DDoS attacks and their alleviation using iptables

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Submitted By
Ravinder Dahiya
(Roll No. 801031025)

Under the supervision of:
Dr. Maninder Singh
Associate Professor
CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*DDoS attacks and their alleviation using iptables* ", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Maninder Singh* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Signature:
Ravinder Dahiya

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Dr. Maninder Singh
Associate Professor
CSED

Countersigned by
Dr. Maninder Singh
Head Computer science
And engineering department
Thapar University
Patiala

Dr. S. K. Mohapatra
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

Many people have shared their time and expertise to help me improvise and accomplish my goal. I would like to sincerely thank my guide, Dr. Maninder Singh, for his constant and felicitous support and guidance. His instant responses to my countless inquiries have been invaluable and motivational. It was a great opportunity to work under his supervision.

Dr. Maninder Singh, Head, CSED, introduced me to the concept of DoS/DDoS attacks and inspired me to choose the current research focus. I would like to thank him for his moral support and research environment he had facilitated for this work.

I wish to thank my family and my friend Mohit Raj for their immense support and encouragement throughout this time without which it would not have been possible to complete this work.

Last but not the least I would like to thank God who have always been with me in my good and bad times.

Ravinder Dahiya
ME SE
801031025

Abstract

Denial-of-Service (DoS) is a network security attack vector that poses a serious challenge to trustworthiness of services deployed on the servers. The aim of DoS attacks is to make services unavailable to legitimate users by flooding the victim with legitimate-like requests and current network architectures allow easy-to-launch, hard-to-stop DoS attacks.

Threat of DoS attacks has become even more severe with DDoS (Distributed Denial-of-Service) attack. It is an attempt by malicious users to carry out DoS attack indirectly with the help of many compromised computers on the Internet. Attackers can compromise a huge number of computers by spreading a computer worm using vulnerabilities in popular operating systems.

This exhausts the victim network of resources such as bandwidth, computing power, etc., the victim is unable to provide services to its legitimate clients and network performance is greatly deteriorated, moreover, with little or no advance warning, a DDoS attack can easily exhaust these resources within a short period of time.

As is said, to fight against this enemy and knowing basics about it, this work discusses some of the methods to do DoS and DDoS attacks by using automated tools as well as by creating compromised machines who become part of the attack unwillingly. The work also explores the efficient packet filtering technique using firewall to defend against DoS/DDoS attacks traffic.

Contents

List of Figures

List of Tables

1. Introduction

The phenomenal growth of the Internet, and its entry into many aspects of daily life, has led to the dependency on its services very often. Unfortunately, it has been shown [1]-[2] that it is easy to disturb the functionality of the Internet by attacking its infrastructure taking advantage of the vulnerable Internet elements and protocols. Denial of Service (DoS) attacks are among the top threats to the Internet infrastructure [3]. These attacks can quickly incapacitate a targeted business, costing victims thousands, if not millions, of dollars in lost revenue and productivity. The objective of this dissertation is to develop efficient countermeasures for these attacks.

The thesis outline is as follows. In this chapter, discussion is on a classification of DoS attacks, their types, some tools to perform DoS attacks and Netfilter firewall Iptables. In Chapter 2, the main research efforts to perform and counter DoS attacks are discussed. In Chapter 3, problem of statement is presented. Chapter 4 focuses on implementation setup and the experimental results and proposed DoS mitigation schemes. Finally, the conclusion and the future work are discussed in Chapter 5.

In this chapter, the problem of DoS attacks focusing on their original causes, and on their various mechanisms is discussed. Also, the challenges associated with this problem are discussed.

1.1. DoS and DDoS attacks

Computer network has undoubtedly become part of today's infrastructure but unfortunately security systems and policies to govern these networks have not progressed as rapidly. With the advance of information and communication technologies, our societies are evolving into global information societies with a ubiquitous computing environment that has made cyber attacks significantly more sophisticated and threatening

[4]. One type of cyber attack that is becoming more prevalent today is that known as a Denial of Service (DoS) attack.

1.1.1. DoS attacks

Information security aims to safeguard confidentiality, integrity and availability of information and information systems. A Denial of Service (DoS) attack is a type of attack focused on disrupting availability. Such an attack can take many shapes, ranging from an attack on the physical IT environment, to the overloading of network connection capacity, or through exploiting application weaknesses.

Gligor defined DoS as follows [21]: "a group of otherwise-authorized users of a specific service is said to deny service to another group of authorized users if the former group makes the specified service unavailable to the latter group for a period of time which exceeds the intended (and advertised) waiting time."

One hundred per cent availability of systems and networks is widely accepted to be unattainable, regardless of diligence or the amount of resources allocated to securing systems against attack. Internet-facing and other networked infrastructure components are at risk of DoS for two primary reasons:

1. Resources such as bandwidth, processing power, and storage capacities are not unlimited and so DoS attacks target these resources in order to disrupt systems and networks.
2. Internet security is highly interdependent and the weakest link in the chain may be controlled by someone else thus taking away the ability to be self reliant.

1.1.2. DDoS attacks

From the beginning the evolution of solutions to resolve the occurrence of attacks promoted the evolution of the attacks itself, and nowadays DoS attacks have been superseded by Distributed Denial of Service (DDoS) attacks, where attackers do not use a single host for their attacks but a cluster of several dozens or even hundreds of computers to do a coordinated strike. The WWW Security FAQ [7] on Distributed Denial of Service

says that: “A Distributed Denial of Service attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the perpetrator is able to multiply the effectiveness of the Denial of Service significantly by harnessing the resources of multiple unwitting accomplice computers which serve as attack platforms. Typically a DDoS master program is installed on one computer using a stolen account. The master program, at a designated time, then communicates to any number of "agent" programs, installed on computers anywhere on the Internet. The agents, when they receive the command, initiate the attack. Using client/server technology, the master program can initiate hundreds or even thousands of agent programs within seconds”.

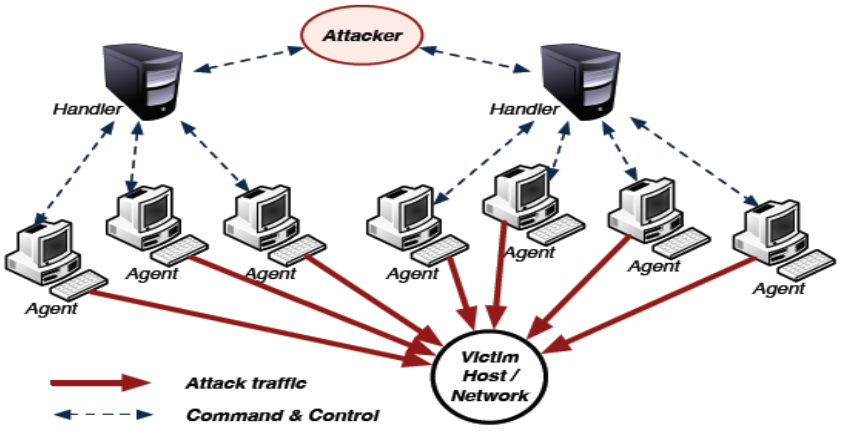


Figure1: Distributed Denial of Service Attack [17].

In February of 2000, one of the first major DDoS attacks was waged against Yahoo.com, keeping it off the Internet for about 2 hours, costing it lost advertising revenue. There are two major reasons making DDoS attacks attractive for attackers. The first reason is that there are effective automatic tools available for attacking any victim , i.e., expertise is not necessarily required. The second reason is that it is usually impossible to locate an attacker without extensive human interaction [5] or without new features in most routers of the Internet [6].

DoS/DDoS attacks have become more sophisticated in the last several years as the level of attack automation has increased. Fully functional attack software like Hping, Mausezahn etc are readily available on the Internet which allow novice users to launch relatively large scale attacks with little knowledge of the underlying security exploits. The advent of remote controlled networks of computers used to launch attacks has changed the landscape and methods that a service provider must use. In the past year, Black Hats have taken theoretical optimizations in worm propagation and applied them to the fastest spreading worm today, Slammer [8]. The financial industry is especially susceptible to DoS/DDoS attacks as millions of consumers are moving to electronic bill payments, purchases and on-line banking.

1.2. Types of DoS/DDoS Attacks

The attacker sends packets directly from his computer(s) to the victim's site but the source address of the packets may be forged. There are many tools available to allow this type of attack for a variety of protocols including ICMP, UDP and TCP. Some common tools include stream2, synhose, synk7, synsend, and hping2**.**

UDP flood attack: A UDP Flood attack is possible when a large number of UDP packets is sent to a victim system. This has as a result the saturation of the network and the depletion of available bandwidth for legitimate service requests to the victim system. UDP Flood attack, the UDP packets are sent to either random or specified ports on the victim system. Typically, UDP flood attacks are designed to attack random victim ports. A UDP Flood attack is possible when an attacker sends a UDP packet to a random port on the victim system. When the victim system receives a UDP packet, it will determine what application is waiting on the destination port. When it realizes that there is no application that is waiting on the port, it will generate an ICMP packet of ''destination unreachable'' to the forged source address. If enough UDP packets are delivered to ports of the victim, the system will go down. By the use of a DoS tool the source IP address of the attacking packets can be spoofed and this way the true identity of the secondary

victims is prevented from exposure and the return packets from the victim system are not sent back to the zombies [9, 10].

ICMP flood attack: ICMP Flood attacks exploit the Internet Control Message Protocol (ICMP), which enables users to send an echo packet to a remote host to check whether it's alive. More specifically during a DDoS ICMP flood attack the agents send large volumes of ICMP_ECHO_ REPLY packets (''ping'') to the victim [9, 10].

These packets request reply from the victim and this results in saturation of the bandwidth of the victim's network connection. During an ICMP flood attack the source IP address may be spoofed.

SYN flood attack: The client system begins by sending a SYN message to the server. The server then acknowledges the SYN message by sending SYN-ACK message to the client. The client then finishes establishing the connection by responding with an ACK message. The connection between the client and the server is then open, and the service-specific data can be exchanged between the client and the server. Here is a view of this message flow:

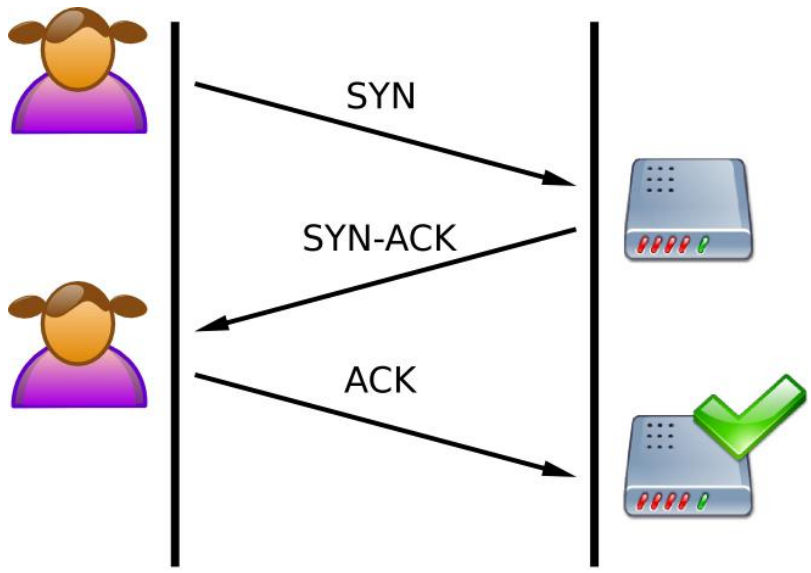


Figure 2: Client server communication [12]

The potential for abuse arises at the point where the server system has sent an acknowledgment (SYN-ACK) back to client but has not yet received the ACK message. This is known as half-open connection. The server has built in its system memory a data structure describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially-open connections.

Creating half-open connections is easily accomplished with IP spoofing. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system.

The half-open connections as shown in Figure 3, data structure on the victim server system will eventually fill; then the system will be unable to accept any new incoming connections until the table is emptied out. Normally there is a timeout associated with a pending connection, so the half-open connections will eventually expire and the victim server system will recover [11].

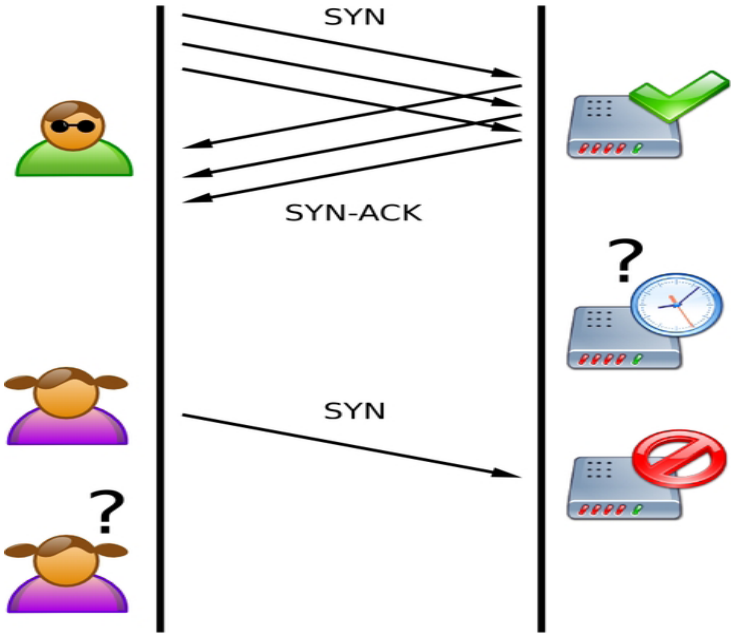


Figure 3: SYN Flood Attack scenario [12].

Smurf attack: The two main components to the smurf denial-of-service attack are the use of forged ICMP echo request packets and the direction of packets to IP broadcast addresses.

The Internet Control Message Protocol (ICMP) is used to handle errors and exchange control messages. ICMP can be used to determine if a machine on the Internet is responding. To do this, an ICMP echo request packet is sent to a machine [14]. If a machine receives that packet, that machine will return an ICMP echo reply packet. A common implementation of this process is the "ping" command, which is included with many operating systems and network software packages. ICMP is used to convey status and error information including notification of network congestion and of other network transport problems. ICMP can also be a valuable tool in diagnosing host or network problems [15].

On IP networks, a packet can be directed to an individual machine or broadcast to an entire network. When a packet is sent to an IP broadcast address from a machine on the local network, that packet is delivered to all machines on that network. When a packet is sent to that IP broadcast address from a machine outside of the local network, it is broadcast to all machines on the target network (as long as routers are configured to pass along that traffic).

In the "smurf" attack, attackers are using ICMP echo request packets directed to IP broadcast addresses from remote locations to generate denial-of-service attacks. There are three parties in these attacks: the attacker, the intermediary, and the victim [16].The intermediary receive an ICMP echo request packet directed to the IP broadcast address of their network. If the intermediary does not filter ICMP traffic directed to IP broadcast addresses, many of the machines on the network will receive this ICMP echo request packet and send an ICMP echo reply packet back. When (potentially) all the machines on a network respond to this ICMP echo request, the result can be severe network congestion or outages. When the attackers create these packets, they do not use the IP address of their own machine as the source address. Instead, they create forged packets

that contain the spoofed source address of the attacker's intended victim. The result is that when all the machines at the intermediary's site respond to the ICMP echo requests, they send replies to the victim's machine. The victim is subjected to network congestion that could potentially make the network unusable. Figure 4 depicts the stepwise process of the attack. Step 1: spoofed packet is crafted and sent to amplifier. Step 2: crafted packet received and forwarded to subnet/network. Step 3: ICMP broadcasts sent to the machines on the network. Step 4: Victim is now flooded with replies it has no clue about.

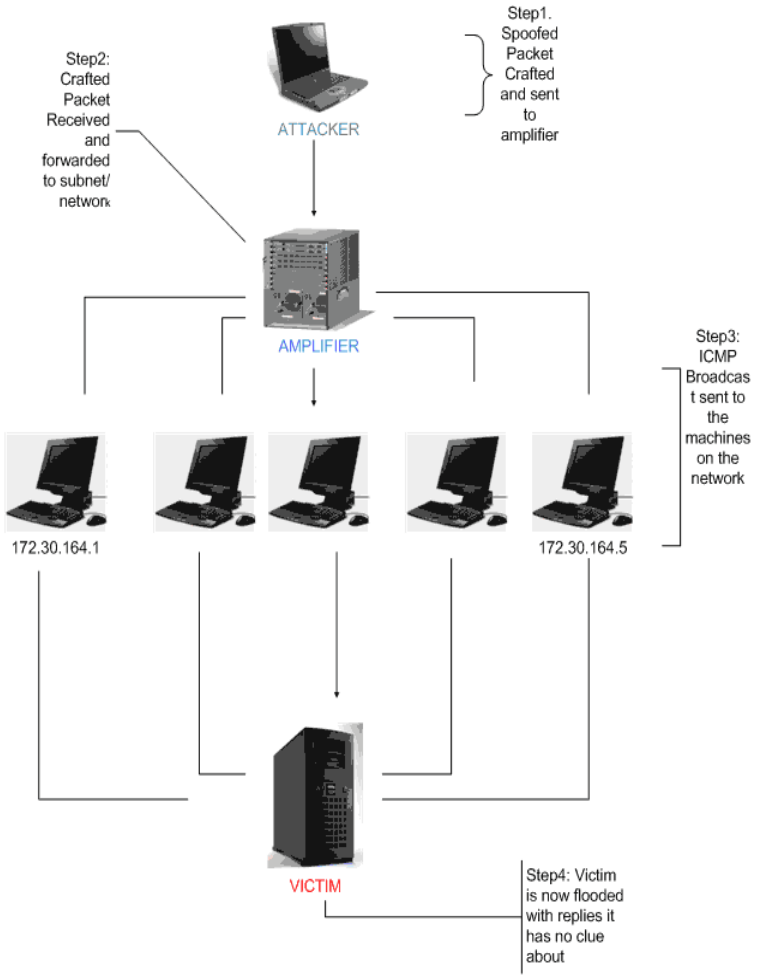


Figure 4: Smurf attack [18, 19].

Ping of Death attack: The Ping of Death attack relied on a bug in the Berkeley TCP/IP stack which also existed on most systems which copied the Berkeley network code. The ping of death was simply sending ping packets larger than 65,535 bytes to the victim. This denial of service attack was as simple as:

ping -l 86600 victim.org

Teardrop attack: This type of denial of service attack exploits the way that the Internet Protocol (IP) requires a packet that is too large for the next router to handle be divided into fragments. The fragment packet identifies an offset to the beginning of the first packet that enables the entire packet to be reassembled by the receiving system. In the teardrop attack, the attacker's IP puts a confusing offset value in the second or later fragment. If the receiving operating system does not have a plan for this situation, it can cause the system to crash [9]**.**

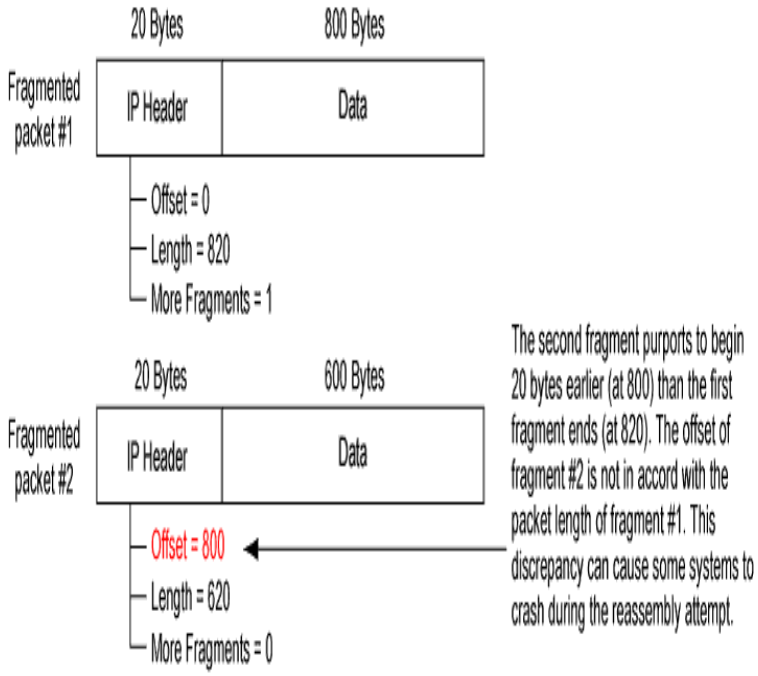


Figure 5: Teardrop Attack [20].

Land attack: The attack involves sending a spoofed TCP SYN packet (connection initiation) with the target host's IP address to an open port as both source and destination.

The reason a LAND attack works is because it causes the machine to reply to itself continuously.

Definition: "A *LAND* attack involves IP packets where the source and destination address are set to address the same device."

Other land attacks have been found in services like Simple Network Management Protocol (SNMP) and Windows 88/tcp (kerberos/global services) which were caused by design flaws where the devices accepted requests on the wire appearing to be from themselves and causing replies repeatedly [22].

1.3. DDoS attack tools

There are various methods employed by the attackers in the various distributed denial of service (DDoS) attacks. There are multiple tools available to accomplish this as discussed below.

Trinoo: The communication between the master and the slaves is using TCP, whereas the communication with the attack daemons is using the UDP packets. The attack implemented is UDP flood [23].

Tribe Flood Network (TFN): Communication between the master and the slaves is using ICMP echo reply packets. The attack could be of smurf, SYN flood, UDP flood and ICMP flood attacks [24].

TFN2K: This is the newer version of the TFN attack and it uses TCP, UDP, ICMP or all three to communicate between the master and the slaves and the communication is encrypted. The attacks implemented are the same as TFN [24].

Stacheldracht: It is based on the TFN attack. The communication between the master and the slave is encrypted and uses TCP and ICMP. In this case also the attacks implemented are the same as TFN [24].

Shaft: This attack is modeled after Trinoo. Communications between the master and the slave is achieved using the UDP packets and the attack implemented is the UDP flood attack. An important feature of this attack is its ability to switch control master servers and ports in real time, thereby making the detection by intrusion detection tools difficult.

HPING: Hping is a free packet generator and analyzer for the TCP/IP protocol. Hping is one of the de-facto tools for security auditing and testing of firewalls and networks, and was used to exploit the Idle Scan scanning technique now implemented in the Nmap port scanner. Like most tools used in computer security, hping is useful to security experts, but there are a lot of applications related to network testing and system administration. Hping is a command-line oriented TCP/IP packet assembler/analyzer. The interface is inspired to the ping(8) unix command, but hping isn't only able to send ICMP echo requests. It supports TCP, UDP, ICMP and RAW-IP protocols, has a traceroute mode, the ability to send files between a covered channel, and many other features. While hping was mainly used as a security tool in the past, it can be used in many ways by people that don't care about security to test networks and hosts. Hping can be used to do many things like[25]:

- Firewall testing.
- Advanced port scanning.
- Network testing, using different protocols, TOS, fragmentation.
- Manual path MTU discovery.
- Advanced traceroute, under all the supported protocols.
- Remote OS fingerprinting.
- Remote uptime guessing.
- TCP/IP stacks auditing.
- Hping can also be useful to students that are learning TCP/IP.

Hping works on the following unix-like systems: Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MacOs X, Windows. The versions of Hping Hping2 and Hping3 are now available. Hping2 is used in this thesis.

1.4. Features of internet that make DoS and DDoS possible.

Internet was designed with functionality and not security in mind. The TCP/IP protocol suite, the most widely used protocol suite for data communication assumes that all the hosts participating in the communication have no malicious intent. There is no security

built into the internet infrastructure to protect hosts from other hosts not regulating their own behavior. For example, the TCP protocol assumes that hosts will reduce the rate of packet transmission on detecting packet losses due to congestion. If a particular host instead does not respond to the congestion conditions, it can easily overwhelm the intermediate links to the destination. Such design opens up the internet to many opportunities for denial of service attacks[4].

Some features of the internet that make DoS/DDoS attacks possible are:

Internet Security is highly dependent: DDoS attacks are launched from hosts whose security has been subverted. No matter how secure a particular host is, it opens itself to the possibility of a DDoS attacks if there are other insecure hosts in the internet which can be used to launch such attacks.

Difficulty in tracing back the attack to the source: Most (if not all) of the internet runs on top of the TCP/IP protocol. The underlying protocol (IP) is basically connectionless in nature. At each intermediate step from the source to the destination, the decision about the next host to forward the packet is made. All such routing decisions are made on the basis of the destination address. It is thus possible to generate packets with incorrect source IP addresses and use them to launch Denial of Service attacks. This technique is known as IP spoofing. Users with sufficient privileges on a system have the ability to fabricate such fake packets. E.g in Linux, raw sockets can be created which enable users (with super user access) to construct all the packet contents and headers for a given packet.

This makes the task of determining the true source of attack very difficult. Apart from the source IP address, the attackers nowadays even randomly change all the headers in an IP datagram, keeping just the destination address constant. This makes dropping of packets based on certain characteristics very difficult, as distinguishing attack packets from legitimate packets becomes difficult. There are also some attacks that rely on illegitimate source addresses to launch a denial of service attack on the hosts whose IP address was used. The Smurf attack is one such example. If on detection of an attack, packets are

dropped solely on the basis of the IP source addresses, then the hosts whose IP addresses were used for the spoofing will suffer from a denial of service.

Limited Resources: The infrastructure of the interconnected hosts and networks is comprised of limited resources. Bandwidth processing power and storage capacities are all targets of Denial of Service attacks. If these resources are increased by substantial investments, it just raises the bar on the degree an attack must reach to be effective. Even if the attack is not able to shut down the victim completely, it may waste its resources, reducing the level of quality as seen by the end users, and making the service provider incur heavy financial losses.

A target rich environment: If the people in the military were to describe the internet today, they would describe it as a target rich environment. There are thousands and thousands of hosts and networks in the internet with vulnerabilities that can be exploited to get access to the machines there. It is therefore easy to gain control of a large number of hosts that can be then used as a spring board to launch DDoS attacks.

Easier to break systems than to make them: It is easier to break the networking infrastructure / protocols than to develop them. All the hosts in the internet, including the intermediate routers expect certain packet formats and traffic behavior. Since, at the time of the design of these software, no one foresaw the system being used for malicious purposes. This can lead to unexpected behavior of the network systems as response to unexpected packets. E.g all routers and hosts allocate buffers in memory while waiting for all the fragments constituting a datagram to arrive. If such a router is sent badly formed fragments indicating a very large datagram, the router will keep on allocating huge amounts of memory till it runs out of memory and cannot process more datagrams. Similar results can be achieved with fragments that indicate negative offsets within its datagram.

1.5. Roots of DoS and defense challenges.

Looking back to the evolution of the Internet, one can explain how the Internet has become ubiquitous. Most of the R&D efforts in the past had been on improving the performance and scalability of Internet protocols. However, little attention was paid to

securing these protocols. This resulted in several vulnerabilities that were exploited to adversary's advantage. Adversaries have always abused the following characteristics of Internet protocols to perform DoS attacks [30].

- Destination oriented routing: The routing protocols were designed to be destination oriented. Packet forwarding is the main task of Internet routers. A router places a packet on the interface that takes it to the next hop on its way to the destination, without bothering about where it came from.
- Stateless nature of the Internet: Routers do not maintain any state information about forwarded packets. Because of this, accountability and computer forensics are difficult to conduct.
- Lack of authenticity over the Internet: Without authentication, malicious Internet users can claim the identities of other users without being easily detected or located.
- Deterministic nature of Internet protocols: This is not a design flaw, but is often necessary to the proper operation of Internet protocols (e.g., TCP connection establishment procedure and TCP congestion control mechanisms).

The first three characteristics have greatly facilitated the task of source address spoofing, which is widely used in DoS attacks. Source address spoofing alters a packets source address so that the packet appears to have come from a source other than the actual sender. An attacker uses source address spoofing for two reasons: to gain access to resources that only accept requests from specific source addresses, or to hide the source of an attack. The fourth characteristic of the Internet contributes directly to a class of DoS attacks that exploits the predictable operation of Internet protocols.

In general, four fundamental challenges contribute to the difficulty of the DoS problem.

Hard prevention: Software bugs, misconfiguration, and human's inherent vulnerability to social engineering all enable a remote attacker to cause a resource to be unavailable to legitimate users [30]. For instance, an attacker may exploit a bug in a packet queue implementation that enables a specially crafted packet to set tail = head, effectively rendering queue capacity to zero.

Weak deterrents: Sending attack packets is almost free, user authentication in order to identify attack sources is hard, and it is easy for the real attack perpetrators to hide behind an army of unwitting zombies that launch the attack on their behalf.

Difficult resource control: Effective resource management and protection ensures that entities (e.g., users and processes) be allocated their resource shares and no entity can gain more than its share. A challenge here is the definition of entities. Entities can be processes, sockets, IP addresses, user tokens, etc. The granularity of an entity determines how much control is available and how much overhead is incurred. For instance, enforcing fair resource allocation between open TCP sockets consumes less resources than enforcing fair-sharing between IP addresses, because the number of open sockets is usually much less than the IP address space[35]. Controlling at the socket level, however, is vulnerable to an attacker with a single IP address gaining all socket resources and leaving nothing for other users. IP spoofing and application-layer proxies reduce the effectiveness of using the IP address as an identifier. A finer-grain control (e.g., based on application-level cookies) leads to a huge state overhead, especially in large services.

Difficult attacker identification: It is hard to come up with a silver-bullet definition of malicious entities to block their access to attacked resources. Many attempts have been made to define characteristics of a malicious entity: sends one-way traffic [27], sends more traffic than normal [26], sends more "heavy" requests than normal [29], appears on system logs only during periods of attack [28], and is controlled by bots (non-human agents controlling compromised machines). These attempts are either specific to particular services or can be bypassed by new attacks. For instance, defense systems that detect non-human bots are only suitable for Web services and other services with "direct" human users. Also, defense systems that detect network-level anomalies are bypassed by service-level attacks that target server resources without causing congestion at the network level.

1.6. Linux Netfilter Firewall

The netfilter/iptables IP packet filtering system is a powerful tool that is used to add, edit and remove the rules that a firewall follows and consists of while making packet filtering

decisions. These rules are stored in special-purpose packet filtering tables integrated in the Linux kernel. Inside the packet filtering tables the rules are grouped together in what are known as chains. The netfilter/iptables IP packet filtering system, although referred to as a single entity, is actually made up of two components: netfilter and iptables.

The netfilter component, also known as kernelspace, is the part of the kernel that consists of packet filtering tables containing sets of rules that are used by the kernel to control the process of packet filtering.

The iptables component is a tool, also known as userspace, which facilitates inserting, modifying and removing rules in the packet filtering tables. By using userspace, you can build your own customized rules that are saved in packet filtering tables in kernelspace. These rules have targets that tell the kernel what to do with packets coming from certain sources, heading for certain destinations or have certain protocol types. If a packet matches a rule, the packet can be allowed to pass through using target ACCEPT. A packet can also be blocked and killed using target DROP or REJECT. There are many more targets available for other actions that can be performed on packets [31].

The rules are grouped in chains, according to the types of packets they deal with. Rules dealing with incoming packets are added to the INPUT chain. Rules dealing with outgoing packets are added to the OUTPUT chain. And rules dealing with packets being forwarded are added to the FORWARD chain. These three chains are the main chains built-in by default inside basic packet-filtering tables. There are many other chain types available like PREROUTING and POSTROUTING and there is also provision for user-defined chains. Each chain can have a policy that defines “a default target”, i.e. a default action to perform, if a packet doesn't match any rule in that chain.

After the rules are built and chains are in place, the real work of packet filtering starts. Here is where the kernelspace takes over from userspace. When a packet reaches the firewall, the kernel first examines the header information of the packet, particularly the destination of the packet. This process is known as routing.

If the packet originated from outside and is destined for the system and the firewall is on, the kernel passes it on to the INPUT chain of the kernelspace packet filtering table. If the

packet originated from inside the system or another source on an internal network the system is connected to and is destined for another outside system, the packet is passed on to the OUTPUT chain. Similarly, packets originating from outside systems and destined for outside systems are passed on to the FORWARD chain.

Next the packet's header information is compared with each rule in the chain it is passed on to, unless it perfectly matches a rule. If a packet matches a rule, the kernel performs the action specified by the target of that rule on the packet. But if the packet doesn't match a rule, then it is compared to the next rule in the chain. Finally, if the packet doesn't match to any rule in the chain, then the kernel consults the policy of that chain to decide what to do with the packet. Ideally the policy should tell the kernel to DROP that packet. Figure 6 graphically illustrates this packet filtering process.

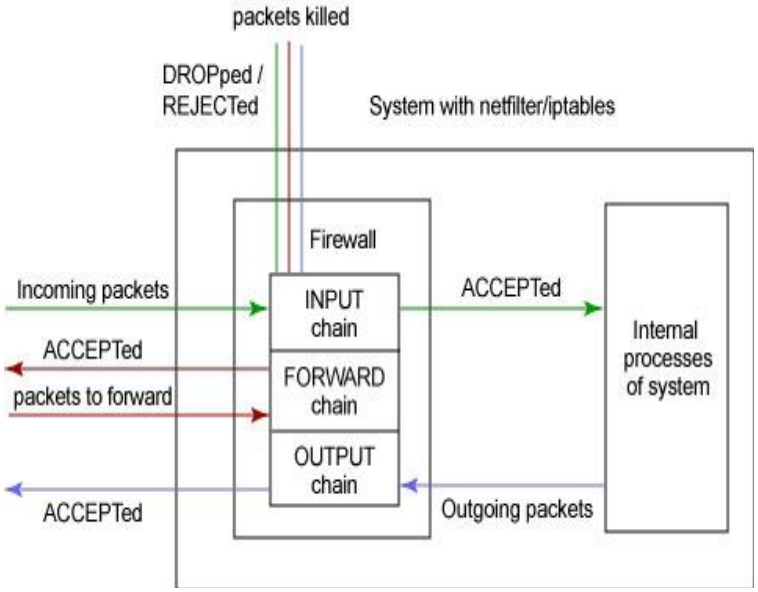


Figure 6: Packet filtering process [32].

Originally, the most popular firewall/NAT package running on Linux was ipchains, but it had a number of shortcomings. To rectify this, the Netfilter organization decided to create a new product called iptables, giving it such improvements as [33]:

- Better integration with the Linux kernel with the capability of loading iptables-specific kernel modules designed for improved speed and reliability.
- Stateful packet inspection. This means that the firewall keeps track of each connection passing through it and in certain cases will view the contents of data flows in an attempt to anticipate the next action of certain protocols. This is an important feature in the support of active FTP and DNS, as well as many other network services.
- Filtering packets based on a MAC address and the values of the flags in the TCP header. This is helpful in preventing attacks using malformed packets and in restricting access from locally attached servers to other networks in spite of their IP addresses.
- System logging that provides the option of adjusting the level of detail of the reporting.
- Better network address translation.
- Support for transparent integration with such Web proxy programs as Squid.
- A rate limiting feature that helps iptables block some types of denial of service (DoS) attacks.

  As it is faster and more secure alternative to ipchains, so iptables has become the default firewall package installed under RedHat and Fedora Linux.

2. Literature Review

2.1. The increasing threat of DoS attacks

DoS attacks are malicious means of denying Internet services to legitimate users or processes. These attacks continue to pose a significant threat for today's Internet as they are growing in number and sophistication. The recent attack incidents confirm the devastating effects of these attacks. For example, in October 2002 eight out of the thirteen root servers were significantly affected by a massive DoS attack [34]. Many other attacks were reported subsequently in 2003 and 2004 (e.g., [35, 36]). In a recent study by D. Moore et. al., [37], the DoS activity in the Internet was estimated using a method called "backscatter". The study found that nearly 4,000 DoS attacks are launched each week. Some systems studied were attacked as often as once per minute, usually with attacks of up to 1, 000 packets per second, though some attacks ran as much as 600, 000 packets per second.

The spread of attack tools and the easy access to them through search engines have contributed to the popularity and criticality of DoS attacks. When coupled with the script driven nature and widespread availability of attack tools, it is relatively simple for the average computer user to create a vast amount of disruption using limited resources. Moreover, attackers are becoming increasingly more creative and are using distributed computing techniques to multiply and amplify their damaging efforts. In fact, DoS attacks are developing more quickly than the defenses used to fight them. This explains why such attacks are easy to conduct, yet difficult to defeat.

Although any system connected to the Internet is considered to be a potential target, most of the reported DoS attacks are against high profile sites that include online gaming sites, Web hosting and to some extent against financial services. The outages caused by these attacks usually lead to significant financial losses. The motive behind launching DoS attacks varies across a spectrum of reasons including economical, political, and personal.

Despite the research efforts done to counter DoS attacks, the fear that future attacks could do even worse damage still exists. Therefore, it is evident that a lot of issues need to be addressed, and a more effective countermeasures need to be developed.

2.2. A glance to the past of DoS and DDoS attacks

In this paragraph, a few of the most notable events of the past regarding DoS attacks are briefly discussed. The information presented here is based on public sources only. In February 1996, CERT published an advisory regarding a User Datagram Protocol (UDP) port DoS attack [38]. In the history of CERT this was the first advisory issuing a clear DoS attack warning. In September 1996, CERT published another advisory alerting of a Transmission Control Protocol (TCP) SYN flooding DoS attack [39]. This particular attack was originally discussed in a well-known hacker magazine Phrack with the source code of the implementation of the attack included [40]. Since then this attack type has received a rather notorious reputation amongst the attacks in the Internet and it is probably the most used DoS attack of all time. Most DDoS attack tools have implemented this attack type and even specific countermeasures have been developed against it. One example of such a countermeasure is SYN cookies [41]. In December 1996, CERT published the third and the last DoS attack advisory of the year [42]. This time the advisory dealt oversized Internet Control Message Protocol (ICMP) echo request packets that when encountered could have halted certain operating systems. This particular attack is often referred as the “Ping of Death”. The year 1996 is a landmark in the history of DoS attacks. Certainly, there must have been flaws in software that might have resulted in DoS before and it is quite reasonable to assume that some underground circles might have known these attacks as well as many others long before public did. However, these three attack types are the first widely noted DoS attacks.

The year 1998 started with an advisory from CERT detailing a so-called “smurf” DoS attack [43]. Even as this particular attack is launched from a single node, it might result in multiple machines sending traffic to the target. Attacks like these are often referred as

amplified DoS attacks. These types of attacks are not equivalent to DDoS attacks. The potency of smurf attack proved to be real. Wired reported several Internet Service Providers (ISP) and Internet Relay Chat (IRC) administrators having serious problems getting their networks online shortly after the year change in January 1998 [44]. A few months later CNET reported that the University of Minnesota was suffering also of a smurf DoS attack (Festa 1998). In July 1999 CERT published reports noting widespread exploitation of RPC vulnerabilities [45,46]. Later on, it was found that in many cases this exploitation phase was followed by DDoS attack tool installation. It was this advisory that first time described two DDoS attack tools publicly [47]. These tools were Trin00 and Tribe Flood Network (TFN). Between these two TFN was more evolved regarding attacking capabilities. Trin00 could mount only a UDP flood, whereas TFN had UDP, ICMP and TCP SYN flood attacks as well as a smurf attack capabilities built-in. In addition, the advisory had a statement, which in a way gave an insight what to expect in the near future, "These tools appear to be undergoing active development, testing and deployment in the Internet." [47]. Only a month later CERT published another advisory describing a new DDoS attack tool named Tribe FloodNet – 2k Edition (TFN2K) [48]. TFN2K was the improved follower of TFN coming from the same author. Even as it contained several new ways to flood the target that were not present at the predecessor, the other new functionality proved to be more Most dramatic improvements were aimed at rendering communication between nodes in DDoS network more obfuscated [49]. The years 1998 and 1999 are significant in the history of DDoS, since during those years the first publicly available DDoS attack tools, such as the previously mentioned Trin00, TFN and TFN2k were developed. During 1998 before these tools were available in the Internet there were conceivably other DDoS attack tools already built, such as fabi and BlitzNet. The exact chronological evolution of DDoS attack tools could not be verified, but in any case, tools that might have existed before Trin00 and TFN did not spread around the Internet as they did. The early DDoS attack tools were probably being developed and known only in small underground cracker and hacker circles. The year 2000 started with another CERT advisory describing a new DDoS attack tool named Stacheldraht and

discussing the developments in the field of DDoS [50]. Stacheldraht, in a similar fashion to TFN was another improved DDoS attack tool based on Trin00 and original TFN. The main improvement was the addition of encryption to communication. This feature was present in the TFN2K as well. In addition, Stacheldraht provided automated update of the agents. A feature yet unseen in the DDoS attack tools. In February 2000, something previously unseen in the history of the Internet occurred. The wave of massive DDoS attacks began. Among many other news sites, BBC News reported that Yahoo! was brought down for three hours [51]. A day later eBay, Buy.com, CNN.com, Amazon.com were all under heavy DDoS attacks as reported by Seattle Post-Intelligencer (2000). The duration of these attacks was similar to the Yahoo! attack and thus let experts to believe they were connected. The magnitude of these attacks was something completely unexpected. For instance, it was noted that in some cases the overall amount of incoming data was over a Gigabit per second [52]. The frenzy the Yahoo! attack bred was certainly justified; if the company who had probably the greatest web resources could be taken down then anything can be taken down. The events escalated into the media and DDoS attacks received the attention they should have received a long time ago. In 2002, a new DDoS attack tool named PUD appeared in the Internet [53]. This particular tool uses a custom coordination method, which is based on peer-to-peer ideology to control its instances. The coordination method is novel to DDoS attack tools and it is much more robust and scalable compared to the traditional derivatives of the agent-handler model. The same coordination method with identical source code was adopted into the Slapper worm [54], which targeted Apache web servers operating on Linux.

Many devastating DDoS attacks have kept occurring in the Internet even after the events of February 2000, although the DDoS attack technology has not evolved much. For instance, in January 2002 one particular attack resulted in a drastic aftermath as a British Internet service provider was forced to close its doors [55]. During the year 2002 also occurred probably the most severe DDoS attack in the history of the Internet, as the thirteen root DNS servers were simultaneously targeted and seven of them were knocked out of the Internet [56]. In March 2003, the Arab satellite television network Al-Jazeera

was forced out of the Internet and some speculation was expressed that the attack might have had something to do with the U.S led war in Iraq. The year 2003 also hinted of the vast power the worm and the DDoS attack technology together hold, as the Blaster worm managed to infect more than 1.4 million computers worldwide [57]. The purpose of the original Blaster worm was to initiate a DDoS attack against the Microsoft, but the attack traffic never reached the original target as was initially designed by the attacker. During the year 2003 it also became apparent that spammers were starting to use DDoS attacks to disable anti-spam blacklists as a means of spreading spam more efficiently [58]. That way the ones who relied on those particular sites on filtering incoming spam were more open to spammers again. Overall, a lot has happened during the past few years in the DDoS field as a whole. DDoS technology has evolved and attacker motivation and identity has faced a shift as well; it is not reasonable anymore to expect that only young misfits, often referred as script-kiddies use DDoS technology in their personal vendettas and quests for glory. The considerably large power of DDoS attacks have been noted everywhere, which is verified by the events discussed previously. Recently the event of businessman hiring a cracker group to perform DDoS attacks against three of his competitors emphasized the notion that today DDoS attacks are being used as tools of achieving more fine-tuned criminal objectives as well [59].

2.3. Targets of DoS.

The targeted victim of a DoS attack can be an end system, a router, an ongoing communication, a link or an entire network, an infrastructure, or any combination of or variant on these. In the case of an end system, the targeted victim can be an application, or an operating system. Note that the term end system corresponds to the terms “Internet host”, “end host”, or simply “host”, where an end host or host is a computer that implements all five layers of TCP/IP protocol stack [60].

DoS on application: In application DoS attacks, an attacker attempts to prevent the application from performing its intended tasks by causing the application to exhaust the finite supply of a specific resource. For example, in an eXtensible Markup Language

(XML) parser DoS attack called Exponential Entity Expansion attack (also known as Billion Laughs attack), an attacker passes to an XML parser a small XML document that is both well-formed and valid, but expands to a very large file [61]. When the parser attempts to parse the XML, it ends up consuming all memory available to the parser application. Usually, the resources for applications are constrained by configuration, such as the maximum number of processes and the maximum number of simultaneous connections that an application can create, to limit the impact of an application DoS on the entire operating system. However, if such limits are not chosen carefully based on the role of a machine (e.g., aWeb server, a database server, or a personal computer etc.), important applications may become the easy targets of DoS.

DoS on operating system: Operating system DoS attacks are very similar to application DoS attacks. However, in application DoS attacks, the operating system may be able to protect other applications from being effected; whereas the problem can be more catastrophic in the case of operating system DoS attacks. A very well-known DoS attack on an operating system is the Transmission Control Protocol (TCP) SYN flooding [62], in which an attacker sends a flood of TCP SYN packets to the victim without completing the TCP handshake, and exhausting victim's connection state memory. Such an attack effects all applications in the operating system that relies on TCP for their communication.

DoS on router: Many of the DoS attacks against an end system can also be launched against an IP router. Additionally, routing protocols can be used to stage a DoS attack on a router or a network of routers [63]. This requires the ability to send traffic from addresses that might plausibly have generated the relevant routing messages. The simplest attack on a router is to overload the routing table with sufficiently large number of routes that the router runs out of memory, or the router has insufficient CPU power to process the routes. More serious DoS attacks on routers that use false route updates can cause blackholing of an entire network address block.

DoS on ongoing communication: Instead of attacking the end system, an attacker may attempt to disrupt an ongoing communication. If an attacker can observe a TCP

connection, then it is relatively easy to spoof packets to either reset that connection or to de-synchronize it so that no further progress can be made [64]. Even if an attacker cannot observe a TCP connection, but can infer that such a connection exists, it is still possible to reset or de-synchronize that connection by sending large number of spoofed TCP reset packets that guess the TCP port number and TCP sequence number.

DoS on links: The simplest form of DoS attack on links is to send enough non-congestion controlled traffic (e.g., UDP traffic) such that a link becomes excessively congested, and legitimate traffic suffers unacceptably high packet loss [63]. Congesting a link might also cause a routing protocol to drop an adjacency if sufficient routing packets are lost, potentially amplifying the effects of the attack. Moreover, it may be possible for an attacker to deny access to a link by causing the router to generate sufficient monitoring or report traffic such that the link is filled. Simple Network Management Protocol (SNMP) traps are one possible vector for such an attack, as they are not normally congestion controlled.

DoS on infrastructure: Many communication systems depend on some underlying infrastructure for their normal operations. Such an infrastructure can be as large as a global domain name system or a global public key infrastructure, or can be as small as a local area ethernet infrastructure or a wireless access point. Effects of infrastructure attacks on the users of that infrastructure can be enormous. For example, Domain Name System (DNS) serves as the phone book for the entire Internet by translating human-friendly hostnames into IP addresses. Denying access to a DNS server effectively denies access to all services, such as Web, email, AFS, public keys and certificates etc, that are being served by that DNS server. The larger the zone a DNS server is responsible for, the bigger the impact of a DoS attack. All 13 root DNS servers were subjected to a very large-scale DoS attack in 2002 [65]. As a result, some root name servers were unreachable from many parts of the global Internet. Some DoS attacks target a common infrastructure that is conjointly used by all hosts in a local network. For example, an attack with access to a subnet may be able to prevent other local hosts from accessing the network by simply exhausting the address pool allocated by a Dynamic Host

Configuration Protocol (DHCP) server [63]. Although such attacks require the ability to spoof MAC address of an ethernet or wireless card, it is quite feasible with certain hardware and operating systems.

DoS on firewalls and IDS: Firewalls are intended to defend the systems behind them against outside threats by restricting data communication traffic to and from the protected systems [66]. Firewalls may also be used in defending against denial of service attacks. Meanwhile, firewalls themselves may become the targets of DoS attacks. Firewalls can be categorized as stateful and stateless, based on whether the firewall holds state for the active flows traversing it, where a flow is a stream of packets sharing IP source and destination addresses, protocol field, and source and destination port numbers. Stateless firewalls generally can be attacked by attempting to exhaust the processing resources of the firewall. In addition to processing power exhaustion, stateful firewalls can also be attacked by sending traffic that causes the firewall to hold excessive state or state that has pathological structure [63]. In the case of excessive state, the firewall may run out of memory, and can no longer instantiate the state required to pass legitimate flows. For most firewalls this will cause denial of service to the systems behind the firewall, since most firewalls are fail-disconnected. In the case of pathological structure, an attacker sends traffic that causes the firewall's data structures to exhibit worst-case behavior. An example of this would be an algorithmic complexity attack on the firewall's forwarding state hash table [67]. Intrusion detection systems (IDSs) suffer from similar problems to that of firewalls. Unlike a firewall, an IDS is normally fail-open, which will not deny service to the systems protected by the IDS. However, it may mean that subsequent attacks that the IDS would have detected will be missed.

2.4. Distributed Denial of Service attack.

Probably the most common definition of a DDoS attack follows the idea of having multiple machines each deploying a DoS attack towards one or more targets [68]. Such a definition is almost correct, however, it fails to include the aspect of coordination between the attacking hosts, which is the most fundamental characteristic of a DDoS attack. For that reason a new definition was formulated. Distributed Denial of Service

(DDoS) attack is a DoS attack, in which a multitude of hosts performs DoS attacks in a coordinated manner to one or more targets. This definition emphasizes three important aspects. First, DDoS is essentially a DoS attack. More accurately, DDoS attacks are a subset of DoS attacks. Second, there must be more than one source attacking. Third, there must be coordination between the attacking hosts. In case either one of these conditions is not met the attack cannot be called as a distributed denial of service attack. In this study the abbreviation DDoS is often used to refer to distributed denial of service attacks. This is an important point to notice, as whereas there is such a thing as denial of service, there is no such a thing as distributed denial of service in the same sense; the service can be denied, but the service cannot be denied distributed unless the service itself is distributed. Only distributed denial of service attack can exist. The word "attack" is appended only when it is considered there is a chance of misunderstanding; For instance, DDoS tool could refer to both DDoS attack- and defence tools. Therefore, the clarifying word would be appended in that particular case. The possibility of arbitrary attackers randomly selecting the same host as a target should also be noted, as this event is not a DDoS attack, although it may seem to be. The view adopted in this research considers this possibility as what it in most simplistic level appears to be; separate attackers engaging in separate DoS attacks against the target, as contingency is hardly a form of coordination. Furthermore, an attack as a concept is singular in regard to its objectives, which means that even in case an attack is an aggregate of other attacks, all of the "sub attacks" strive for the shared objectives. Random target selection is not a shared objective. Therefore, in case multiple attackers each randomly select the same target the resulting attack cannot be reasonably stated as a shared attempt in achieving a shared objective amongst the attackers.

2.5. Distributed Denial of Service network.

DDoS is always about multiple DoS attacks targeted to one or more specific destinations. Coordination is a crucial part of DDoS. Coordination of multiple hosts in turn implies the existence of some sort of a network structure, which could be titled as DDoS network. However, for the purposes of this study, no suitable definition for such network was

found. Therefore, a tentative definition for DDoS network was created. DDoS network is a network of hosts that are being controlled by a same static entity using the same control interface to administrate DDoS attacks. The "control interface" refers to the specific methods the entity can use to control the hosts. Essentially, the control interface is sort of a protocol, which the network apprehends. The notion of the static entity, whether or not it consists of several subjects refers to the controller of the network. The combination of the control interface and the network controller serve as a network identifier. In case these qualifiers were not determined, it would be practically impossible to identify a network and state which hosts belong to which network. Furthermore, the purposes of the network must include the administration of DDoS attacks. Otherwise the definition would come overly broad, as it would cover many seemingly different computer networks as well.

2.6. Term definitions.

Host is a computer connected to a network [69]. This particular term is preferred and commonly used when referring to a networked computer. In this study this practise was followed.

DDoS attack software is a program or a set of programs that contributes in some way to the functionality of a DDoS network.

End-host is the last node or the node with the least amount of authority in a DDoS network responsible of attacking. In other words, end-host is the node that performs the actual attack as ordered by some other node with higher authority in the same network. DDoS networks most often consist of nodes with different tasks amongst each other. Certain nodes may only be coordinating other nodes, which ultimately inject the attack traffic. End-host is always the host responsible of injecting the attack traffic. DDoS networks usually consist of large number of end-hosts and may additionally consist of some other nodes responsible of various other tasks, such as the coordination of the end-host activity.

DDoS zombie / DDoS slave / DDoS daemon / DDoS host / DDoS agent is an endhost computer program. Depending of the author, most commonly one of these terms is used

to refer to the end-host program. Different terms could be used as well. In this study, *DDoS agent* was used, as it appeared to be quite commonly used.

DDoS server / *DDoS handler* / *DDoS master* is an intermediate computer program in the DDoS network, which controls a set of DDoS agents and is usually being directly controlled by a cracker or another intermediate computer program. As in the case of end-host programs, all these terms are commonly used in the literature. Again, other terms could be used as well. In this study, *DDoS handler* was used.

DDoS client is a computer program, which a cracker uses to control DDoS handlers and DDoS agents. These programs provide the user interface to the DDoS network.

Cracker is an individual, who by his / her action seeks either to cause damage or to gain an unauthorized access to a computer system or systems. Some define a cracker as an individual who attempts to gain an unauthorized access to a computer system [69]. Some others define a cracker as an individual who seeks to cause damage in computer systems. In this study the term cracker is used to refer to both.

Hacker is an individual who enjoys exploring the details of programmable systems and how to stretch their capabilities [69]. The term hacker is widely used and often many groups of people perceive it differently. Many mistakenly confuse the terms hacker and cracker with each other.

IP spoofing is a technique of masquerading a source address of the Internet protocol (IP) to appear as something else than it really is.

Packet is "the unit of data sent across a network."Packet" is a generic term used to describe a unit of data at any layer of the OSI protocol stack, but it is most correctly used to describe application layer data units.

Datagram is a "self-contained, independent entity of data carrying sufficient information to be routed from the source to the destination computer without reliance on earlier

exchanges between this source and destination computer and the transporting network" [69].

Worm is a program that propagates itself over a network and simultaneously reproducing itself [69]. The term is often used instead of *automated intrusion agent*, which is another term used occasionally to refer to a similar program.

Trojan is a standalone malicious file or program that does not attempt to inject itself into other files unlike a computer virus and often masquerades as a legitimate file or program. Trojan horses can make copies of them, steal information, or harm their host computer systems [69].

2.7. Issues relating to law and liability.

The aim of this paragraph is to take a brief glance to the most basic issues regarding cybercrime and DDoS. The motivations for this type of discussion are the facts that DDoS is a severe act of cybercrime, and as the actual attackers behind DDoS are rarely caught and the damages suffered can be significant the questions of law and liability become important issues. The victims may seek compensation from some other party regardless were the actual attackers caught or not. Moreover, the victims may be subsidized by a jurisprudentially valid case.

In this section the problems regarding cybercrime is presented and then some of the most important arguments of liability are briefly discussed. The purpose is not to provide legal advices. For such matter, a qualified attorney should be consulted.

2.7.1. Legislation and cybercrime

The issue of cybercrime has been troubling the legal environment for quite some time. One of the most profound problems lies in the controversial natures of technology and legislation. As Chen et al. eloquently put it, "It usually takes laws months, if not years, to be developed, approved and implemented. Technology, on the other hand, seems to change in a matter of days, or sometimes even hours." [70]. In addition, cybercrime by its very nature is highly global. The culprit of an attack occurred in the United States could

very well be somewhere in the Asia, which emphasizes that legislative cooperation between countries is a definite requirement to catch and prosecute cyber criminals accordingly. Unfortunately, global legislative cooperation appears to be still in its infancy with rough times ahead. Legislation regarding cybercrime may be inadequate in western countries; for all that, some developing countries may not have any legislation concerning the issue at all. Furthermore, some countries might even be harboring criminals, which it is not reasonable to expect that even the basis for global cooperation concerning this issue be in near prospects.

Besides the legal problems, acquiring adequate technology and resources for monitoring and upholding the law can be costly and problematic. Consequently, even if cybercrime legislation were in order the high economic costs to which many countries simply cannot afford may prove to be yet another voluminous obstacle to cross. Nonetheless, international cooperation is probably the only way to approach the problem efficiently.

2.8. Distributed Denial of Service network models

Many models can be adopted to do DoS attacks. Here we have discussed some of them.

2.8.1. Network models and model selection

The term DDoS network was quite loosely defined in chapter two to be a set of hosts that are controlled by the same entity, share the same control interface and which purpose is the administration of DDoS attacks. Ultimately DDoS network is as any other network of hosts, which communicate amongst each other and work for a common goal. The unique aspects of DDoS networks are that some members have to be capable of performing DDoS attacks and every member is controlled by the same static entity. Nevertheless, when considering suitable network models for DDoS the common network theories apply well for the most parts, since the purpose of the DDoS network model is to determine the locations and the relationships of its nodes. The malicious usage, however, does favors traits, which only some network models are able to provide. The dimensions between different network models are numerous, ranging from fast creation to efficient, comprehensive and secure communication. Since it is not practical to attempt to cover the

suitability of every network model to DDoS network administration the focus in this study is only in the four models contemporary publicly available DDoS attack tools use. These models are the agent-handler model, IRC-based model, scattered model. In this paragraph these network models and some of their variations are presented.

Agent-handler model: Most commonly DDoS networks are formed using the agent-handler model or some slight variation of it. The model is illustrated in Figure 7. In essence, the attacker communicates directly with one or more handlers, which communicate directly with the agents. The agents in turn perform the attacks.

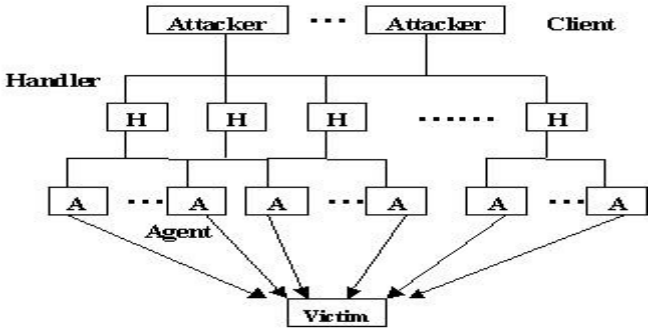


Figure 7: Agent-handler model [71].

IRC-based model: Topologically IRC-based model is a derivative of agent-handler model. However, as it has many notable properties due to IRC being the coordination medium that are not present in the basic agent-handler model it is discussed separately. The IRC-based model is illustrated in Figure 8. According to Oikarinen [72], "The server forms the backbone of IRC, providing a point to which clients may connect to talk to each other, and a point for other servers to connect to, forming an IRC network." Hence, IRC network is a collection of IRC servers around the Internet that together form a teleconferencing system. Oikarinen defines "client" as anything connecting to a server that is not another server. A client in IRC network is identified by a unique "nickname" that is at most nine letters long. Oikarinen defines "channel" as a named group of one or more clients, which will all receive messages addressed to that channel. Therefore, in

principle, a client connects to a server that relays client's messages to other servers, which host the other clients the message was addressed. These servers then forward the message to the clients they are hosting. If thought in terms of agent-handler model, the handlers in IRC-based model are the IRC servers that the attacker commands through an IRC server. The IRC servers relay the commands to the agents, which are hosts connected to the IRC servers. The IRC-based model could be seen as a derivative of the basic agent handler model with the handler depth being two and the first handler level containing a single handler, being the "handler of handlers" or master handler in other words. With this handler the attacker communicates. However, it is important to note that it is not restricted in anyway what IRC servers the agents can use. The agents could be connected to the same IRC server (master handler) as the attacker.

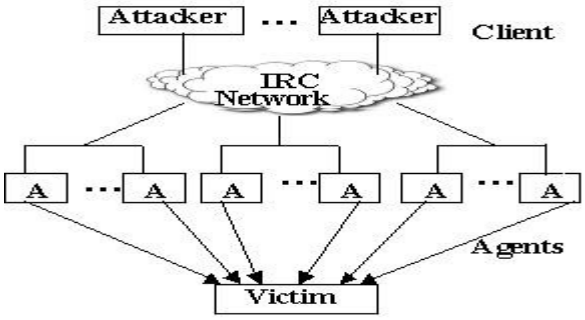


Figure 8: IRC-based model [71].

DDoS attack tools use IRC and its channels as a meeting point for agents and attackers. In essence, the attacker implements the minimal features required to communicate appropriately with an IRC server along with the packet flooding functionality into the agents. The DDoS agent in this scenario is a simple IRC client that connects to an IRC server, joins a specified channel and begins waiting for commands. The attacker sends messages to the channel following a format the agents understand. All agents in the channel receive all the messages sent to the channel and act only in case the message

meets the criteria required. For instance, the criteria could be a regular expression matching the nickname of the agent. DDoS attack tools that use IRC as a communication backbone emerged publicly in 2001.

The model has several major advantages over the basic agent-handler model:

- Traffic observation is difficult.
- IRC provides working communication backbone with a lot of functionality.
- Dynamic handler change is an easy and fast operation.

Traffic observation is difficult mostly due to the usage of common IRC ports and the popularity of the IRC. IRC is definitely one of the most used virtual chatting environments today and therefore traffic destined for established IRC ports, such as TCP 6667-6669 are usually allowed to pass firewalls, routers and other devices capable of traffic filtering. Moreover, due to the global popularity of the IRC the amount of IRC traffic is likely to be quite large almost regardless of the observation point. Pinpointing the few messages attackers send to the channels is overly difficult without some additional information of what exactly to search. Furthermore, intercepting traffic streams known to belong to a popular chatting environment, especially without evidence of malice, brings privacy issues forth. Based on that, it seems that the IRC as a communication backbone is a fine way to masquerade the control traffic of DDoS networks. IRC provides working communication backbone with a lot of functionality. This lessens the implementation overhead of the agent code to the attacker and provides plenty of usable functionality. A good example is the channels of the IRC, which make controlling the agents extremely easy, as the attacker is only required to send messages to the channel. Dynamic handler change can be accomplished by connecting to a different IRC server, which makes it a very easy and fast operation. Changing handlers dynamically makes detecting the presence of DDoS networks more difficult and hence aggravates the network seizure, as the server operators have less time to notice anything possibly suspicious.

However, the IRC-based model has few negative aspects as well:

● Large message propagation area.

● Non-existent protection against network seizure and sensitive information disclosure in case of presence exposure.

● Limited scalability.

As stated in the specification of the IRC protocol [72], messages are broadcast and processed at all IRC servers that host any of the clients to whom the messages were addressed. Added to that, clients receive every message destined to the channels they have joined. This type of message propagation could compromise information confidentiality and increase the possibility of DDoS network seizure, since there are numerous possibilities where a third party could intercept the DDoS control messages. As an illustration, consider that a third party has obtained a copy of the agent binary. Executing the binary and consequently capturing the network traffic the agent sends and receives will give the third party information of the identity of the attacker (the nickname and the IP address from which the attacker is connected to the IRC) and of all operations the attacker commences using that particular channel or channels the agent is connected. Once the third party has learned the details of the IRC servers, channels, channel keys and so on, the third party can seize the network by obtaining control of the channels the attacker uses. Anyone who is able to send messages to the channels is practically in control of all the agents in those channels, assuming the commands the agents responds to are known. Setting the channel modes “secret”, “key” and “moderated” on [72] is in essence all that can be done to protect the channel from unauthorized viewing, access and usage. In addition, IRC implements a channel mode “invite-only”, which can be used to grant channel access to only those that have been specifically invited to join the channel by their nickname. The downside is that this option forces explicit invitation of each agent to the channel, which is not completely problem-free either. The moderated flag will prevent unprivileged users sending messages into the channel even if they were able to join the channel. Hence, the flag could prevent the loss of control of the agents. However, even in presence of moderated channel, a successful channel intruder receives channel messages normally and thus the intruder might be aware of actions the attacker

performs if the attacker commands the participants of the channel publicly. As a minimum measure, it is thus essential to retain the channel key as confidential to maintain some kind of information confidentiality. The problem with the channel key is that it either must be inserted into the agent code or dynamically transmitted to the agents and furthermore, it must be transmitted in clear text to the server when joining the channel. From this follows that there are multiple possibilities of learning the channel key and subsequently gaining access to the channel. Statically coded string is easily obtained from the object code and network traffic capture would reveal unencrypted channel key. Thus, it would be preferable that the agent executable would never be inspected by a third party, which suggests that choosing agents for the network is a careful operation to assure the owners of the compromised hosts are unlikely to notice the misuse.

A significant problem the IRC-based model faces is its limited scalability. Even though the IRC specification does not state an upper bound for the number of clients a channel can hold, large channels should not be used in controlling DDoS networks. A channel with several thousand participants is likely to raise the interest of IRC operators who are responsible of administrating IRC servers. Dividing agents amongst multiple channels as well as multiple servers is a solution, but raises control overhead. Consequently, very large DDoS networks are hard to administrate using this model. So IRC can be used to have large number of compromised machines called zombies, creating botnets (discussed in 2.9), and attacking victims through them.

Scattered model: Scattered model differs vastly from the other contemporary DDoS network models due to its complete lack of intercommunication, which means that the nodes of the scattered network are completely unaware of each other. Even the attacker most often has no knowledge concerning the location of the nodes. Topologically the nodes are arbitrarily located and they are linked only to the target via the attack traffic, otherwise there is usually no other traffic originated. In other words, a DDoS network following the scattered model is build of independent agents. From this follows that coordination of such DDoS network has to be handled through other means, such as

using static instructions placed in the agent code or via “drop-zone” -based approaches that do not require intercommunication. Scattered model is illustrated in Figure 9.

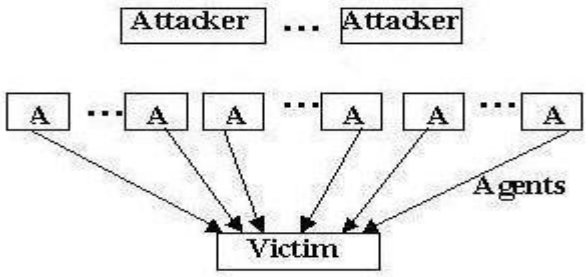


Figure 9: Scattered model [71].

The scattered model has emerged through constantly increasing use of automated intrusion agents (AIA), which are often referred as computer worms. There are very few reasons why attackers would choose this model without automating the intrusions, as will be demonstrated ahead.

To simple purposes the plain scattered model using static coordination appears suitable. Some of the attributes the model has include

- Low implementation overhead,
- Low risk of disclosing the identity of the attacker,
- Significant difficulty of network seizure,
- High scalability and
- Rapid network creation with AIA technology.

The differences and the characteristics of these models were discussed in this paragraph and the main advantages and disadvantages of each model are summarized into Table 1.

Model	*Advantages*	*Disadvantages*
Agent-handler	• Rapid and efficient message propagation, • well defined structure and • enables partial mobilization	• Maintenance overhead, • increased possibility of identity exposure and • high handler dependence
IRC-based	• Traffic observation is difficult, • provides a working backbone and much functionality as well as • dynamic handler change is easy and fast	• Futile message propagation, • poor protection against network seizure and disclosure of sensitive information as well as • limited scalability
Scattered	• Implementation is easy, • low risk of identity exposure, • significant difficulty of network seizure, • rapid network creation using AIA technology and high scalability	• Does not scale to sophisticated purposes, • short life span and • generally low usability

Table 1: The Advantages and Disadvantages of the DDoS Network Models.

2.9. Botnets

These days, online computers, especially those with a high bandwidth connection, have become a desirable target for attackers. Attackers can gain control of these computers via direct or indirect attacks. Direct attacks refer to sending packets containing a malicious payload that exploits a vulnerable computer, for example, an unpatched Windows home PC. Generally, these attacks are conducted via automated software so that the number of compromised computers can be maximized in a short period. The requirement for launching direct attacks is that publicly available services on the targeted computers contain software vulnerabilities. For example, the Blaster Worm spread by exploiting a vulnerability in the Remote Procedure Call (RPC) service [73], which allowed malicious code to be executed in the remote host. Unfortunately, this kind of vulnerability occurs frequently and has been increasing. According to CERT-2011 [74], the number of vulnerabilities reported in 2005 was 5,990, which is 35 times the number in 1995. Statistics as shown in Figure 10 shows the number of vulnerabilities reported each year after 2005.

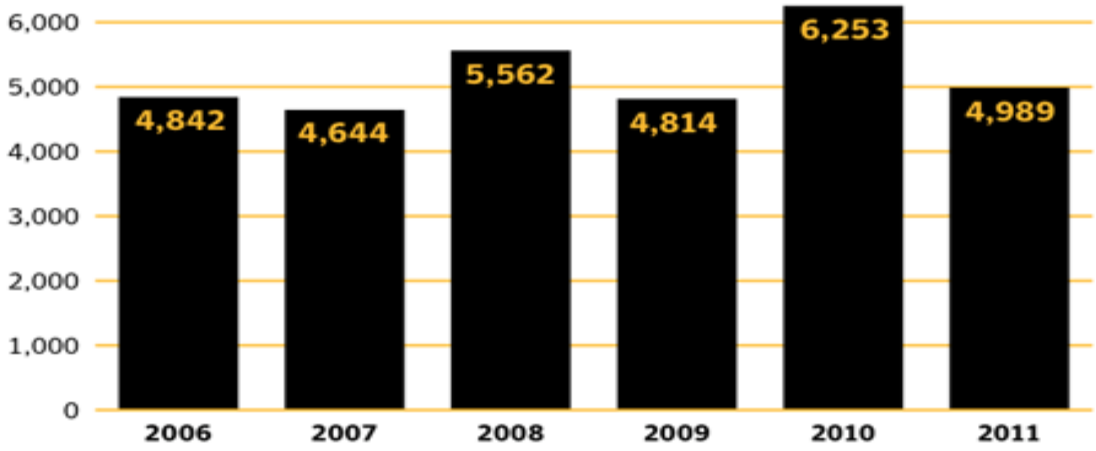


Figure 10: The number of vulnerabilities reported each year according to CERT [74].

In contrast, indirect attacks can exploit insecure actions that may be performed by users. These attacks generally require human interaction. For example, users are decoyed to open a malicious HTML file that exploits a vulnerability in Microsoft Internet Explorer, or to install free software with malicious software embedded. Once these attackers have compromised a computer, they install a "bot", which is another name for a "zombie". The term bot (derived from the word robot) is used in industry jargon to describe an automaton or automated process in both the real world and the computer world. A bot generally supports a communication channel with the attacker, as well as the ability to execute particular tasks, for example, launching DoS attacks, according to the attacker's instructions.

2.9.1. Botnet communication

A common way for attackers to control the bots is to use Internet Relay Chat (IRC) channels. IRC is a form of real-time communication over the Internet. It is mainly designed for group (many-to-many) communication in discussion forums called channels, but also allows one-to-one communication. Once installed in the compromised computers, the bot will automatically join a specific IRC channel on an IRC server, and wait there for further instructions. These compromised computers that can be managed by

the attacker through the IRC channel are called a botnet. In fact, IRC channels are not the best solution for an attacker to communicate with the bots in terms of efficiency and robustness. With increases in botnet size, IRC channels are likely to be congested. Moreover, relying on IRC servers for communication creates a single-point-of-failure. In fact, removing the IRC server used by the botnet has proved to be an effective DoS attack defense approach. There are two main reasons that explain why IRC-based bots are so popular. First, IRC servers are freely accessible to the public, and they are easy to set up. Second, many attackers are familiar with IRC communication [75]. However, future botnets can use non-IRC-based communication, for example, by making use of decentralized and encrypted peer-to-peer communication.

2.9.2. Botnet function

Botnets can be used for a wide variety of purposes. Nevertheless, DDoS attack capability is a common feature of botnet software [75]. Generally, each type of botnet software contains a set of flooding mechanisms, such as SYN flood, ICMP flood, and HTTP flood. A set of sophisticated configuration commands are provided to control the attack parameters, such as sending rate and packet size. Another important feature of botnets is the ability to update software from a remote server. In this way, an attacker can fix existing software bugs and add new functions into the botnet software. For example, an attacker can instruct all bots to download a new type of flooding mechanism to defeat a DDoS protection system. Hence, the botnet owner has the capability to design a specific attack for a particular target, and maximize the similarity between attack traffic and legitimate traffic. Attackers are now using open source software development methodologies to improve the effectiveness of botnet software by making it easier for multiple contributors to develop and test new software features.

2.10. Wireshark – A packet sniffing tool

The Wireshark is an open source network protocol analyzer, otherwise known as packet sniffer. It can capture live network traffic or read from a file and translate the data to be presented in a format the user can understand. Network Analyzers such as Wireshark are

invaluable tools for administrators to diagnose and troubleshoot problems with, but are also used by intruders to obtain unauthorized information [76].

Wireshark isn't an intrusion detection system. It will not warn when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Wireshark might help you figure out what is really going on. Wireshark will not manipulate things on the network, it will only "measure" things from it. Wireshark doesn't send packets on the network or do other active things (except for name resolutions, but even that can be disabled).

Distributions of Wireshark are available for a wide range of UNIX and Linux platforms as well as Windows. To actually capture the packets from the network requires a packet capture library like WinPcap (on Windows) or Libpcap (on Linux). Wireshark is sponsored by CACE, developers of the WinPcap library. The packet driver used will vary depending on the exact UNIX, Linux or Windows platform running Wireshark on. It has a GUI front-end, and many more information sorting and filtering options. It allows the user to see all traffic being passed over the network by putting the network interface into promiscuous mode. Given below in Figure 11 is the Wireshark in action.

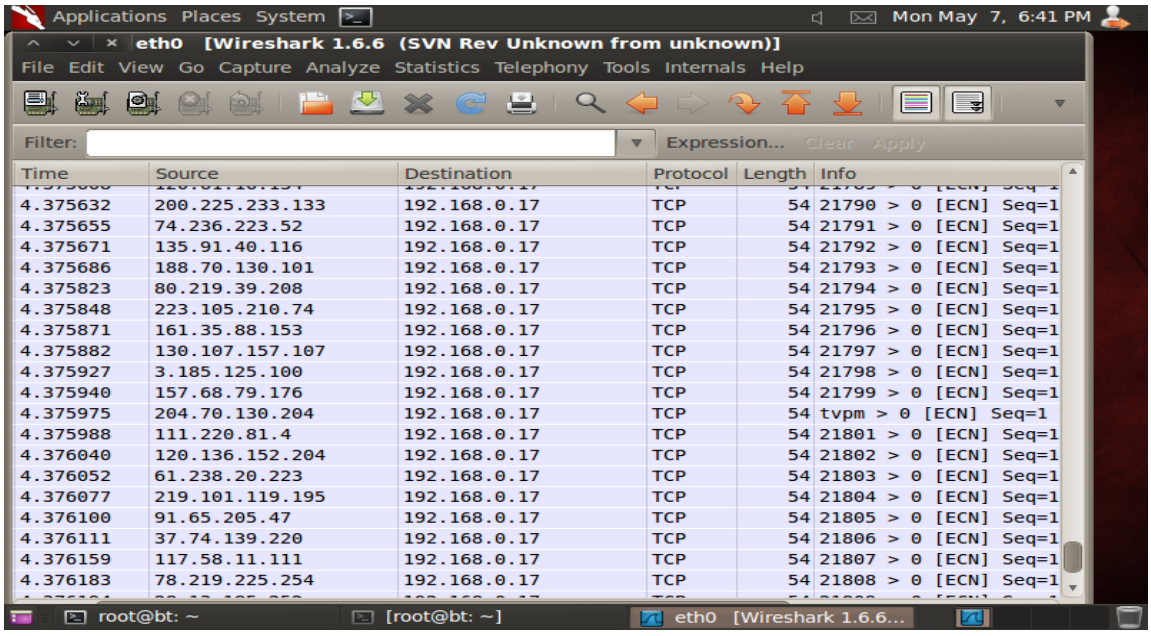


Figure 11: Wireshark - Network Protocol Analyzer.

3. Problem Statement

DoS/DDoS attacks are a big threat to the Internet. They decrease the service quality of Internet services. It matters because the Internet is now becoming a critical resource whose disruption has financial implications or even dire consequences on human safety. An increasing number of critical services are using the Internet for daily operation. A DoS attack may mean losing a bid on an item you want to buy or losing your customers for a day or two while you are under attack, therefore it is important to have means to prevent or at mitigate them.

There is a constant arms race between the attackers and the defenders. As soon as there are effective defenses against a certain type of attack, the attackers change tactics finding a way to bypass this defense.

In addition, since absolute protection is not feasible, developing effective defense framework involves an often complicated set of trade-offs.

Objectives:

1. To study and investigate various DoS and DDoS attack vectors.
2. To establish workbed to launch and study DoS and DDoS attacks.
3. To design and develop a stateful technique to countermeasure DoS and DDoS.
4. To verify and validate attack vectors and technique developed experimentally.

4. Implementation details and results

In this chapter work has been done on creating the environment to perform DDoS attacks using packet generating tool Hping and also by creating botnet using IRC-based model and a Trojan named Illusion BOT. All the experimentation was performed in the real environment by connecting different machines through a switch. Then using the network protocol analyzer Wireshark capturing of Live traffic on victim machine was done. Then offline analyses of these network data packets was done and after that on the basis of analyses on the packets the defense framework to log/block/allow the network traffic using IPtables is purposed and after blocking traffic further capturing and analyzing of live traffic using Wireshark to verify that our defense framework works to mitigate the DoS attacks remarkably.

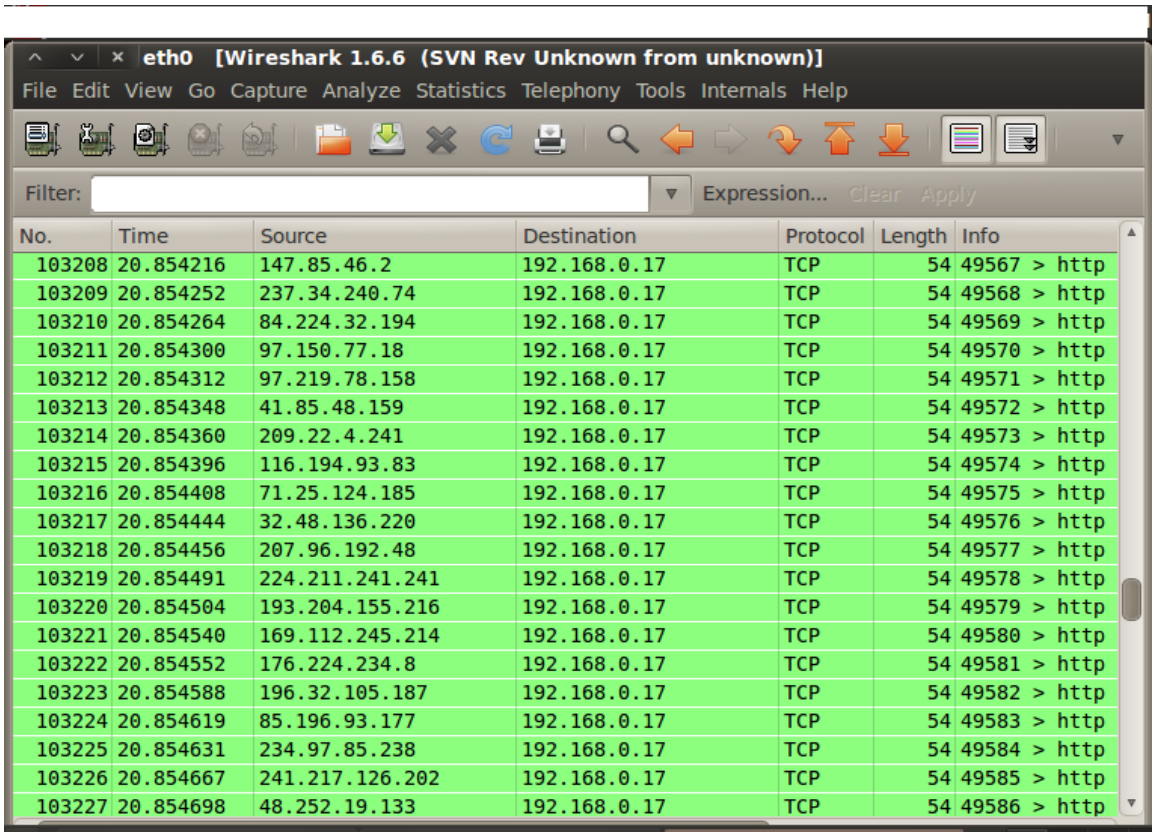
4.1. Implementation setup and experimentation results.

- Set up four machines and installed with following operating systems:
  - **Windows xp**
  - **BackTrack 5r1**
  - **CentOs 5/Redhat 5/Redhat 6**
  - **Window 7/Window 8**
- All the machines were connected using Cisco switch Catalyst Express 500 series.
- DHCP server was set up on the machine with Redhat 5. All the machines were provided with the ip addresses within range 192.168.0.1 to 192.168.0.64.
- Network protocol analyzer tool wireshark (version 1.6.6) was installed on all the machines to capture all the live traffic.
- IRC server(UnrealIRC 13.2.9) was set up on the machine with window 7 operating system having ip address 192.168.0.21.
- IRC clients were set up on the machines having Window xp and Window 8.
- BackTrack 5r1 was installed on the attacker machine(192.168.0.29) and CentOs5/Redhat5/Redhat6 on the victim's machine(192.168.0.17).

- Using Hping2 installed by default in BackTrack, victim's machine was flooded with:
- SYN flood using command:

  #hping2 –i u10 –S –p 80 - -rand-source 192.168.0.17: using **u10** with **i** ten SYN flag packets in one micro second and **- -rand-source** is used to use random source addresses that is spoofed ones.
- ICMP flood using command:

  #hping2 –i u10 -1 –rand-source 192.168.0.17: using **-1** hping2 sends icmp request packets.
- Using network traffic analyzer wireshark, attack traffic was captured on the victim's machine. Given are Figure 12 and Figure 15 showing SYN flood and ICMP flood snapshots on the victim as captured by wireshark respectively.



No.	Time	Source	Destination	Protocol	Length	Info
103208	20.854216	147.85.46.2	192.168.0.17	TCP	54	49567 > http
103209	20.854252	237.34.240.74	192.168.0.17	TCP	54	49568 > http
103210	20.854264	84.224.32.194	192.168.0.17	TCP	54	49569 > http
103211	20.854300	97.150.77.18	192.168.0.17	TCP	54	49570 > http
103212	20.854312	97.219.78.158	192.168.0.17	TCP	54	49571 > http
103213	20.854348	41.85.48.159	192.168.0.17	TCP	54	49572 > http
103214	20.854360	209.22.4.241	192.168.0.17	TCP	54	49573 > http
103215	20.854396	116.194.93.83	192.168.0.17	TCP	54	49574 > http
103216	20.854408	71.25.124.185	192.168.0.17	TCP	54	49575 > http
103217	20.854444	32.48.136.220	192.168.0.17	TCP	54	49576 > http
103218	20.854456	207.96.192.48	192.168.0.17	TCP	54	49577 > http
103219	20.854491	224.211.241.241	192.168.0.17	TCP	54	49578 > http
103220	20.854504	193.204.155.216	192.168.0.17	TCP	54	49579 > http
103221	20.854540	169.112.245.214	192.168.0.17	TCP	54	49580 > http
103222	20.854552	176.224.234.8	192.168.0.17	TCP	54	49581 > http
103223	20.854588	196.32.105.187	192.168.0.17	TCP	54	49582 > http
103224	20.854619	85.196.93.177	192.168.0.17	TCP	54	49583 > http
103225	20.854631	234.97.85.238	192.168.0.17	TCP	54	49584 > http
103226	20.854667	241.217.126.202	192.168.0.17	TCP	54	49585 > http
103227	20.854698	48.252.19.133	192.168.0.17	TCP	54	49586 > http

Figure 12: SYN flood on 192.168.0.17 as captured by wireshark (random source addresses).

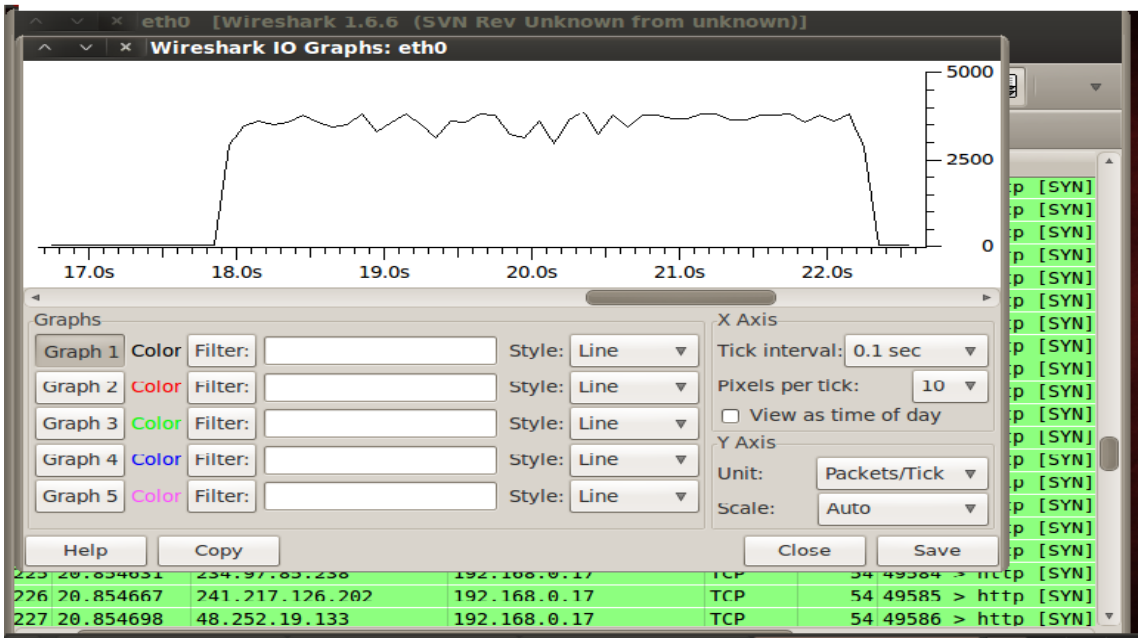


Figure 13: IO graph generated by wireshark showing duration for which the attack was performed(17.8 sec to 22.3 sec).

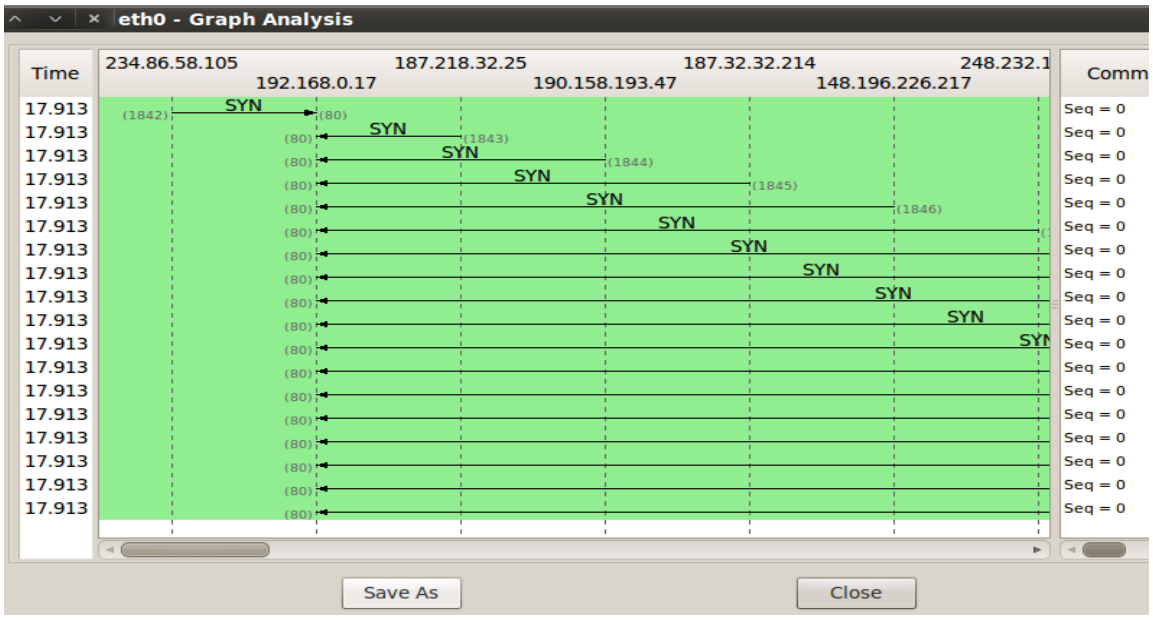


Figure 14: Flow graph of SYN flood generated by wireshark.

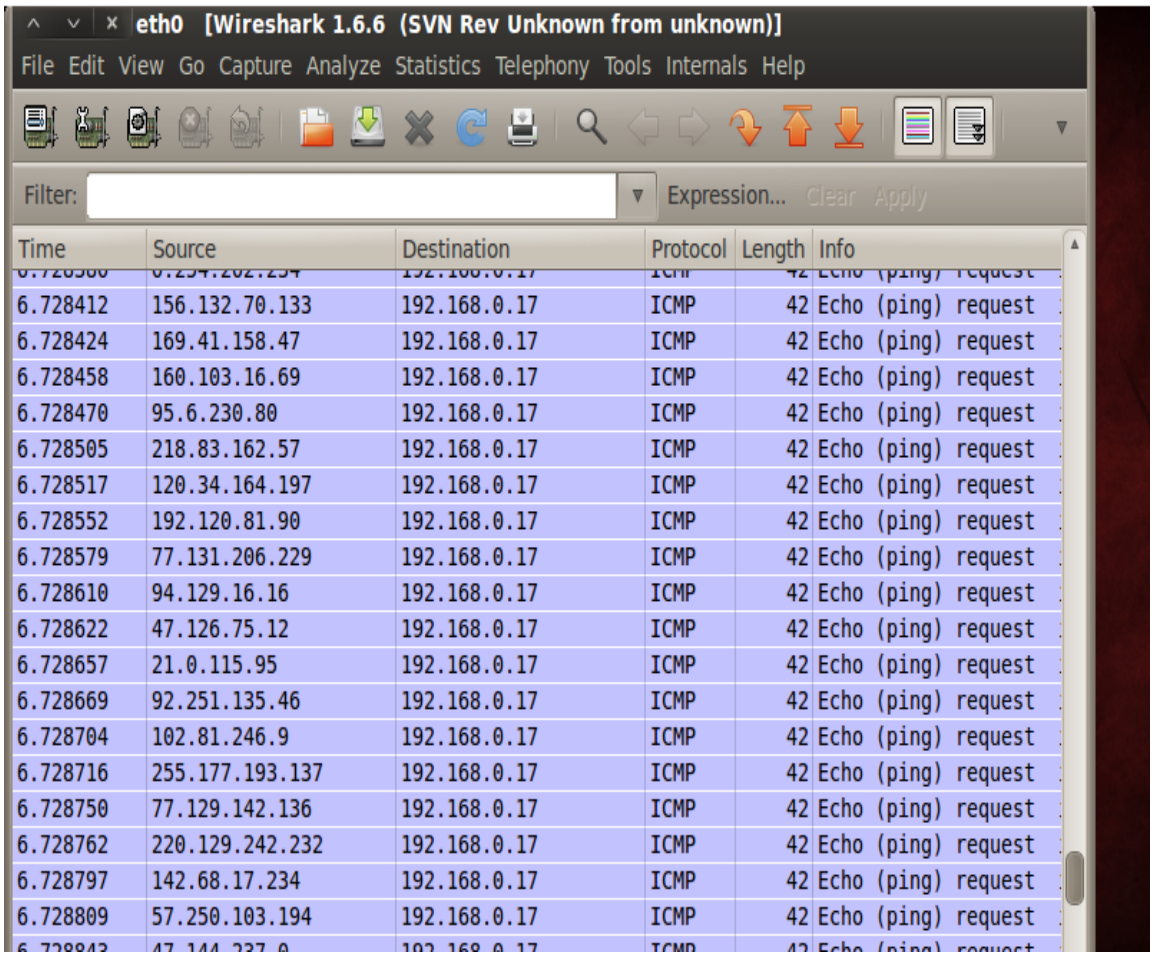


Figure 15: ICMP flood on 192.168.0.17 as captured by wireshark (random source addresses).

- IRC server (Dahiya.com) started on window 7 machine with ip 192.168.0.21 and the IRC clients on window xp and window 8 were connected Ravinet IRC network to the server joining them on the same channel named casual.
- Illusion BOT Trojan was build using Illusion Maker with the configuration shown in Figure 16.

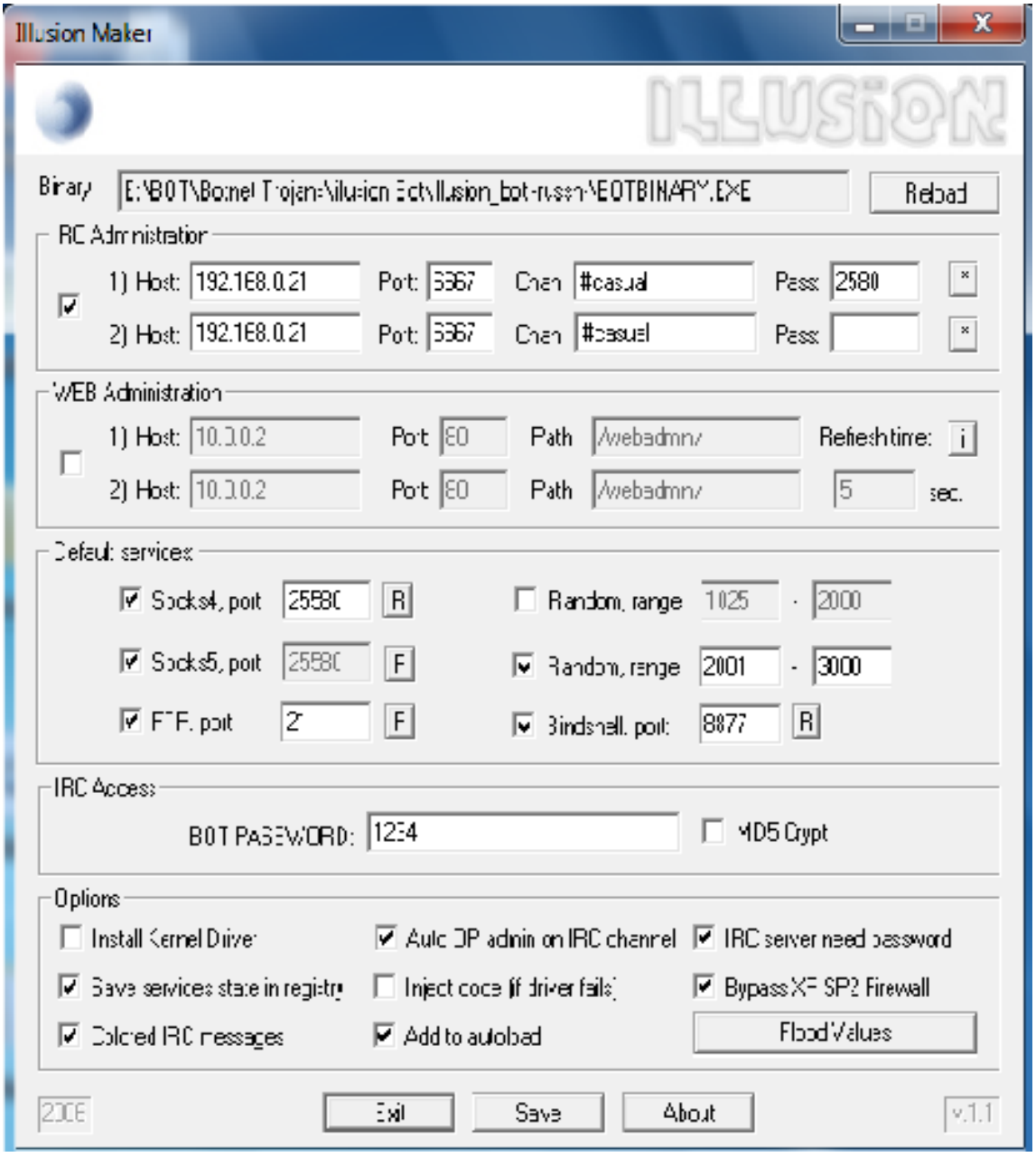


Figure 16: Illusion Maker to create Trojan BinaryBot.

- Trojan was executed on the window 7(192.168.0.21) and window xp(192.168.0.29). Through the Trojan client R5 on window 8(192.168.0.26) was able to get hold on window 7 and window xp and started SYN flood attack on 192.168.0.1 through them as shown in Figure 17.

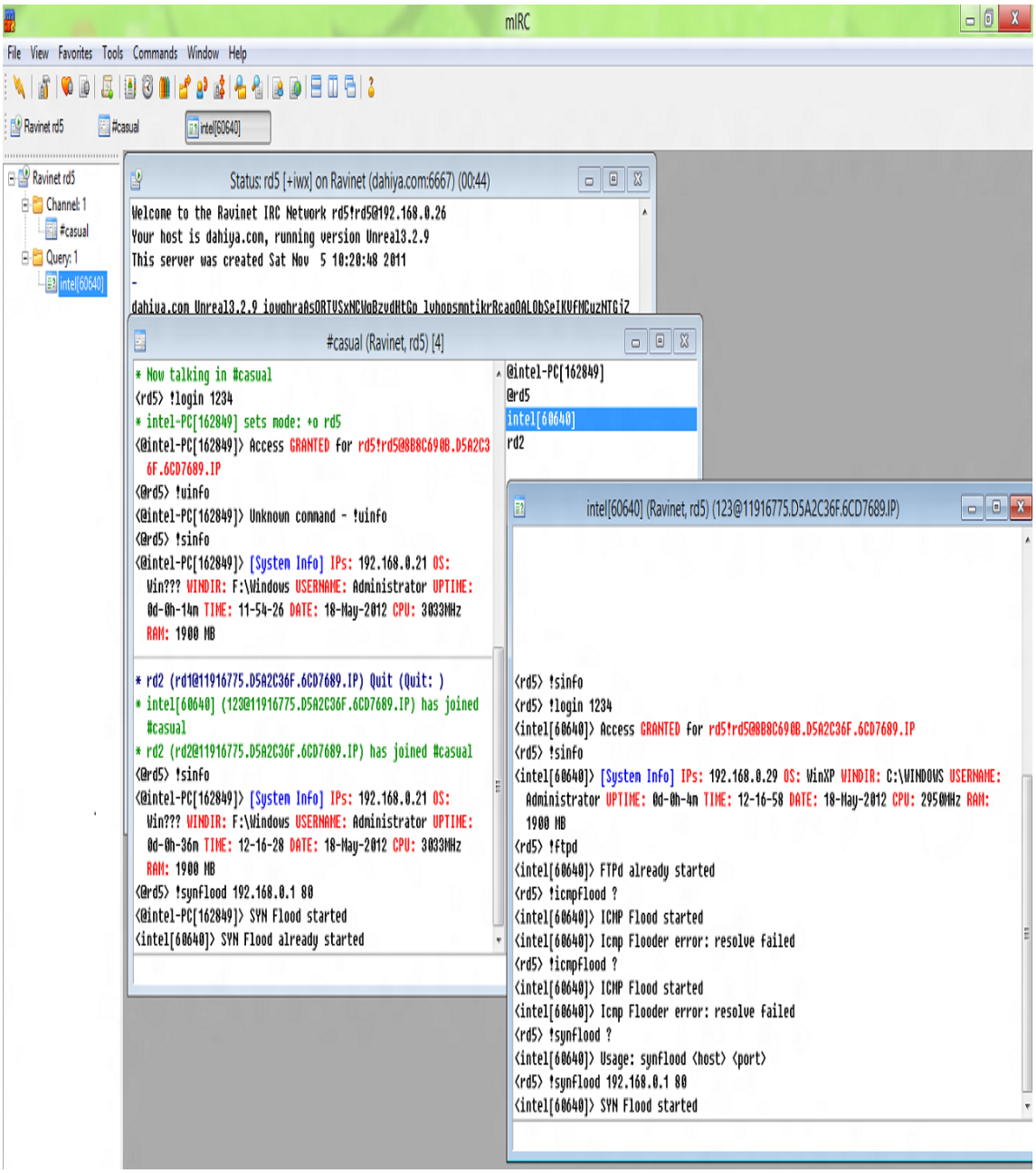


Figure 17: IRC client rd5 performing SYN flood through window 7(intel-PC[162849]) and window xp(intel[60640]).

- After performing the attacks some of the iptables rules were applied on the victim machine 192.168.0.17 and 192.168.0.1 having CentOs 5 and Redhat 5 installed respectively, to remarkably mitigate these attacks.

First rule : Limit NEW traffic on port 80:

iptables -A INPUT -p tcp - -dport 80 -m state - -state NEW -m limit - -limit 50/minute - -limit-burst 200 –j ACCEPT

Lets break that rule down into intelligible chunks.

-p tcp - -dport 80 : Specifies traffic on port 80.

-m state NEW : This rule applies to NEW connection.

-m limit - -limit 50/minute - -limit-burst 200 -j ACCEPT : This is the essence of preventing DOS. - -limit-burst is a bit confusing, but in a nutshell 200 new connections (packets really) are allowed before the limit of 50 NEW connections (packets) per minute is applied.

Similarly, limit can be imposed on ICMP requests.

Second rule: Limit established traffic:

iptables -A INPUT -m state - -state RELATED,ESTABLISHED -m limit - -limit 50/second - -limit-burst 50 -j ACCEPT.

This rule applies to RELATED and ESTABLISHED all traffic on all ports, but is very liberal (and thus should not affect traffic on port 22 or DNS). If one understood the 1st rule, one should understand this one as well.In summary, 50 ESTABLISHED (and/or RELATED) connections (packets really) are allowed before the limit of 50 ESTABLISHED (and/or RELATED) connections (packets) per second is applied.

Third rule: Log the input traffic:

iptables -A INPUT -j LOG - -log-level 4.

Similarly we can apply these rules on output chain also according to the requirements.

- These ips were logged as well(experiment was performed on Redhat based, Centos). First of all set your Log Daemon to log the IPTABLES

  vi /etc/syslog.conf

  Add the following line at the end of the file

  kern.warning/var/log/iptables.log

  touch/var/log/iptables.log

Restart the System Log Service

/etc/init.d/syslog restart (On Redhat based,CentOs)

- Then the logged ips were added to TCPWRAPPER (/etc/hosts.deny). Command to extract only the source ip from the log files:

  /bin/cat /var/log/iptables.log | awk '{print $9}' | cut -f2 -d "=" >>/root/rd.txt

  The above line will grep the SOURCE ip from the log and append to rd.txt. Now the ips not matching the rules indicated in the iptables firewall will be dropped but will be logged in rd.txt.

- Next to run this command as a frequent interval with the help of CRON

  vi/etc/crontab

  add following line in the file:

  */1 * * * * root /bin/cat /var/log/iptables.log | awk '{print $9}' | cut -f2 -d "=" >>/root/rd.txt

  Here the script will run in every minute. The file will grow up rapidly to heavy size if your server has heavy traffic. So clean up the file in frequent intervals.

- Next to add these IPs in the hosts.deny file

  vi/etc/hosts.deny

  add following line in the file:

  SSHD:/root/rd.txt

  Now the ips can be denied by the system.

5. Conclusion and Future Scope

5.1. Conclusion

As we have seen, distributed DoS attacks are a genuine threat that cause serious damage to many Internet users. The losses being suffered have escalated from being merely annoying to actually being debilitating and disastrous for some users. As long as DDoS attacks prove effective in achieving such aims, attackers are likely to continue using them. Until we find a reasonable defense against some kinds of DDoS attacks, we should expect to see their incidence, power, and seriousness increase because network bandwidth, processor speed, and number of available systems that can be attacked and compromised all continue to increase, as does the sophistication of attacker tools for compromising computers and using them to attack. Many different defense mechanisms are typically needed to mitigate DoS attacks, but it is not cost-effective to blindly choose a large set of defense mechanisms against DoS attacks. In this work DDoS attacks are performed both by using automated tool as well as by setting up the IRC based model. Also the capability of firewall is explored to defend against this attack.

To determine whether the network traffic is legitimate or not, a firewall relies on a set of rules it contains that are predefined by a network or system administrator. These rules tell the firewall whether to consider as legitimate and what to do with the network traffic coming from a certain source, going to a certain destination, or having a certain protocol type.

In this thesis, work has been done on capturing the live traffic using the network protocol analyzer Wireshark and on the basics of analyzed data packets, some rules using IP tables to allow/deny the network traffic on the basis of the traffic rate by any IP address of the computer sending the packets were listed.

Some of the system files(of Redhat and CentOs) have been explored which can help mitigate against DDoS Attack as they can be used to log the traffic coming to the system. Also it has been proposed how to deny the suspicious traffic on the basis of the content of the packet sent by the attacker by using other system files.

5.2. Future scope

The Netfilter/IPtables system is ideal for Linux system administrators, network administrators, and home users who want to configure firewalls according to their specific needs, save money on firewall solutions, and have total control over IP packet filtering hence even though this thesis provide ample defense against DoS attack, many areas of defense mechanism remain for further research.

In this thesis work it could not experiment the advanced features of IPtables such as NAT, IP masquerading, packet redirect, IPtables has the ability to REDIRECT packets like IP Chains does, however it also has a generalized DNAT feature that allows arbitrary changing of the destination IP address and port number. Thus, it can actually disguise where packets of a given service go.

Also, in this thesis we could not experiment the IPtables scripts that can be proposed for DDoS attack

Another active area of research could be to study other Trojans used by attackers to build botnets like NetBot Attacker. Also the DDoS tools like PoisionIVY , Shark and Sprut DoS tool could be studied that how they are used by attackers to get hold of other systems. Along with them DoSHTTP and LOIC for HTTP flood attack could be studied and analyzed to explore how the stuff works.

In this thesis work the sniffer that was used to capture the live traffic was wireshark. There are many other sniffers available that could be used instead of wireshark like Colasoft Capsa, CommonView, NetworkView, OmniPeek Network, PromiScan etc.

Also instead of IRC administration, WEB host administration can be used to build botnet.

References

[1] CERT Coordination Center, "Denial of Service Tools," *http://www.cert.org/advisories/CA-1999-17.html*,15 March, 2005.

[2] CERT Coordination Center, "IP Denial-of-Service Attacks," *http://www.cert.org/advisories/CA-1997-28.html*, 15 March, 2005.

[3] A. Chakrabarti and G. Manimaran, "Internet Infrastructure Security: A Taxonomy," in Proc. IEEE Network, vol.16, no.6, pp.13-21, Nov/Dec. 2002.

[4] Cisco Systems, "Distributed Denial of Service Attacks", The Internet Protocol Journal, Volume 7, 4 November, 2005.

[5] "Yahoo on Trail of Site Hackers", Wired.com, Feb 8, 2000. *http://www.wired.com/news/business/0,1367,34221,00.html*, 15 May, 2003.

[6] David K. Yau, John C. S. Lui, and Feng Liang, "Defending Against Distributed Denial of Service Attacks with Max-min Fair Server-centric Router Throttles," Quality of Service, 2002 Tenth IEEE International Workshop, pp. 35-44, 2002.

[7] Lincoln Stein & John Stewart, "WWW Security FAQ: Securing Against Denial of Service," *www.w3.org/Security/Faq/wwwsf6.html*, 02 Feb, 2003.

[8] David Karig and Ruby Lee, "Remote Denial of Service Attacks and Countermeasures," Princeton University Department of Electrical Engineering Technical Report CE-L2001-002, October 2001.

[9] Stephen M. Specht and Ruby B. Lee, "Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures," Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems, 2004 International Workshop on Security in Parallel and Distributed Systems, pp. 543-550, Sept. 2004.

[10] Paul J. Criscuolo. "Distributed Denial of Service Trin00, Tribe Flood Network, Tribe Flood Network 2000, And Stacheldraht CIAC-2319," Department of Energy Computer Incident Advisory Capability (CIAC), UCRL-ID-136939, Rev.1, Lawrence Livermore National Laboratory, 14 Feb, 2000.

[11] Kevin J. Houle, "Trends in Denial of Service Attack Technology," CERT Coordination Center, Carnegie Mellon Software Engineering Institute. Oct 2001. *www.nanog.org/mtg-0110/ppt/houle.ppt*, 14 Mar, 2003.

[12] "SYNFlood," *http//:en.m.wikipedia.org/wiki/SYN_flood.*

[14]TFreak,"smurf.c,"*www.phreak.org*Oct1997, www.phreak.org/archives/exploits/denial/smurf.c, 6 May, 2003.

[15] Martin, Michael J., "Router Expert: Smurf/Fraggle Attack Defense Using SACLS", Networking Tips and Newsletters,*www.searchnetwork.techtarget.com*,Oct,2002. *http://searchnetworking.techtarget.com/tip/1,289483,sid7_gci856112,00.html* May 6, 2003.

[16] Thomas Dubendorfer & Arno Wagner, "Past and Future Internet Disasters: DDoS attacks: Survey and Analysis," a talk in the seminar on "Security Protocols and Applications," April 8, 2003.

[17] "DDoS," *http://www.egyhacks.net/2011/01/what-is-ddos-attack-and-how-does-it.html*.

[18] "Smurf IP Denial-of-Service Attacks," CERT Advisory CA-1998-01, *http://www.cert.org/advisories/CA-1998-01.html*, 5 January, 1998.

[19]"SmurfAttack,"*http//:en.m.wikipedia.org/wiki/Smurf_attack.*

[20]"UnderstandingTeardropAttacks,"*http://www.juniper.net/techpubs/software/junos-es/junos-es93/junos-es-swconfig-security/understanding-teardrop-attacks.html#id33015*.

[21] Yu, C.-F. and Gligor, "A formal specification and verification method for the prevention of denial of service," IEEE Symposium on Security and Privacy, 187-202, 1988.

[22] "Land Attack," *http://en.wikipedia.org/wiki/LAND*.

[23] David Dittrich, "The DoS Project's 'trinoo' distributed denialofserviceattacktool,"*http://www.securityfocus.com/templates/archive.pike?list=1&date=1999-121&msg=Pine.GUL.4.20.9912071041410.9470-100000@red7.cac.washington.edu*.

[24]David Dittrich, "The 'Tribe Flood Network' distributed denial of service attack tool, "*http://www.securityfocus.com/templates/archive.pike?list=1&date=1999-1201&msg=Pine.GUL.4.20.9912071044490.9470-100000@red7.cac.washington.edu*.

[25] "HPING," *http://HpingActiveNetworkSecurityTool.htm.*

[26] J. Yan, S. Early, R. Anderson, "The XenoService – A distributed defeat for distributed denial of service," In *Proceedings of ISW 2000*, October 2000.

[27]Arbor Networks, "PeakFlow DoS for Hosting Providers Datasheet," *http://www.arbornetworks.com/up_media/up_files/PFDoS_ServProv_1.6.pdf*.

[28] D. Moore, H. Xiao, "Cisco quality of service and DDoS," *http://www.mitre.org/support/papers/tech_papers_01/moore_cis co/index.shtml*.

[29] F. Lau, S. H. Rubin, M. H. Smith, and Lj. Trajkovic, "Distributed denial of service attacks," In *Proceedings of 2000 IEEE International Conference on Systems, Man, and Cybernetics*, October 2000.

[30] Cisco, "Strategies to protect against Distributed Denial of Service Attacks," *http://www.cisco.com/warp/public/707/newsflash.html*.

[31] "Iptables," http://*www.linuxsecurity.com/resource_files/firewalls/IPTables-Tutorial/iptables-tutorial.html*.

[32] "IptablesTutorial1.2.2," *http://iptables-tutorial.frozentux.net/iptables-tutorial.html*.

[33]"QuickHOWTO,"*http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO.pdf*.

[34]BizReport, "DDoS Attacks Still Pose Threat to Internet,"Nov. 2003.*http://www.bizreport.com/news/5413/*, 10 February, 2005.

[35] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *ACM* Computer Communications Review (CCR), 31(3), July 2001.

[35] R. Lemos, "Attacks disrupt some credit card transactions," Sep. 2004, *http://news.zdnet.com/2100-1009 22-5378217.html*, 10 February, 2005.

[36] J. Leyden, "Online payment firm in DDoS drama," Nov. 2004. *http://www.theregister.co.uk/2004/11/03/protx ddos attack/*, 10 February, 2005.

[37] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial of Service Activity," in Proc. USENIX Security Symposium, Washington D.C., August 2001.

[38] CERT Coordination Center, "CERT Advisory CA-1996-01 UDP Port Denial-of Service Attack," *http://www.cert.org/advisories/CA-1996-01.html*,Feb 1996.

[39] CERT Coordination Center, "CERT Advisory CA-1996-21TCP SYN Flooding and IP Spoofing Attacks," *http://www.cert.org/advisories/CA-1996-21.html*,Sep 1996.

[40] Phrack Magazine, "ProjectNeptune," *http://www.phrack.org/phrack/48/P48-13,* no. 48, file 13, Sep. 1996.

[41] Bernstein D., "SYN cookies*", http://cr.yp.to/syncookies.html*, Sep. 1996.

[42] CERT Coordination Center, "CERT Advisory CA-1996-26 Denial-of-Service Attack via Ping," *http://www.cert.org/advisories/CA-1996-26.html, Dec. 1996.*

[43] CERT Coordination Center, "CERT Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks," *http://www.cert.org/advisories/CA-1998-01.html,* Jan. 1998.

[44]GlaveJ.,"SmurfingcripplesISPs,"*http://www.wired.com/news/technology/0,1282,9506,00.html*, Wired News, 7 Jan., 1999.

[45] CERT Coordination Center, "CERT Incident Note IN-1999-04," *http://www.cert.org/incident_notes/IN-99-04.html*, Dec 1999.

[46] CERT Coordination Center, "CERT Incident Note IN-1999-05," *http://www.cert.org/incident_notes/IN-99-05.html*, Dec 1999.

[47] CERT Coordination Center, "CERT Incident Note IN-1999-07," *http://www.cert.org/incident_notes/IN-99-07.html*, Dec 1999.

[48] CERT Coordination Center, "CERT Advisory CA-1999-17 Denial-of-Service Tools," *http://www.cert.org/advisories/CA-1999-17.html*, Dec 1999.

[49] TribeFloodNetwork(TFN), *http://packetstormsecurity.org/groups/mixter/tfn.tgz*, Aug 1999.

[50] CERT Coordination Center, "CERT Advisory CA-2000-01 Denial-of-Service Developments," *http://www.cert.org/advisories/CA-2000-01.html*, Jan 2000.

[51] BBC News, "Yahoo brought to standstill," *http://news.bbc.co.uk/1/hi/sci/tech/635048.stm*, 9 Feb., 2000.

[52] Garber L., "Denial-of-Service Attack Rip the Internet," IEEE Computer, vol. 33, no. 4, pp. 12-17, Apr 2000.

[53] "Peer-to-peer UDP Distributed Denial of Service (PUD)," *http://www.packetstormsecurity.org/distributed/pud.tgz*, 2002.

[54] "Slapper, alias apache-worm, bugtraqworm, Modap," *http://packetstormsecurity.org/0209-exploits/bugtraqworm.tgz*, *2002*.

[55] The Register, "Cloud Nine blown away, blames hack attack," *http://www.theregister.co.uk/2002/01/22/cloud_nine_blown_away_ blames,* Tue. 22 Jan, 2002.

[56] Washingtonpost.com, "Attack on Internet Called Largest Ever," *http://www.washingtonpost.com/ac2/wpdyn?pagename=article&contentId=A828-2002Oct22&notFound=true*, 22 Oct, 2002.

[57] Berghel H., "Malware Month," Communications of the ACM, Vol. 46, no. 12, pp. 15-19, Dec. 2003.

[58] Tynan D., "Sobig May Be Working for the Spammers," http://www.pcworld.com/news/article/0,aid,112261,00.asp, PCWorld, 29 Aug, 2003.

[59] Poulsen K., "FBI Busts alleged DDoS Mafia," *http://www.securityfocus.com/news/9411*, Security Focus, 26 Aug, 2004.

[60] Bellovin S., "Security Problems in the TCP/IP Protocol Suite," ACM Computer Communications Review, Vol. 19, no. 2, pp. 32-48, Apr. 1989.

[61] Sundaram A., "An Introduction to Intrusion Detection," *http://www.acm.org/crossroads/xrds2-4/intrus.html*, 1996.

[62] CERT/CC. 1996b, "Tcp syn flooding and ip spoofing attacks," CERT Advisory CA-1996-21.

[63] Handley, M., Rescorla, E., and IAB, "Internet Denial-of-Service Considerations," RFC 4732 (Informational), 2006.

[64] Joncheray, L. " A simple active attack against TCP," In Proceedings of the 5th conference on USENIX UNIX Security Symposium - Volume 5.2-2.

[65] Vixie, P., Sneeringer, G., and Schleifer, M, "Events of 21-oct-2002," *http://c.root-servers.org/october21.txt*, 2002.

[66] Shirey, R., " Internet Security Glossary," Version 2. RFC 4949 (Informational), 2007.

[67] Crosby, S. A. and Wallach, "Denial of service via algorithmic complexity attacks," In Proceedings of the 12th conference on USENIX Security Symposium - Volume 12, D. S. 2003.

[68] Mirkovic J., Martin J. and Reiher P., "A Taxonomy of DDoS Attacks and DDoS defence Mechanisms," UCLA Computer Science Department, Technical report no. 020018, *http://www.lasr.cs.ucla.edu/ddos/ucla_tech_report_020018.pdf*, *2002*.

[69] Free On-Line Dictionary of Computing (FOLDOC), http://foldoc.doc.ic.ac.uk/foldoc/index.html, 1993.

[70] Clark D., Sollins K., Wroclawski J. and Braden R., "Tussle in Cyberspace: Defining Tomorrow's Internet," Proceedings of the 2002 conference on Applications, technologies, architechtures and protocols for computer communications, pp. 347-356, Aug. 2002.

[71] Spech S. and Lee R., "Taxonomies of Distributed Denial of Service Attacks, Tools and Countermeasures," Princeton University Department of Electrical Engineering, Technical report CE-L2003-004, May 2003.

[72] Oikarinen J., "RFC 1459 – Internet Relay Chat Protocol," *http://www.faqs.org/rfcs/rfc1459.html*, May 1993.

[73] P. Barford and V. Yegneswaran, "An Inside Look at Botnets," Advances in Information Security, 27:171-191, March 2007.

[74] CERT., "The Uncleanliness Vector: Histories of HostileActivity," *http://www.cert.org/research/2006research-report.pdf*, pp.40, 2006.

[75] The Honeynet Project, "Know your Enemy: Honeynets," www.honeynet.org/papers/honeynet/, 2005.

[76] Ulf Lamping, Richard Sharpe, and Ed Warnicke, "Wireshark User's Guide: for Wireshark 1.6.6," NS Computer Software and Services P/L, 2005.

List of Publications

Ravinder Dahiya, Dr. Maninder Singh "DDoS attacks and their alleviation using iptables", International Journal of Information & Network Security (IJINS). **ISSN:** 2089-3299 (communicated).