this attack. One of the most talked about attack happened in the year 2000, February 7 when yahoo servers were crashed. The famous internet site was unavailable for several hours which affected the business of yahoo considerably. Buy.com, e-bay and CNN were the other giant companies that were attacked, the very next day after yahoo. E-bay is an online bank that undertakes millions of transactions online. The site was completely inaccessible which incurred huge loss to the company. The downtime was calculated as three hours for yahoo and the other websites were down for several more hours. There were also other companies like ZDnet, Etrade and Excite that were bombarded with more than 1 gigabit per second of data which made the server isolated from legitimate requests [2].

Apart from using DoS attacks as a single weapon against the companies, attackers use it along with other destructive security vulnerabilities. One of the most interesting incidents took place in the year 2008, July 2 when Revolutionary Armed Forces of Colombia was attacked using *Denial of Service and Man in the Middle attacks (MITM)*. The attackers were later found to be **Colombian National Forces (CNF)** who carried out series of DoS and MITM attacks in order to bring out their hostages. **Revolutionary Armed Forces of Colombia** also called **FARC** had to release fifteen most crucial hostages including Betancourt Ingrid (a famous social leader), 3 American citizens and 11 other members. This is one of the most talked about incident in the history of security industry that freed several hostages without a single arm or ammunition being involved [41].

subramanirao@yahoo.com

CHAPTER 3 DENIAL OF SERVICE IN DETAIL

Denial of Service attack is generally carried out with large number of systems attacking a specific victim. Such an attacking network is called the **Botnet**. A Botnet is formed by thousands of slave systems usually termed as the **Zombies**. The attacking systems are often controlled and manipulated by a remote attacker who makes use of these compromised machines. Most of the times, the real owner of a compromised machine is not aware of the malicious activities.

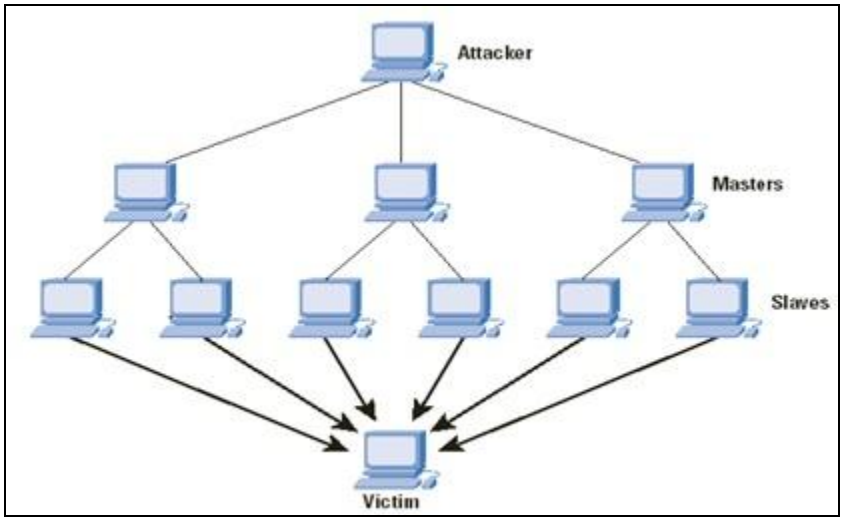

Figure 2: Master Slave architecture in a BOTNET

A broad classification of this attack involves two types [16]. First type is to attack the user and make him/her not use the service. The other type is to attack the server itself. The latter is more dangerous as it could affect millions of legitimate clients that could starve without getting served.

3.1 Types of attacks

There are several flavors of Denial of Service that could disrupt a normal service. The attacking methods are classified into two methods according to Erikson Jon [1].

- First type would be to flood the network not leaving enough bandwidth for the legitimate packets to get through. This could also be termed as Flooding.
- The other method is to crash a hardware or software item and make it inoperable. Web servers, routing devices, DNS look up servers are the common targets that could be crashed during an attack.

This project has investigated both the scenarios and has analyzed its effects. The DDoS paper published by Lee Garber talks about the mechanisms involved in some common attack types. Following are the most basic attacking methods employed so far [2].

subramanirao@yahoo.com

3.1.1 Smurf attack:

This attack works on the mechanism of flooding the victim's bandwidth. In this method, the attacker sends a large number of ICMP echo requests to a broadcast address. All the ICMP messages have spoofed source address as that of victim's IP address. Eventually all the reply messages target and flood the victim's address.

3.1.2 Ping Flood and Ping of Death:

Ping flood is similar to Smurf wherein the victim is bombarded with thousands of ping packets. In Ping of death, the victim is sent corrupt packets that could crash the system [3]. Smurf and ping floods are very easy to craft and any novice attacker could do it with ease. The following command in a Linux terminal could launch an attack [17].

Attacker# ./sing -echo -s 1024 -S ddos-1.example.com 192.168.81.255
Singing to 192.168.81.255 (192.168.81.255): 16 data bytes

There are enough effective defense mechanisms against Smurf and Ping attacks on the internet lately. However, these attacks could cause considerable damage in small Local Area Networks.

3.1.3 TCP SYN flood:

The above described methods works on consuming the bandwidth space whereas this attack aims at exploiting server CPU memory. Whenever a host attempts to connect to a server, a three way handshake protocol is established before any actual data transfer occurs. Firstly, the host sends a SYN packet to initiate the handshake. The server then replies with an Acknowledgement packet. At last the host again needs to send a SYN ACK packet to establish a successful connection. But attackers leave the handshake half open by not sending the last SYN ACK. Such a half open state is stored in the server's memory and the server keeps waiting for the host to send the final packet. When thousands of such half open connections are initiated, the server runs out of memory and crashes. It will not be able to serve the legitimate clients as its memory is dumped with forged fake packets [5].

3.1.4 UDP flood:

UDP flooding is similar to ping flood. Here instead of ping packets, UDP packets are bombarded against the server. UDP could be a lot more effective than ICMP in smaller networks as the size of the UDP packets are enormous. The packet size could be set up to 65000 bytes which could easily flood a given Ethernet network when multiple zombies are set up. This project has analyzed all the above described attacks and has brought down some interesting observations.

subramanirao@yahoo.com

CHAPTER 4 REAL TIME IMPLEMENTATION OF DoS

4.1 Environment Setup

A suitable test bed environment is created for carrying out the attack and measuring its attributes. This LAN environment is isolated from internet and other neighboring networks to avoid any accidental damage. An **Apache httpd 2.3.12-beta server** is 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. A tool called **Hyenae packet generator** is used in this project to create forged packets and dump the server. This tool is a platform independent network packet producer that can initiate DDoS, DoS and MITM attacks. This tool also has the ability to group zombies and trigger a remote attack [15].

Cisco Netflow Configuration for measuring Network traffic rate

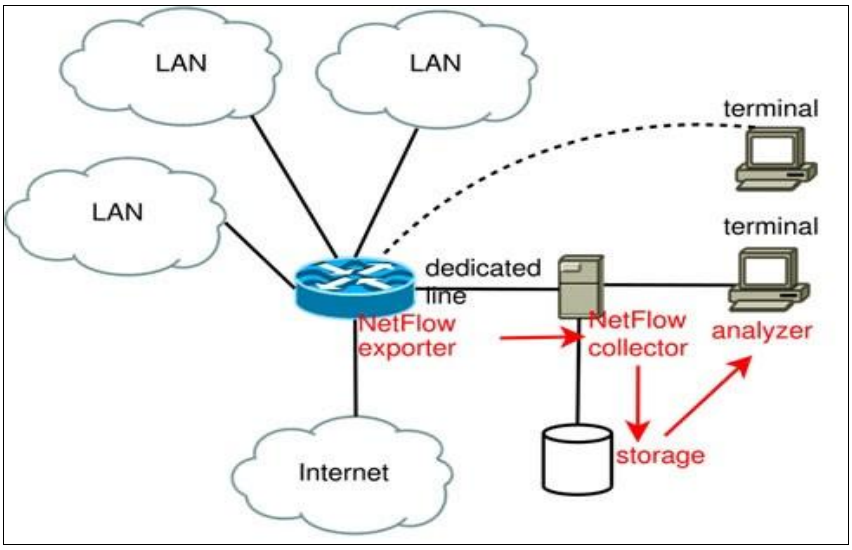

Figure 3: Cisco Netflow architecture to measure network traffic [19]

Netflow is a traffic monitoring protocol developed by Cisco in order to capture and analyze the traffic passing through Cisco devices. This protocol works not just with Cisco devices but also with other brands of network devices. Netflow is one of the essential components adapted by every company to monitor their network traffic. This software has also done some remarkable work in identifying and mitigating DDoS. *Internet Protocol Flow Information Export* (IPFIX) is a similar piece of software that was created by the IETF group for the same network monitoring purpose [18]. Netflow registers the details of the packets passing through a router in a log file which can be accessed from a remote system for analysis. It monitors traffic in accordance with various metrics like IP source address, destination address, Type of service, Source port, Destination port etc. This traffic information is gathered and is sent to the traffic collector which then forwards it to the analyzer. In this project analysis, Netflow is one of the software that collects traffic information to show the variation between normal and attacked scenario. The commands used to configure Netflow are listed in Appendix 1 towards the end of this report [9].

subramanirao@yahoo.com

4.2 Network Setup

The LAN environment is set up and tested for two different topologies. These topologies are individually tested under different attack scenarios and compared. The first one is for testing DoS directly against a server and the latter is against the network.

- Figure 4 given below is designed to make sure that the traffic is manipulated majorly by the server rather than overloading the network path. There are two switches and a router connected linearly that can withstand the fake traffic produced by the hyenae tool. This scenario exploits the server and can make it to crash when enormous amount of packets are released. During a SYN flood this network scenario would take the entire server's CPU memory as it cannot hold all the half open SYN connections eventually forcing the server to crash.

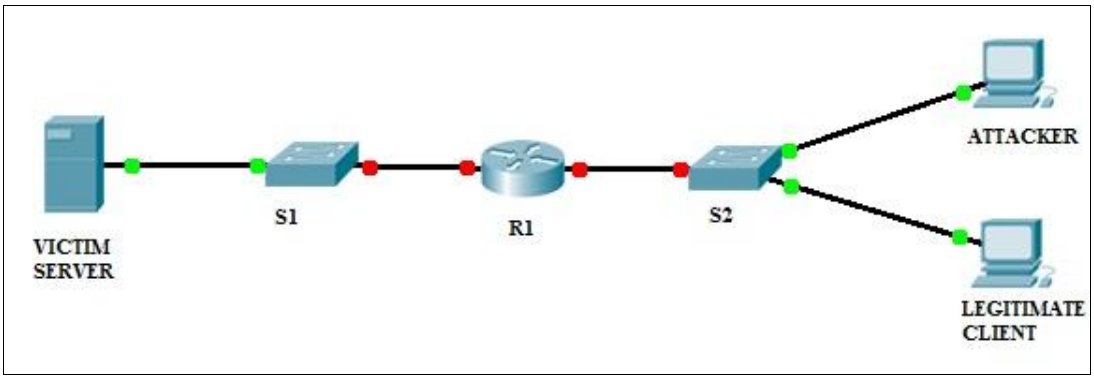

Figure 4: Network Scenario 1 with single router (Constraint on the server)

- Figure 5 given below represents another network scenario that is primarily focused in dumping the network rather than forcing the server to crash. The arrow mark in the figure shows where exactly the bottleneck could occur. All the fake packets that are generated would try and rush into the network to reach the destination server thereby giving very less probability for the legitimate packets to get through. Here the routers are the primary decision makers that needs to stop the forget packets from entering into the destination network rather than just forwarding packets.

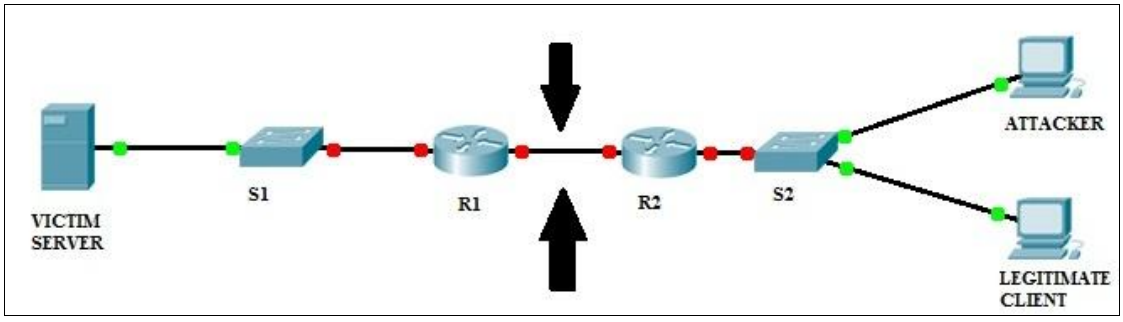

Figure 5: Network Scenario 2 with two routers (Constraint over the network)

subramanirao@yahoo.com

CHAPTER 5 ATTACK AND DETECTION PHASE

5.1 Attack Strategies

The attack from hyenae packet generator could be initiated from the Linux terminal. The following commands decide what kind of attack to be launched against the server. The characters and the keywords used for various attacks are first explained below.

'P' denotes that the packet size of 1000 bytes is to be generated by this command. The packet size can be adjusted according to the need but the highest size is chosen here to do the maximum damage.

The symbols **'S and D'** denote the source and destination addresses respectively.
'A 4' denotes the IP version 4. The attacks can also be performed with IP version 6 by changing 'A4' to 'A6'.

The **'percentage symbol'** in the command is used to aim at random ports when attacking. The attack could be made even more specific by changing it to port 80 instead of %%.

- This below command when entered on the terminal generates random UDP packets and targets against the server's IP address 192.168.1.2. We also need to provide the mac address of the server in order to launch this attack. In the following commands

  UDP flood → hyenae -I 1 -a udp -p 1000 -A 4 -s 00:02:B3:94:9E:DF-192.168.2.2@%% -d 00:10:A7:0F:2F:04-192.168.1.2@%%

- The following command is employed to launch of ICMP instead of UDP. It is exactly as same as the previous command but just differs with the destination address

  ICMP flooding → hyenae -I 1 -a icmp-echo -A 4 -s 00:02:B3:94:9E: DF -192.168.1.2 -d ff:ff:ff:ff:ff:ff-255.255.255.255

- This command invokes fake TCP connections and forces the server to hold all the half open SYN connections. The server's CPU memory tries and holds as much connection as possible, but eventually crashes out of memory.

  SYN flooding → hyenae -I 1 -a tcp -f s -A 4 -s 00:02:B3:94:9E:DF-192.168.2.2@%% -d 00:10:A7:0F:2F:04-192.168.1.2@%%

  The following table shows the list of parameters specified here in the Hyenae tool for carrying out each attack.

Attack	Payload Size	IP version	Source		Destination		Port
			MAC address	IP address	MAC address	IP address	
UDP	1000	IPV4	00:02:B3:94:9E:DF	192.168.2.2	00:10:A7:0F:2F:04	192.168.1.2	Random
ICMP		IPV4	00:02:B3:94:9E:DF	192.168.2.2	ff:ff:ff:ff:ff:ff	255.255.255.255	
TCP		IPV4	00:02:B3:94:9E:DF	192.168.2.2	00:10:A7:0F:2F:04	192.168.1.2	Random

subramanirao@yahoo.com

Table 1: Attack specifications used in the Hyenae packet generator

From Figure 6, we can observe that nearly five thousand traffic pattern packets have been exported to the Netflow collector. This information is always exported as UDP packets for many convenience reasons. We can see from Figure 5 that a total of 779 UDP datagrams have been exported. But there is always a danger of losing UDP packets as they are connectionless and does not guarantee Quality of Service. Hence Cisco came with the idea of **'flow sequence number'** that is attached to the packets to make sure that they are not lost during network congestion [10].

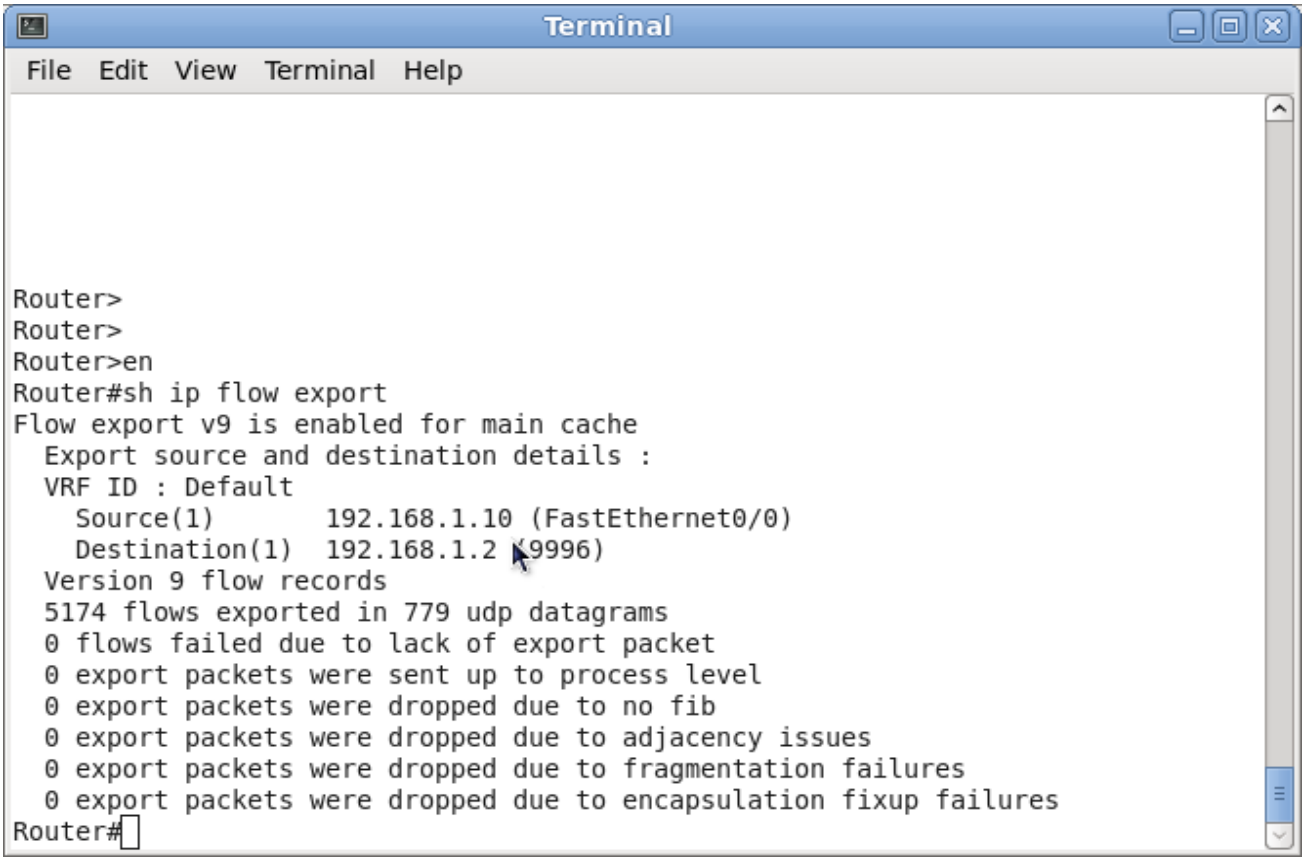

Figure 6: Screenshot of Netflow export during ICMP attack

5.2 Detecting DoS using Wireshark

This report highlights various feasible solutions for detecting Distributed Denial of Service. The following sections describe how detection could be done using various tools and commands. **Wireshark** or ethereal are the packet monitoring tools that capture traffic entering or exiting a specific port. This software is an open source analyzer that breaks down into finer details of the packets [11]. The best feature of this tool is that it can capture live traffic for analysis. The traffic pattern could also be stored for future detailed analysis.

subramanirao@yahoo.com

5.2.1 Analysis of UDP and ICMP traffic in Wireshark

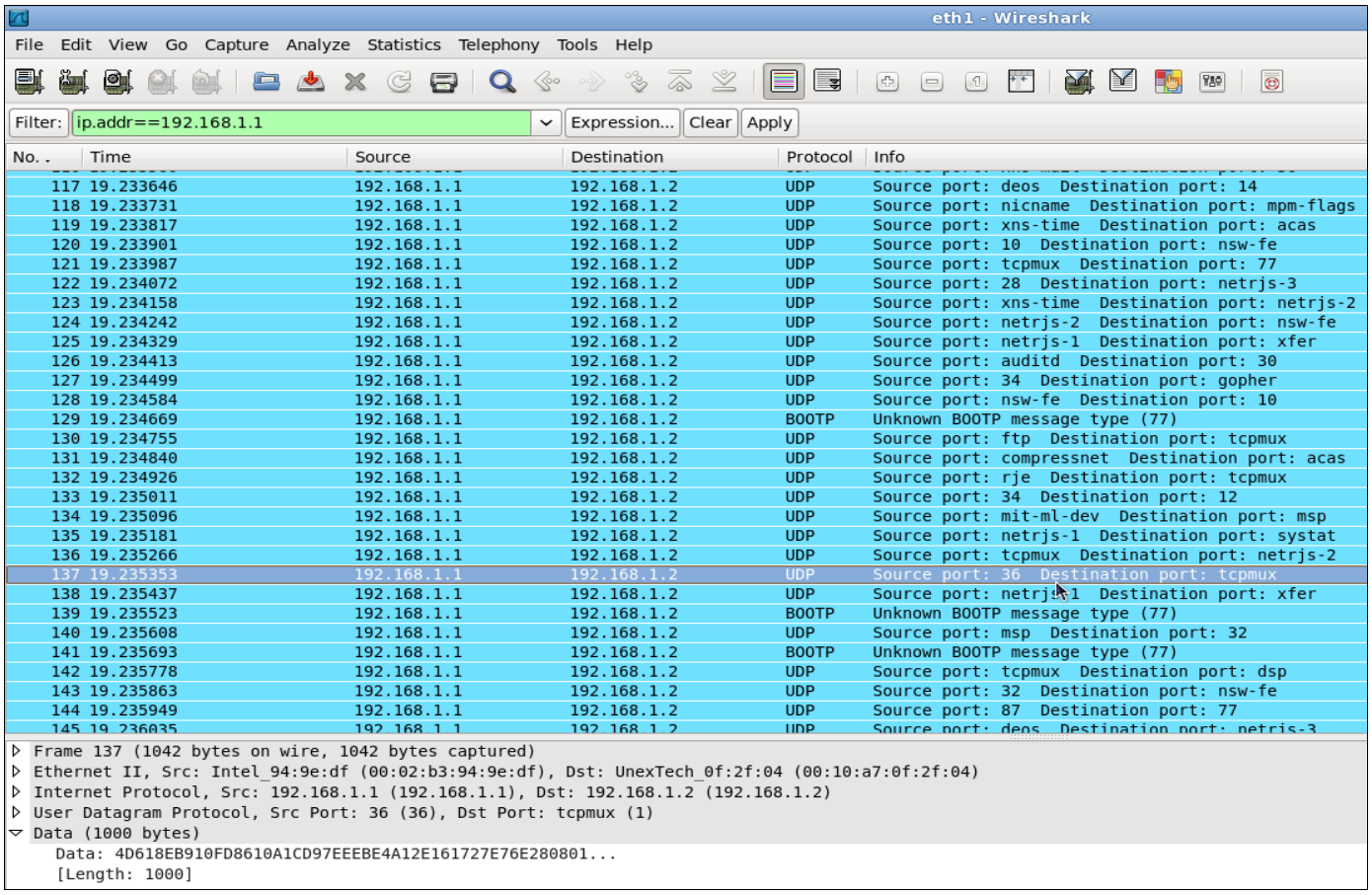

Figure 7: Screenshot of Wireshark during UDP flood attack

The above Figure 7 shows the UDP packets captured during an UDP flood attack against the server. These UDP packets are all forged with byte size of 1000. It could be seen clearly from the figure that all packets are destined towards the IP address 192.168.1.2 which is the server's IP address. The attacker is also from the same network thus making a direct impact over server's performance.

Figure 8 given below is obtained from wireshark which represents the graph for UDP traffic. Wireshark has various options and functionalities that can be used to draw many interesting graphs and deductions. This graph could be found under the 'Statistics' tab in wireshark. The following graph shows us when the attack has exactly started. The attack started at 12.945 seconds when it was triggered by the hyenae tool using the command line interface. The UDP traffic that was initially around zero kbps suddenly shoots up to 1400 kbps flooding the entire network. But a standard 10 Base T Ethernet cable has a capacity of 10 Mbps that can withstand this traffic. Ethernet cables with 100 Mbps and 1 Gbps are also available. According to the first network scenario as shown in figure 3, a standard Ethernet cable with 10 Mbps would able to withstand this attack traffic of 1400 kbps. But the same attack could be strengthened if a number of zombies attack at the same time towards the server. When such a cohesive attack was attempted the server crashed without being able to handle the continuous packets.

subramanirao@yahoo.com

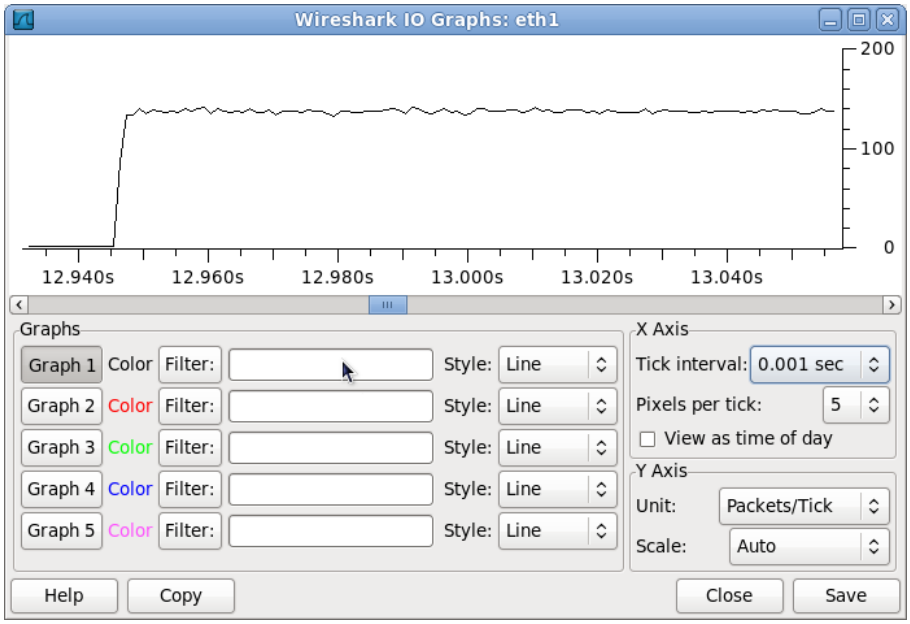

Figure 8: Network traffic rate monitored during an UDP packet

5.2.2 Analysis of TCP SYN packets

Wireshark has the ability to capture any kind of packet passing through a given port. When Ethernet port 1 was monitored during a TCP SYN flood attack, thousands of SYN requests were captured. The observation is much similar to Figure 7 except that we get SYN packets instead of UDP packets.

The time taken by a packet to reach the destination and return is called the **Round trip time (RTT)** [12]. This RTT is an important metric for establishing a TCP connection. When a packet exceeds its RTT, the packet is considered to be lost and thus it is retransmitted in a TCP connection. Since retransmissions aggregates Denial of Service, TCP SYN flood is often an attack used by the attackers. The TCP RTT graph could be obtained in the wireshark by filtering the IO graph with the key phrase *"tcp.analysis.ack_rtt"*.

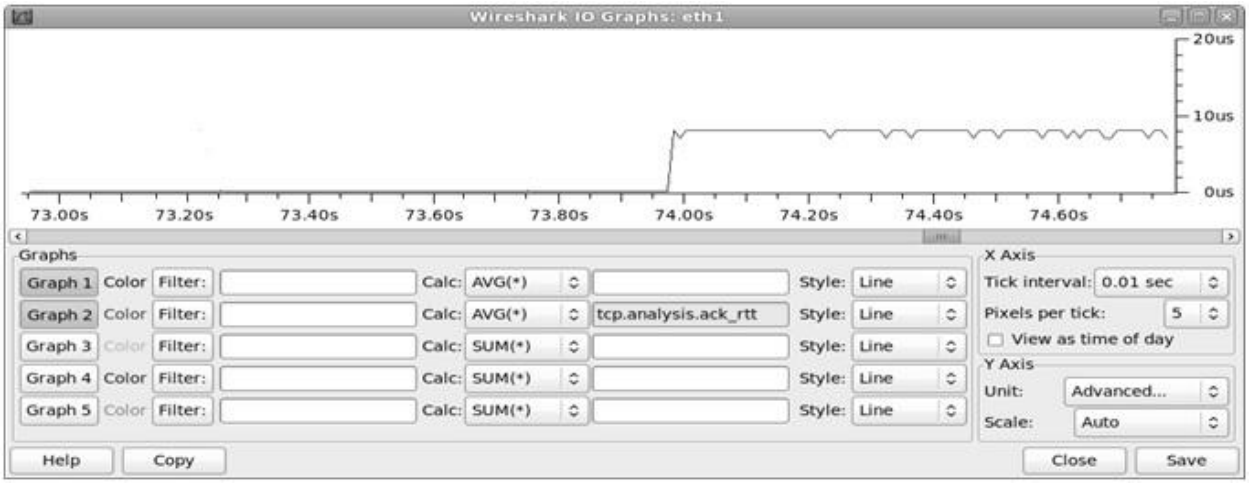

Figure 9: IO graph for RTT during TCP SYN flood attack

subramanirao@yahoo.com

From the above Figure 9, it is evident that when there is no attack, there is no TCP traffic. Thus the reading is close to zero when the attack is not live. When the attack is initiated, the RTT increases to up to 10 microseconds which stays almost constant till the end of the attack.

The TCP connections are always established with a three way handshake as explained earlier in this report. The handshakes are supposed to happen in accordance to the following flow diagram.

Client (SYN) →Server
Client ←(SYN, ACK) Server
Client (ACK) →Server

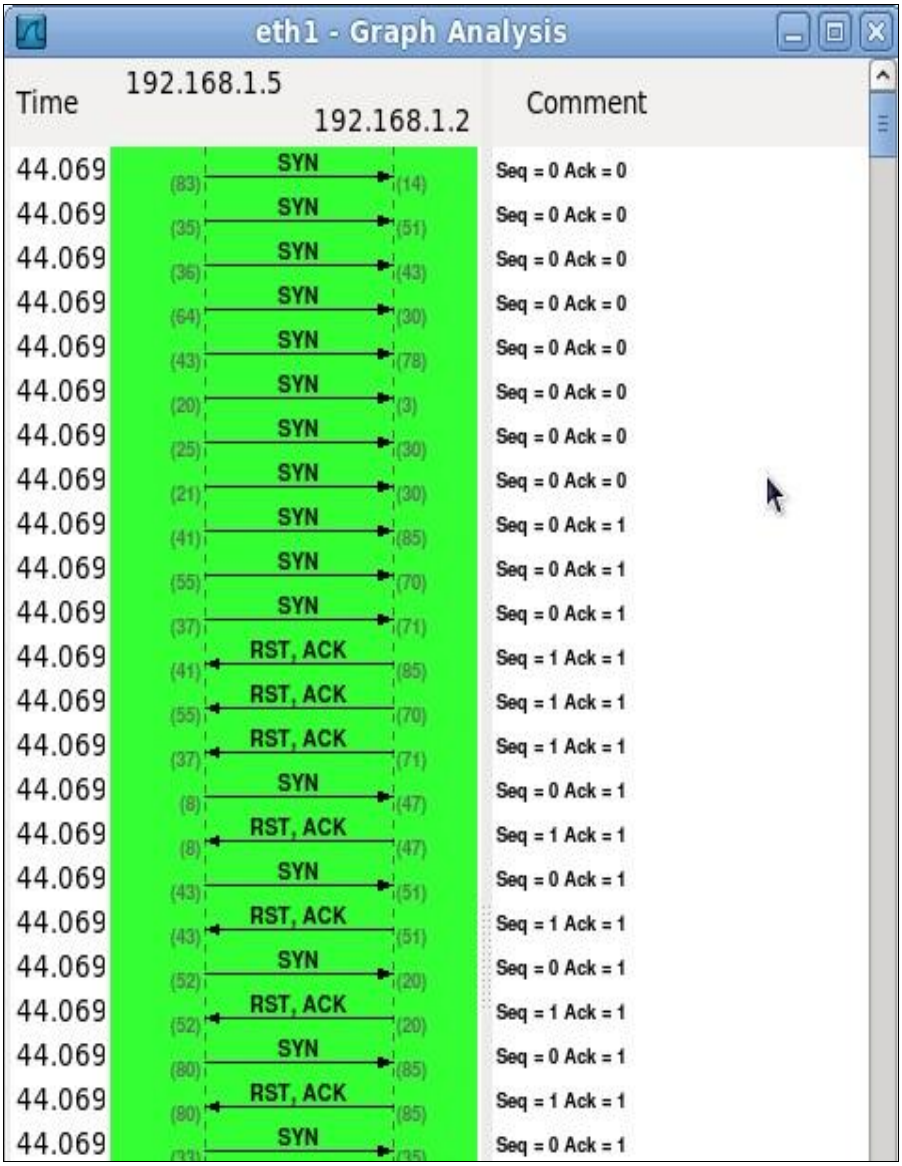

Figure 10: TCP SYN requests during attack

When the number of SYN requests increase beyond the server's threshold, it stumbles without establishing a proper connection. It is evident from the above Figure 10 that the host 192.168.1.5 is continuously sending forged **TCP SYN** packets with Sequence and Acknowledgement number 0. Since, the server 192.168.1.2 is not able to establish TCP

subramanirao@yahoo.com

connections for all the requests; it is forced to send the *'RST, ACK'* packet. **RST** is the Reset packet that is sent to a host requesting it to reset the connection request. However, when hundreds of packets are shot unceasingly towards the server, the RST packets are not very effective. It could be seen from the figure that there is continuous bombarding of packets during the 44.069th millisecond. The figure 10 shown is just a part of the actual reading obtained from Wireshark. However, there were more than several hundreds of packets that erupt in a single millisecond.

5.3 Detecting DoS using Netflow

The UDP exports from Netflow acts as a strong source for detecting traffic pattern. The exports could be sent every second to have the most recent activity of the network. The Netflow export gives us a finer detail of the 'type of traffic' flowing across a device. The Netflow collects data according to the *SNMP counters* that updates itself on a regular basis [13].

Netflow is capable of monitoring traffic on any direction, but in practice, today's potential network run Netflow only on the ingress direction for efficient output [14]. Netflow protocol could also be turned on for all the interfaces like how it is done in real time scenarios. But for this project traffic is exported only from the interface fast Ethernet f0/1 as the UDP export should not add to the existing flood. From this interface, a sampled stream of data is taken and the line card CPU calculates the flow pattern. The flow export is created by grouping packets with headers having similar fields. These similar fields are called the **flow key**. The traffic pattern is stored in a flow cache until the connection is terminated using a FIN or RST packet in TCP or until the timer expires. Then the flow information is exported as UDP datagrams to a Netflow collector that aggregates data.

During a combined UDP/TCP flood, the following graph was collected from the Netflow data analyzer that shows the traffic pattern. From figure 11, it could be seen that the UDP traffic occupies majority of the portion thereby doing the maximum damage to the server. The TCP connection occupies only quarter of the total traffic as the entire network is occupied by UDP datagrams of 1000 bytes each. It could be observed that when TCP and UDP attacks are carried out at the same time, UDP conquers the network path more than TCP.

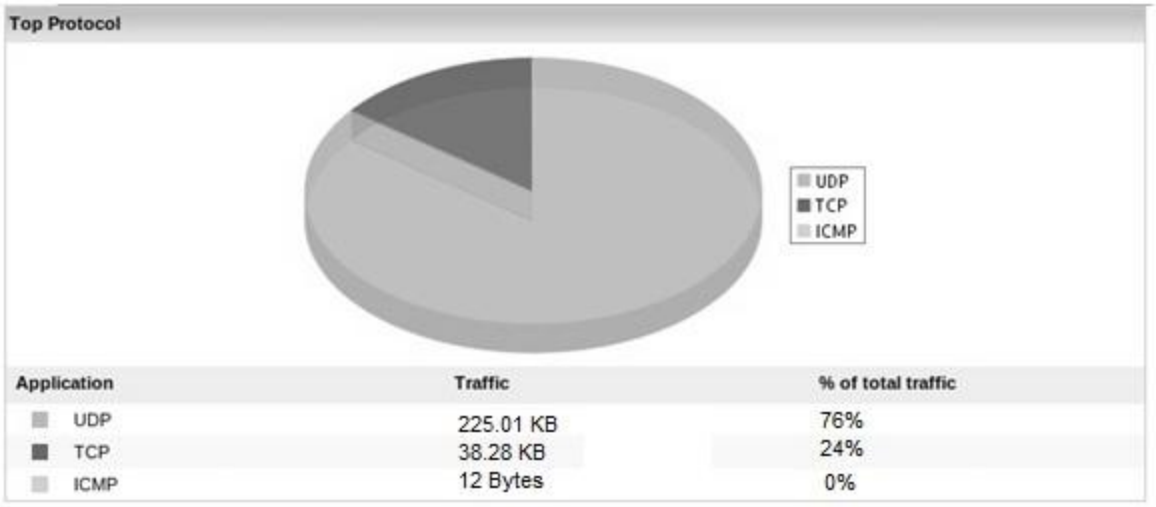

Figure 11: Netflow graph during combined UDP/TCP attack

subramanirao@yahoo.com

5.3.1 Determining the start and end time of an attack

The exact time when the attack starts is also available with the post processing of the UDP packets. From figure 12 of Netflow, it could be seen that the traffic pattern shoots up to 1.5 kbps at 13.10 and lasted until 13.20 hours. This small attack for ten minutes was carried out to check if Netflow can detect the duration of the attack. This Cisco software could be really handy to find the attacks as and when it is happening in a real time network. The attack is originated from the host with IP address 192.168.1.10 and that particular machine is also listed out here in this report. Thus, Netflow also helps in trace back techniques in finding out the origin of the attack.

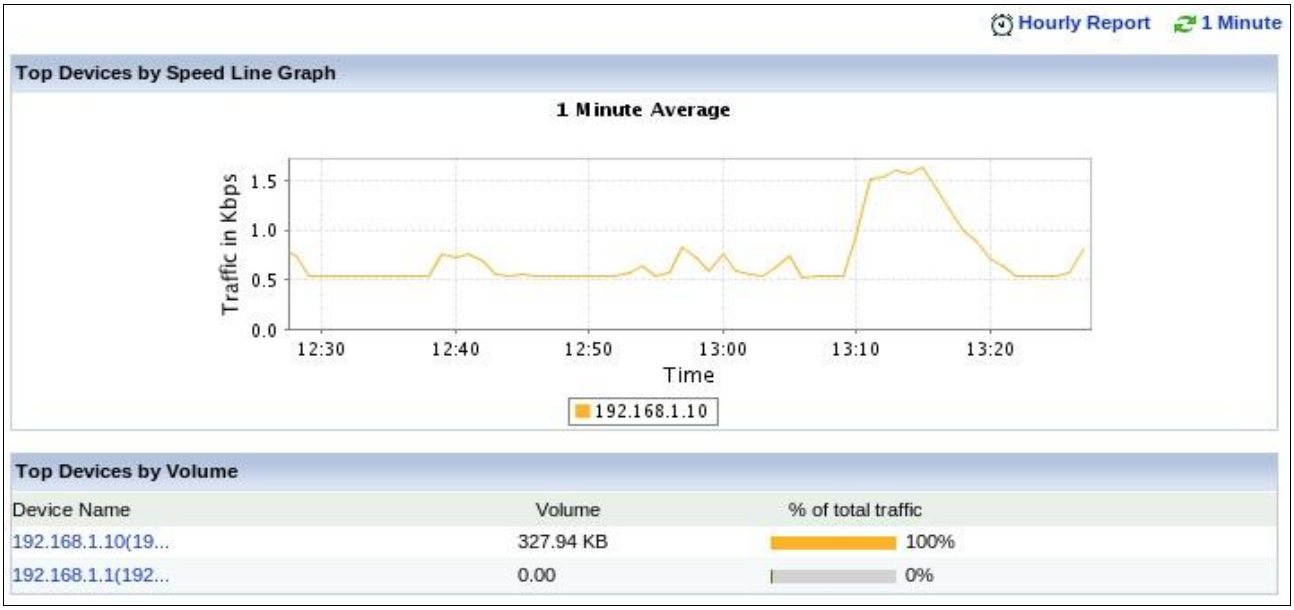

Figure 12: Capturing Start and End time of an attack using Netflow export

subramanirao@yahoo.com

CHAPTER 6 DEFENDING DENIAL OF SERVICE

6.1 Mitigating DoS using Access Control Lists (ACL)

Access Control Lists are the set of rules that are applied to a machine in order to control permissions. This project aims at applying Access Control lists on the Cisco routers in order to stop specific set of IP packets. Such a list provides protection to a network as it controls the traffic moving in and out of that point.

For instance, when an ACL is applied on a router, the incoming IP packets are checked if they satisfy the ACL table before entering. When a packet conforms to an existing rule present in a router, various options like deny, accept, reject etc could be performed. According to an IEEE paper by Alex, Eric and Chad, today's internet backbone is prone to millions of network vulnerabilities which demands more complex ACL rules [30]. Thus the ACL table grows in its size, degrading the network performance thus also making it harder to manage. This paper also comes out with an interesting idea called the **'ACL compressor'** which can reduce the size of the ACL tables and still follow the same semantics. This paper has also brought a remarkable outcome in its experimental results. The experimental results show that it can compress an ACL almost half its size when ACL compressor is used.

In this project, we use these following ACL rules in order to stop traffic from the attacking network. These rules are entered into the Router's command line interface so that it is applied to all the incoming traffic that passes this router.

Conf t
Access list 1 192.168.2.2
Interface f0/1
ip access-group 1 in

The first command ***'conf t'*** is used to configure the terminal of Cisco router. Then on the next line of the command line interface, the ***'Access list'*** is created. The Access list is also given a number and here it is 1. We can create many Access lists for a specific router and apply on any of its port. On the third line, command ***'interface f0/1'*** is used to enter that specific interface and make necessary changes. When the final command ***'ip access-group 1 in'*** is entered, the access-list 1 is invited to the interface f0/1 where all the ip packets from 192.168.2.2 are denied. It should also be noted that the ip packets from all the hosts of 192.168.2.0 are denied.

It could be seen that after applying the above mentioned ACL rules there is no traffic reaching the apache server from the attacking network. The attacking network 192.168.2.0 is completely blocked and is given no access for the webpage. This would form the basic mitigation step and thus complex ACL rules could be developed from this. It could be noted that the network 192.168.1.0 can still access hosts in 192.168.2.0 which is the attacking network. It is interesting to see the results from wireshark and other traffic monitoring tools how the traffic instantly stopped after applying ACL.

However there is one major loophole in this defense mechanism. None of the hosts from network 192.168.2.0 is able to reach the main database. In our test bed environment, the actual attacker is only 192.168.2.2, whereas the other hosts 192.168.2.3 and 192.168.2.4 are legitimate and need connection to the server. This ACL concept thus blocks service to a complete network

subramanirao@yahoo.com

rather than just blocking the attacker. If this is assumed to be a Distributed Denial of Service or Reflected Denial of Service attack, the ACL table would have to been long thus disconnecting various networks from the server. Apparently, the other legitimate clients existing in these networks would also be devoid of server connection. Hence this would not be an ideal solution and might need enhancement when the attackers are spread across various networks. The following suggestion of 'rate limiting traffic' seems better as it allows the legitimate hosts in the attacking network to connect to the server.

6.2 Mitigation using Rate limiting

Unlike Access Control Lists, rate limiting techniques does not separate the attacking network completely off the victim. Instead it places a cap or sets up a threshold limit of traffic that the server would be able to withstand. This method is adopted by most of the data providers as it proves to be extremely effective and saves the network components from permanent denial of service. However this cannot be an ideal solution as it still permits controlled traffic from attacking system as well. The following commands limit the traffic from the attacking network 192.168.2.0 to certain level. The best feature of this technique is that the network administrator is capable of deciding how much traffic to be let inside the network. This traffic rate depends on the size of the company, the traffic it can withstand and the server's processing capacity.

Cisco applies rate limiting to the traffic in the name of **Committed Access Rate (CAR)** and **Distributed Committed Access rate (DCAR)** [32]. The rate limiting rules could be applied to incoming or even on the outgoing traffic in a particular interface. The two common keywords used in the rate limiting commands are the *'conform action'* and *'exceed action'*. When an IP packet conforms or exceeds a specific rule, various decisions could be made on it. The decisions vary according to the network requirements like *'deny', 'allow', 'continue', 'drop'* etc. Once rate limiting rules are applied it could also be verified later using commands on the Cisco Command Line Interface. The command 'show int rate-limit' gives us the details of CAR applied to that corresponding interface. The following diagram from Cisco website shows us the basic structure of rate limiting commands.

As shown in the below figure 13, we can use *'rate limit'* or *'no rate limit'* to apply or remove this CAR policy from a specific interface. The *'input' or 'output'* keywords defines if the action needs to be performed on the incoming or outgoing traffic. When this rule has to apply to a pre configured access list, the keyword *'access group'* is used. Thus the policy is applied only to the packet that satisfies the access list. Every access list created in the Cisco router could be given a number for identification and the same can be used here in rate limiting using the command *'ACL index'*. The rate of traffic could be determined by the network administrator and could be specified along with this command in bits per second. *'Burst normal'* and the *'Burst maximum'* are the compulsory keywords specified to handle the traffic fluctuations and maintain steady flow of packets respectively. As discussed above, the *'action'* keyword decides how the packet has to be treated in that specific interface.

The below figure 13 is obtained from the Cisco website that describes the functionality of every parameter that could be used in a rate limiting command. Unlike other complex defense mechanisms, rate limiting is easier to manage and maintain. From the figure below, it could be seen that rate limit could be applied or removed from an interface using a single command. Rate

subramanirao@yahoo.com

limiting could be applied either on a single interface or could also be extended to multiple interfaces in complex network scenarios.

rate-limit {**input** | **output**} [**access-group** [**rate-limit**] ***acl-index***] ***bps burst-normal burst-max*** **conform-action** *action* **exceed-action** *action*

no rate-limit {**input** | **output**} [**access-group** [**rate-limit**] ***acl-index***] ***bps burst-normal burst-max*** **conform-action** *action* **exceed-action** *action*

Syntax Description

input	Applies this CAR traffic policy to packets received on this interface.
output	Applies this CAR traffic policy to packets sent on this interface.
access-group	(Optional) Applies this CAR traffic policy to the specified access list.
rate-limit	(Optional) The access list is a rate-limit access list.
acl-index	(Optional) Access list number.
bps	Average rate in bits per second. The value must be in increments of 8 kbps.
burst-normal	Normal burst size in bytes. The minimum value is bps divided by 2000.
burst-max	Excess burst size in bytes.
conform-action	Action to take on packets that conform to the rate limit.
action	Action to take on packets. Specify one of the following keywords: • **continue**—Evaluate the next **rate-limit** command. • **drop**—Drop the packet. • **set-prec-continue** *new-prec*—Set the IP precedence and evaluate the next **rate-limit** command. • **set-prec-transmit** *new-prec*—Set the IP precedence and transmit the packet. • **transmit**—Transmit the packet.
exceed-action	Action to take on packets that exceed the rate limit.

Figure 13: Cisco Rate-limiting [31]

Using the knowledge of above command structure specified by Cisco, the following commands were implemented in this project.

Conf t
int f0/1
rate-limit input 8000 2000 4000 conform-action transmit exceed-action drop

The first command brings the router to configuration mode which was also used when applying Access lists. Then entry is made into the interface f0/1 using command int f0/1 where the incoming traffic from various networks should be policed. Here the average traffic is defined as 8000 bps which is tolerable by the apache server used in this project. With this amount of traffic it will be able to serve all the clients connected to it. It was also recorded from the network monitors that the amount of traffic was around this range and did not cross more than 8500 bps even during attack. The burst maximum is defined as 4000 bytes which is a standard traffic with

subramanirao@yahoo.com

Ethernet topology. The network traffic obeying these conditions is transmitted and remaining is dropped.

These commands were successful in stopping the Denial of Service attack and was confirmed when the output was studied using wireshark and other network monitors. This seemed to be an effective solution in saving the bandwidth as well the apache server. In this project, the host 192.168.2.2 is the only attacker and other hosts from all the networks were designed harmless. However, the traffic is limited to all the networks including 192.168.3.0 which is legitimate and deserves full access. Hence in the following section, a promising and clever solution is suggested that proves effective in most of the small scale networks.

6.3 Combining Rate limit and Access Control features

Unlike the other two solutions, this method suggested here proves effective in stopping the bad traffic while giving full access to the legitimate traffic. Here the features of Access control lists and rate limiting are combined to form new set of protocols that governs the traffic flow better. The figure below best describes the operation of each scenario attempted in stopping Denial of Service. From the figure, it is clear that there are three networks 192.168.1.0, 192.168.2.0 and 192.168.3.0. Out of the hosts in all the networks, the only host that attacks the server is 192.168.2.1. This project examines three different mitigation techniques and correlates the result to find the best one.

The first method employed here to block the attack is using an ‘ACL rule’ in the Cisco router. This method is indeed the most effective as it stops the traffic from attacking network completely. However, it is to be understood that the other host 192.168.2.2 and 192.168.2.3 is legitimate and needs access to the server. Hence this cannot be an ideal solution and thus demands enhancements. For the given scenario with only three networks and fewer hosts, this technique could be satisfying. However, many networks and hosts will be kept out of server connection when the topology has a lot of sub-networks. Particularly, this idea fails miserably in the event of Distributed Denial of Service attacks as the zombies are spread across different networks.

From the second diagram in the figure it could be seen that on applying rate limiting values, a filter common to all the networks is applied at the routing entry terminal. This filter not only stops traffic from a specific network but also applied to all the three. All the three networks 192.168.1.0, 192.168.2.0 and 192.168.3.0 have the same treatment in entering the router’s interface. This rule could be applied when ‘IP trace back’ is not possible and the attacking network could not be found. Instances in the past like the Burma DDoS attack were not aware of the source of attacker and thus suffered heavy network damage [20]. This technique could thus save the network from being a victim of PDoS and also allowing limited access to the legitimate clients [27].

The third diagram could be seen as a hybrid of the earlier techniques giving a better defense mechanism. Here both the Access control and the limiting filters are used together giving the network an enhanced solution. Thus the legitimate networks 192.168.1.0 and 192.168.2.0 are given complete access. The network 192.168.2.0 which has the attacking machine in it is not given the complete access. However, it is given partial access in order to save the other legitimate clients that deserve the server connection.

subramanirao@yahoo.com

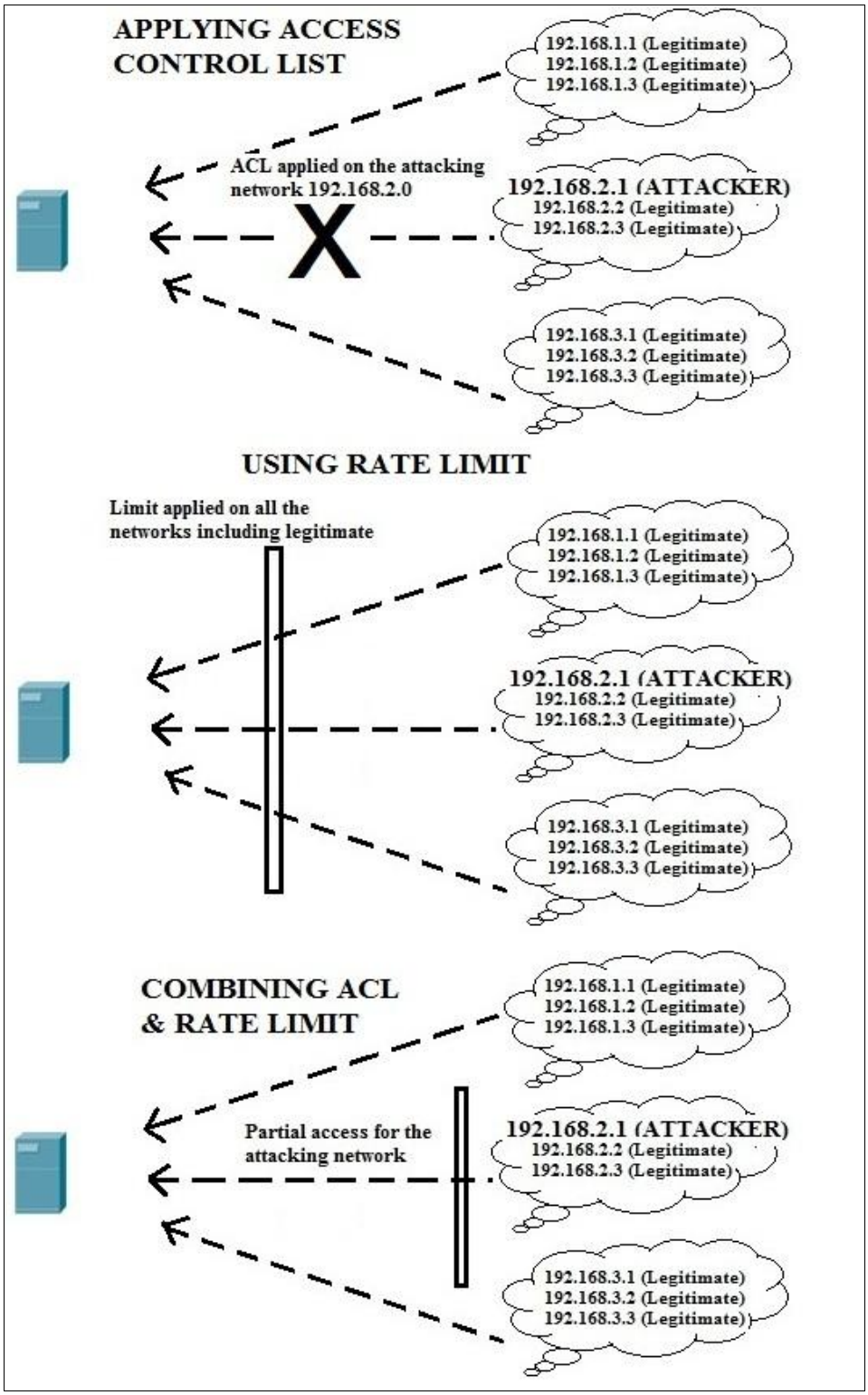

Figure 14: ACL, Rate limiting and Hybrid defense strategies

subramanirao@yahoo.com

The combination of these two strategies is done in two different styles and is described below.

Method 1

Conf t
access-list 1 permit 192.168.2.1
int f0/1
ip access-group 1 in
int f0/1
rate-limit input access-group 1 8000 2000 4000 conform-action transmit exceed-action drop

Usually, the attacking traffic is denied access to enter the router interface. On the contrary, this method uses the command 'permit 192.168.2.1' which is the attacker. Similar to the previous procedures, the access list is given an index number. It is then applied to the interface f0/1 by calling the 'access group' along with the corresponding ACL index number. After the attacking traffic is sent in, rate limiting command is used here to reduce its traffic rate to just 8000 bps. Thus the other two networks 192.168.1.0 and 192.168.3.0 get complete access while the attacking network gets partial. The other hosts 192.168.2.2 and 192.168.2.3 were able to reach the server and are able to obtain the webpage from it.

However when server response time was measured, it took longer for these networks to load the page compared with the hosts in the other networks. The following command was used to find the server's response time and it could be noted that the response time shoots up during an attack.

wget -p http://10.0.0.1

10.0.0.1 Here in the command denotes the IP address of the apache web server. This is a terminal command which is used in the Linux platform to check the server's performance. The same technique of combining ACL and rate limiting was also done in another method and is described below.

Method 2

Conf t
ip access-list extended Client2Server
permit ip host 192.168.2.1 host 10.0.0.1
class-map match-any Client2Server
match access-group name Client2Server
policy-map CAR
class Client2Server
police 8000 4000 2000 conform-action transmit exceed-action drop!
interface FastEthernet0/1
service-policy input CAR

This method employs two exclusive features of Cisco called the **Class map and Policy map** that proves very powerful in designing networks. With these features, the incoming traffic could be verified with a pre configured set of guidelines and corresponding decisions could be made.

Class maps are generally employed in segregating traffic flow information. This could be done using various parameters that define an IP packet. Packets could be distinguished for class

subramanirao@yahoo.com

map using parameters like source address, destination address, source port, destination port, protocol used etc. It can also be classified in accordance to the application layer information it holds. The parameters like http *URL, cookie, header length, content* etc could be used to class map the incoming packets [33]. Policy maps are the ones which correspond to a specific class map and perform necessary action on it.

In the above commands, the first command '*conf t'* is to bring the router into configuration mode. In the next line, instead of creating an access list with index number, an extended access list is created with a name (*client2server*) assigned to it. Every class map classifies or filters traffic using specific parameters. Here the traffic flowing from 192.168.2.1 (attacker) to 10.0.0.1 (server) is separated and mapped to client2server. In the fourth line the command *'class map match any client2server'* helps in mapping the related traffic flowing from the attacker to the victim. In the event of Distributed Denial of Service, many class maps can thus be created and appropriate policy maps could be constructed to take necessary actions. The next command *'match access group name'* picks out the necessary class map out of the pool to apply further action on it. The command *'policy-map CAR'* is entered in the sixth line of Command Line Interface which makes the router to apply rate limit on the class map traffic. The keyword **CAR** stands for Committed Access Rate which is responsible for limiting traffic. In the following line which issues the command '*police*' limits the traffic to 8000 bits per second. Whereas in the second scenario, traffic was controlled using the keyword *'rate limit'*. The final command *'service policy input CAR'* applies the policy rule (rate limiting) on the interface f0/1.

Thus the third mechanism is applied here in this project using two different techniques. The latter one is advanced and could be used in the complex network scenarios. The class map and policy map strategies are efficient procedures in withstanding the Distributed Denial of Service. However, for smaller networks, the first method holds good as it does exactly the same as latter.

6.4 Automatic command insertion using SSH

The commands used above could also be inserted into the Cisco router without manually updating it. This automatic command insertion technique could be done into the router which makes a network stronger and secure. For this to happen, the incoming and outgoing traffic rate should be continuously monitored and register every anomaly. A code is specially developed in using ***shell scripting*** that could insert commands in a router automatically, when the traffic crosses certain defined threshold limit. The shell program in given in *Appendix 2*, inserts the ACL automatically into whereas the program listed out in *Appendix 3* inserts rate limiting parameters.

6.4.1 'Expect' – A command line tool for interactive applications

Expect is a command line tool that is developed for the UNIX platform and could be used with many applications. There are many powerful tools developed exclusively for UNIX platforms but expect is the tool that can communicate with live applications [42]. These expect scripts are constructed in such a way that interactive systems could be handled without any actual interaction from the user. It makes easier for the network administrators to handle enormous requests or queries. The '***expect script***' is constructed with predefined set of solutions that will be fed when an interactive application prompts and expects a reply. A simple expect script could be constructed for password logins and the script would work only when the correct

subramanirao@yahoo.com

password is supplied. Rather than using only for passwords, expect could be used for a variety of arguments.

6.4.2 Program Structure

Dr. Martin Reed developed an automated script using 'expect tool' that can defend Distributed Denial of Service without human intervention. This code is given in the Appendix 2, which uses **Secure Shell** concept to send commands to the router. Secure shell is similar to telnet that creates a communication platform between two network components. Secure Shell is considered safer as it encrypts the data before sending it to the receiver.

The code developed by Dr. Reed consists of 8 blocks of 'expect commands' followed by a logical loop [43]. Each block expects certain predefined parameters to be passed and which are already stored in the program. Every block of 'expect code' is explained in the below table.

Expect blocks used in the shell coding [43]:

expect –c	'c' stands for command and this block expects a command to be followed. The spawn SSH command is then fed into the program and the system initiates a SSH connection with the Router.
expect word	The router's enable password is sent here in this block
expect *	The command 'enable' is sent
expect *word	In this block, the console password is sent via SSH
expect$prompt	Here enable command is passed to enter the router's privileged mode
expect*#	The command 'configure terminal' is sent to the router to enter configuration mode
expect*\(config\)#	The actual ACL command 'access-list 1 deny' is inserted into the router
Expect *\(config-if\)#	The router is brought back to global mode by sending the command 'Logout'

Table 2: Expect command structure used in the shell program

Logical loop design implemented in the shell coding:

Following this expect structure, a logical loop is constructed that counts the number of times an IP address repeats itself. The code is set in such a way that any IP address repeating itself for more than 10 times will be added to the Access Control List and will be denied access to the server [43]. This code works without any human intervention and can stop Distributed Denial of Service by its own.

A similar code is also developed that can insert rate limiting commands into the Cisco router using the same SSH concept. This code proves effective in having a constant traffic rate towards the server. The second code is given in Appendix 3 and works in an automated manner similar to the former code.

subramanirao@yahoo.com

CHAPTER 7 OBSERVATIONS AND RESULTS

7.1 Performance measurement of the Apache server

There are a variety of tools that are employed here in this research to measure the performance of networks accurately. **Bandwidth monitor** and **IPtraf** are few of the tools used here in this project. The tools need to be varied according to the operating system in which the testing is done. There are also exclusive Linux commands that could be very handy in getting the results. The performance here is determined on the basis of network traffic, server's response time, content download time from the website, round trip time etc.

7.2 Measuring Denial of Service using Bandwidth Monitor

The figure 15 is a screenshot from the Network Bandwidth monitor tool that gives us the traffic details when the attack was active. It could be seen from the figure that the network traffic is 3 Mbps when tested for topology 1 (as in figure 4). But when the traffic load was tested for topology 2 (as given in Figure 5) the load increased twice and reached up to 6 Mbps.

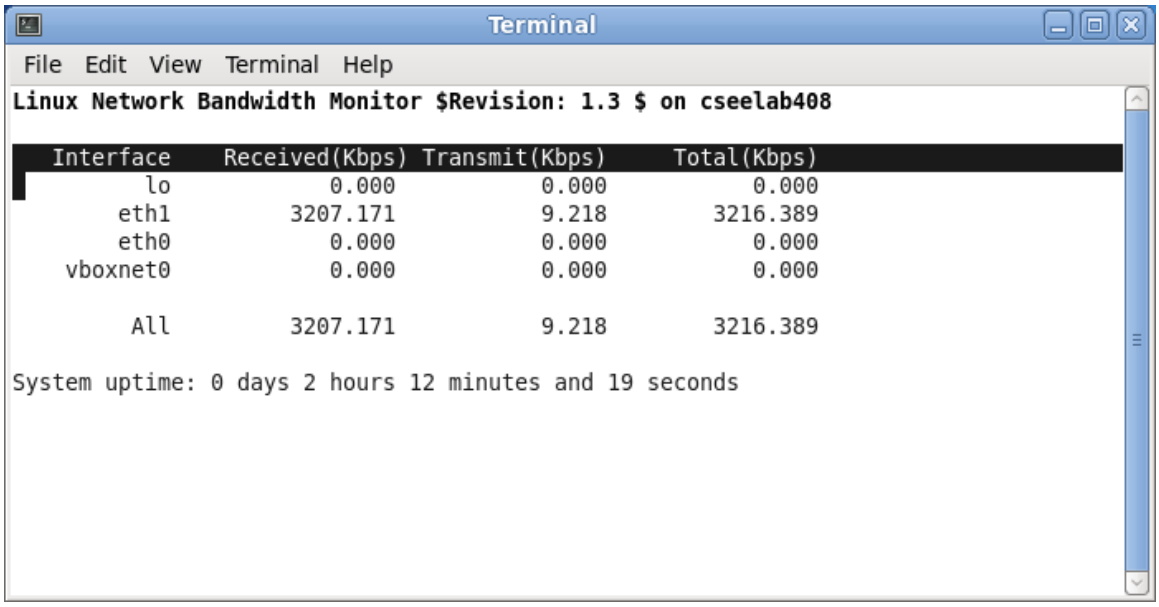

Figure 15: Screenshot from Bandwidth Monitor showing Network traffic rate

	Network topology 1	Network topology 2
With Attack	3216.389 kbps	6498.342 kbps
Without Attack	241 kbps	387 kbps
With ACL	497 kbps	568 kbps
With rate limit	812 kbps	879 kbps

Table 3: Network traffic rate during various scenarios

It could be noted from the readings that the traffic rate considerably reduces after applying mitigation strategies. The Access control list is able to reduce the traffic better than rate limiting as it entirely stops traffic coming from all the hosts in attacking network.

subramanirao@yahoo.com

7.3 Measuring Distributed Denial of Service using IPtraf

Unlike the previous environment, now the environment is set up in such a way that it has an attacker in every individual network. These networks work together in flooding the server thus leading to Distributed Denial of Service.

The following figure 16 from IPtraf shows the complete traffic information when ICMP, SYN and UDP attacks were carried out together. The reading shot up to 60 megabytes of traffic per second which is more than enough to get this network unstable.

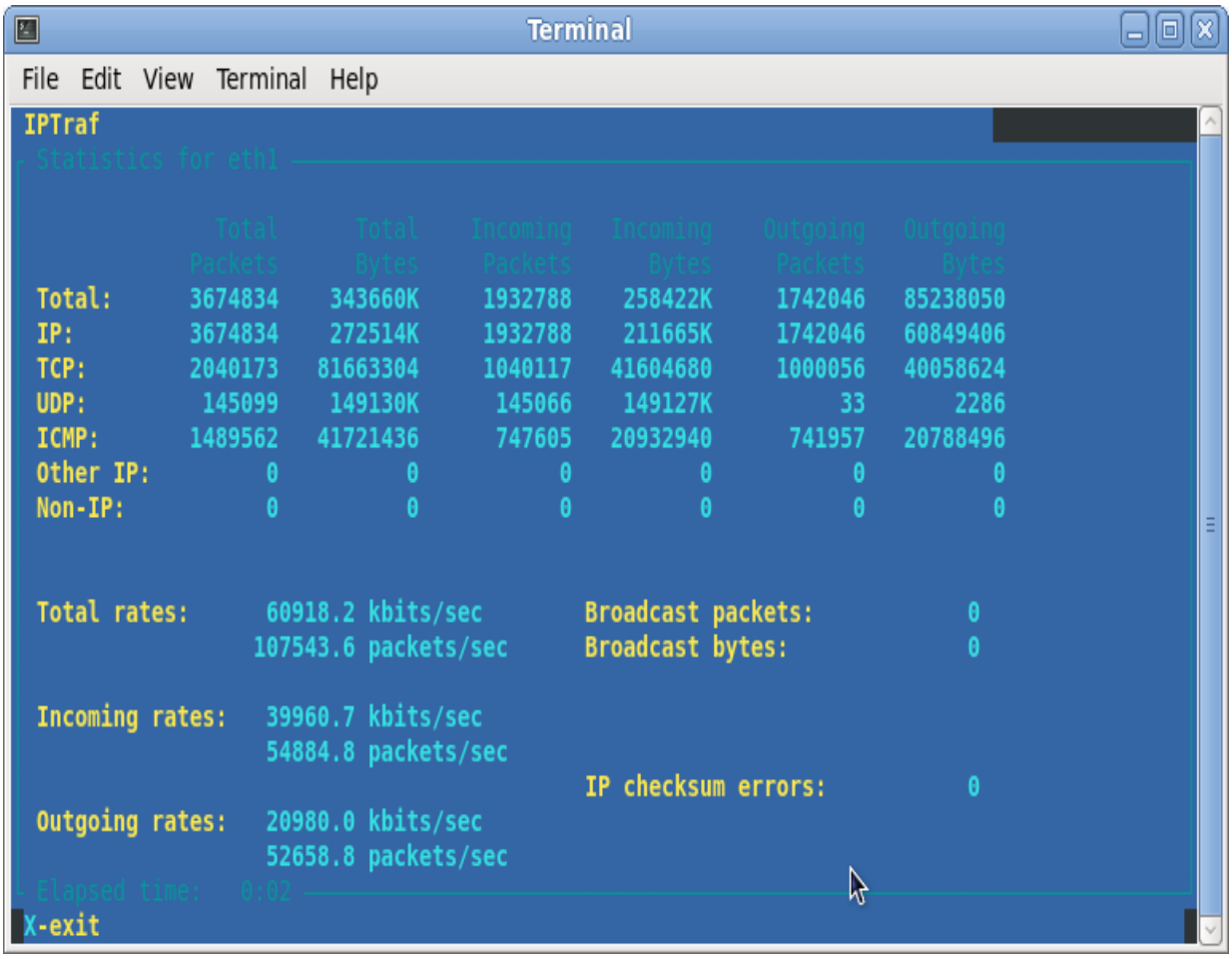

Figure 16: Screenshot of IPTraf showing DDoS attacks

	Network topology 1	Network topology 2
With 1 attacker	3.45 Mbps	6.74 Mbps
DDoS (Many attackers)	54 Mbps	60.92 Mbps

Table 4: Network traffic during Distributed Denial of Service

subramanirao@yahoo.com

7.4 Measuring Server response time

One of the very important metric to be measured in an attack would be server's response or loading time. The server's performance drastically changes during an attack and could be measured using the appropriate commands.

7.4.1 Using time curl

Using the following time curl command, the **response of time of the server** could be exactly determined. This command sends a HTTP packet to the server and measures the time it takes to process. In this following command, HTTP packets are sent to the server's IP address and the processing time is noted under different scenarios.

time curl -I http://192.168.1.1 | grep HTTP

As shown in the below figure, the response time during is approximately 9.3 seconds during an UDP attack. However, when the defense mechanisms (Rate limiting) are deployed, the processing time reduces considerably to 4.279 seconds. Any delay up to 8 seconds is accepted according to the internet standards and is known as the **'Network's 8 second rule'.** This rule points out that the response time should be less than 8 seconds to make internet browsing comfortable. It also brings out an interesting result that a user is distracted and looses patience if he is made to wait more than 8 seconds [34]. The response time drops down to 1.008 seconds when the attack is mitigated using ACL rules. In today's real time networks, the response time for a webpage is lesser than one second even though it runs complex java applications with embedded videos.

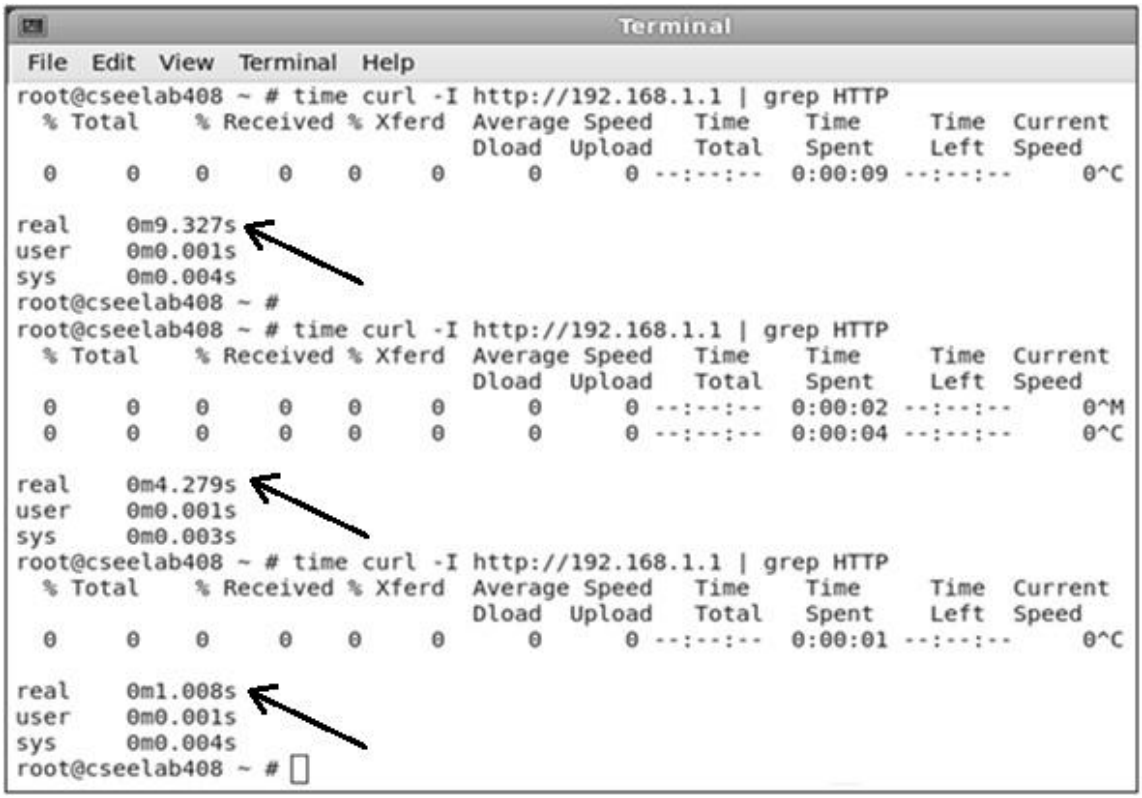

Figure 17: Response time with attack, with mitigation and without attack

subramanirao@yahoo.com

The following Table shows the response time of server under different scenarios. This is the same reading obtained from the screenshot of Linux terminal when applying time curl command. It could be noted that the server's response time is 'one second' without any attack. The table also shows how the response time varies when an UDP attack is implemented and also after Cisco rate limiting is applied.

Scenarios	**Server response time**
During UDP attack	9.327 sec
After applying Rate limiting	4.279 sec
Without an attack	1.008 sec

Table 5: Response time obtained using time curl command under different scenarios

7.4.2 Using WGET Utility

Wget is a powerful command that could be used in the Linux terminal in finding the **time taken by a client to download server's data**. It is a non interactive command line tool that is developed specially for the protocols like HTTP, HTTPS, and FTP etc [35]. It depends on how much information the server holding on its web page. This tool is helpful to find if there is a delay in downloading content due to a Denial of Service attack. It is to be understood that if the data to be downloaded is a large file, the reading might not be precise and accurate. For this reason, the website is loaded with an image file of small size and is requested by the client under different scenarios.

It could be seen from the below table 6 and figure 13that the time take to download content from the server increases several folds when the attack is active. In the below command shown in figure 13, IP address of the server follows after the keyword 'WGET' in order to determine the time taken for downloads.

Time taken to download data from server	
With attack	6.8 seconds
Without attack	0.834 seconds
With ACL	1.452 seconds
With Rate limiting	4.545 seconds

Table 6: Download time during various scenarios

The below Figure 18 shows is the screenshot of download time during an ICMP attack. The total download time taken to obtain all information from the server is observed as 6.8 seconds. Initially the 'Index' file took 1.4 seconds to get downloaded and the dos.jpg image file took 5.4 seconds to download. Thus a total of 6.8 seconds (1.4 + 5.4) was observed. However, the download time decreased considerably when the defense procedures were followed. The rate limiting technique reduced the time by nearly 2 seconds whereas ACL rules managed to decrease the download time by 5 seconds. When this command was run without any attack and the download time was noted to be only 0.834 seconds.

subramanirao@yahoo.com

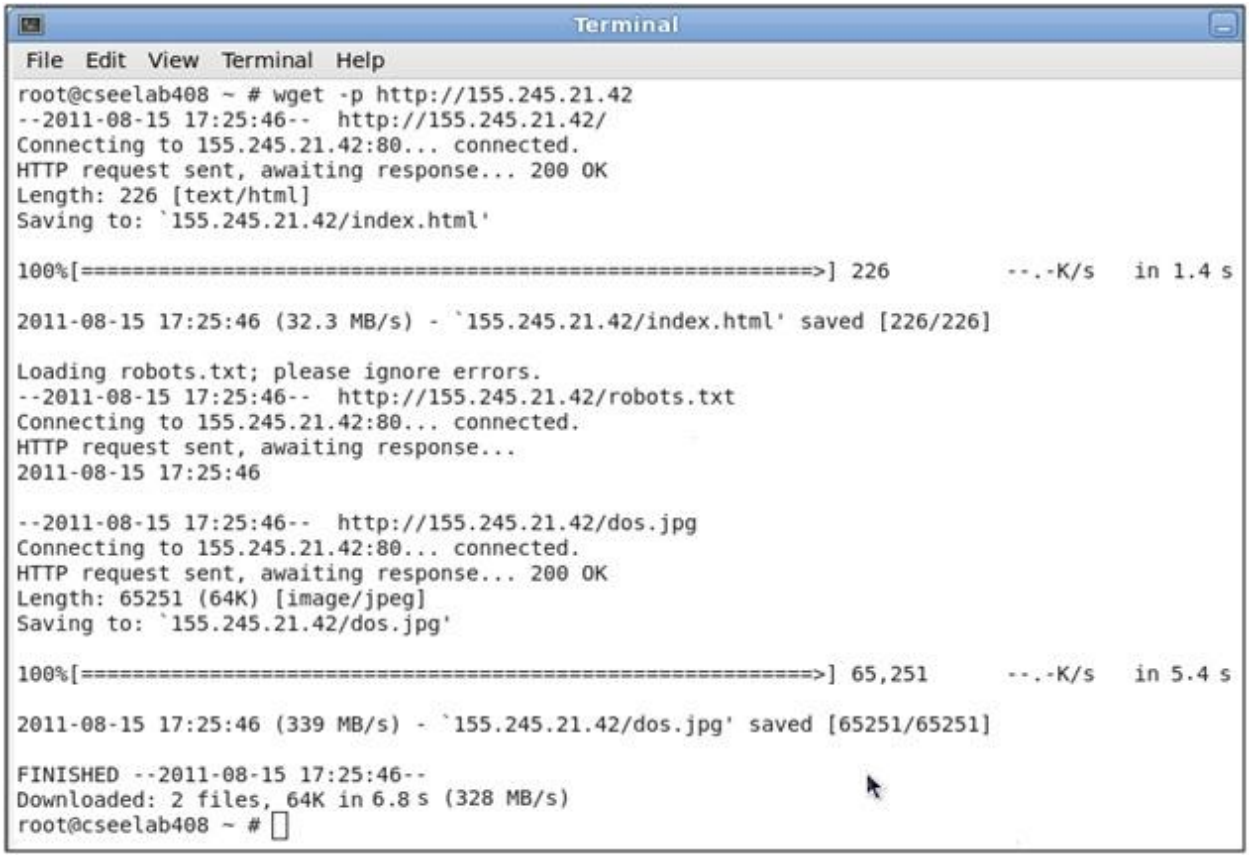

Figure 18: Wget screenshot showing download time

7.4.3 Using Apache benchmarking tool

Apache benchmarking tool is specially designed by the apache to measure the performance of the Apache HTTP servers [36]. This tool is independent of OS platforms and is very effective in measuring the server's processing capability. In this project, every host was installed with this tool and performance was tested with and without attacks.

The command 'ab' when entered on the Linux terminal starts the Apache benchmarking tool. The following command was used in this project and the following observations were made.

ab -n 1000 -c 5 http://155.245.21.42/

The mean request time is obtained as 0.834 seconds when the network is free from flooding. However, when the attack was initiated, the reading changed to 8.782 seconds. But after applying defense mechanisms like ACL and rate limiting commands on the Cisco router, the request time dropped down considerably.

Mean Request Time	
With attack	8.782 seconds
Without attack	0.834 seconds
With ACL	1.093 seconds
With Rate limiting	6.985 seconds

Table 7: Mean Request Time

subramanirao@yahoo.com

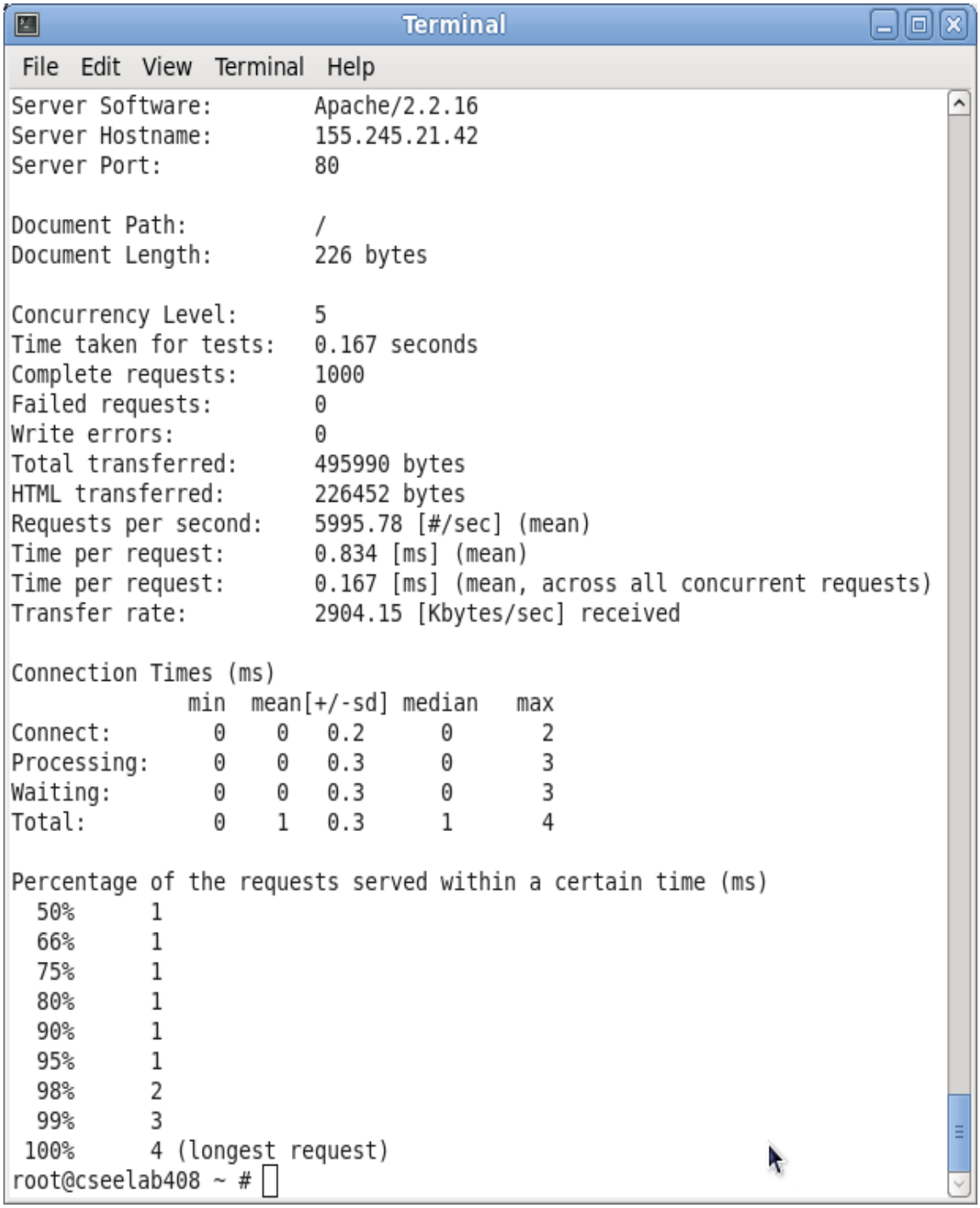

Figure 19: Mean request time using Benchmarking tool

The above screenshot was captured from the Apache benchmarking tool when measuring the mean request time from the server. The mean request time of 0.834 was observed when all the attacks were turned off.

subramanirao@yahoo.com

7.5 Netstat commands in measuring SYN, UDP and ICMP attacks

Netstat is a UNIX tool that could be used from the command line terminal and is used for various operations [37]. It is used to list all the inbound and outbound traffic connections of a UNIX machine. The same was employed in this project to measure the magnitude of the attacks generated. The following command was used to determine the number of active TCP and UDP connections the server holds. The readings in table 8 shows the values obtained, when the command was executed with and without an UDP attack.

netstat -anp |grep 'tcp\|udp' | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort –n

With attack (UDP)		**Without attack**	
45845	192.168.2.1	4	192.168.2.1
1	192.168.2.2	8	192.168.2.2
2	192.168.3.1	1	192.168.3.1
12	192.168.3.2	10	192.168.3.2

Table 8: Number of SYN connections

The second command shown here also employs netstat, but shows only the number of active SYN connections. This command was used in the Linux command line terminal during TCP SYN flood attack.

netstat -an | grep "SYN_RCVD"

The number of awaiting SYN connection was 847512 during the attack which completely drained out the server memory. The server was not able to hold all the pending SYN connections in its CPU memory space and eventually crashed. However, when the attack was stopped, the number of awaiting SYN connections dropped down to 6.

netstat -s -t | more

The above command was used to pull out the entire details of the existing, awaiting and pending connection requests. This command is very powerful and could be used directly from the linux terminal. This command brings out the entire information about all the incoming and outgoing connections the system is currently handling. This is one of the best commands a network administrator can use in order to gain complete knowledge about the system. It gives an overall picture of the machine and helps in debugging the problem in an effective manner. The Netstat could be altered in various ways according to the need and important system information could be deduced out of it.

The figure 21 below shows the results yielded with the above command which gives the complete listing of all the connection status. At the initial stages of ‘SYN flood attack’, the number of active connection showed 546 and then rose up to 54785 during the peak. This feature also showed the number of passive and failed connections. When the attack rate was increased, the number of failed connections increased considerably including the legitimate requests. This command also highlights the amount of message that are being queued and the number of packets that gets dropped due to TCP congestion.

subramanirao@yahoo.com

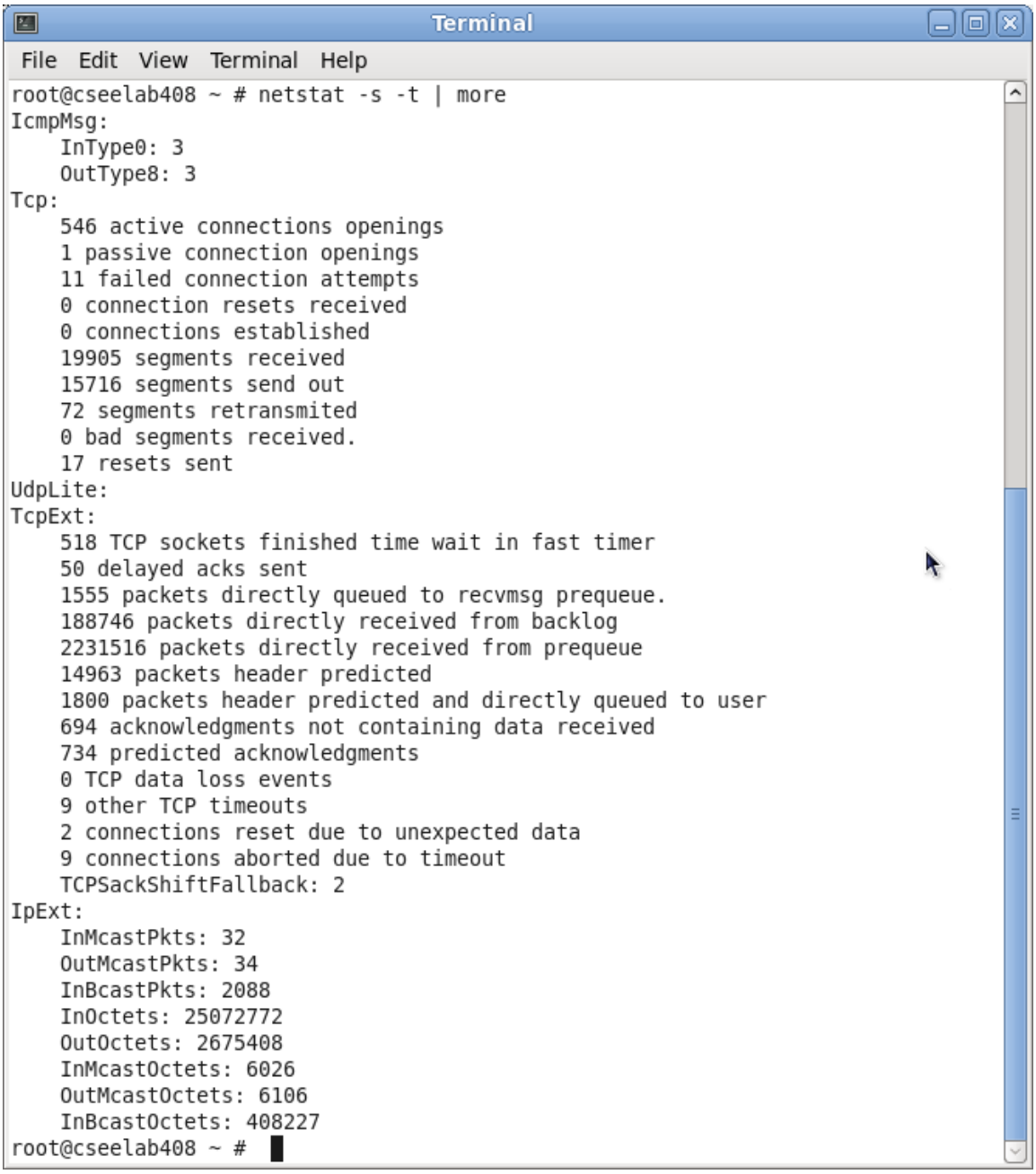

Figure 20: Netstat TCP command

The following command was used during the ICMP ping of death attack and was used to measure the number of ICMP ping requests sent to the server.

While:; do netstat -s| grep -i icmp | egrep 'received|sent' ; sleep 1; done

Even though the ICMP ping requests could be monitored and measured using the wireshark tool, this particular command gives the exact number of packets used. This output from the terminal increments itself in real time and shows the exact number of ICMP ping request obtained. As shown in the screen shot below, the number of ping packets started from 3 and came up to 354842 during the 'ICMP ping flood attack'.

subramanirao@yahoo.com

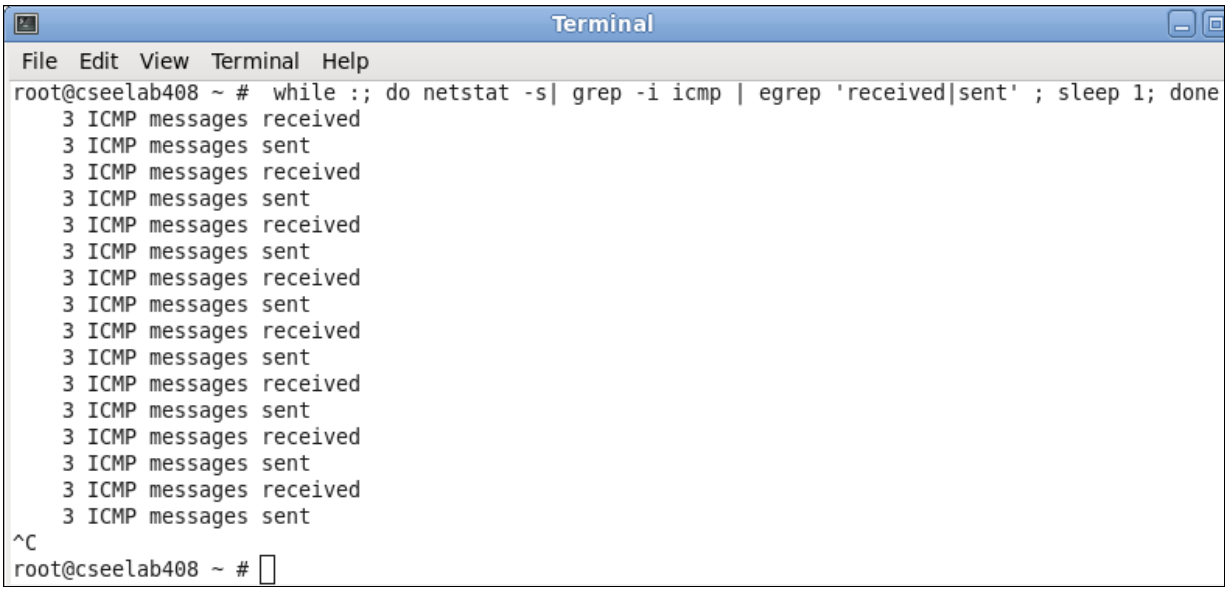

Figure 21: Netstat ICMP command

subramanirao@yahoo.com

CHAPTER 8 CONCLUSIONS AND KEY FINDINGS

8.1 Inference of this project

There are a few important inferences deduced out of this project experiment and is discussed below with corresponding observations.

(i) The topology of a network decides the amount of attack traffic passing through it.

More number of network elements in the path results in potential bottlenecks leading to higher traffic rates whereas with fewer network elements the network is free from traffic bursts.

These were the readings taken during UDP, ICMP and SYN flood attacks.

	Network Topology 1	**Network Topology 2**
	IP packet traffic	
UDP attack	2458 KB/s	6845KB/s
Ping of death	5452 KB/s	87598KB/s
SYN flooding	2541 KB/s	5413 KB/s

Table 9: Traffic during UDP, TCP and ICMP attacks

From the above table it could be understood that the network traffic rate is almost doubled with network topology with two routers when compared to a network with single router. When a path with more number of networking devices is attacked, the load on the network increases leading to packet loss and retransmissions. However, the server is still safe from getting damaged by the forged or malformed packets.

(ii) More number of network elements in the path keeps the server safe from exploits and physical damages

The attacks carried out with the Network topology 1 crashed the Apache server several times damaging all the data saved on it. Whereas, in the second network topology, there was no server crash observed. However in the latter scenario, there is a potential risk of networking equipments like routers and switches getting damaged. Hence an important deduction was made out of this test bed environment. If a company would like to secure its data more than the networking devices, it should employ enough routers and switches in the network path. However, that would definitely cost more than a simple network and might lead to delayed response from the server due to routing and switching. Hence, there is an important tradeoff between security and performance that a company must consider before designing its network structure.

(iii) Combined rate limiting and ACL is a better solution

It could be seen from chapter 6 that ACL rules keeps an entire network out of access which proves highly inefficient when there are a large number of legitimate clients from an attacking network. Also during Distributed Denial of Service, a larger number of networks are blocked thus allowing minimal hosts to obtain service.

Rate limiting is comparatively better than ACL, yet allows traffic from the attacker to the reach the server. This enables the attacker to inject worms and viruses in the server which could lead to possible exploits like buffer overflow, dangling pointers etc. Once the attacker is

subramanirao@yahoo.com

successful in exploiting the weaknesses of the server, he can install a '**rootkit'** and use it for malicious purposes. The attacker might also create a **'backdoor'** and seal the vulnerabilities which he earlier used to gain access. Thus rate limiting does not seem to be an ideal solution against Denial of Service attacks.

The hybrid concept proves better than the previous two strategies giving the server enhanced protection. It also provides wider access allowing legitimate clients from the attack network to establish connection with the server. In this technique legitimate clients in the attack network are given access with controlled rate.

8.2 Future work

This project gives the reader adequate knowledge on how to implement Denial of Service attacks. It highlights various attack tools and the ways to identify the same in a given network. It also suggests basic mitigation strategies that could be adopted in order to defend attacks.

However, serious challenges arise when IPv6 needs to be established globally and transition from version 4 to version 6 has to be done. IPv6 introduces six optional headers like Routing header, Authentication header etc [38]. In spite of providing better security with authentication, encryption and encapsulation techniques, IPv6 also brings out serious complications. The following two types of Denial of Service attacks could be implemented if IPv6 is used.

Routing Header Denial of Service [39]:

Experts say DDoS attacks could be strengthened 88% more in IPv6 when compared to the IPv4. In IPv4, the path taken by the attack packets can be either one way (TCP, UDP and other attacks) or two ways (ICMP traffic). But in IPv6, the attack packets could be made to oscillate between the routers endlessly. Thus the network would be constantly occupied by these forged packets and can lead to a powerful DDoS attack. Routing header is an extension header that dictates a packet to visit the compulsory routers on path. Attackers can easily forge this Routing header (RH0) and make a packet wander back and forth between two routers.

Denial of Service attack exploiting IPv6 mobility [40]:

IPv6 has come out with a revolutionary idea of mobile IP that was not possible in the former versions. Regular IP is designed to serve only the stationary users. A user is forced to change his IP address when he changes his geographical network. But with the advent of IPv6, a user can change his geographical location moving to different networks and can still hold to a single IP address. This is achieved by the extension headers provided in IPv6. The original IPv6 address is stored in the extension header whereas an additional temporary address is held in the IP header. The temporary address keeps changing when the user is mobile but the original IP address remains unchanged. An attacker can easily change this temporary IP address and carry out spoof attacks.

Since IPv4 is in the verge of extinction and is getting replaced by IPv6, the future work should focus on stopping DDoS in IPv6. In spite of having additional headers and options for enhanced security in IPv6, it is still prone to various flavors of Denial of Service attacks. Thus, additional research work should be emphasized on IPv6 security and proper mitigation techniques should be introduced for existing vulnerabilities.

subramanirao@yahoo.com

REFERENCES

1. Liu, S, “Surviving Distributed Denial-of-Service Attacks,” IEEE Computer, vol. 11, no. 5, pp. 51–53, Sep. 2009.
2. Lee, Garber, “Denial-of-service attacks rip the Internet,” IEEE Computer, vol. 33, no. 4, pp. 12–17, Apr. 2000.
3. Schuba, C.L.; Krsul, I.V.; Kuhn, M.G.; Spafford, E.H.; Sundaram, A.; Zamboni, D.; , "Analysis of a denial of service attack on TCP," Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on , vol., no., pp.208-223, 4-7 May 1997
4. J. Elliott, “Distributed denial of service attacks and the Zombie ant effect,” IT Professional, pp. 55–57, Mar.–Apr. 2000.
5. Khaled M. Elleithy, “Denial of Service Attack Techniques: Analysis, Implementation and Comparison,” IIISCI journal, vol. 3, no.1, pp. 66-71.
6. X. Geng and A. B. Whinston, “Defeating distributed denial of service attacks,” IT Professional, vol. 2, no. 4, pp. 36–42, July–Aug. 2000.
7. Cheung, S.; , "Denial of service against the Domain Name System," Security & Privacy, IEEE , vol.4, no.1, pp. 40- 45, Jan.-Feb. 2006.
8. Macia-Fernandez, G.; Diaz-Verdejo, J.E.; Garcia-Teodoro, P.; , "Mathematical Model for Low-Rate DoS Attacks Against Application Servers," Information Forensics and Security, IEEE Transactions on , vol.4, no.3, pp.519-529, Sept. 2009.
9. Solar winds. (2009, April 11) Enabling Net Flow and Net Flow Data Export (NDE) on Cisco Catalyst Switches (2nd ed.) [Online]. Available: http://www.solarwinds.com/documentation/Netflow/docs/OrionNetFlowSwitches.pdf
10. Cisco. (2011, August 4) Cisco IOS Release 12.0 Switching Services Configuration Guide (Release 12.0) [Online]. Available: http://www.cisco.com/en/US/docs/ios/12_0/switch/configuration/guide/xcovntfl.html
11. Dabir, A.; Matrawy, A.; , "Bottleneck Analysis of Traffic Monitoring using Wireshark," Innovations in Information Technology, 2007. IIT '07. 4th International Conference on , vol., no., pp.158-162, 18-20 Nov. 2007
12. Fred Halsall, “Computer Networking and the Internet,” 5th ed., South Asia: Dorling Kindersley., Licenses of Pearson Education in South Asia. pp. 6,7 and 31
13. J. Case, M. Fedor, M. Schoffstall, and J. Davin, “A simple network management protocol (SNMP),” IETF, RFC 1157, May 1990.
14. Lee, M.; Duffield, N.; Kompella, R. R.; , "Opportunistic Flow-Level Latency Estimation Using Consistent NetFlow," Networking, IEEE/ACM Transactions on , vol.PP, no.99, pp.1, 0
15. Robin Richter (2009 January 26) Open source software – Hyenae Packet Generator (0.35 version) [Online]. Available: http://hyenae.sourceforge.net/
16. Erikson, Jon (2008). *HACKING the art of exploitation* (2nd edition ed.). San Francisco: No Starch Press. p. 251. ISBN 1-59327-144-1.

subramanirao@yahoo.com

17. Darren Hoch. (2004 November) Analysis of Network Denial of Service, UUASC - 11/04 [Online]. Available: http://www.ufsdump.org/papers/uuasc-november-ddos.pdf

18. Cisco. (2007, October) Cisco IOS Netflow, Introduction to Cisco IOS Netflow - A Technical Overview [Online]. Available: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html

19. Brad Reese. (2007 November 11) Cisco invention NetFlow appears missing in action as Cisco invests into the network behavior analysis business [Online]. Available: http://www.networkworld.com/community/node/22284

20. Craig Labovitz. (2010 November 3) Attack Severs Burma Internet: Arbor Networks, [Online]. Available: http://asert.arbornetworks.com/2010/11/attac-severs-myanmar-internet/

21. Markoff, John (August 13, 2008). "Before the Gunfire, Cyberattacks". *The New York Times*. Retrieved 2008-08-12. [Online]. Available: http://www.nytimes.com/2008/08/13/technology/13cyber.html?em

22. Shachtman, Noah (2009-06-15). "Activists Launch Hack Attacks on Tehran Regime". *Wired*. Retrieved 2009-06-15. [Online]. Available: http://www.wired.com/dangerroom/2009/06/activists-launch-hack-attacks-on-tehran-regime/

23. Schneider, D.; , "Network defense gone wrong [Update]," *Spectrum, IEEE* , vol.48, no.2, pp.11-12, February 2011 doi: 10.1109/MSPEC.2011.5693060

24. Kuzmanovic, A.; Knightly, E.W.; , "Low-rate TCP-targeted denial of service attacks and counter strategies," *Networking, IEEE/ACM Transactions on* , vol.14, no.4, pp.683-696, Aug. 2006
doi: 10.1109/TNET.2006.880180

25. Jun Xu; Wooyong Lee; , "Sustaining availability of Web services under distributed denial of service attacks," *Computers, IEEE Transactions on* , vol.52, no.2, pp. 195- 208, Feb. 2003
doi: 10.1109/TC.2003.1176986

26. Chang, R.K.C.; , "Defending against flooding-based distributed denial-of-service attacks: a tutorial," *Communications Magazine, IEEE* , vol.40, no.10, pp. 42- 51, Oct 2002
doi: 10.1109/MCOM.2002.1039856

27. ARS technical, Joel hrushka. (2008, August) "Phlashing" attacks could render network hardware useless [Online]. Available: http://arstechnica.com/security/news/2008/05/phlashing-attacks-could-render-network-hardware-useless.ars

28. VeriSign. (2009, April 11) DDoS Protection Services, Network Intelligence [Online]. Available: http://www.verisigninc.com/en_US/products-and-services/network-intelligence-availability/ddos/index.xhtml

29. Arbor Networks. (20011, August 11) Peakflow SP: Traffic Anomaly Detection [Online]. Available: http://www.arbornetworks.com/peakflowsp

subramanirao@yahoo.com

30. Liu, A.; Torng, E.; Meiners, C.; , "Compressing Network Access Control Lists," *Parallel and Distributed Systems, IEEE Transactions on* , vol.PP, no.99, pp.1, 0 doi: 10.1109/TPDS.2011.114

31. Cisco. Cisco IOS Release 12.0 Quality of Service Solutions Command Reference, Quality of Service Commands (r-z) [Online]. Available: http://www.cisco.com/en/US/docs/ios/12_0/qos/command/reference/qrcmdr.html#wp1017761

32. Robichaud, Y.; Changcheng Huang; Jinmei Yang; Peng, H.; , "Access delay performance of resilient packet ring under bursty periodic class B traffic load," *Communications, 2004 IEEE International Conference on* , vol.2, no., pp. 1217- 1221 Vol.2, 20-24 June 2004

33. Cisco. Configuring Class Maps and Policy Maps (Chapter 4), Cisco Application Control Engine Module Administration Guide [Online]. Available: http://www.cisco.com/en/US/docs/interfaces_modules/services_modules/ace/v3.00_A1/configuration/administration/guide/mapolcy.pdf

34. Dr. Ken Guild. University of Essex, "Converged Networks & Services," (Chapter 1) IP Convergence & Service Level Agreements, pp 22, 15-March 2011.

35. Micah Cowan, (last edited 2011-06-01 07), "The Wget Wgiki" [Online]. Available: http://wget.addictivecode.org/

36. Apache, (2011) ab - Apache HTTP server benchmarking tool (Version 2.0) [Online]. Available: http://httpd.apache.org/docs/2.0/programs/ab.html

37. Vigna, G.; Kemmerer, R.A.; , "NetSTAT: a network-based intrusion detection approach," *Computer Security Applications Conference, 1998, Proceedings., 14th Annual* , vol., no., pp.25-34, 7-11 Dec 1998 doi: 10.1109/CSAC.1998.738566

38. Dr Nick Zakhleniuk. University of Essex, "IP Networking and Applications," (Chapter 3) IPv6 Extension Headers (EHs) - IPv6 Security, March 2011.

39. J. Abley, Afilias and P. Savola, "Deprecation of Type 0 Routing Headers in IPv6," IETF, RFC 5095, December 2007.

40. Sierra, J.M.; Ribagorda, A.; Munoz, A.; Jayaram, N., "Security protocols in the Internet new framework," Proceedings IEEE 33rd Annual 1999 International Carnahan Conference on Security Technology, 1999. vol., no.pp.311-317, 1999.

41. Viecco, C.; Camp, J.; , "A Life or Death InfoSec Subversion," *Security & Privacy, IEEE* , vol.6, no.5, pp.74-76, Sept.-Oct. 2008

42. Libes, D,; ," Using expect to Automate System Administration Tasks," National Institute of Standards and Technology, pp1-2, 21 Jan 1992

43. Dr. Martin Reed, "University of Essex," Computer Science and Electronic Engineering (School of), 23 Sep, 2011

44. Picture in Index page is taken from the Symantec tutorial video.

subramanirao@yahoo.com

APPENDICES

Appendix 1 [9]

```
victim#conf t
victim(config)#int f 0/0
victim(config-if)#ip route-cache flow
victim(config-if)#ex
victim(config)#ip flow-export destination 192.168.1.2 7776
victim(config)#ip flow-export source f 0/1
victim(config)#ip flow-export version 9
victim(config)#snmp-server ifindex persist
victim(config)#ex
victim#write memory
```


subramanirao@yahoo.com

Appendix 2 [43]

```
#!/bin/bash
limit=10
function acl() {
    echo "I will stop $1"
prompt="*#"

expect -c "
spawn ssh -l cisco 192.168.1.1
sleep 2

expect {
        "*word:"
       {
send "cisco"\r
       }
       }
sleep 1

expect {
       "*"
       {
send "en"\r
       }
       }
sleep 1

expect {
       "*word:"
       {
send \"cisco\"
send \r
       }
       }
sleep 1

expect {
       "$prompt"
       {
send \"ena\"
send \r
       }
       }
sleep 1
```


subramanirao@yahoo.com

```
expect {
       "*#"
       {
send \"conf t\"
send \r
       }
       }
sleep 1

expect {
       \"*\(config\)#\"
       {
send \"access-list 1 deny $1\"
send \r
send \"int f0/1\"
send \r
send \"ip access-group 1 in\"
send \r
       }
       }
sleep 1

expect {
       \"*\(config-if\)#\"
       {
send \"do logout\"
send \r
       }
       }
sleep 1

expect {
       "*#"
       {
send \"logout\"
send \r
       }
       }
expect  eof "
}

#netstat -anp |grep 'tcp\|udp' | awk '{print $5}' | cut -d: -f1 | sort
| uniq -c | sort -n > temp.dat
saveifs=$IFS
IFS=" "
```


subramanirao@yahoo.com

```
cat temp.dat | while read line; do
    arr=($line)
    count=${arr[0]}
    ip=${arr[1]}

    if [ $ip == "0.0.0.0" ]; then
    echo "should ignore $ip"
    else
    if (( count > limit )); then
        echo "stopping $ip"
        acl $ip
    else
        echo "not stopping $ip"
    fi
    fi
done
IFS=$saveifs
exit
```


subramanirao@yahoo.com

Appendix 3 [43]

```
#!/bin/bash
limit=10
function acl() {
    echo "I will stop $1"
prompt="*#"

expect -c "
spawn ssh -l cisco 192.168.1.1
sleep 2

expect {
        "*word:"
       {
send "cisco"\r
       }
       }
sleep 1

expect {
       "*"
       {
send "en"\r
       }
       }
sleep 1

expect {
       "*word:"
       {
send \"cisco\"
send \r
       }
       }
sleep 1

expect {
       "$prompt"
       {
send \"ena\"
send \r
       }
       }
sleep 1
```


subramanirao@yahoo.com

```
expect {
       "*#"
       {
send \"conf t\"
send \r
       }
       }
sleep 1

expect {
       \"*\(config\)#\"
       {
send \"access-list 1 deny $1\"
send \r
send \"int f0/1\"
send \r
send \" rate-limit input 8000 2000 4000 conform-action transmit
exceed-action drop \"
send \r
       }
       }
sleep 1

expect {
       \"*\(config-if\)#\"
       {
send \"do logout\"
send \r
       }
       }
sleep 1

expect {
       "*#"
       {
send \"logout\"
send \r
       }
       }
expect  eof "
}

#netstat -anp |grep 'tcp\|udp' | awk '{print $5}' | cut -d: -f1 | sort
| uniq -c | sort -n > temp.dat
saveifs=$IFS
```


subramanirao@yahoo.com

```
IFS=" "
cat temp.dat | while read line; do
    arr=($line)
    count=${arr[0]}
    ip=${arr[1]}

    if [ $ip == "0.0.0.0" ]; then
    echo "should ignore $ip"
    else
    if (( count > limit )); then
        echo "stopping $ip"
        acl $ip
    else
        echo "not stopping $ip"
    fi
    fi
done
IFS=$saveifs
exit
```


subramanirao@yahoo.com