EARLY WARNING SYSTEM FOR DISTRIBUTED DENIAL OF SERVICE ATTACKS

Project Report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

Computer Science & Engineering

By

Adarsh Singh(121214)

Priyamvada Garg(121319)

under the Supervision of

Ms. Ramanpreet Kaur

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

CERTIFICATE

Candidate's Declaration

We hereby declare that the work presented in this report entitled **"Early Warning System For DDOS Defence"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat has an authentic record of my own work carried out over a period from August 2015 to May 2016 under the supervision of **Ramanpreet Kaur** (Assistant Professor, Department of Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Adarsh Singh(121214) Priyamvada Garg(121319).

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

(Supervisor Signature)

Ms. Ramanpreet Kaur

Assistant Professor

Information Technology

Dated: 26/05/2016

ACKNOWLEDGEMENT

We would like to express my special thanks of gratitude to our Supervisor Ms. Ramanpreet Kaur who gave us the golden opportunity to do this wonderful project on the topic Early Warning System For Distributed Denial of Service Attacks, which also helped us in doing a lot of Research and we came to know about so many new things We are really thankful to them.

Secondly we would also like to thank our parents and friends who helped us a lot in finalizing this project within the limited time frame.

TABLE OF CONTENT

LIST OF ABBREVATIONS

DDoS----------Distributed Denial of Services

DoS----------Denial of Service

TCP----------Transmission Control Protocol

SYN----------Synchronization

ACK----------Acknowledgement

UDP----------User Datagram Protocol

NOP----------No Processing

SSl ----------Secure Socket Layer

CN ----------Cluster Node

JVM ----------JAVA Virtual Machine

IDE ----------Integrated Development Environment

JRE ----------JAVA Run Time Environment

JDK ----------Java Development Kit

GHz ----------Giga Hertz

MB ----------Mega Bytes

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

DDoS attack is one of the most dangerous attacks in the network community to flood the network traffic so that the genuine and authentic users cannot access the services delivered by the servers and legitimate sources. Recently, there has been much excitement in the attackers over using social networks to flood the packets so that the choke of network services can be implemented.

A number of schemes have been proposed, but they differ greatly in the algorithms they use and in the networks upon which they are evaluated. As a result, the research community lacks a clear understanding of how these schemes compare against each other, how well they would work on real-world social networks with different structural properties, or whether there exist other (potentially efficient) ways of defense against distributed denial of service. In the event of a DDoS strike, one or more users from a network location send the data packets or signals and overload the network limit. Due to this reason the, network gets busy and is not able to deliver the service to other users.

This attack is generally implemented on reservation, e-commerce, bidding, social media and many related websites so that the contenders cannot access the same service for which the other users are willing. The DDoS attack can be implemented on wireless sensor network, twork or Wi-Fi environment and is having very adverse effects on the bandwidth side. In this work, an effective algorithm for avoidance of distributed denial of service is proposed so that such attack can be pushed back or detected to carry on the network services without halt. This approach enhances the quality of service and overall performance of the network

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Distributed denial-of-service (DDoS) attacks are a real-and growing-threat to businesses worldwide. Designed to elude detection by today's most popular tools, these attacks can quickly incapacitate a targeted business, costing victims thousands, if not millions, of dollars in lost revenue and productivity. By adopting new purpose-built solutions designed specifically to detect and defeat DDoS attacks, businesses can keep their business operations running smoothly.

DDoS attacks are weapons of mass disruption. Unlike access attacks that penetrate security perimeters to steal information, DDoS attacks paralyze Internet systems by overwhelming servers, network links, and network devices (routers, firewalls, etc.) with bogus traffic.

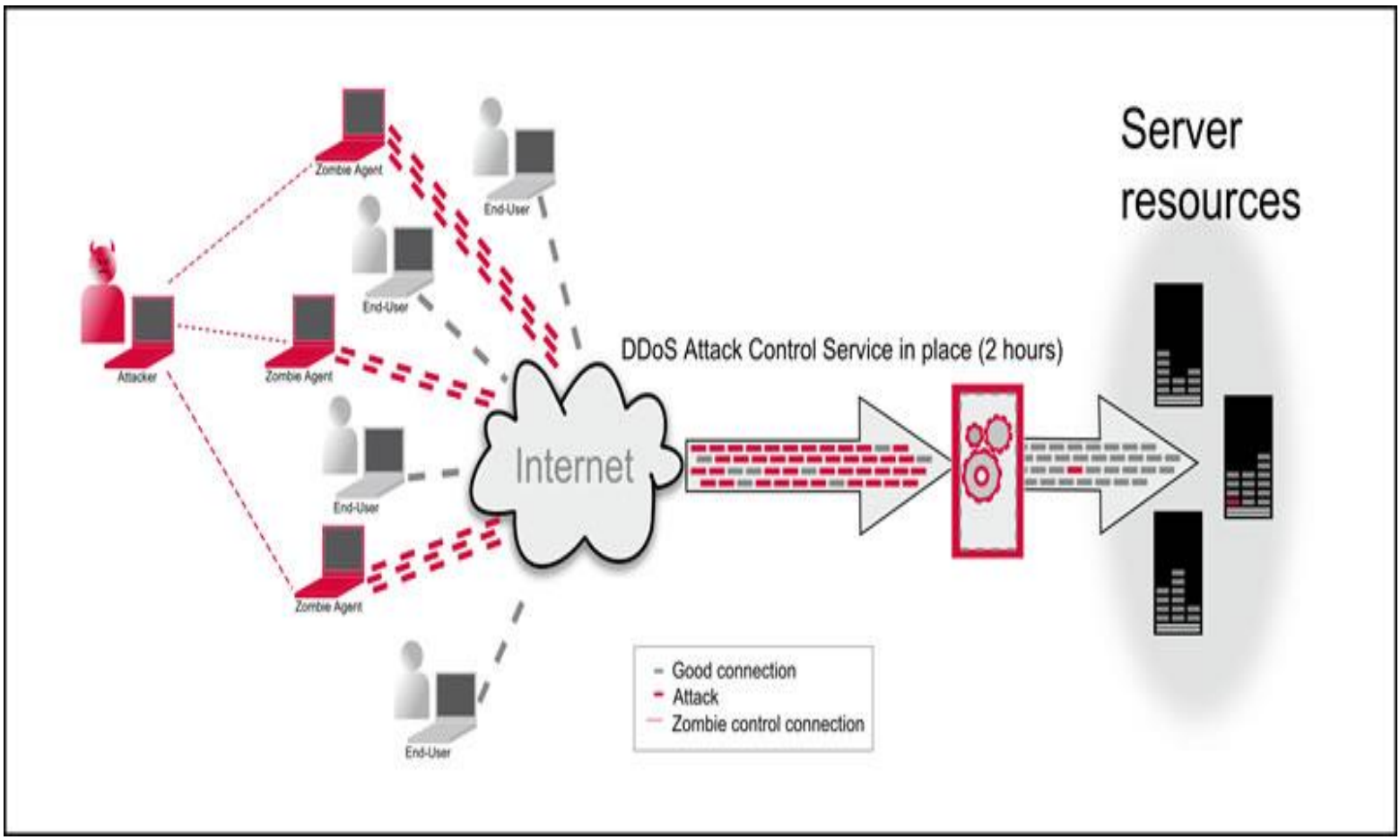
DDoS is emerging as the weapon of choice for hackers, political "hacktivists," cyber-extortionists, and international cyber-terrorists. Easily launched against limited defenses, DDoS attacks not only target individual Websites or other servers at the edge of the network-they subdue the network itself. Attacks have begun to explicitly target the network infrastructure, such as aggregation or core routers and switches, or Domain Name System (DNS) servers in a provider's network. In October 2002, a harbinger of future large-scale attacks was a crude DDoS attack that affected 8 of the 13 root DNS servers, critical systems serving as the roadmap for virtually all Internet communications.

The growing dependence on the Internet makes the impact of successful DDoS attacks-financial and otherwise-increasingly painful for service providers, enterprises, and government agencies. And newer, more powerful DDoS tools promise to unleash even more destructive attacks in the months and years to come.

Because DDoS attacks are among the most difficult to defend against, responding to them appropriately and effectively poses a tremendous challenge for all Internet-dependent organizations. Network devices and traditional perimeter security technologies such as firewalls and intrusion detection systems (IDSs), although important components of an overall security strategy, do not by themselves provide comprehensive DDoS protection. Instead, defending against the current DDoS onslaught threatening Internet availability

requires a purpose-built architecture that includes the ability to specifically detect and defeat increasingly sophisticated, complex, and deceptive attacks.

Figure 1
DDoS Attack



WHY DDOS A THREAT?

In today's world everything is online so if the attacker decides to attack then he can make everything go down which can affect our daily life like going down of reservation website
Can lead to turmoil as no reservation can be performed so DDoS has the ability to bring our daily life to halt.

THE IMPACT OF DDOS ATTACK

The impact of a successful DDoS attack is widespread. Site performance is severely compromised, resulting in frustrated customers and other users. Service-level agreements (SLAs) are violated, triggering costly service credits. Company reputations are tarnished, sometimes permanently. Lost revenue, lost productivity, increased IT expenses, litigation costs-the losses just keep mounting.

The numbers are staggering. Estimates from Forrester, IDC, and the Yankee Group predict the cost of a 24-hour outage for a large e-commerce company would approach US$30 million. A spate of DDoS attacks against Amazon, Yahoo, eBay, and other major sites in February 2000 caused an estimated cumulative loss of US$1.2 billion, according to the Yankee Group. And in January 2001, Microsoft lost approximately US$500 million over the course of a few days from a DDoS attack on its site. Clearly, businesses must take steps to protect themselves from these malicious attacks by shoring up defenses at their multiple points of vulnerability (refer to Figure 1).

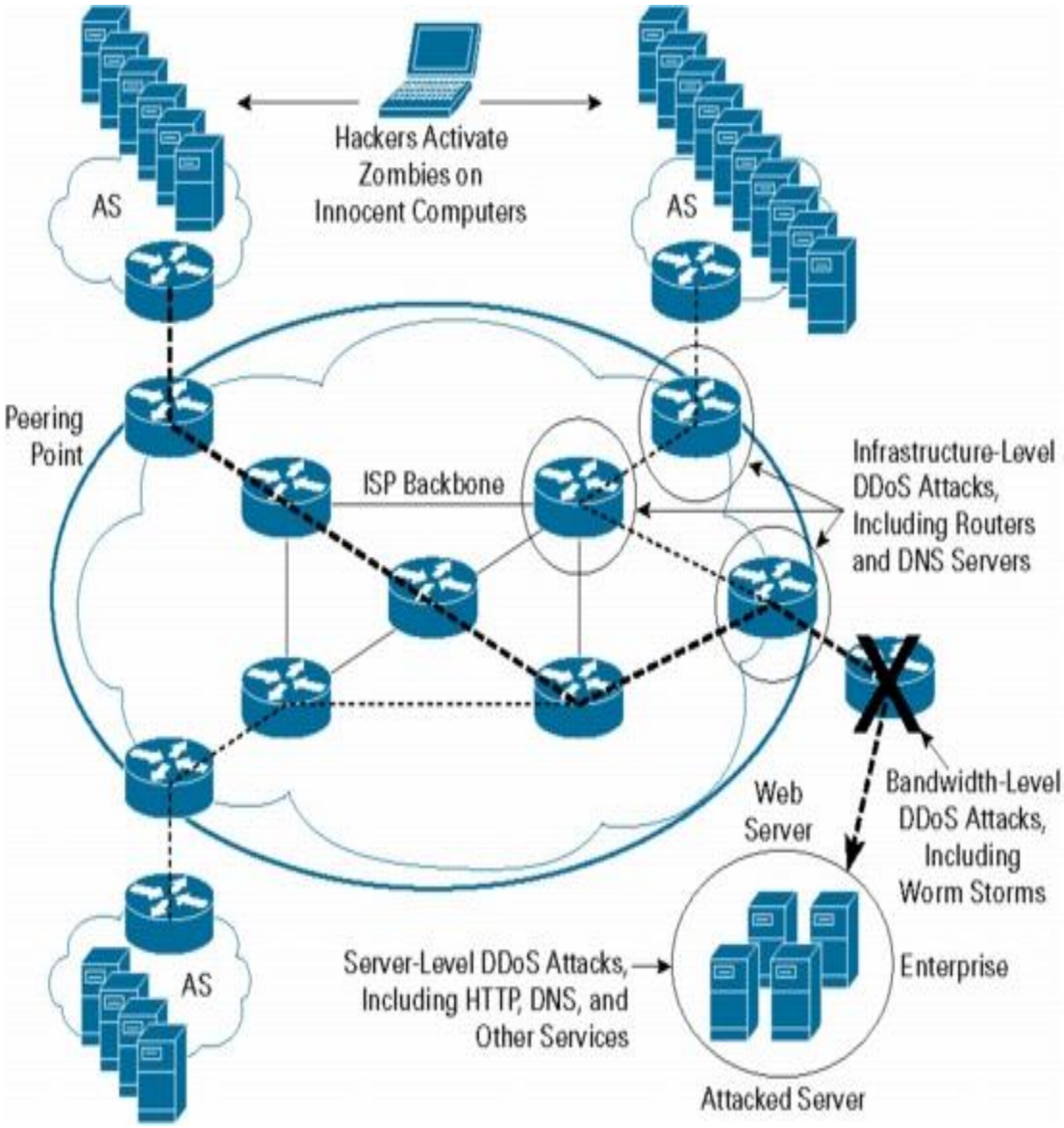


Figure 2
Multiple Points of Vulnerability and Failures

SYMPTOMS

The United States Computer Emergency Readiness Team (US-CERT) defines symptoms of denial-of-service attacks to include:

-Unusually slow network performance (opening files or accessing web sites)

-Unavailability of a particular web site

-Inability to access any web site

-Dramatic increase in the number of spam emails received—(this type of DDoS attack is considered an e-mail bomb)

-Disconnection of a wireless or wired internet connection

-Long term denial of access to the web or any internet services

If the attack is conducted on a sufficiently large scale, entire geographical regions of Internet connectivity can be compromised without the attacker's knowledge or intent by incorrectly configured or flimsy network infrastructure equipment.

TYPES OF ATTACK

The different types of attacks are as follows

- Denial of Service (DoS)
- Jamming
- SYN flooding

DENIAL OF SERVICE (DOS) ATTACKS

In this type of attacks, the attacker injects enormous amount of junk packets into the network which leads to the loss of network resources and causes congestion among the wireless networks.

Prevention mechanism

Attack prevention methods try to stop all well known signature based and broadcast based DDoS attacks from being launched in the first place or edge routers, keeps all the machines over Internet up to date with patches and fix security holes. Attack prevention schemes are not enough to stop DDoS attacks because there are always vulnerable to novel and mixed attack types for which signatures and patches are not exist in the database.
Techniques for preventing against DDoS can be broadly divided into two categories: (i) General techniques, which are some common preventive measures i.e. system protection, replication of resources etc. that individual servers and ISPs should follow so they do not become part of DDoS attack process. (iii) Filtering techniques, which include ingress filtering, egress filtering, router based packet filtering, history based IP filtering, SAVE protocol etc.

A. General Techniques

1) Disabling unused services

The less there are applications and open ports in hosts, the less there are chance to exploit vulnerabilities by attackers. Therefore, if network services are not needed or unused, the services should be disabled to prevent attacks, e.g. UDP echo, character generation services

2) Install latest security patches

Today, many DDoS attacks exploit vulnerabilities in target system. So removing known security holes by installing all relevant latest security patches prevents re-exploitation of vulnerabilities in the target system.

3) Disabling IP broadcast

Defense against attacks that use intermediate broadcasting nodes e.g. ICMP flood attacks, Smurf attacks etc. will be successful only if host computers and all the neighboring networks disable IP broadcast.

4) Firewalls

Firewalls can effectively prevent users from launching simple flooding type attacks from machines behind the firewall. Firewalls have simple rules such as to allow or deny protocols,

ports or IP addresses. But some complex attack e.g. if there is an attack on port 80 (web service), firewalls cannot prevent that attack because they cannot distinguish good traffic from DoS attack traffic.

5) Global defense infrastructure

A global deployable defense infrastructure can prevent from many DDoS attacks by installing filtering rules in the most important routers of the Internet. As Internet is administered by various autonomous systems according their own local security policies, such type of global defense architecture is possible only in theory.

6) IP hopping

DDoS attacks can be prevented by changing location or IP address of the active server proactively within a pool of homogeneous servers or with a pre-specified set of IP address ranges . The victim computer's IP address is invalidated by changing it with a new one. Once the IP addresses change is completed all internet routers will be informed and edge routers will drop the attacking packets. Although this action leaves the computer vulnerable because the attacker can launch the attack at the new IP address, this option is practical for DDoS attacks that are based on IP addresses. On the other hand, attackers can make this technique useless by adding a domain name service tracing function to the DDoS attack tools.

B. Filtering Techniques

1) Ingress/Egress filtering

Ingress Filtering, proposed by Ferguson et al., is a restrictive mechanism to drop traffic with IP addresses that do not match a domain prefix connected to the ingress router. Egress filtering is an outbound filter, which ensures that only assigned or allocated IP address space leaves the network. A key requirement for ingress or egress filtering is knowledge of the expected IP addresses at a particular port. For some networks with complicated topologies, it is not easy to obtain this knowledge.
One technique known as reverse path filtering can help to build this knowledge. This technique works as follows. Generally, a router always knows which networks are reachable via any of its interfaces. By looking up source addresses of the incoming traffic, it is possible to check whether the return path to that address would flow out the same interface as the packet arrived upon. If they do, these packets are allowed. Otherwise, they are dropped.
Unfortunately, this technique cannot operate effectively in real networks where asymmetric Internet routes are not uncommon. More importantly, both ingress and egress filtering can be applied not only to IP addresses, but also protocol type, port number, or any other criteria of importance. Both ingress and egress filtering provide some opportunities to throttle the attack

power of DoS attacks. However, it is difficult to deploy ingress/egress filtering universally. If the attacker carefully chooses a network without ingress/egress filtering to launch a spoofed DoS attack, the attack can go undetected. Moreover, if an attack spoofs IP addresses from within the subnet, the attack can go undetected as well. Nowadays DDoS attacks do not need to use source address spoofing to be effective. By exploiting a large number of compromised hosts, attackers do not need to use spoofing to take advantage of protocol vulnerabilities or to hide their locations. For example, each legitimate HTTP Web page request from 10,000 compromised hosts can bypass any ingress/egress filtering, but in combination they can constitute a powerful attack. Hence, ingress and egress filtering are ineffective to stop DDoS attacks.

2) Router based packet filtering

Route based filtering, proposed by Park and Lee, extends ingress filtering and uses the route information to filter out spoofed IP packets. It is based on the principle that for each link in the core of the Internet, there is only a limited set of source addresses from which traffic on the link could have originated.
If an unexpected source address appears in an IP packet on a link, then it is assumed that the source address has been spoofed, and hence the packet can be filtered. RPF uses information about the BGP routing topology to filter traffic with spoofed source addresses. Simulation results show that a significant fraction of spoofed IP addresses can be filtered if RPF is implemented in at least 18% of ASs in the Internet. However, there are several limitations of this scheme. The first limitation relates to the implementation of RPF in practice. Given that the Internet contains more than 10,000 ASs, RPF would need to be implemented in at least 1800 ASs in order to be effective, which is an onerous task to accomplish. The second limitation is that RPF may drop legitimate packets if there has recently been a route change. The third potential limitation is that RPF relies on valid BGP messages to configure the filter. If an attacker can hijack a BGP session and disseminate bogus BGP messages, then it is possible to mislead border routers to update filtering rules in favor of the attacker. RPF is effective against randomly spoofed DoS attacks. However, the filtering granularity of RPF is low. This means that the attack traffic can still bypass the RPF filters by carefully choosing the range of IP addresses to spoof. Hence, RPF is ineffective against DDoS attacks. The router-based packet filter is vulnerable to asymmetrical and dynamic Internet routing as it does not provide a scheme to update the routing information.

3) History based IP filtering
Generally, the set of source IP addresses that is seen during normal operation tends to remain stable. In contrast, during DoS attacks, most of the source IP addresses have not been seen before. Peng et al. relies on the above idea and use IP address database (IAD) to keep frequent source IP addresses. During an attack, if the source address of a packet is not in IAD, the packet is dropped. Hash based/Bloom filter techniques are used for fast searching of

IP in IAD. This scheme is robust, and does not need the cooperation of the whole Internet community.
However, history based packet filtering scheme is ineffective when the attacks come from real IP addresses. In addition, it requires an offline database to keep track of IP addresses. Therefore, Cost of storage and information sharing is very high.

4) Capability based method
Capability based mechanisms provides destination a way to control the traffic directed towards itself. In this approach, source first sends request packets to its destination. Router marks (pre-capabilities) are added to request packet while passing through the router. The destination may or may not grant permission to the source to send. If permission is granted then destination returns the capabilities, if not then it does not supply the capabilities in the returned packet. The data packets carrying the capabilities are then send to the destination via router. The main advantage achieved in this architecture is that the destination can now control the traffic according to its own policy, thereby reducing the chances of DDoS attack, as packets without capabilities are treated as legacy and might get dropped at the router when congestion happens .
However, these systems offer strong protection for established network flows, but responsible to generate a new attack type known as DOC (Denial of Capability), which prevents new capability-setup packets from reaching the destination, limits the value of these systems. In addition these systems have high computational complexity and space requirement.

5) Secure overlay Service (SOS)
Secure Overlay Service proposed by Keromytis et al. defines an architecture called secure overlay service (SOS) to secure the communication between the confirmed users and the victim. All the traffic from a source point is verified by a secure overlay access point (SOAP). Authenticated traffic will be routed to a special overlay node called a beacon in an anonymous manner by consistent hash mapping. The beacon then forwards traffic to another special overlay node called a
secret servlet for further authentication, and the secret servlet forwards verified traffic to the victim. The identity of the secret servlet is revealed to the beacon via a secure protocol, and remains a secret to the attacker. Finally, only traffic forwarded by the secret servlet chosen by the victim can pass its perimetric routers.
Secure Overlay Service (SOS) addresses the problem of how to guarantee the communication between legitimate users and a victim during DoS attacks. SOS can greatly reduce the likelihood of a successful attack. The power of SOS is based on the number and distribution level of SOAPs. However, wide deployment of SOAPs is a difficult DoS defense challenge. Moreover, the power of SOS is also based on the anonymous routing protocol within the overlay nodes. Unfortunately, the introduction of a new routing protocol is in itself another security issue. If an attacker is able to breach the security protection of some overlay node,

then it can launch the attack from inside the overlay network. Moreover, if attackers can gain massive attack power, for example, via worm spread, all the SOAPs can be paralyzed, and the target's services will be disrupted.

6) SAVE: Source Address Validity Enforcement

Li et al. have proposed a new protocol called the Source Address Validity Enforcement (SAVE) protocol, which enables routers to update the information of expected source IP addresses on each link and block any IP packet with an unexpected source IP address. The aim of the SAVE protocol is to provide routers with information about the range of source IP addresses that should be expected at each interface. Similarly to the existing routing protocols, SAVE constantly propagates messages containing valid source address information from the source location to all destinations. Hence, each router along the way is able to build an incoming table that associates each link of the router with a set of valid source address blocks. SAVE is a protocol that enables the router to filter packets with spoofed source addresses using incoming tables. It overcomes the asymmetries of Internet routing by updating the incoming tables on each router periodically.
However, SAVE needs to change the routing protocol, which will take a long time to accomplish. If SAVE is not universally deployed, attackers can always spoof the IP addresses within networks that do not implement SAVE. Moreover, even if SAVE were universally deployed, attackers could still launch DDoS attacks using non spoofed source addresses.
Table II summarizes filtering techniques for DDoS attacks prevention.
To conclude, attack prevention aims to solve IP spoofing, a fundamental weakness of the Internet. However, as attackers gain control of larger numbers of compromised computers, attackers can direct these “zombies” to attack using valid source addresses. Since the communication between attackers and “zombies” is encrypted, only “zombies” can be exposed instead of attackers. According to the Internet Architecture Working Group, the percentage of spoofed attacks is declining. Only four out of 1127 customer-impacting DDoS attacks on a large network used spoofed sources in 2004. Moreover, security awareness is still not enough, so expecting installation of security technologies and patches in large base of Internet seems to be an ambitious goal in near future. To add on, there exists no way out to enforce global deployment of a particular security mechanism. Therefore, relying on attack prevention schemes is not enough to stop DDoS attacks.

Filtering Technique	Benefits	Limitations
Ingress/ Egress	-Prevents IP Spoofing	-Need global development -Attacks with real IP addresses can not be prevented
RPF (Route based Packet Filtering)	-Work well with static routing	-Problem when dynamic routing is used -Need wide implementation to be effective
History based	-Does not require cooperation of whole Internet Community. -Gives priority to the frequent packets in case of congestion or attack	- Ineffective when the attacks come from real IP addresses - Requires an offline database to keep track of IP addresses - Depend on information collected
Capability based	-Provides destination a way to control the traffic it desires -Incremental deployment	-Attacks against the request packets can not prevented (e.g. ROC attack) -High computational complexity and space requirement
SOS	-Works well for communication of predefined source nodes	-Solution has limited scope e.g. not applicable to web servers -Require introduction of a new routing protocol that itself another security issue
SAVE	-Filtering improperly addressed packets is worthwhile -incremental deployment	-During the transient period valid packets can be dropped

Table1: Summary of Filtering Techniques for DDoS prevention

NETWORK JAMMING OF DOS ATTACK

Jamming is known as the DoS attack that affect communication between two nodes, the main goal of jamming is to block the valid user's like sender and receiver from transmitting and receiving packets, jamming is divided into two types

- Physical jamming attacks
- Virtual jamming attacks

Physical jamming is caused by continuous transmission of packets to the receiver or by causing packet collision at the receiver. Physical jamming is also known as radio jamming, radio jamming is simple attack causing more disrupt to the authorized users. Jammers causing this attack block the authorized users from accessing the wireless channel by controlling the wireless medium. The nodes trying to communicate strangely waiting for the carrier sense timing of the channel to become idle. This put the nodes into list of larger exponential back off period.

Virtual jamming is most often possible at the MAC (Medium Access Control) layer, causing affects on Rate to send (RTS) frames, or Clear to send (CTS) frames, or data frames. One of the advantage of this attack is it consumes less power than comparing to physical or radio jamming. In virtual jamming the malicious node try send RTS command continuously on the transmission with more number of times. In this process the malicious node blocks the transmission limited amount of power. This attack more dangerous than that of physical attack, by sending false frames it will disturb other node from accessing for certain period of time. To prevent and secure the network from jammer or from hidden attacker who causes the network jamming RTS and CTS method is implemented, this mechanism minimizes the attacker node from handshaking process.

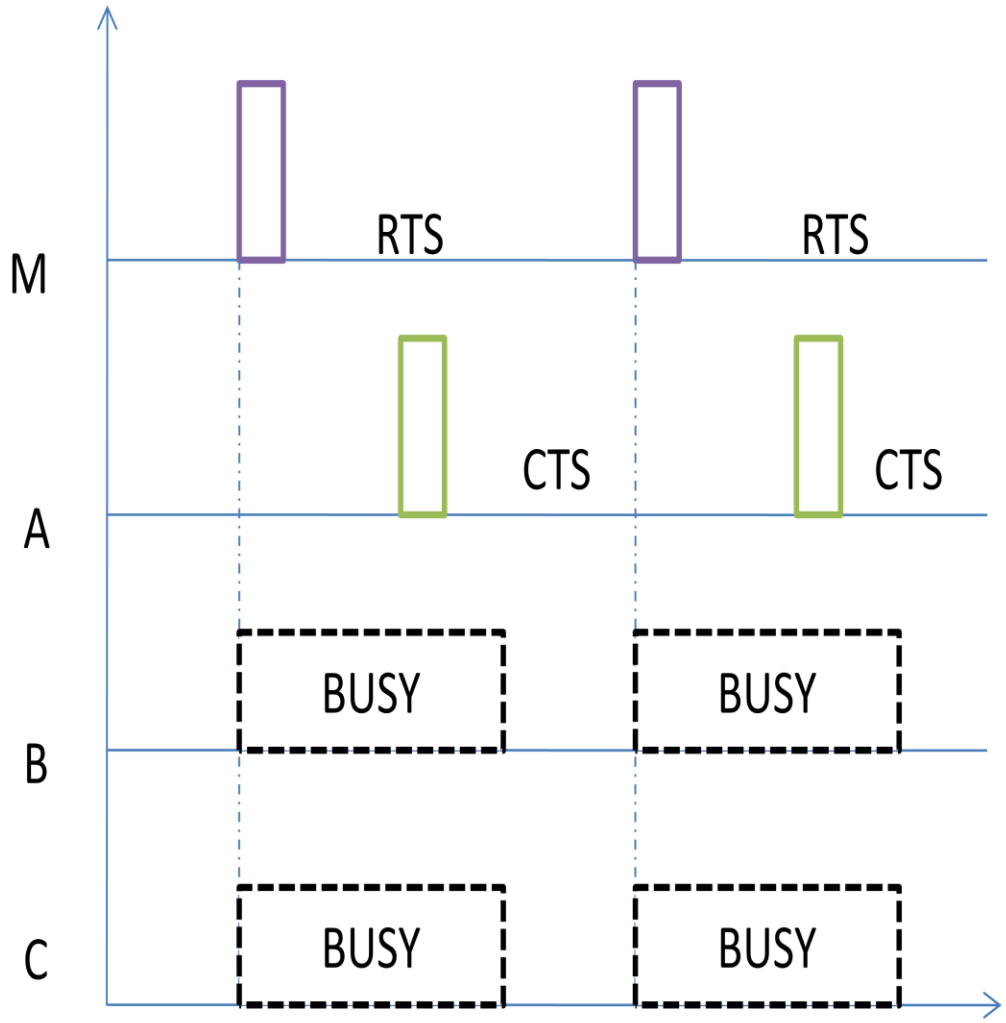


Figure 3: Virtual Jamming

In this fig 4, A B and C are the authorized nodes and M is the malicious node starts sending false RTS request to node A with large frame, when B and C receives this packets they are jammed for certain period of time.

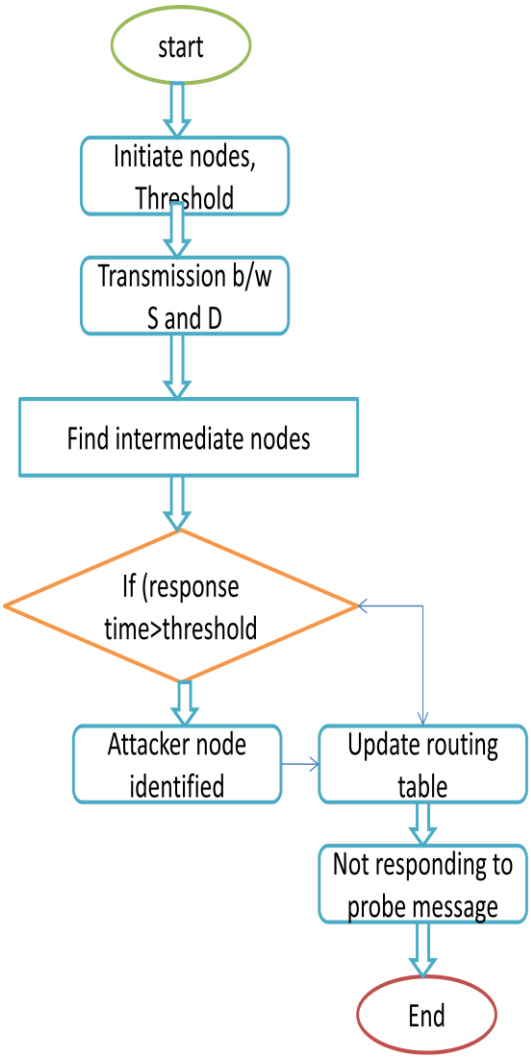


Figure 4: Jamming prevention flowchart

SYN FLOODING ATTACK

In this attack Synchronization (SYN) flooding , a malicious node sends enormous amount of synchronization packets to the affected node and by faking the address of synchronization packets. An SYN-ACK message was sent out from affected node after it receives SYN packets from the attacker, without getting any response from malicious node, the half open request remains in the affected node.

The victim node stores this connection in fixed size table while it waits for the acknowledgement, with all these pending connection, the affected node not be able to accept any other valid attempts to open a connection. Normally the half open connection automatically expires at certain period of time, but the malicious node continuously sends the packets before the previous connection expires.

Some of the counter measures to prevent SYN flooding attacks are

- Filtering
- Firewalls and proxies

Filtering

The filtering techniques are described in the RFC 2827. The most effective technique to prevent SYN flooding is the ingress filtering. It prevents the spoofed IP address. But this technique is not reliable because it is not universally accepted.

Firewalls and proxies

These firewalls and proxies prevent the SYN flooding attack from network attacker using two techniques

- Spoofing initiators SYN-ACK
- Spoofing ACK to the listeners

Non Attacker Firewall behaviour

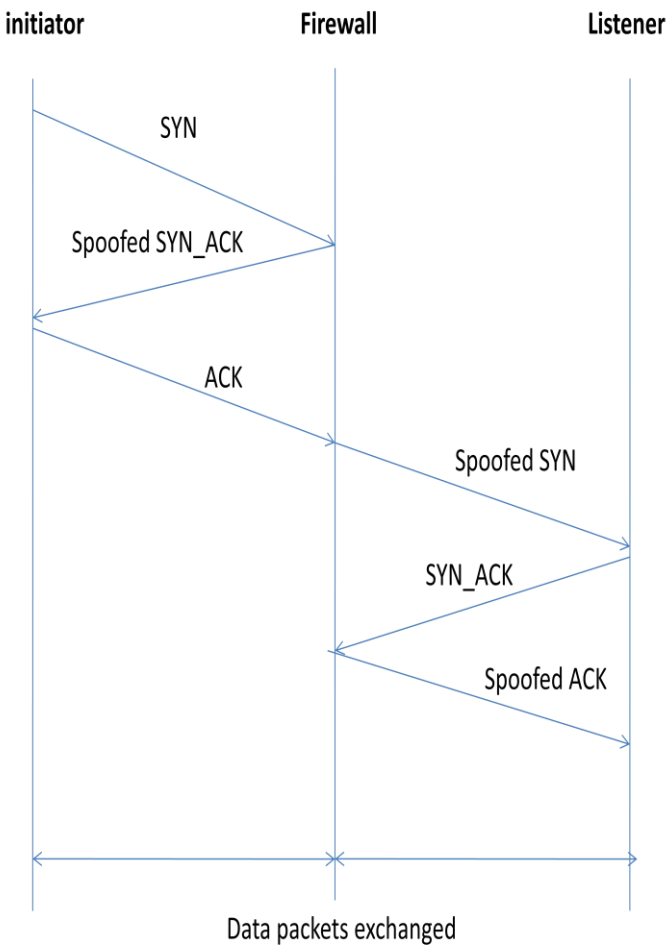


Figure5: Non attacker firewall behaviour

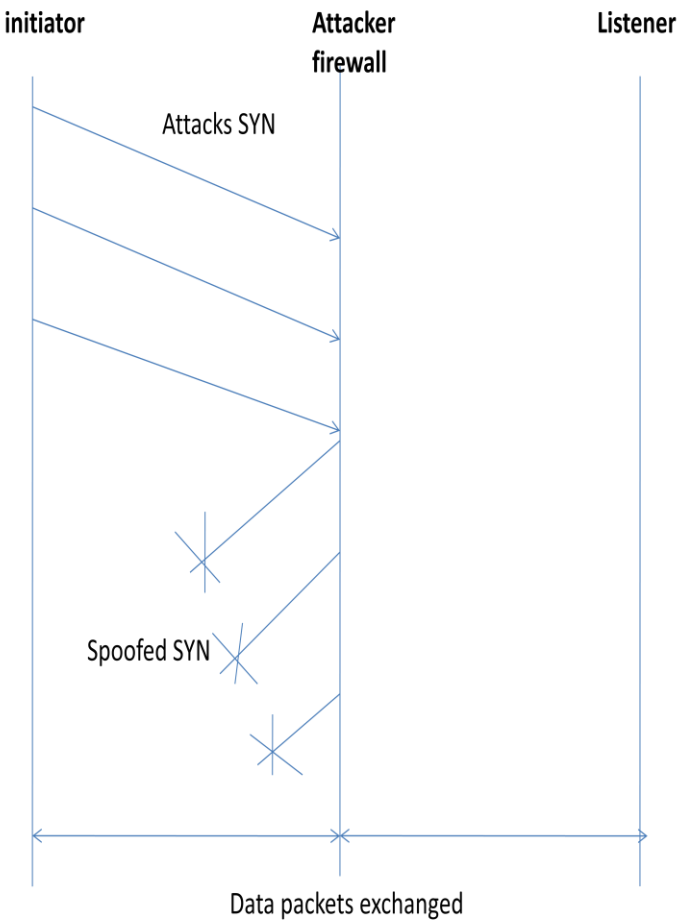


Figure6: Attacker firewall behavior

These are the techniques proposed to overcome SYN flooding attacks.

ATTACK TECHNIQUES

A denial-of-service attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. There are two general forms of DoS attacks: those that crash services and those that flood services.

The most serious attacks are distributed and in many or most cases involve forging of IP sender addresses (IP address spoofing) so that the location of the attacking machines cannot easily be identified, nor can filtering be done based on the source address.

Advanced Persistent DoS (APDoS)

An APDoS is more likely to be perpetrated by an advanced persistent threat (APT): actors who are well resourced, exceptionally skilled and have access to substantial commercial grade computer resources and capacity. APDoS attacks represent a clear and emerging threat needing specialized monitoring and incident response services and the defensive capabilities of specialized DDoS mitigation service providers. This type of attack involves massive network layer DDoS attacks through to focused application layer (HTTP) floods, followed by repeated (at varying intervals) SQLI and XSS attacks. Typically, the perpetrators can simultaneously use from 2 to 5 attack vectors involving up to several tens of millions of requests per second, often accompanied by large SYN floods that can not only attack the victim but also any service provider implementing any sort of managed DDoS mitigation capability. These attacks can persist for several weeks- the longest continuous period noted so far lasted 38 days. This APDoS attack involved approximately 50+ petabits (51,000+ terabits) of malicious traffic. Attackers in this scenario may (or often will) tactically switch between several targets to create a diversion to evade defensive DDoS countermeasures but all the while eventually concentrating the main thrust of the attack onto a single victim. In this scenario, threat actors with continuous access to several very powerful network resources are capable of sustaining a prolonged campaign generating enormous levels of un-amplified DDoS traffic.

APDoS attacks are characterised by:

- Advanced reconnaissance (pre-attack OSINT and extensive decoyed scanning crafted to evade detection over long periods)
- Tactical execution (attack with a primary and secondary victims but focus is on Primary)

- Explicit motivation (a calculated end game/goal target)
- Large computing capacity (access to substantial computer power and network bandwidth resources)
- Simultaneous multi-threaded OSI layer attacks (sophisticated tools operating at layers 3 through 7)
- Persistence over extended periods (utilising all the above into a concerted, well managed attack across a range of targets])

Attack tools

A wide array of programs are used to launch DoS-attacks.

In cases such as MyDoom the tools are embedded in malware, and launch their attacks without the knowledge of the system owner. Stacheldraht is a classic example of a DDoS tool. It utilizes a layered structure where the attacker uses a client program to connect to handlers, which are compromised systems that issue commands to the zombie agents, which in turn facilitate the DDoS attack. Agents are compromised via the handlers by the attacker, using automated routines to exploit vulnerabilities in programs that accept remote connections running on the targeted remote hosts. Each handler can control up to a thousand agents.

In other cases a machine may become part of a DDoS attack with the owner's consent, for example, in Operation Payback, organized by the group Anonymous. The LOIC has typically been used in this way. Along with HOIC a wide variety of DDoS tools are available today, including paid and free versions, with different features available. There is an underground market for these in hacker related forums and IRC channels.

UK's GCHQ has tools built for DDoS, named PREDATORS FACE and ROLLING THUNDER.

Application-layer floods

Various DoS-causing exploits such as buffer overflow can cause server-running software to get confused and fill the disk space or consume all available memory or CPU time.

Other kinds of DoS rely primarily on brute force, flooding the target with an overwhelming flux of packets, oversaturating its connection bandwidth or depleting the target's system

resources. Bandwidth-saturating floods rely on the attacker having higher bandwidth available than the victim; a common way of achieving this today is via distributed denial-of-service, employing a botnet. Another target of DDoS attacks may be to produce added costs for the application operator, when the latter uses resources based on Cloud Computing. In this case normally application used resources are tied to a needed Quality of Service level (e.g. responses should be less than 200 ms) and this rule is usually linked to automated software (e.g. Amazon CloudWatch) to raise more virtual resources from the provider in order to meet the defined QoS levels for the increased requests.The main incentive behind such attacks may be to drive the application owner to raise the elasticity levels in order to handle the increased application traffic, in order to cause financial losses or force them to become less competitive. Other floods may use specific packet types or connection requests to saturate finite resources by, for example, occupying the maximum number of open connections or filling the victim's disk space with logs.

A "banana attack" is another particular type of DoS. It involves redirecting outgoing messages from the client back onto the client, preventing outside access, as well as flooding the client with the sent packets. A LAND attack is of this type.

An attacker with shell-level access to a victim's computer may slow it until it is unusable or crash it by using a fork bomb.

A kind of application-level DoS attack is XDoS (or XML DoS) which can be controlled by modern web application firewalls (WAFs).

Denial-of-service Level II

The goal of DoS L2 (possibly DDoS) attack is to cause a launching of a defense mechanism which blocks the network segment from which the attack originated. In case of distributed attack or IP header modification (that depends on the kind of security behavior) it will fully block the attacked network from the Internet, but without system crash.

Distributed attack

A distributed denial-of-service (DDoS) attack occurs when multiple systems flood the bandwidth or resources of a targeted system, usually one or more web servers.[7] Such an attack is often the result of multiple compromised systems (for example a botnet) flooding the targeted system with traffic. A botnet is a network of zombie computers programmed to

receive commands without the owners' knowledge When a server is overloaded with connections, new connections can no longer be accepted. The major advantages to an attacker of using a distributed denial-of-service attack are that multiple machines can generate more attack traffic than one machine, multiple attack machines are harder to turn off than one attack machine, and that the behavior of each attack machine can be stealthier, making it harder to track and shut down. These attacker advantages cause challenges for defense mechanisms. For example, merely purchasing more incoming bandwidth than the current volume of the attack might not help, because the attacker might be able to simply add more attack machines. This after all will end up completely crashing a website for periods of time.

Malware can carry DDoS attack mechanisms; one of the better-known examples of this was MyDoom. Its DoS mechanism was triggered on a specific date and time. This type of DDoS involved hardcoding the target IP address prior to release of the malware and no further interaction was necessary to launch the attack.

A system may also be compromised with a trojan, allowing the attacker to download a zombie agent, or the trojan may contain one. Attackers can also break into systems using automated tools that exploit flaws in programs that listen for connections from remote hosts. This scenario primarily concerns systems acting as servers on the web. Stacheldrahtis a classic example of a DDoS tool. It utilizes a layered structure where the attacker uses a client program to connect to handlers, which are compromised systems that issue commands to the zombie agents, which in turn facilitate the DDoS attack. Agents are compromised via the handlers by the attacker, using automated routines to exploit vulnerabilities in programs that accept remote connections running on the targeted remote hosts. Each handler can control up to a thousand agents. In some cases a machine may become part of a DDoS attack with the owner's consent, for example, in Operation Payback, organized by the group Anonymous. These attacks can use different types of internet packets such as: TCP, UDP, ICMP etc.

These collections of systems compromisers are known as botnets / rootservers. DDoS tools like Stacheldraht still use classic DoS attack methods centered on IP spoofing and amplification like smurf attacks and fraggle attacks (these are also known as bandwidth consumption attacks). SYN floods (also known as resource starvation attacks) may also be used. Newer tools can use DNS servers for DoS purposes. Unlike MyDoom's DDoS mechanism, botnets can be turned against any IP address. Script kiddies use them to deny the availability of well known websites to legitimate users.[13] More sophisticated attackers use DDoS tools for the purposes of extortion – even against their business rivals.[14]

Simple attacks such as SYN floods may appear with a wide range of source IP addresses, giving the appearance of a well distributed DoS. These flood attacks do not require completion of the TCP three way handshake and attempt to exhaust the destination SYN queue or the server bandwidth. Because the source IP addresses can be trivially spoofed, an attack could come from a limited set of sources, or may even originate from a single host. Stack enhancements such as syn cookies may be effective mitigation against SYN queue flooding, however complete bandwidth exhaustion may require involvement.

If an attacker mounts an attack from a single host it would be classified as a DoS attack. In fact, any attack against availability would be classed as a denial-of-service attack. On the other hand, if an attacker uses many systems to simultaneously launch attacks against a remote host, this would be classified as a DDoS attack.

It has been reported that there are new attacks from Internet of Things (IoT) which have been involved in denial of service attacks. In one noted attack that was made peaked at around 20,000 requests per second which came from around 900 CCTV cameras.

UK's GCHQ has tools built for DDoS, named PREDATORS FACE and ROLLING THUNDER.

DDoS extortion

In 2015, DDoS botnets such as DD4BC grew in prominence, taking aim at financial institutions. Cyber-extortionists typically begin with a low-level attack and a warning that a larger attack will be carried out if a ransom is not paid in Bitcoin. Security experts recommend targeted websites to not pay the ransom. The attackers tend to get into an extended extortion scheme once they recognize that the target is ready to pay.

HTTP POST DoS attack

First discovered in 2009, the HTTP POST attack sends a complete, legitimate HTTP POST header, which includes a 'Content-Length' field to specify the size of the message body to follow. However, the attacker then proceeds to send the actual message body at an extremely slow rate (e.g. 1 byte/110 seconds). Due to the entire message being correct and complete, the target server will attempt to obey the 'Content-Length' field in the header, and wait for the entire body of the message to be transmitted, which can take a very long time. The attacker establishes hundreds or even thousands of such connections, until all resources for incoming

connections on the server (the victim) are used up, hence making any further (including legitimate) connections impossible until all data has been sent. It is notable that unlike many other (D)DoS attacks, which try to subdue the server by overloading its' network or CPU, a HTTP POST attack targets the *logical* resources of the victim, which means the victim would still have enough network bandwidth and processing power to operate. Further combined with the fact that Apache will, by default, accept requests up to 2GB in size, this attack can be particularly powerful. HTTP POST attacks are difficult to differentiate from legitimate connections, and are therefore able to bypass some protection systems. OWASP, an open source web application security project, has released a testing tool to test the security of servers against this type of attacks.

Internet Control Message Protocol (ICMP) flood

A smurf attack relies on misconfigured network devices that allow packets to be sent to all computer hosts on a particular network via the broadcast address of the network, rather than a specific machine. The attacker will send large numbers of IP packets with the source address faked to appear to be the address of the victim. The network's bandwidth is quickly used up, preventing legitimate packets from getting through to their destination. Normally used against small clients or servers, without ISP security.

Ping flood is based on sending the victim an overwhelming number of ping packets, usually using the "ping" command from Unix-like hosts (the -t flag on Windows systems is much less capable of overwhelming a target, also the -l (size) flag does not allow sent packet size greater than 65500 in Windows). It is very simple to launch, the primary requirement being access to greater bandwidth than the victim.

Ping of death is based on sending the victim a malformed ping packet, which will lead to a system crash on a vulnerable system.

Nuke

A Nuke is an old denial-of-service attack against computer networks consisting of fragmented or otherwise invalid ICMP packets sent to the target, achieved by using a modifiedping utility to repeatedly send this corrupt data, thus slowing down the affected computer until it comes to a complete stop.

A specific example of a nuke attack that gained some prominence is the WinNuke, which exploited the vulnerability in the NetBIOS handler in Windows 95. A string of out-of-band data was sent to TCP port 139 of the victim's machine, causing it to lock up and display a Blue Screen of Death (BSOD).

Peer-to-peer attack

Attackers have found a way to exploit a number of bugs in peer-to-peer servers to initiate DDoS attacks. The most aggressive of these peer-to-peer-DDoS attacks exploits DC++. With peer-to-peer there is no botnet and the attacker does not have to communicate with the clients it subverts. Instead, the attacker acts as a "puppet master," instructing clients of large peer-to-peer file sharing hubs to disconnect from their peer-to-peer network and to connect to the victim's website instead.

Permanent denial-of-service attacks

Permanent denial-of-service (PDoS), also known loosely as phlashing, is an attack that damages a system so badly that it requires replacement or reinstallation of hardware. Unlike the distributed denial-of-service attack, a PDoS attack exploits security flaws which allow remote administration on the management interfaces of the victim's hardware, such as routers, printers, or other networking hardware. The attacker uses these vulnerabilities to replace a device's firmware with a modified, corrupt, or defective firmware image—a process which when done legitimately is known as *flashing*. This therefore "bricks" the device, rendering it unusable for its original purpose until it can be repaired or replaced.

The PDoS is a pure hardware targeted attack which can be much faster and requires fewer resources than using a botnet or a root/vserver in a DDoS attack. Because of these features, and the potential and high probability of security exploits on Network Enabled Embedded Devices (NEEDs), this technique has come to the attention of numerous hacking communities.

PhlashDance is a tool created by Rich Smith (an employee of Hewlett-Packard's Systems Security Lab) used to detect and demonstrate PDoS vulnerabilities at the 2008EUSecWest Applied Security Conference in London.

sReflected / spoofed attack

A distributed denial-of-service attack may involve sending forged requests of some type to a very large number of computers that will reply to the requests. Using Internet Protocol address spoofing, the source address is set to that of the targeted victim, which means all the replies will go to (and flood) the target. (This reflected attack form is sometimes called a "DRDOS".)

ICMP Echo Request attacks (Smurf Attack) can be considered one form of reflected attack, as the flooding host(s) send Echo Requests to the broadcast addresses of mis-configured networks, thereby enticing hosts to send Echo Reply packets to the victim. Some early DDoS programs implemented a distributed form of this attack.

Many services can be exploited to act as reflectors, some harder to block than others. US-CERT have observed that different services implies in different amplification factors, as you can see below:

UDP-based Amplification Attacks

Protocol	Bandwidth Amplification Factor
NTP	556.9
CharGen	358.8
DNS	up to 179 [31]
QOTD	140.3
Quake Network Protocol	63.9

BitTorrent	4.0 - 54.3
SSDP	30.8
Kad	16.3
SNMPv2	6.3
Steam Protocol	5.5
NetBIOS	3.8

Table 2: UDP based Amplification Attacks

DNS amplification attacks involve a new mechanism that increased the amplification effect, using a much larger list of DNS servers than seen earlier. SNMP and NTP can also be exploited as reflector in an amplification attack.

R-U-Dead-Yet? (RUDY)

RUDY attack targets web applications by starvation of available sessions on the web server. Much like Slowloris, RUDY keeps sessions at halt using never-ending POST transmissions and sending an arbitrarily large content-length header value.

Slow Read attack

Slow Read attack sends legitimate application layer requests but reads responses very slowly, thus trying to exhaust the server's connection pool. Slow reading is achieved by advertising a very small number for the TCP Receive Window size and at the same time by emptying clients' TCP receive buffer slowly. That naturally ensures a very low data flow rate.

Sophisticated low-bandwidth Distributed Denial-of-Service Attack

A sophisticated low-bandwidth DDoS attack is a form of DoS that uses less traffic and increases their effectiveness by aiming at a weak point in the victim's system design, i.e., the attacker sends traffic consisting of complicated requests to the system. Essentially, a sophisticated DDoS attack is lower in cost due to its use of less traffic, is smaller in size making it more difficult to identify, and it has the ability to hurt systems which are protected by flow control mechanisms.

Teardrop attacks

A teardrop attack involves sending mangled IP fragments with overlapping, over-sized payloads to the target machine. This can crash various operating systems because of a bug in their TCP/IP fragmentation re-assembly code. Windows 3.1x, Windows 95 and Windows NT operating systems, as well as versions of Linux prior to versions 2.0.32 and 2.1.63 are vulnerable to this attack.

(Although in September 2009, a vulnerability in Windows Vista was referred to as a "teardrop attack", this targeted SMB2 which is a higher layer than the TCP packets that teardrop used).

One of the fields in an IP header is the “fragment offset” field, indicating the starting position, or offset, of the data contained in a fragmented packet relative to the data in the original packet. If the sum of the offset and size of one fragmented packet differs from that of the next fragmented packet, the packets overlap. When this happens, a server vulnerable to teardrop attacks is unable to reassemble the packets - resulting in a denial-of-service condition.

Telephony denial-of-service (TDoS)

Voice over IP has made abusive origination of large numbers of telephone voice calls inexpensive and readily automated while permitting call origins to be misrepresented through caller ID spoofing.

According to the US Federal Bureau of Investigation, telephony denial-of-service (TDoS) has appeared as part of various fraudulent schemes:

- A scammer contacts the victim's banker or broker, impersonating the victim to request a funds transfer. The banker's attempt to contact the victim for verification of the transfer

fails as the victim's telephone lines are being flooded with thousands of bogus calls, rendering the victim unreachable.

- A scammer contacts consumers with a bogus claim to collect an outstanding payday loan for thousands of dollars. When the consumer objects, the scammer retaliates by flooding the victim's employer with thousands of automated calls. In some cases, displayed caller ID is spoofed to impersonate police or law enforcement agencies. scammer contacts consumers with a bogus debt collection demand and threatens to send police; when the victim balks, the scammer floods local police numbers with calls on which caller ID is spoofed to display the victims number. Police soon arrive at the victim's residence attempting to find the origin of the calls.

Telephony denial-of-service can exist even without Internet telephony. In the 2002 New Hampshire Senate election phone jamming scandal, telemarketers were used to flood political opponents with spurious calls to jam phone banks on election day. Widespread publication of a number can also flood it with enough calls to render it unusable, as happened with multiple +1-area code-867-5309 subscribers inundated by hundreds of misdialed calls daily in response to a popular song 867-5309/Jenny.

TDoS differs from other telephone harassment (such as prank calls and obscene phone calls) by the number of calls originated; by occupying lines continuously with repeated automated calls, the victim is prevented from making or receiving both routine and emergency telephone calls.

Related exploits include SMS flooding attacks and black fax or fax loop transmission.

SMURFING

In a smurfing attack, a network amplifier is used create a flood of traffic to target a victim system. The attack begins with a ping packet sent to some system, which supports direct broadcast messages known as a network amplifier. A network amplifier is usually a system on the Internet with an incorrect configured network. The source address of the packet is spoofed to be that of the victim system. Spoofing is a way for the attacker to send messages to IP address, which says that the message was from a trusted host. By doing this, all the ping responses are sent to the victim system. Using the network amplifier with 50 hosts, 50 packets can be sent to the victim by just sending one packet. Network amplifier will receive

packet by packet until the maximum amount of traffic is sent. This is because the network amplifier itself has a fixed bandwidth connection to the Internet. At the end, the attack will be traced back to the network amplifier and not the attacker.

Smurf attacks rely on a directed broadcast to create a flood of traffic for a victim on a particular IP address. An IP address is made of host address and network address. If the host part of address is all 1's then the packet is destined for broadcast address of the network. For example, if the network IP address of the network were 10.1.0.0 with net mask of 255.255.0.0, the broadcast IP address for the network would be 10.1.255.255. Using 255 consecutively means there is a message for network IP address because host contains 16 consecutive 1s. This in turn will cause every machine on destination LAN to read the packet and send a response.

The packets sent by the attacker are ICMP ECHO REQUESTS. Normally if the packet's destination network router allows direct broadcasts, all destination LANs will receive the packet. Once received, these machines will then send a ping response. By sending 1 packet, thousands of response packets can be sent. If the first ping response were from spoofed address then all ping responses from the network would be sent to the spoofed address. The number of response packets will increase with more machines on the network that allow direct broadcasting. Using this idea an attacker can conduct a smurfing attack.

A similar attack to smurfing is the fraggle attack. Fraggle is similar that the attacker sends packets through network amplifier but differ by using UDP ECHO packets rather than ICMP ECHO packets. The attack begins with packets sent to IP broadcast address. The destination is UDP port set to a service, which can send the response. The service that receives the packet just sends the packet back exactly as received. By doing this, all machines will echo UDP traffic back causing a flood of the victim's system.

UDP FLOODING

User Datagram Protocol (UDP) is a connectionless protocol. When sending data packets through UDP, no handshake is required between the sender and receiver. The receiving party will receive packets to process. If a large number of UDP packets are sent, this could cause the victim system to be saturated. This in turn would reduce the bandwidth amount available for legitimate users on the system.

When the attacker uses UDP flood attack, UDP packets are sent to either random or specified ports on a victim system. Most of the time they are sent to random ports. When the packets are sent, it causes the victim system to process the incoming data. The system then has to determine which application sent the request. If no applications were running on targeted port, the victim system would send out ICMP packet indicating the destination port is unreachable. As with smurfing, UDP flooding uses spoofed IP address when sending the attacking packet. By doing this, the return packets are sent to another system with spoofed address and not sent back to zombie systems. Another side effect of UDP flood attacks is that these attacks can fill the bandwidth connection around the victim system causing those systems to experience problems with their connectivity.

TCP SYC ATTACK

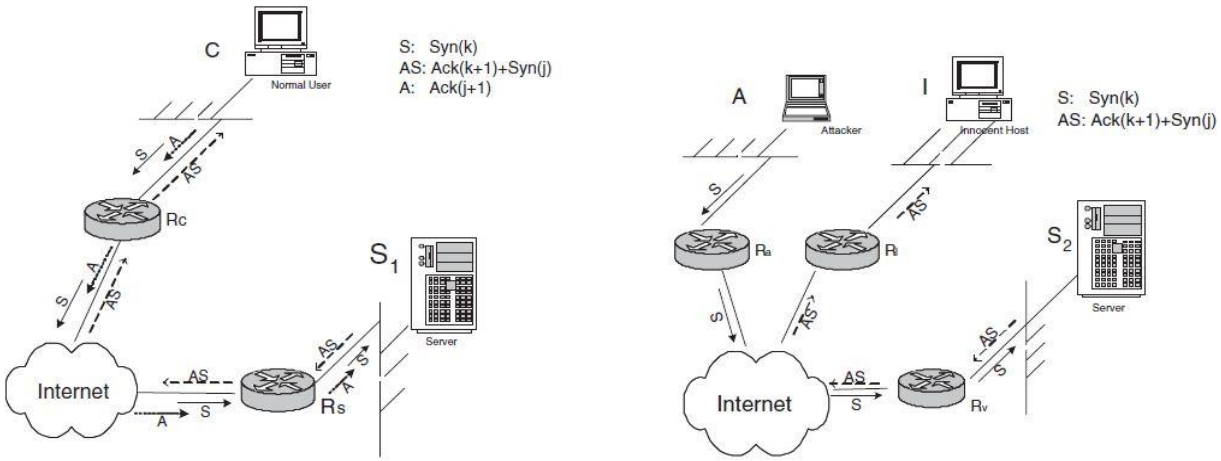
A TCP SYC attack is a denial of service attack in which attacker deliberately violates the three way handshake and open a large number of half open TCP/IP connections. Potential targets for this attack are any system connected to Internet that provides TCP based network services. Some examples include a web server, FTP server, or mail server.

When a TCP connection is made to a system providing a service (server), the client and server exchange a set sequence of message known as three-way handshake. The client's system begins by sending SYN (synchronization) message to the server. The server then

acknowledges the message by sending SYN-ACK message to client. The client then finishes establishing the connection by responding with an ACK message.

Problems arise when the server system has sent an acknowledgment (SYN-ACK) back to client, but has not received the final ACK message. This is called the half opened connection. In memory, the server has a built in data structure describing all pending connection. This data structure is finite size and can be made to overflow by creating lots of partially opened connections. When a large volume of SYN requests are being processed by a server and none of the ACK+SYN responses are returned, the server begins to run out of processor and memory resources

In a TCP SYN attack, the attacker instructs the zombies to send some bogus TCP SYN request to a victim server in order to tie up the server's processor resources. This in turn, prevents the server from responding to legitimate requests. The source address of SYN packet sent by the attacker is spoofed thus hiding the identity of the attacker.



(a) Normal three-way handshake (b) Spoofed three-way handshake

Figure7. The process of the TCP three-way handshake

PUSH + ACK ATTACK

In the TCP protocol, packets that are sent to a destination are buffered within the TCP stack and when the stack is full, the packets get sent on to the receiving system. However, the sender can request the receiving system to unload the contents of the buffer before the buffer becomes full by sending a packet with the PUSH bit set to one. PUSH is a one-bit flag within the TCP header. TCP stores incoming data in large blocks for passage on to the receiving system in order to minimize the processing overhead required by the receiving system each time it must unload a non-empty buffer.

The PUSH + ACK attack is similar to a TCP SYN attack in that its goal is to deplete the resources of the victim system. The attacking agents send TCP packets with the PUSH and ACK bits set to one. These packets instruct the victim system to unload all data in the TCP buffer (regardless of whether or not the buffer is full) and send an acknowledgement when complete. If this process is repeated with multiple agents, the receiving system cannot process the large volume of incoming packets and it will crash.

1.2 PROBLEM STATEMENT

Distributed Denial of Service (DDoS) attacks are under research from a long time since the inception of cracking the network infrastructure.

There are number of methods to hack and destroy the network resources but most of these cracks are detectable and easy traceable. In case of DDoS attack, it is very difficult to identify the type of attack because this attack choke or freeze the network services delivery.

It is very important to devise a new algorithm so that easily detection of the DDoS attack and done to improve the quality of service. If DDoS attack is implemented on the network, the quality of service degrades drastically.

A DDoS attack exploits or utilizes the distributed nature of the network, with hosts owned by disparate entities around the world.

In this work, we propose and implement a new algorithm making use of tracking the source identity as well as the limit on sending packets so that the load balancing can be implemented.

Using this approach, any source or node will be able to send the limited number of packets and with dynamic encryption.

CLASSICAL APPROACH OF DDOS ATTACK

Here we underline the process of DDoS attack takes place. The key point is that DDoS attacks work in a way that the attack and attacker are well concealed from being caught for their actions.

Step 1

The DDoS attack operates through a client machine by hacking into weakly secured computers. This is done by searching and finding well-known defects in standard network service programs and commonly weak configurations in known operating systems. But before that attacker can start, the attacker scans these systems looking for vulnerabilities. Unfortunately, this phase very much favors the attackers. The attacker uses computer systems and network port openings to gain access. The more ports that are open, the more points of vulnerability.

To determine which ports are open on a given system, a program called port scanner is used. A port scanner runs through a series of ports to see which ones are open. Usually a machine in TCP/IP stack has 65,535 TCP ports and 65,535 UDP ports. The number of ports combined has a potential doorway into the system. Normally, major services listen on fixed port number with the list of open ports on a target system. Using this information, the attacker can get an idea of which services are in use by checking RFC 1700, "Assigned numbers".

In the Windows environment, one good scanner is called Scan port. This is a fairly basic port scanner but it enables the attacker to specify both the range of addresses and range of ports to scan. On the Unix side, the best scanner is Nmap. This program scans for open ports by sending packets to the target system to interact with each port. What type of packets is sent and how does interaction happen depend on type of scan being conducted. Some of the types of scan are as follows.

TCP Connect: Completes the three-way handshake with each scanned port.
TCP Syn: Only sends the initial SYN and awaits SYN-ACK response to determine if the port is open.
UDP scan: Sends a UDP packet to target ports to determine if a UDP service is listening.
Ping: Sends ICMP Echo request to every machine on the target network, for locating live hosts.

After the vulnerability scan is done on the target system, a list of vulnerabilities is given to the attacker could exploit. The reason behind the scan is to automate the process of connecting to a target system and checking to see if the vulnerabilities are present.

Another scan tool called Nessus scans random IP addresses to find a known vulnerability. After the scan, a list of victim systems is created that shares the same common vulnerability.

Step 2

After the scan, the attacker chooses a number of machines to be involved in the attack. These systems are also known as handlers or masters. Now the attacker can find a way to gain access and have significant control over these machines. Most common method is using Stack Based buffer overflow attack. Any application or operating system component that is poorly written could have this problem. A buffer overflow attack happens when an attacker tries to store too much information in an undersized receptacle. Buffer overflow takes advantage of the way in which data is stored by computer programs. When a program calls a subroutine, the function variable and the subroutine returns address pointers stored in a logical data structure known as stack. A stack is a portion of memory, which stores information about the current program needs and contains the address where the program returns after the subroutine has completed execution.

When the buffer is overflowed, the data placed there goes into neighboring variable space and eventually into the pointers space. To cause the attacker's code to be executed, the attacker precisely tunes the amount and content of data to cause buffer overflow and stack to crash. The data the attacker sends usually consists of machine specific byte code to execute a command plus a new address for return pointer. This address points back into the address space of stack, causing the program to run the attackers instruction when it returns from the subroutine. To help improve the odds that the return pointer will jump to a good place to begin executing the attacker's code, attackers will often prepend a series of NOP (no processing) instruction to their machine level code. A key point is that attacker code will run at whatever privileges the software that is exploited is running at. In most cases, attacker tries to exploit program running as root or administrator privilege. So attacker can easily install backdoor on a system in this way.

The captured machines are now instructed to control another set of captured machines. These are called the agents or daemons. By doing this, it ensures a measure of cautiousness on the part of the attacker. Now it is very difficult and impossible to track and find the actual attacker on the Internet. The attacker comprises more systems until the risk of being captured is almost impossible. At the end, the attacker knows the addresses of all the nodes and stores them in a file on his control system. This is later used to attack the victim.

Step 3

After the attacker breaks into the system, they want to be able to get back into victim's system whenever they want. They could achieve this by installing a backdoor entry as in step 2 or by installing a rootkit (very common in Unix operating system). A rootkit is like a trojan key system files on an operating system. The attacker can replace the login program by overwriting it, but it would be obvious someone messed up the system so a legitimate user could not gain access. To avoid this, the attacker could add some feature into existing login program like allowing someone to have root access without prompting for a password; it

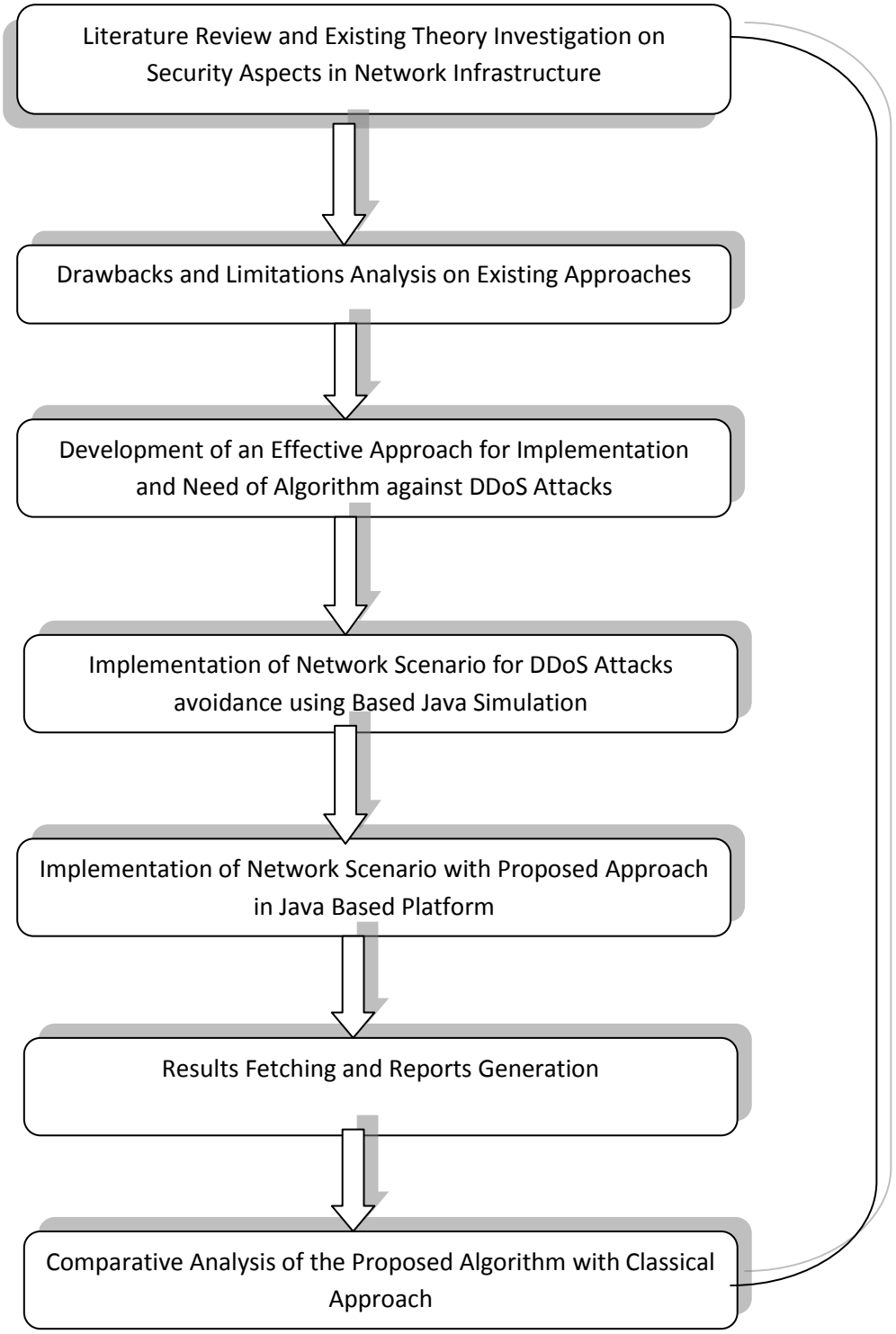
would be hard for the administrator to detect their system has been comprised. In general, rootkit provide false information or lie to the administrator to hide what the attacker is doing. The rootkit masks attack activity going on the background.

So finally the actual attack takes place. The attacker on his computer using client software sends instructions to the handlers or nodes to launch a particular attack. These attacks come from variety of different flooding attacks against specific victim.

1.3 OBJECTIVES

1. To analyze assorted network attacks and their impact on the quality of service and higher uptime of the network resource
2. To perform a comparative analysis of various attacks and their penetration levels
3. To investigate the Distributed Denial of Service (DDoS) Attack and its effect on any network infrastructure
4. To propose a new and effective algorithm to identify and cope up with DDoS attack
5. To perform the implementation in Advance Java Based Environment using Socket Programming so that the effect and avoidance against DDoS attacks can be accomplished.
6. To perform a detailed comparative and pragmatic analysis between the classical and proposed approach of avoiding the DDoS attacks.

1.4 METHODOLOGY



1.5 ORGANIZATION

Chapter 1 highlights and underlines the assorted network vulnerabilities and attacks. In this chapter, the introduction to various security loopholes and attacks are covered. The key focus on DDoS attack is done so that the proposed work can be defended.

The detailed literature review from the research paper, books, journals and conferences are done in **Chapter 2.** In this chapter, the extracts from assorted research papers on DDoS attacks are taken and depicted.

Chapter 3 covers the system development which is the key aspect of this work. In this chapter, the proposed model, algorithm and related parameters are emphasized.

The simulation of implementation results with the relative performance analysis is shown in **Chapter 4**. In this chapter, the simulation results and screenshots are revealed to depict and defend the proposed work

Chapter 5 ends with the detailed conclusion and scope of the future work which guides the upcoming students and research scholars to enhance the current work with higher efficiency and effectiveness.

CHAPTER 2
LITERATURE SURVEY

Working On

SYN Flooding

For completion, justification and solving the problem definition, a number of research papers, magazines, journals and online links are investigated in details.

In this chapter, the details of research papers and journals are specified from where we have analyzed the content and formulated the problem.

A number of research scholars and scientists has written a number of research papers and found excellent results. This section underlines all those research papers and their extracts

Ayman Mukaddam et al. proposed the utilization of both Round Trip Time (RTT) and Hop Count to detect IP Spoofing. RTT is the difference in time between the time a packet is sent and the times is corresponding reply is received. This is a cumbersome technique when packets transmitted are lost in the network and are to be re-transmitted. RTT is influenced by the distance between the sender and the receiver, link bandwidth and the queuing behaviour of the nodes. This technique tries to eliminate the weakness of the HCF technique and relies both on HCF and RTT technique instead of only on HCF. Now the attackers have to guess both Hop Count and RTT for the spoofed packet to be considered legitimate. Since, these variables are independent, the probability of guessing both the parameters correctly is lower than the probability of guessing only Hop Count correctly. Also, both parameters cannot be spoofed easily as they are path and load dependent.

Xia Wang et al. focussed on the elimination of the execution caused by the DDoS Attack and tracking its attack source. They have used filters at the intermediate node on the basis of some fixed Hop Count threshold. So, by using the variation of Hop Count Filtering technique, they are not protecting the end systems only but the whole network is protected from traffic congestion.

Krishna Kumar et al. proposed to detect IP spoofing by checking both the Hop Count and the path Identification (PID) at every router. The PID is inserted in each IP Packet in the identification field. If both the Hop Count and the PID match, then the packet is considered legitimate otherwise, the routers start attack detection process. The algorithm requires a shared key between every pair of adjacent routers.

B.R. Swain et al. proposed a probability based HCF technique over conventional HCF Technique resulting in the saving of Computational Time. Usually, in conventional HCF 90% of erroneous packets are dropped but in their case, 80% to 85% of packets will be dropped with the reduction in memory overhead. Unlike the HCF technique that checks every packet for its legitimacy, they check the packets till they reach n malicious packets. After that m packets are allowed unchecked. Their packet analysis is based on probability of packet arrival p, number of malicious packets n and number of legitimate packets m.

Haining Wang et al. proposed HCF to remove IP packets at the very start of network processing. He considered two HCF States in his work which are ‘learning’ state and ‘filtering’ State. HCF works in ‘learning’ state under normal conditions and watch for abnormal TTL behaviours without discarding any packets. After detecting an attack, mechanism switches to ‘filtering’ State to discard IP packets with mismatched Hop Counts. This HCF technique has been used at the victim side. HCF is an important technique to remove the randomly spoofed IP traffic or random IP Spoofing. But, attacker may also find the effective way by creating an effective IP2HC table to overcome HCF.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 ALGORITHM

VOID Outgoing Packets Process (INPUT: P) {
if P is a Syn(k) packet then
R=hash(source IP, destination IP)
Create R
Set R state = syn
else if P is a Ack(j+1) packet then
R=hash(source IP, destination IP)
if R is found in the Hash Table AND R state == ack then
Release R
else if R is found in the Hash Table AND R state == suspicious then
Release R
end if
end if
}

VOID Incoming Packets Process (INPUT: P) {
if P is Ack(k+1)+Syn(j) packet then
R=hash(destination IP, source IP)
if R is found in Hash Table AND R state == syn then
Set R state=ack
else if R is not found then
Create R
Set R state= suspicious
end if
end if
}

PROBLEM IN EXISTING SYSTEM

- The classical approach is not efficient in terms of security and integrity.
- The approach is required to be updated so that the attacks should not happen by any medium
- The existing system is not efficient in terms of the packet loss, jitter and throughput.

- There should be a specialized mechanism and algorithmic approach for avoidance of DDoS attacks in network infrastructure.
- Securing network is great challenge for many years due to the absence of proper infrastructure and its open type of network.
- Previous security measures in the network are not effective in the challenging world with advancement in technology.
- Many layers often prone to attacks man in middle attack or multilayer attack, so proposal should concentrate on this layers.
- The proper intelligent approach of securing the network has not yet discovered.
- Comparing the security issues of wireless network with wired network, wired network has the proper infrastructure for forward and receiving packets, whereas in wireless network there is no proper infrastructure and it is accessible by both authorized users and hackers.
- In this wireless network there is no particular design to monitor the traffic and accessibility, these leads to third party intervention like malicious users.

A specialized harmful attack that takes the advantage of these characteristics is the DDoS attack, whereby the malicious node illegitimately claims multiple identities. Sybil attack exceedingly disrupt assorted operations of the mobile ad hoc networks such as data aggregation, voting, fair resource allocation scheme, misbehavior detection and routing mechanisms etc. There is the specific requirement to develop and simulate the algorithmic approach for mechanism against the denial of service attacks.

3.2 HARDWARE REQUIREMENTS

The minimum requirements needed to perform operations are

- Intel Pentium Processor at 2 GHz or Higher
- RAM 256MB or more
- Hard disk capacity 10GB or more

3.3 SOFTWARE REQUIREMENTS

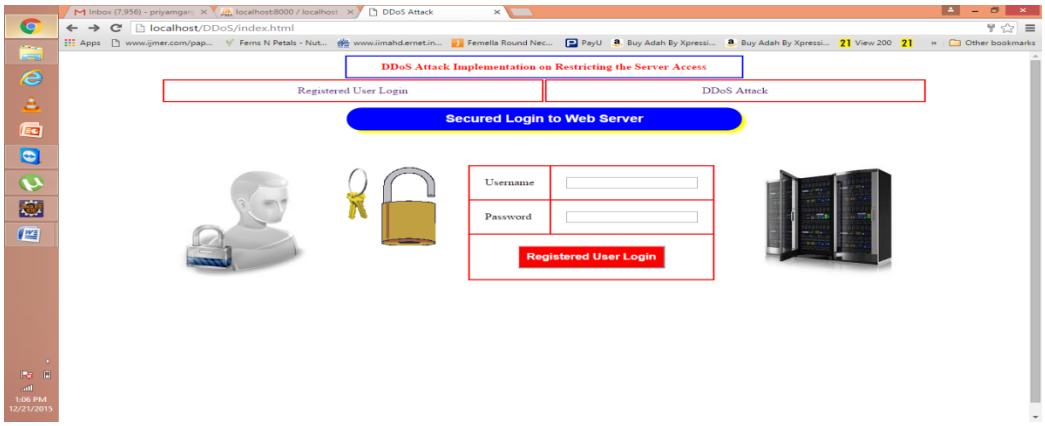
The software required to perform the implementation are

- Windows or Linux Operating System (Ubuntu, Fedora)
- JDK 8
- Eclipse / NetBeans IDE
- Dia - The Diagram Editor
- Notepad++

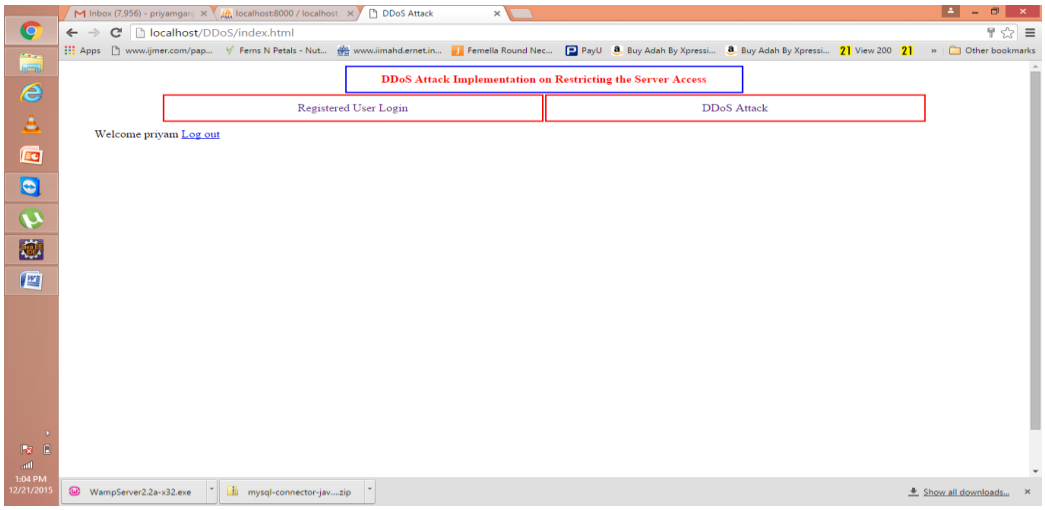
CHAPTER 4

PERFORMANCE ANALYSIS

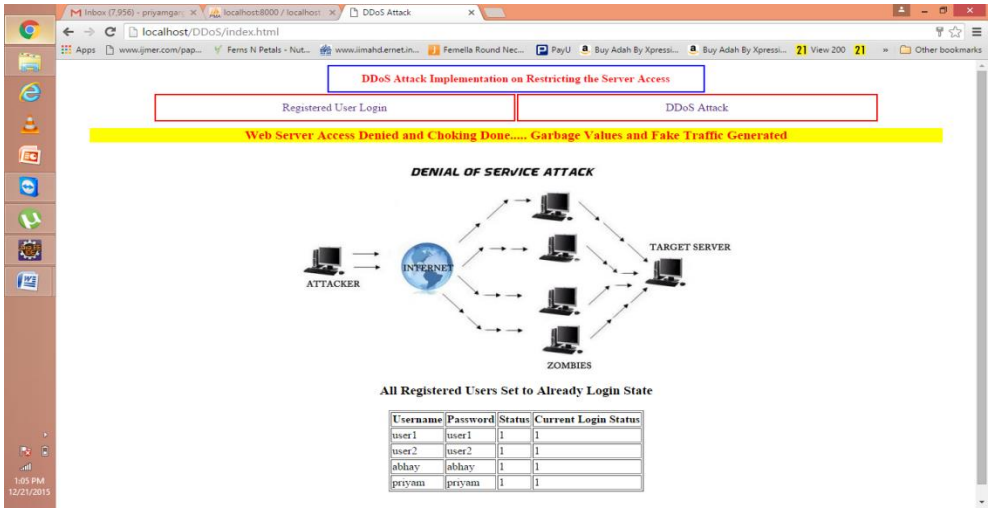
1. DDoS Attack implementation on retrieving the server Access:-



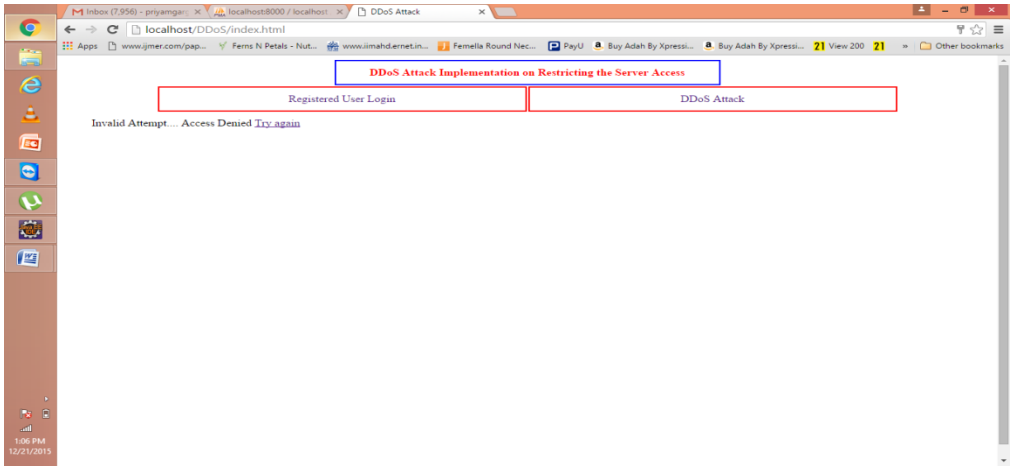
2. After login with username and password; it gets login with a message on the screen as shown below



3. When we click on the DDoS Attack ,access on the web server is denied and current login state is set to one as shown below and after that the legitimate user can not access…



4. When the DDoS Attack is on the currently registered user can not login or their access is denied , and the message is shown on the screen as shown below:-



293 MB Data Packets Transmitted from Anonymous IP

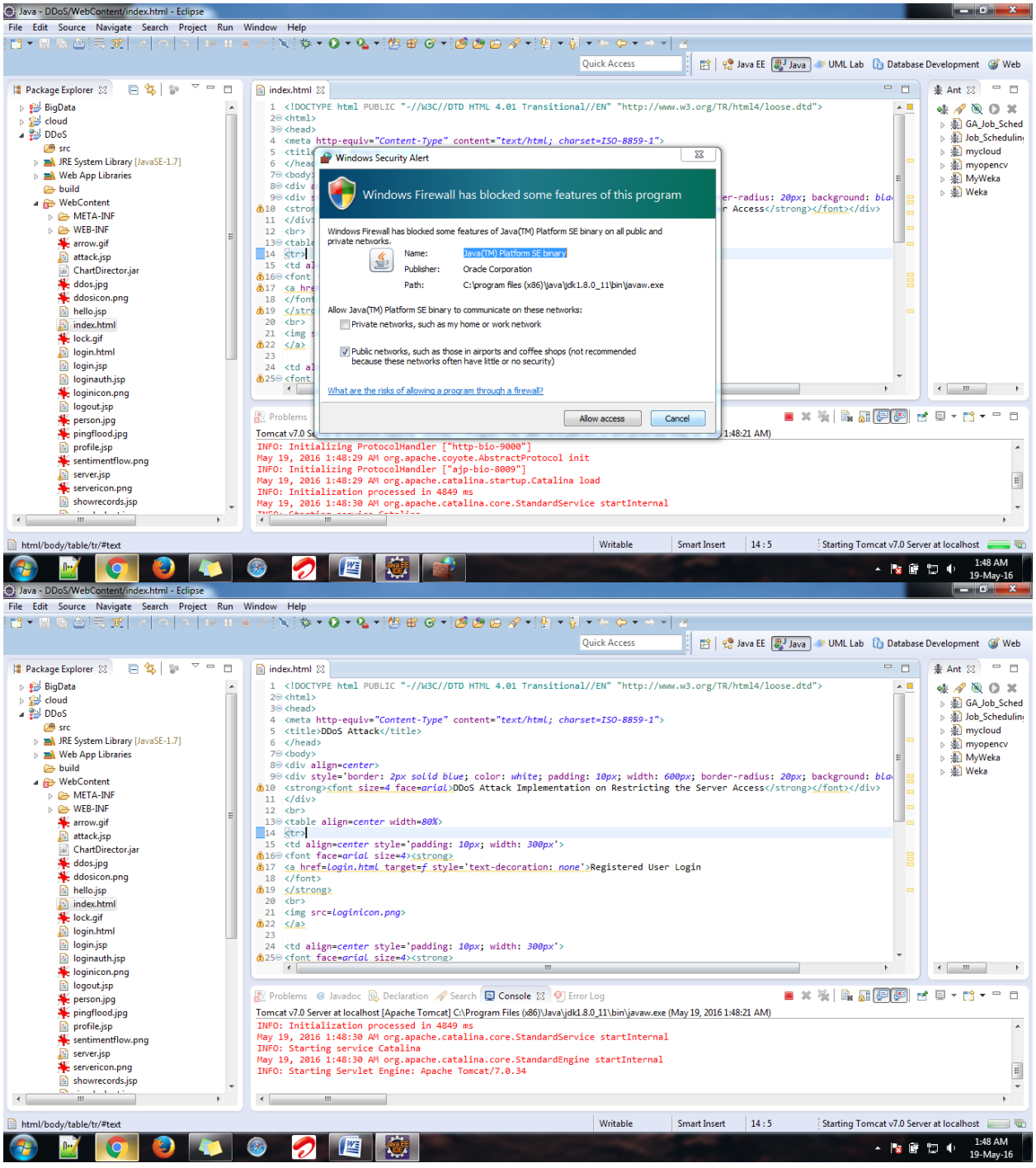
2343 MB Data Packets Transmitted from Anonymous IP – DDoS Attack

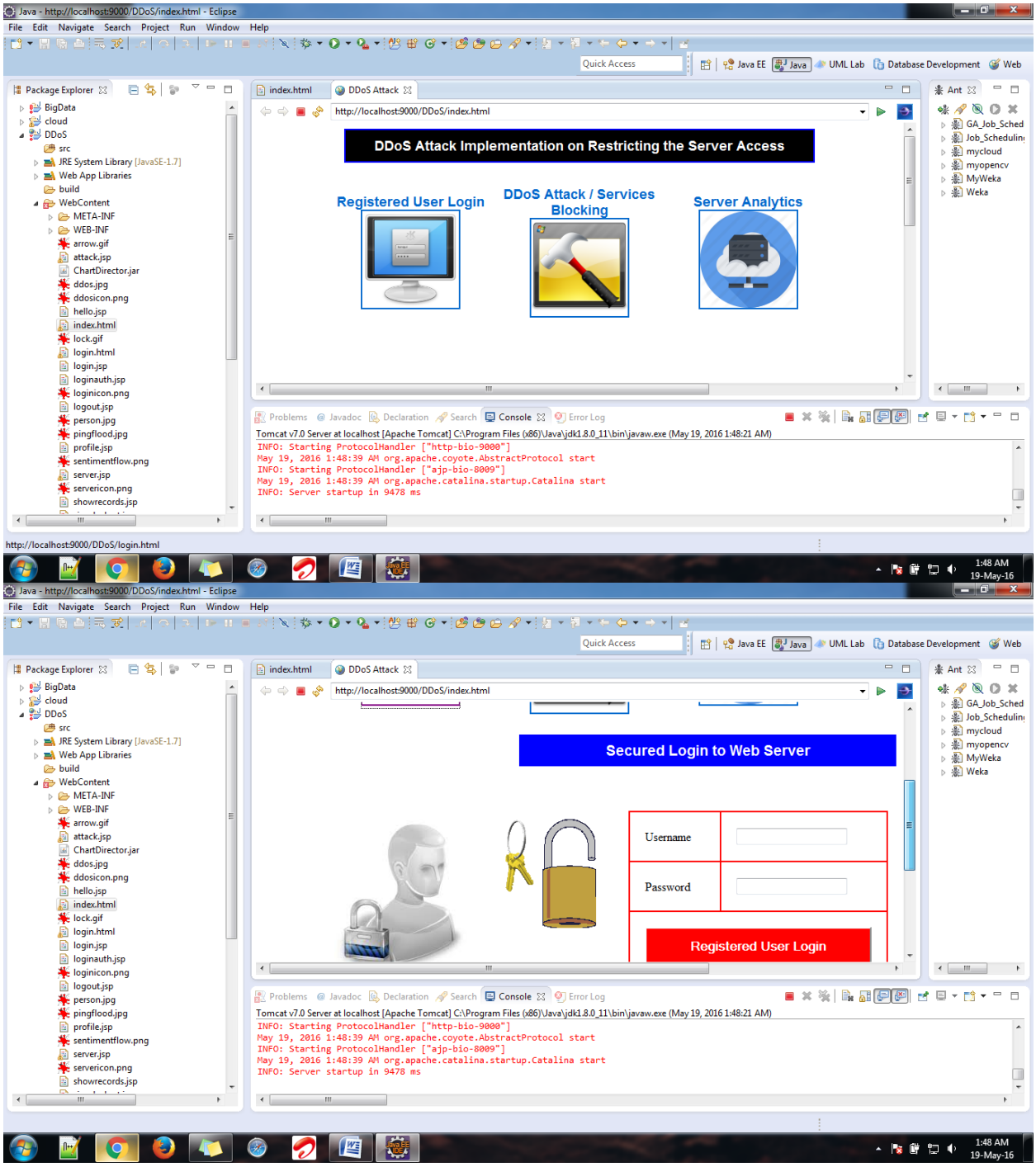
24 MB Data Packets Transmitted from Registered IP – DDoS Attack

3493 MB Data Packets Transmitted from Anonymous IP – DDoS Attack

232342 KB Data Packets Transmitted from IP 2.203.211.22– DDoS Attack

433 MB Data Packets Transmitted from IP 230.22.1.1– DDoS Attack


Java - DDoS/WebContent/index.html - Eclipse
File Edit Source Navigate Search Project Run Window Help
Windows Security Alert
Windows Firewall has blocked some features of this program
Windows Firewall has blocked some features of Java(TM) Platform SE binary on all public and private networks.
Name: Java(TM) Platform SE binary
Publisher: Oracle Corporation
Path: C:\program files (x86)\java\jdk1.8.0_11\bin\javaw.exe
Allow Java(TM) Platform SE binary to communicate on these networks:
Private networks, such as my home or work network
Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)
What are the risks of allowing a program through a firewall?
Allow access
Cancel
INFO: Initializing ProtocolHandler ["http-bio-9000"]
May 19, 2016 1:48:29 AM org.apache.coyote.AbstractProtocol init
INFO: Initializing ProtocolHandler ["ajp-bio-8009"]
May 19, 2016 1:48:29 AM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 4849 ms
May 19, 2016 1:48:30 AM org.apache.catalina.core.StandardService startInternal
INFO: Starting service Catalina
May 19, 2016 1:48:30 AM org.apache.catalina.core.StandardEngine startInternal
INFO: Starting Servlet Engine: Apache Tomcat/7.0.34
Tomcat v7.0 Server at localhost [Apache Tomcat] C:\Program Files (x86)\Java\jdk1.8.0_11\bin\javaw.exe (May 19, 2016 1:48:21 AM)
1:48 AM
19-May-16


Java - http://localhost:9000/DDoS/index.html - Eclipse
File Edit Navigate Search Project Run Window Help
Package Explorer
BigData
cloud
DDoS
src
JRE System Library [JavaSE-1.7]
Web App Libraries
build
WebContent
META-INF
WEB-INF
arrow.gif
attack.jsp
ChartDirector.jar
ddos.jpg
ddosicon.png
hello.jsp
index.html
lock.gif
login.html
login.jsp
loginauth.jsp
loginicon.png
logout.jsp
person.jpg
pingflood.jpg
profile.jsp
sentimentflow.png
server.jsp
servericon.png
showrecords.jsp
index.html
DDoS Attack
http://localhost:9000/DDoS/index.html
DDoS Attack Implementation on Restricting the Server Access
Registered User Login
DDoS Attack / Services Blocking
Server Analytics
Ant
GA_Job_Sched
Job_Schedulin
mycloud
myopencv
MyWeka
Weka
Problems Javadoc Declaration Search Console Error Log
Tomcat v7.0 Server at localhost [Apache Tomcat] C:\Program Files (x86)\Java\jdk1.8.0_11\bin\javaw.exe (May 19, 2016 1:48:21 AM)
INFO: Starting ProtocolHandler ["http-bio-9000"]
May 19, 2016 1:48:39 AM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["ajp-bio-8009"]
May 19, 2016 1:48:39 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 9478 ms
http://localhost:9000/DDoS/login.html
1:48 AM
19-May-16
Secured Login to Web Server
Username
Password
Registered User Login

CHAPTER 5

CONCLUSION AND FUTURE WORK

Distributed denial of service (DDoS) is a foremost hazard to the Internet these days. Valid users have a hard time connecting to the servers that are open to DDoS attacks. These attacks impersonate normal clients and consume the system resources at extensive scale, in this manner considerably refusing service to the normal nodes. This work proposed a DDoS defense solution which consigns distrust evaluation to a timestamp in fraction to its divergence from normal behavior and uses a DDoS resistance approach to choose whether and at what time the timestamp is utilized. Using an experimental simulation in Java, we verified the strength of DDoS attack as well as the effectiveness of DDoS defense capability to prevent it.

This work focus specifically on detection and avoidance of DDoS attacks on the network infrastructure whether it is concerned with the sniffing of packets or stealing the actual identify of the genuine node. However, there are number of techniques to address and remove this issue, but our approach is efficient enough because of the fact that this proposed approach is relying on the security trust architecture. It means there is a trust between the data transmission channel. The concept of mobile node and server agents are integrated to remove any probability of the attack. In the proposed approach implementation, better results in terms of jitter, packet loss and throughput are obtained in different simulation scenarios. In the proposed work, the current proposed approach can be further enhanced using unique approach and excellent technique for optimization on multiple parameters.

Integration of following techniques can be performed for making the current technique more efficient

- Bat Algorithmic Approach
- Swarm Intelligence
- Artificial Neural Networks
- Simulated Annealing

REFERENCES

[1] Ayman Mukaddam, Imad H. Elhajj, "Round Trip Time to Improve Hop Count Filtering," IEEE Symposium on Broadband Networks and Fast Internet, American University of Beirut, Lebanon, pp. 66-72, 28-29, May, 2012.

[2] A Wang, Xia, Li Ming, Li Muhai, "A scheme of distributed hop-count filtering of traffic," International Communication Conference on Wireless Mobile and Computing, pp. 516-521, 7-9 Dec.2009.

[3] B. Krishna Kumar, P.K. Kumar, R. Sukanesh, "Hop Count Based Packet Processing Approach to Counter DDoS Attacks," International Conference on Recent Trends in Information, Telecommunication and Computing, PET Engineering College, Thirunelvelli, India, pp. 271-273, 12-13, March, 2010.

[4] H. Wang, C.Jin and K. Shang, "Defense Against Spoofed IP Traffic Using Hop-Count Filtering," IEEE Transaction on Networking, vol. 15 (1), pp. 40-53, February, 2007.

[5] Biswa Ranjan Swain, Bibhudatta Sahoo, "Mitigating DDoS attack and Saving Computational Time using a Probabilistic approach and HCF method," IEEE International Conference on Advance Computing, NIT, Rourkela, India, pp. 1170-1172, 6-7, March 2009.

[6] B.B. Gupta, R. C. Joshi, and Manoj Misra, "Distributed Denial of Service Prevention Techniques" International Journal of Computer and Electrical Engineering, Vol. 2, No. 2, April, 2010 1793-8163.

[7] Bin Xiao, Wei Chen, and Yanxiang He, "A Novel Technique for Detecting DDoS Attacks at Its Early Stage**".**

APPENDICES

Index

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>DDoS Attack</title>

</head>

<body>

<div align=center>

<div style='border: 2px solid blue; color: red; padding: 10px; width: 500px;' align=center>

<strong>DDoS Attack Implementation on Restricting the Server Access</strong></div>

</div>


<table align=center width=80%>

<tr>

<td align=center style='border: 2px solid red; padding: 10px; width: 300px'>

<a href=login.html target=f style='text-decoration: none'>Registered User Login</a>

<td align=center style='border: 2px solid red; padding: 10px; width: 300px'>

<a href=attack.jsp target=f style='text-decoration: none'>DDoS Attack</a>

</table>
```

```
<div align=center>
<iframe name=f src="" height=600 width=1200 frameborder=0></iframe>
</div>

</body>
</html>
```

Attack

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>

<html>
<head>
<title>DDoS Attack Implementation on Restricting the Server Access</title>
</head>
<body>

<div align=center>
<h3 style='background-color: yellow; color: red'>Web Server Access Denied and Choking Done..... Garbage Values and Fake Traffic Generated</h3>
</div>
```

```
<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/ddos" user="root"  password=""/>

<sql:update dataSource="${snapshot}" var="result">
update user set currentlogin='1'
</sql:update>

<div align=center>
<img src=ddos.jpg>
</div>

<sql:query dataSource="${snapshot}" var="result">
SELECT * from user;
</sql:query>

<h3 align=center>All Registered Users Set to Already Login State</h3>
<table border="1" align=center>
<tr>
  <th>Username</th>
  <th>Password</th>
  <th>Status</th>
  <th>Current Login Status</th>
```

```
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
   <td><c:out value="${row.username}"/></td>
   <td><c:out value="${row.password}"/></td>
   <td><c:out value="${row.status}"/></td>
   <td><c:out value="${row.currentlogin}"/></td>
</tr>
</c:forEach>
</table>
</body>
</html>
```