uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Yao Chen

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Master of Computer Science

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

A Novel Marking-based Detection and Filtering Scheme Against Distributed Denial of Service Attack

TITRE DE LA THÈSE / TITLE OF THESIS

Amiya Nayak

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

A. El Saddik

Nicola Santoro

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

A Novel Marking-based Detection and Filtering Scheme Against Distributed Denial of Service Attack

by

Yao Chen

A thesis submitted to
the Faculty of Graduate and Postdoctoral Studies
in partial fulfilment of
the requirements for the degree of
Master of Computer Science

Supervisor: **Dr. Amiya Nayak**

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering
University of Ottawa
Ottawa, Ontario, Canada

©Copyright
2006, Yao Chen

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-494-14890-X
Our file Notre référence
ISBN: 0-494-14890-X

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

The Denial of Service (DoS) attack, including Distributed Denial of Service (DDoS) Attack, has become one of the major threats to the Internet today. The victim's resources are exhausted so that its services are disrupted under the DoS attack. Spoofed packets, in which the source IP addresses are forged, are usually used by attackers to implement the attacks or disguise their actual locations.

In this thesis, we investigate DoS attack, analyze some existing defense mechanisms, and compare their strengths and weaknesses. Then, we present a novel Marking-based DDoS Attack Detection and Filtering (MDADF) scheme. The MDADF system can distinguish and filter out spoofed IP packets by maintaining a record of the legitimate users and their markings. The system also functions as a DDoS attack detector. We evaluate the performance of this under various conditions in a simulated environment. The results demonstrate that the system is effective in defending against massive DDoS attacks, even when only 20% of the routers on the Internet participate in the marking process. The system is specially effective against IP-spoofed attacks, which are the most difficult to control, although it works well even under randomized attacks. Moreover, the system detects the occurrence of an attack quite quickly and precisely.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Dr. Amiya Nayak, for his invaluable guidance, enthusiastic encouragement, and strong support during my Masters studies.

I also want to thank Shantanu Das for his valuable suggestions, generous help and great support on my thesis work. I am grateful to Ming Xie for his constant help during my study period.

Finally, I feel extremely grateful to my parents, Jianjun Chen and Huizhu Zhang, my dear sister Zhang Chen, and my boyfriend Yuan Xue. Without their selfless support, encouragement and love, I would never have had the opportunity to pursue my studies.

Contents

1 Introduction **1**

1.1 Overview 1

1.1.1 The Internet 1

1.1.2 Denial of Service (DoS) Attack 2

1.2 Motivation 4

1.3 Objectives 6

1.4 Thesis Contributions 7

1.5 Thesis Outline 7

2 Denial of Service Attacks **8**

2.1 Introduction 8

2.1.1 Distributed Denial of Service (DDoS) Attack 9

2.1.2 Distributed Reflected Denial of Service (DRDoS) Attack 10

2.1.3 Internet Relay Chat (IRC) Based DDoS Attack 11

2.2 Overview of Major DoS Attacks 12

2.2.1 Smurf Attack 12

2.2.2 Fraggle Attack 13

2.2.3 Ping of Death Attack 13

2.2.4 Teardrop and Other Fragmentation Attacks 14
2.2.5 Land Attack 15
2.2.6 TCP SYN Flood Attack 15
2.2.7 Naptha Attack 16
2.2.8 UDP Flood Attack 17
2.2.9 ICMP Flood Attack 17
2.2.10 TCP Flood Attack 18
2.3 Classification of DoS Attacks 18
2.4 DDoS Attack Tools 19
2.4.1 Trinoo 19
2.4.2 Tribal Flood Network (TFN) 19
2.4.3 Stacheldraht 20
2.4.4 Shaft 20
2.4.5 TFN2K 20
2.4.6 Mstream 21
2.4.7 Trinity 21
3 Classification of Denial of Service Defense Mechanisms 22
3.1 Detection of DoS Attacks 22
3.1.1 Common Detection Strategies 22
3.1.2 Common Reaction Strategies 23
3.2 Prevention of DoS Attacks 24
3.3 Specific Defense Mechanisms 25

3.3.1 Ingress/Egress Filtering . . . 25
3.3.2 D-WARD . . . 25
3.3.3 PacketScore . . . 27
3.3.4 Source IP Address Monitoring . . . 28
3.3.5 Combating SYN Floods . . . 29
3.3.6 Route-Based Packet Filtering . . . 30
3.3.7 Neighbor Stranger Discrimination . . . 32
3.3.8 Pushback . . . 33
3.3.9 Redirecting Legitimate Users . . . 34
3.3.10 Proactive Server Roaming . . . 36
3.4 Classification of Defense Mechanisms . . . 38
3.5 Summary of Defense Mechanisms . . . 38
4 Identifying the Attacker **43**
4.1 Packet Marking . . . 44
4.1.1 Probabilistic Packet Marking (PPM) . . . 44
4.1.2 Deterministic Packet Marking (DPM) . . . 53
4.2 Message Traceback . . . 57
4.3 Logging . . . 58
4.3.1 CenterTrack . . . 58
4.3.2 Single Packet Traceback . . . 59
4.4 Traffic Observation . . . 60
4.5 Summary of Attack Identification Schemes . . . 61

5 A Novel Scheme for Preventing Denial of Service Attacks **64**

5.1 Marking Scheme 65

5.1.1 Computing the Packet Marking 66

5.1.2 Inserting Order information 67

5.2 Filtering Scheme 68

5.2.1 Basic Filtering Scheme 69

5.2.2 Route Change Consideration 71

5.2.3 Complete Filtering Scheme 72

5.3 Pushback Implementation 76

6 Analysis and Evaluation of the Proposed Scheme **78**

6.1 Practical Implementation 78

6.1.1 Internet Dataset 78

6.1.2 Simulation of Normal Internet traffic 79

6.1.3 Simulation of DDoS Attack traffic 79

6.1.4 Packet Filtering 80

6.1.5 Check List Verification Process 82

6.1.6 Parameter Selection 83

6.2 Results 85

6.2.1 Performance under Spoofed Attack 85

6.2.2 Performance under Random Attack 87

6.2.3 Attack Detection Time 88

6.3 Analysis 89

7 Conclusion & Future Work **93**

7.1 Summary 93

7.2 Future Work 95

List of Figures

2.1 DDoS Attack Model 10

2.2 DRDoS Attack Model 11

2.3 IRC Based DDoS Attack Model 11

3.1 Classification of Defense Mechanisms 39

4.1 Bit-Interleaving IP Address with Hash Value [42] 46

4.2 XOR Segments of Two Adjacent Addresses [42] 47

5.1 Internet Path Lengths Distribution (a) Skitter Map and Internet Map [51], (b) Three Skitter Monitors [52] 66

5.2 The Marking Scheme 68

5.3 The System Structure 69

5.4 The structure of the Filter Table and Check List 72

5.5 The structure of the Pushback List 77

6.1 Ratio of Accepted Packets vs. Different Amount of Attackers under Spoofed Attack 86

6.2 Acceptance Ratio Gap vs. Different Participation Percentage under Spoofed Attack 87

6.3 Ratio of Accepted Packets vs. Different Filter Table Size 88

6.4 Ratio of Accepted Packets vs. Different Number of Attackers under Random Attack 89

6.5 Acceptance Ratio Gap vs. Different Participation Percentage under Random Attack 90

List of Tables

3.1 Summary of Defense Mechanisms 40

4.1 Summary of Attack Identification Schemes 63

6.1 Accepting Ratio of Packets with Different Probability in Normal Time . . 87

6.2 Accepting Ratio of Packets with Different Probability in Attack Time . . . 88

6.3 Attack Detection Time for Different Number of Attackers 91

6.4 Attack Detection Time by Different Participation Rate 91

Chapter 1

Introduction

1.1 Overview

1.1.1 The Internet

The Internet, named ARPANET (Advance Research Project Agency Network), was created in 1969. It was designed for openness and scalability and was only used by researchers or universities for research purposes. In the Internet design, the communication between the sender and the receiver is through packets, which include source and destination addresses in their headers. Network security did not get any attention at the beginning. To make the message transmission cheap and fast, any user could send packets to another without any authentication of the network. Also, there was no special route for any two users to exchange packets. Instead, Internet resources are shared among network users, so that packets sent between two users may take various paths at different times. Receivers have to process the packets and authenticate their identities by themselves when necessary.

Until the late 1990s, the Internet was still used by limited users. However, in recent years, the Internet has experienced rapid growth. The number of computers connected

to the Internet has increased from about 0.3 million in 1990 to more than 353 million in the middle 2005 [29]. At the same time, the number of users has also increased to nearly 1 billion by November 2005 [30]. The Internet is no longer a tool for scientists or researchers. It has penetrated into many aspects of our life. Many important services, such as government information, power, banking, transportation, and medicine, are available through the Internet. Many companies rely on the Internet to provide information to their customers or partners, especially online businesses. Apart from the computer literate, a lot of today's Internet users have little knowledge about computers, networks, or security issues. They use the Internet for web-browsing, communication and entertainment, but are not concerned about securing their computers. Therefore, many attackers have taken advantage of this point. The number of reported Internet security incidents has increased from 252 in 1990 to 137,529 in 2003, and a lot more in last two years due to the use of automated attack tools [14]. The attacks include email viruses, computer worms, and the Denial of Service (DoS) attacks, which are most aggressive.

1.1.2 Denial of Service (DoS) Attack

A Denial of Service (DoS) attack is an attack with the purpose of reducing or eliminating the availability of a service to its legitimate users. One kind of DoS attack exploits the vulnerabilities, which can be the default configuration of a device or the bug in a software, a network protocol, or an operation system. The other kind of DoS attack aims at exhausting the consumable resources of the victim, such as the bandwidth of a network, the computation ability and memory capability of a computer.

The first kind of attack is always performed by sending certain specially modified packets to the target. The malicious packets will make the victim's computer/network malfunction, stick, or even shut down. Fortunately, vulnerabilities that exist in computers and networks are limited; they are patched, once they have been exploited by attackers, so that no further attack can make use of it again. Therefore, this kind of attack can be avoided in a great extent by updating computers in time.

In comparison, the second kind of attack is much more difficult to defend. It works by sending a large number of packets to the target, so that some critical resources of the victim are exhausted and the victim can no longer communicate with other users. Initially, to perform this kind of DoS attack, the attacker needed a more powerful machine than the victim – faster processor and wider bandwidth. Therefore, it is difficult for a single attacker to generate enough packets to deny service to the websites with adequate resources.

The new form of attack is the Distributed Denial of Service (DDoS) attack that uses thousands of compromised computers across the Internet to simultaneously attack a victim. In today's Internet, many computers have weak security measures to protect against intruders, while many users have not realized the importance of patching the vulnerabilities in their operating systems and software. Therefore, a large number of exploitable machines exist for the attackers to launch DDoS attacks, so that DDoS attacks can be executed by the attacker with limited resources against a large, sophisticated site. The attackers in DDoS attacks always modify the source addresses in their attack packets to hide their identity, or even pretend to be the legitimate users. More seriously, the tools

that the attackers use are more sophisticated and automated. These attack tools are readily available in many attacker forums and can be used by anybody, or even by any computer illiterate, to attack any website.

1.2 Motivation

Nowadays, the Internet has become an essential resource in our daily lives. Therefore, the DoS and DDoS attacks, whose targets range from individual users to large organizations, are getting more devastating.

- Many online businesses do their transactions using the Internet. Once their websites are disrupted by DoS attacks, they not only lose revenue but also credibility with their users.

- More and more important messages or public information are provided in the Internet. If these web-servers are suspended by DoS attack, it will bring people inconvenience or property loss.

- To some more critical Internet services, such as the Internet's Domain Name System (DNS) which translates the alphabetic human-readable web-addresses into IP addresses, the damage caused by DoS attack will be huge. Many websites will be unreachable because DNS cannot translate addresses for them.

DDoS attacks are quite common in the Internet now. Many Internet websites, including CNN, Yahoo, eBay and Amazon, have suffered from DDoS attacks.

Due to the seriousness of this situation, many mechanisms have been proposed to deal with DoS and DDoS attacks. These techniques are mainly in three fields:

- **Attack prevention** – This method aims at improving the security level of a computer system or network, preventing the attacks from happening, or enhancing the resistance to attacks.

- **Attack detection and reaction** – Attack detection intends to detect the occurrence of DoS attacks as soon as possible without misclassifying and disrupting legitimate traffic. After detection, the victim must react to the attack. To reduce the damage, it needs to identify attack packets, drop them, and protect its normal service to legitimate users.

- **Attack source identification** – This method tries to find out the sources of attacks. DDoS attackers usually use spoofed source IP addresses in their packets. By locating their actual addresses, the attacks can not only be stopped, it will also be possible to identify the attackers.

Although different defense mechanisms have been proposed, many issues still remain before any practical implementations on the Internet.

- The implementation cost should be low. Before the threats of DoS/DDoS attacks have been fully recognized by all the Internet users, it is difficult to get cooperation from them in defending the attack, especially when the investment is costly.

- Since attack traffic is usually indistinguishable from legitimate traffic, the ability

of timely detection and immediate reaction is important. Also, the conditions for attack warning must be set carefully to reduce the misinformation.

- Attack packets need to be differentiated precisely from the legitimate packets; otherwise, the refused packets from legitimate users become a kind of denial of service, which may serve the purpose of the attackers.

- The source identification need be quite efficient. Otherwise, it is hard to trace the multiple attackers existing in DDoS attack.

A good mechanism should take these problems into consideration. However, due to the complexity of DDoS attack, these problems still cannot be solved well enough. Therefore, a low-cost, efficient and precise scheme is still the aim of many researchers.

1.3 Objectives

The main objective of this thesis is to investigate the existing DoS/DDoS attack defense schemes and develop a more efficient mechanism, taking the practical issues into consideration.

Due to the distributed nature and devastating effect of DDoS attacks, any defense is hard for a single user or organization to achieve. It needs the cooperation from users in the Internet. However, most of them have low motivation to invest in extra equipments or services, so a practical solution must have low cost. In addition, to minimize the damages caused by DDoS attacks and to make sure the victim has enough ability to defend during attacks, the scheme should be able to detect attacks quickly and react properly.

Therefore, our aim is to develop a low-cost scheme that can detect an attack quickly and protect legitimate users.

1.4 Thesis Contributions

In this thesis, we first study the evolution of DoS attacks, major DoS attack types, DDoS attack tools, and propose a classification for them based on their properties.

Then, we categorize typical solutions to DoS attacks according to their objectives and compare different mechanisms, outlining their advantages and disadvantages.

Finally, we propose a novel Marking-based DDoS Attack Detection and Filtering (MDADF) scheme. It uses the packet marking technique to distinguish spoofed packets. It can detect high volume DDoS attacks no matter whether or not they use spoofed packets. We also take the path change into consideration, so that the services to legitimate users are guaranteed even if their paths become different.

1.5 Thesis Outline

This thesis is composed of seven chapters. Chapter 1 gives an overview of the Internet and DoS attack. Chapter 2 introduces the development and types of DoS attacks. Chapter 3 reviews some DoS detection and prevention schemes. Chapter 4 describes the techniques used in identifying the attacker. Chapter 5 presents a novel marking-based DDoS attack detection and filtering scheme. Chapter 6 gives a detailed analysis and evaluation of the proposed scheme. Finally, Chapter 7 concludes the thesis and outlines future work.

Chapter 2

Denial of Service Attacks

2.1 Introduction

DoS attacks started around the early 1990's with the purpose of denying the availability of a service to its legitimate users. The attack method can be exploiting vulnerabilities or flooding a network. Simple tools were used to generate and send packets from a single source to attack a single victim.

In early days, attackers often tested the network manually to find out vulnerable hosts and installed DoS tools onto them. Recently, the compromise process has been much more automated. Attackers use random scanning or selective scanning [27] to look for the vulnerable hosts. Random scanning, which is highly automated, usually searches random IP addresses for certain particular exploitable vulnerabilities in the operating systems or softwares. Selective scanning looks for vulnerabilities within the IP addresses known in advance and is less automated.

To automatically propagate among vulnerable hosts, DoS attacks install attack tools on the compromised hosts and use them as the attacking machines for further compromise. Central source propagation, back-chaining propagation, and autonomous propagation [27]

are three main propagation models. In the central source propagation model, the attack code is stored in a central location and downloaded by the newly compromised hosts. In the back-chaining propagation model, the attack code is transferred from the attacking machine to a newly compromised host. In the autonomous propagation model, the attack code is installed directly on a host when it is compromised, so that no file needs to be downloaded from external source.

Network worms have been used in the automatic scanning, exploitation, deployment, and propagation process of DoS attack tools. With the first appearance of "Ramen Worm" using the back-chaining propagation model in January 2001, many more worms have gradually emerged, such as the "1i0n Worm" using central source model and the self-propagating worms: "Code Red", "Code Red II" and many email viruses, including recently "MyDoom" and "Netsky".

In the evolution of DoS attack, the attack types have developed from the initial DoS attack to the distributed DoS attack, distributed reflector DoS attack, and the Internet Relay Chat (IRC) based DoS attack. Since the number of compromised systems determines the impact of a DoS attack, the deployment and propagation method is still developing.

2.1.1 Distributed Denial of Service (DDoS) Attack

Today, the most common attack type is the DDoS. It is a large-scale attack launched simultaneously by attackers through a large number of compromised computers on the Internet.

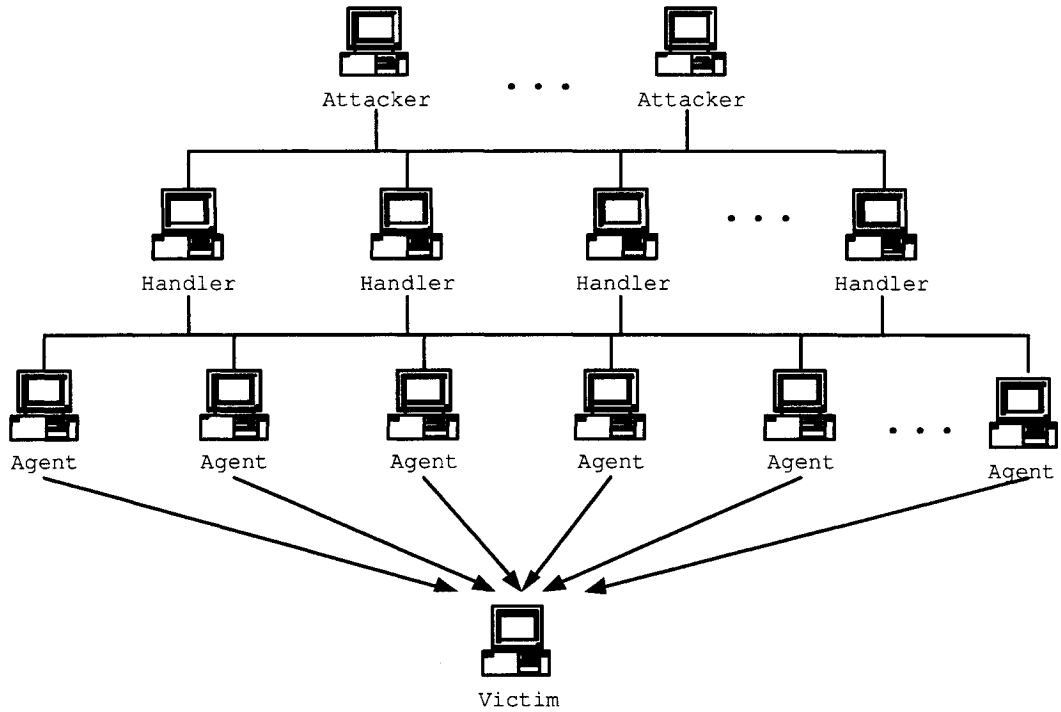

Figure 2.1: DDoS Attack Model

In the DDoS attack, the attacker scans the Internet to find out a number of vulnerable hosts as the handlers, connects to them and installs attack tools. Each handler is then used to further scan. The hosts compromised by handlers are called agents and responsible for generating the attack packets against the victim. The agents register their IP addresses with the handlers and accept commands from them.

2.1.2 Distributed Reflected Denial of Service (DRDoS) Attack

In the DRDoS attack, attackers do not use the compromised hosts to flood the victim directly, but through some other hosts called reflectors. They control the compromised hosts to send a large number of packets to many reflectors with spoofed source IP addresses of the victim, and all the reflectors send responds to the victim.

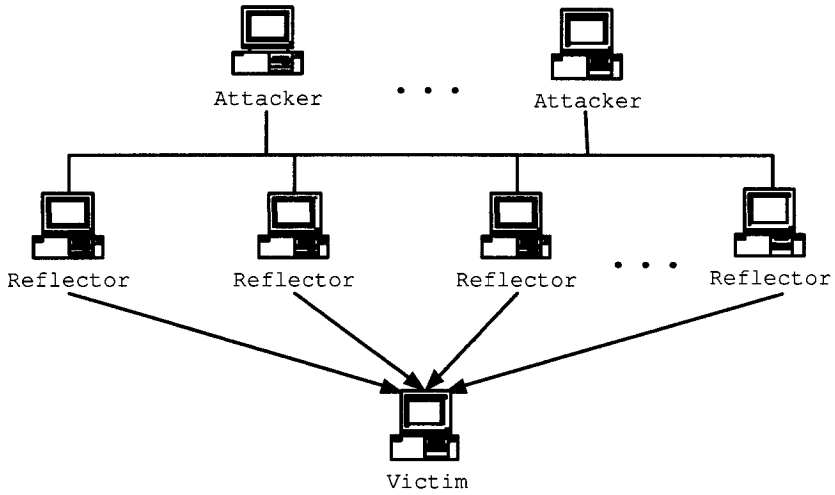

Figure 2.2: DRDoS Attack Model

Though the reflectors are innocent, they can amplify the attack effect. For example, a TCP server will send out SYN/ACK message repeatedly if it does not receive a response from the victim, thus the amount of attack packets simply grows many times. Also, the attack path becomes unclear due to the participation of reflectors.

2.1.3 Internet Relay Chat (IRC) Based DDoS Attack

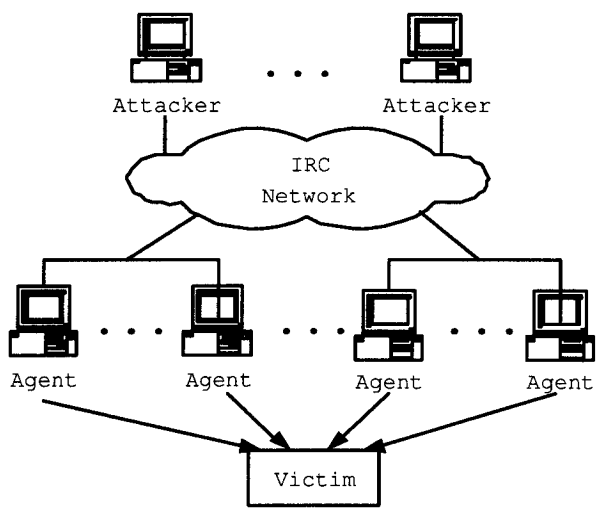

Figure 2.3: IRC Based DDoS Attack Model

In an ordinary DDoS attack, a handler usually keeps a list of agents that it can control, and an agent also stores the IP addresses of handlers that it can communicate with. Thus, once one of the handlers or agents is exposed, a large number of handlers and agents, even the whole attack network, can be discovered. The communications between attacker, handlers and agents can also be detected.

To make the traditional attack methods more sophisticated, distinctive, and harder to detect, the attacker began to use IRC protocols and networks as the platform of DDoS attack [27]. IRC is a multi-user, online chatting system, in which users can connect and communicate with another in realtime. An attacker can simply connect to an IRC server and control the hosts in the IRC channel as the agents, and the IRC communication channel acts as the handler. Since the attacker controls the agents through a legitimate network service and the agents do not listen to any port, they are not easily detectable. Even the IRC server can be identified if one or some of the agents are exposed, in order to dig out the whole DDoS network, the detector still needs to have the ability to know all the agents connected the server, which is especially difficult in private IRC networks. Therefore, it is easy for an attacker to set up IRC-based DDoS networks, while the networks are difficult to detect.

2.2 Overview of Major DoS Attacks

2.2.1 Smurf Attack

ICMP (Internet Control Message Protocol) message, which exchanges control messages and reports errors between hosts in the Internet, is utilized by the Smurf attack. ICMP

includes two kinds of packets: ICMP echo request, which is used to determine whether or not a host in the Internet is responding, and ICMP echo reply which is returned by that host if it receives the request. "Ping" is a common application of ICMP messages.

In the Smurf attack, the attacker sends ICMP echo request messages, such as ping, to a broadcast address with spoofed source IP address of the victim. All the hosts in this broadcast network respond to these messages and send ICMP echo reply packets to the victim, which floods the victim with a large number of packets.

2.2.2 Fraggle Attack

The Fraggle attack is similar to the Smurf attack, but uses UDP (User Data Protocol) echo packets instead of ICMP echo packets. UDP does not set up connections before data are sent and normally uses the broadcast method to transmit data, thus gives the attacker the opportunity to use broadcast addresses to attack the victim.

A variation of the Fraggle attack [46] utilizes the port supporting character generation. The attacker sends UDP echo packets, with the spoofed source address of the victim, to the character generation ports of hosts through the broadcast address. Each of these hosts sends a character to the victim, and the victim will reply back with an echo packet to the character generation port again. This process iterates, then a loop is formed. Therefore, this kind of Fraggle attack causes more serious damage to the victim.

2.2.3 Ping of Death Attack

In the Ping of Death attack [34], the attacker utilizes a property of IP packet, that it can be broken down into smaller segments for transmission, to make the victim crashed by

an illegal size IP packet. After an IP packet is broken down into segments, the attacker can intentionally combine an invalid offset to the last segment. Then when the segments are reassembled on the victim side, the size of the packet is more than the allowed maximum 65536 bytes. The oversized packet will overflow the buffer, thus the victim will crash, reboot or be inoperative.

2.2.4 Teardrop and Other Fragmentation Attacks

The Fragmentation attack [2] also takes advantage of fragmented IP packets, as Ping of Death, but exploits the vulnerability in the offset field of fragment. Normally, when a packet is sliced into fragments, the offset is set to 0 in the first fragment and a flag in its header is set to ON to indicate more fragments will be sent; the next fragments are sent with the offset field containing the sum of all previously sent fragments' lengths; and the flag is set to OFF in the last fragment to tell the destination the finish of fragments transmission.

Teardrop is a kind of fragmentation attack which sends fragmented packets with overloading offsets. The victim cannot reassemble these malformed fragments according to their offsets and eventually crashes or reboots.

Other illegal fragment configurations can also be exploited by the attacker. For example, the attacker can send a large number of fragmented packets without the end packets or with unmatched offsets, which forces the victim to allocate a lot of resources to keep the received packets and eventually causes resource exhaustion.

2.2.5 Land Attack

In the Land attack [10, 11], the attacker sends to the victim a TCP SYN packet in which both the source and destination addresses are the victim's address. Such a packet will lock the victim's system. If the target is a router, a routing loop may be formed to consume the bandwidth. The attacker can also use the port number of certain TCP service provided by the victim as the source and destination ports in the attack packets, causing very high CPU utilization on the victim side.

2.2.6 TCP SYN Flood Attack

The TCP SYN attack aims at consuming a server's TCP connection table to prevent other users from establishing new TCP/IP connections. A TCP connection is usually set up through a three-step handshake process. First, a user requests to set up a connection by sending a SYN (Synchronize) packet to the server. Second, the server replies to this packet with an ACK (Acknowledgement) and its own SYN request, both of which constitute a SYN/ACK packet, and the server keeps a record of this incomplete connection in a connection table. Third, after the user receives the SYN/ACK packet, an ACK is sent back to confirm. When the ACK is received, the server removes this record from the connection table. The handshake process now completes and the data exchange begins using the established connection between the user and the server.

In the TCP SYN attack [9], the attacker sends a large number of SYN packets to the victim with random spoofed source IP addresses. Since the source IP addresses are incorrect or nonexistent, it is impossible for the victim to receive ACK messages

after sending out SYN/ACK packets. The connection table has limited memory space, and some systems keep the uncompleted connections for a relatively long timeout period before clearing them out. Therefore, the victim easily runs out of memory and processor resources during the SYN attack, and requests for setting up connections from legitimate users will be lost. A TCP SYN attack is difficult to deal with because the SYN packets are a part of legitimate traffic that cannot be filtered thoroughly.

2.2.7 Naptha Attack

Naptha is a set of DoS vulnerabilities in TCP/IP services explored by BindView's RAZOR Security Team [12, 18]. A TCP connection has a total of eleven states: LISTEN, SYN SENT, SYN RECVD, ESTABLISHED, LAST ACK, FIN WAIT-1, CLOSE WAIT, FIN WAIT-2, CLOSING, TIME WAIT and CLOSED. When a TCP connection has been set up, it is in the ESTABLISHED state, and the user can begin communication with the server. If the user wants to finish the connection, he sends a FIN message to the server, and changes to the FIN WAIT-1 state. The server replies with an ACK and subsequently changes to CLOSE WAIT. The state of the user then becomes FIN WAIT-2 and TIME WAIT later when the FIN message from the server is received. Finally, the user responds with an ACK, and the server closes this connection.

In the Naptha attack, the attacker does not send any data to the server in the ESTABLISHED state, whereas it keeps sending more SYN messages to establish a large number of connections with the server. Also, the attacker may send a FIN message to the server to finish the connection and change its state to FIN WAIT-1, but does not reply the ACK

from the server. In this case, the server has to wait for its reply and it maintains the connection until timeout. Using both methods, the attacker can consume a large amount of the server's resources.

A Naptha attack can be implemented anonymously and it is not easily detected because the TCP connection is an essential part of normal traffic. Moreover, the attacker can establish and respond to large numbers of TCP connections by using very few resources of itself.

2.2.8 UDP Flood Attack

In a UDP flood attack [8], the UDP packets are sent to random ports of the victim. When the victim receives a packet, it will try to determine what application has sent out the request. When the victim has no application running on that port, it will send back an ICMP message "destination port unreachable". The source addresses of these UDP packets are always spoofed to hide their identities. If a large number of UDP packets are sent to the victim, its bandwidth will be exhausted.

2.2.9 ICMP Flood Attack

The ICMP flood attack is formed by large volumes of ICMP echo request messages sent to the victim. As a result, the victim does not have the ability to process all requests and respond quickly, so that the bandwidth will be occupied and legitimate traffic cannot go through. The source addresses in these packets may be spoofed.

2.2.10 TCP Flood Attack

TCP flood acts similarly to the UDP and ICMP flood, which sends a large number of TCP packets to flood the victim.

2.3 Classification of DoS Attacks

Basing on the properties of the aforementioned attacks, they can be classified into three categories:

1. **Protocol Attacks**

   Smurf and Fraggle attacks make use of the design of commonly used network protocols to amplify their attack effects.

2. **Vulnerability Attacks**

   Ping of Death, Teardrop, and Land attacks exploit the weaknesses in network software implementations to crash the victim.

3. **Bandwidth Depletion Attacks**

   TCP SYN flood, Naptha, UDP flood, ICMP flood, and TCP flood are the examples of bandwidth depletion attacks, in which the attacker sends a large number of packets to consume the resources of the victim. Protocol attacks can also be classified as bandwidth depletion attacks.

2.4 DDoS Attack Tools

Several DDoS tools began to surface in 1999. The best known DDoS tools include Trinoo, Tribal Flood Network (TFN), its derivative called TFN2k, Stacheldraht, Shaft, Mstream, and Trinity. Since the available vulnerabilities in IP packets and Internet protocols are limited, most of the attack types are still used and work well. However, the control of compromised hosts and the attack effectiveness of attack tools have developed. All of these tools use handler/agent architecture, which allows a single attacker to launch attack by simultaneously conducting many machines, but their approaches are different.

2.4.1 Trinoo

Trinoo [22] is the first and simplest DDoS tool which appeared in mid 1999. In this tool, the attacker establishes a TCP connection to the handlers using telnet and controls them through the TCP connection. Communication between the Trinoo handler and agents is via unencrypted UDP packets. Both the handlers and agents are password protected to prevent the Trinoo network from being controlled by others. The attack is initialized by the agents simultaneously and floods one or more targets with UDP packets. Trinoo does not use spoofed source IP addresses.

2.4.2 Tribal Flood Network (TFN)

The TFN attack tool [24] appeared after Trinoo. In additional to the UDP flood attacks used in Trinoo, a TFN can also generate TCP SYN flood, ICMP flood, and Smurf attacks. In TFN, the communication between the attacker, handlers, and agents

is accomplished using ICMP echo request and ICMP echo reply packets. The absence of TCP or UDP based communication between the handlers and agents makes the traffic difficult to be detected. TFN has the capability to generate packets with random source IP addresses and source ports, and the packet size can be altered.

2.4.3 Stacheldraht

Stacheldraht [23], which means "barbed wire" in German, is based on the source code of TFN and it combines features of the Trinoo and TFN. It can also generate UDP flood, TCP SYN flood, ICMP flood and Smurf attacks. It improves TFN by encrypting the communication between the attacker, handlers, and agents.

2.4.4 Shaft

Shaft [20, 21] is similar to Trinoo and it uses UDP to communicate between handlers and agents. The attacker uses a telnet program to connect to and control the handlers. The access to handlers and agents is password protected. Shaft has the ability to generate TCP SYN, UDP, ICMP flood packets or their combination. The agent controls the packet sizes, attack type, duration of the attack, and a set of target victims.

2.4.5 TFN2K

TFN2K [3] is a variant of TFN. It not only generates the flood attacks as in TFN, but also uses malformed or invalid packets to crash the victim. The control commands are encrypted, instead of the clear-text form in TFN. TCP, UDP, and ICMP packets are used randomly, so the detection and packet filtering become inefficient and difficult to achieve.

2.4.6 Mstream

Mstream [15, 25] uses TCP ACK packets, whose source information and destination ports are random, to flood one or more target victims. The communication and remote control from the attacker to the handler is accomplished via an unencrypted TCP connection. Communication between handlers and agents uses unencrypted UDP packets. Access to the handler is password protected, but the password is in clear-text and it is not encrypted.

2.4.7 Trinity

Trinity [36] is more complex than previous tools, in which the control and communication to compromised machines are via Internet Relay Chat (IRC) channels or America Online Inc.'s ICQ online chat service. Once Trinity connects to the IRC channel, the agents there will wait for the commands from it. Since the agents do not listen to any port, they are difficult to detect. Trinity is capable of launching several types of flooding attacks on a victim, including SYN, UDP, and other floods.

Chapter 3

Classification of Denial of Service Defense Mechanisms

3.1 Detection of DoS Attacks

3.1.1 Common Detection Strategies

Two techniques are usually used to detect DoS/DDoS attacks: *anomaly detection* and *pattern detection.*

Anomaly detection is a method used most frequently in DoS/DDoS detection. Since some of the traffic attributes are usually stable to a user or network in normal conditions, anomaly detection uses this property to detect certain behavior which is different with that in normal conditions and looks it as an indication of the occurrence of attack. In order to detect abnormality, the behaviors under normal conditions must be known by the system before it begins to detect attacks. Therefore, a learning phase is necessary and the learning results need to be stored. The storage, computation overhead and/or transmission delay are inevitable with this approach, when comparing current traffic with that under normal conditions. In addition, if the data used during the learning time is not precise enough, the detection result will not be accurate.

Pattern detection is another kind of detection approach. The properties of detected attacks are stored and used to compare with the current traffic. This method can identify known attacks quite easily and precisely, but cannot deal with new attacks.

3.1.2 Common Reaction Strategies

After detecting a DoS/DDoS attack, there are four possible ways to alleviate the effect of an attack or confront the attack traffic: *rate-limiting*, *packet filtering*, *reconfiguration* and *agent identification*.

- Rate limiting is to limit the rate of suspicious flow. Since it is always difficult for a system to distinguish good packets from attack packets, especially when the packet is from a user who initiates a connection when an attack is occurring, the new user has to compete with attackers to use resources until it is considered good and protected. Therefore, the main problem of rate-limiting is how to set the limit so that certain unprivileged packets are allowed to pass through while attack traffic is blocked as much as possible. Also, passing packets through the rate-limiting process causes delay in transmission.

- Packet filtering is to filter out the attack flow completely. The mechanisms using this method must have the ability to recognize attack flow precisely; otherwise, legitimate traffic will be affected. The attacker can also use this method to deny service to certain host by spoofing packets from that address.

- Reconfiguration means protecting service to privileged users by changing system/network parameters, topology, or resources in the event of an attack. However, similar with

rate limiting, it is difficult for the new user to get normal service in the beginning. Also, the changing process is complex and the new configuration must be kept secret from attackers.

- The purpose of agent identification is to identify the actual source(s) of attack, even spoofed addresses are used in attack packets. Then the attack can be prevented from continuing, and the attackers can be prosecuted.

3.2 Prevention of DoS Attacks

A good defense against DoS attack is to enhance the resistance of a system or network and protect the service from being denied. Besides some general security mechanisms, such as firewall, intrusion detection systems (IDS), access control and updating with security patches, *resource multiplication* and *resource accounting* can also be used.

- Resource multiplication means providing enough resources to withstand DoS threats. Though this method is useful in reducing the effect of DoS attack, multiple resources bring heavy overhead on expense and management. Also, if the attack traffic is quite aggregate, the multiple resources may still not be enough to resist.

- Resource accounting means limiting the access to resources basing on the user's privileges. The verification of user's identity is quite important here.

3.3 Specific Defense Mechanisms

3.3.1 Ingress/Egress Filtering

Ingress filtering is proposed to defend DoS attacks at the victim network. The filter prevents packets with illegitimate source addresses coming into the internal network. The illegitimate address can belong to one of the internal network that is impossible to come from outside or it is not one of the users' addresses, all of which the network has known. However, it is not effective when the spoofed address is one of the legitimate users'.

Egress filtering is a simple but useful technique deployed at source networks. When a packet comes out of a network, the egress filter will check its source address to make sure the address is inside the network. If the address belongs to somewhere else, it's forged and should be dropped. Therefore, egress filtering can prevent internal hosts from involving in attacks. Though an attacker can still spoof the address of someone else in the same local network, the scope of addresses can be spoofed is much smaller.

3.3.2 D-WARD

Mirkovic et al. [37] proposed a D-WARD (DDoS Network Attack Recognition and Defense) system, which is deployed at the border routers of source networks (stub networks or ISP networks), to defend DDoS attacks by autonomously detecting and stopping attacks from flowing out.

D-WARD consists of observation and throttling components. The observation component monitors two-way traffic between the source network and others in the Internet. It observes their communications and looks for signs of communication difficulties, such

as the decrease of response packets or longer response time. The traffic with each address are observed and periodically compared with the predefined normal traffic model. The comparison results are passed to the throttling component. If the comparison reveals the possibility of a DDoS attack, D-WARD imposes a rate limit on the suspicious outgoing flow according to its offensive. If a suspicious flow is confirmed to be an attack, D-WARD limits its rate further. Otherwise, good behavior leads to a recovery of the traffic rate allowed for the flow.

D-WARD can effectively detect many common DDoS attacks and limit their rates. However, it needs longer detection time if the attack increases its sending rate slowly. Also, during an attack, the new user has to compete for the bandwidth and suffer packet losses until it establishes a good credit. Moreover, an attacker can deny the outgoing bandwidth to legitimate users by preventing the response packets from reaching the D-WARD system or disguising its packets among legitimate packets to form aggressive traffic, so that D-WARD will think the legitimate user is generating a DDoS attack and limit the rate of relevant outgoing flows. An attacker can also intentionally avoid detection by D-WARD. One of the methods is disguising respond packets from certain legitimate users so that D-WARD will not consider its flowing out packets as attack flows, though many more resources has to be gathered by the attacker to achieve this goal. In the other way, the attacker uses UDP packets, which always do not require responds, to form an attack by spoofing a small number of addresses and distributing the attack sufficiently, so that the D-WARD cannot detect abnormity.

3.3.3 PacketScore

Kim et al. [33] proposed a PacketScore architecture which automatically detects DDoS attacks with the cooperation of a DDoS control server (DCS) and several distributed Detecting-Differentiating-Discarding routers (3D-Rs). It estimates the legitimacy of packets according to the scores of their attributes and filters packets basing on the score distribution and congestion level of the victim.

The 3D-Rs compute the scores for suspicious packets by comparing their attributes with the victim's normal traffic and estimate their legitimacy. The DCS receives traffic statistics from 3D-Rs and decides whether an attack is ongoing. The attributes of a packet used in detection include: packet size, protocol type, TCP flag pattern, server port number, TTL value, 16-bit source IP prefix, etc. These values are usually stable under normal conditions, then the unusual attribute values possibly imply the occurrence of an attack. The score, conditional legitimate probability (CLP), of each packet is computed basing on some "scorebooks", which map the attribute values to different scores and are updated periodically or after obvious changes. When the amount of incoming packets exceeds the victim's ability to deal with, a fraction of them are dropped. The fraction is determined using a cumulative distribution function (CDF) of the CLP, which is influenced by the victim's congestion level and the offensive of the attack flow.

This PacketScore mechanism is scalable and easy to be hardware implemented. However, it has the limitation that it cannot detect attacks which have low traffic volume.

3.3.4 Source IP Address Monitoring

A source IP address monitoring (SIM) approach is proposed by Peng et al. [41] to detect highly distributed DoS attacks by deploying monitors in the perimeter of a network to observe the growth of new IP addresses. This approach can be deployed at both the source or victim network. Because all the valid IP addresses should be within the same network, attacks can be easily detected at the source.

In the training phase of this approach, legitimate IP addresses are added to an IP address database and kept updated. To detect the attacks, several statistics, such as the number of new IP addresses, about incoming traffic in a certain period are collected. Since a large number of hosts are usually utilized by attackers, and most of their addresses are new to the victim, a DDoS attack can be detected by analyzing these statistics data. A cumulative sum method is used to detect attacks. If the accumulated value of the new IP addresses is larger than the attack detection threshold, it indicates that an attack is ongoing. Since a monitor can only see part of the attack traffic, the increase in the number of new IP addresses may not be enough to be a proof of attack. Two or more monitors can share their beliefs about potentially suspicious traffic to get more precise detection of attacks. Thus, each monitor broadcasts when detects a suspicious traffic, and include information obtained from other monitors in its evaluation.

Using accumulated value in the attack detection, the SIM mechanism can detect the attack with either explosive or constant rate. There are two issues if using multiple monitors: how to combine and when to share beliefs, so the cost in information sharing

and the delay in confirmation are introduced. Also, this mechanism is not effective when only a few of new IP addresses are used by the attacker.

3.3.5 Combating SYN Floods

Gemona et al. [26] proposed an architecture to protect the victim from SYN flood attacks. It uses online passive monitoring techniques to observe traffic coming into the network and acts actively when an attack is detected to close the corresponding connections.

This architecture includes a connection layer which contains an entry for each open connection and a host layer above it which maintains a table of the hosts in local network. A monitor identifies which connection each received packet belongs to, updates the connection's state, and keeps account of the number of connections in each state for each host. The connections are distinguished according to their source/destination IP addresses and port numbers. By recording the number of connections and their states, especially those in open TCP connections, the monitor can identify whether a host is under an attack and transfers from a passive mode to active when an attack happens. Then, it filters out the connections belonging to the attack and allows legitimate connections to complete. In a SYN attack, the attacker sends only SYN requests to the victim to exhaust its connection queue. Then, if a host can send and receive data normally, the packets sent from it are unlikely to be part of the SYN flood attack and this host can be looked as a safe host. For the packets from an unknown host, a processing queue is held. The packets in this queue will be forwarded in the normal way, but will be monitored and looked as suspect.

If a previous packet in this queue does not get a response before a latter one, it is treated as part of the attack and a reset packet is generated to close this connection. Otherwise, this host and all future connections from it are treated as safe. Moreover, the Pushback mechanism (see Section 3.3.8) can be used so that the upstream routers can help to limit the rates of SYN packets.

This architecture protects the server from exhausting all its connection queue so that the communication with legitimate users will not be disrupted. The first time user can also receive normal services after the completion of the first connection. Some SYN flood packets could come through, but should not affect the service seriously and would be cleaned up quickly. However, it requires storage spaces to record information for each connection, each host and the processing queue. There is also administration cost in checking each packet.

3.3.6 Route-Based Packet Filtering

Park and Lee [39] proposed a route-based distributed packet filtering (DPF) approach which uses the reachability of a source address to proactively filter out a large fraction of spoofed packets before they reach the victim and also reactively find the origins of the unfiltered spoofed packets within a small scope. Its effect will be significant if it is implemented on less than 20% of Internet autonomous system (AS) sites. The network topology and the location of filters have strong effect on this approach.

Packet filtering controls on the data coming in/out a router or firewall basing on the source/destination addresses and port numbers of packets. The border router of

an AS checks the source address of a packet (regardless the destination address), using the reachability constrains imposed by routing and network topology, to see whether it is possible for this packet to arrive on its incoming link. However, a filter can only detect the spoofed packets whose source addresses belonging to absolutely another link. If a packet spoofs an address in the same link with it, it cannot be detected. Thus, the cooperation of ASs is needed. In order to minimize the overhead and improve the efficiency, DPF should be implemented on the ASs which cover the most remaining uncovered links. In the power-law structure which represents the Internet AS topology, the connectivity of ASs is centralized at some centers while most other nodes are loosely connected, so these centers become the best places to implement DPF. By using proactive filtering, most of the spoofed packets can be filtered and the resources can be used by attackers become quite limited. Also, the attack site can be localized much easier in a small scope using the reactive filtering.

The DPF approach requires the cooperation of autonomous systems, so its implementation is still a problem. The attacker can still spoof IP address, though within a much smaller scope. Sometimes there are multiple paths to route a packet from its source to destination, then it will be easier for attackers to send spoofed packets and the sources become more difficult to locate. The efficiency of DPF decreases in this condition. Another critical problem for implementing DPF is that the current IP routing protocols are destination-based, while DPF needs to check the source addresses of packets. Therefore, a protocol that can efficiently provide the precise source information is needed to implement the route-based DPF.

3.3.7 Neighbor Stranger Discrimination

Aaaraj et al. [1] proposed a neighbor stranger discrimination (NSD) approach to alleviate the effect of large scale DDoS attacks. In NSD, the router, called "signing router" or "NSD router", performs signing and filtering besides routing.

All networks are divided into two classes by the NSD router: neighbors and strangers. If the packets from a network reach the NSD router directly without passing through other NSD routers, this network is considered as neighbor network. Two NSD routers are neighbor routers if the packets between them do not need to transit other NSD routers. Therefore, a packet received by an NSD router must either from a neighbor networks, or from a neighbor router. Each NSD router keeps an IP addresses list of its neighbor networks and a signatures list of its neighbor routers. When an NSD router receives a packet, if its source address is in the IP addresses list, the packet is legitimate. Otherwise, the NSD router will check the Identification field of this packet, which stores the 16-bit signature of neighbor routers. If the signature is in its signatures list, this packet has already been authenticated by one of the neighbor routers and it is legitimate. For a legitimate packet, the NSD router writes its own signature in the Identification field of the packet and forwards it. If a packet satisfies neither of the conditions, it is looked as illegitimate and dropped. The IP addresses list and the signature list are learned by each router under normal conditions, when no attack is happening. In the learning phase, the NSD router signs its signature to packets with 0 in the Identification field, which means the packets have never been signed, and saves their source address in the IP addresses

list. In this way, all packets will be signed by the first NSD routers they pass. Thus, each NSD router can have the signatures of neighbor routers and store them in the signatures list.

The packet with spoofed source address is difficult to pass through the NSD routers. However, the number and location of signing routers have influence on the effect of this approach. If the IP addresses list and the signature list are not precise enough or the deployment is not wide, large numbers of legitimate packets will be dropped. Therefore, more central routers should be used. These are storage overhead to each NSD router and delay in transmission, caused by checking the IP addresses or signatures of packets and signing them.

3.3.8 Pushback

An aggregation detection algorithm, pushback, was proposed by Ioannidid and Bellovin [31] which is based on the IP destination addresses of packets. In the Pushback scheme, a router notifies its upstream routers when it detects an attack, then the upstream routers drop such packets so that the legitimate traffic will be affected less.

When the congestion level of a network is high, such as the total incoming bandwidth is 1.2 times of the output bandwidth, the pushback algorithm begins to match the destination address of each dropped packet against the routing table, and selects the longest matching prefix, which constitutes the congestion signature. In the router, a rate limiter is added to decide whether a packet is dropped or forwarded. It takes effect when the incoming traffic exceeds a threshold and drops the packets matching the congestion sig-

natures. The excess packets are also dropped by the output queue. A pushback daemon receives dropped packets from both the rate limiter and output queue. It analyzes the number of dropped packets, determines whether there is an attack going on and how to react it. Then it updates the parameters of the rate limiter and informs upstream routers periodically. It also listens to requests from downstream routers. If dropping the packets matching this congestion signature cannot effectively reduce the incoming traffic, it is likely that more than one attack is happening. This algorithm needs to be performed repeatedly to find out more prefixes. If no such prefix can be found out, it means the traffic is not caused by an attack, but by an increase of general traffic. Multiple detection algorithms can be used together to increase the accuracy of detection.

Pushback can be used as complementary to many other DoS detection approaches. However, the effectiveness of pushback lies on a precise inference of the congestion signature. Also, the pushback requests must be authenticated to avoid being spoofed by attackers. There are storage cost for the dropped packets, computation cost for finding out the prefix, and the overhead caused by the messages sending between routers.

3.3.9 Redirecting Legitimate Users

Xu and Lee [50] designed a defense system in which the victim can distinguish legitimate traffic by using HTTP redirect message and MAC when an attack occurs. To the attack packets using true IP addresses, they will be rate-limited or even blocked according to their behaviors later. In this way, the services for legitimate users are ensured to be available. The protected website is connected to the Internet through a firewall. Some

upstream routers help to filter certain attack packets by getting instructions from the firewall.

When an attack occurs, the system will redirect its users to a pseudo-IP address and port number by sending standard HTTP URL redirect messages, which include secret MAC keys. After receiving the redirect message, the legitimate user will have the MAC inside the destination IP addresses of its packets. The secret MAC key is only shared by the perimeter routers and the firewall, so that the perimeter routers can check the validity of MACs. When the server receives a SYN packet from an unknown address, it sends back a SYN+ACK packet with the SYN cookie, acting as the other kind of MAC. The user having correct SYN cookies in its packet will receive the redirect message. The packets with spoofed source addresses will not receive the MACs, then will be filtered by the perimeter routers. By using the redirect message and SYN cookie, the packets with genuine sources are distinguished and the availability of services are sustained for them. To avoid the case that the MAC known by attackers, the key can be changed regularly, by deriving from the previous key using a secure hash function. For the attack packets using true IP addresses that cannot be distinguished in the previous step, Deficit Round Robin (DRR) is used as the packet scheduling algorithm. If a user sends packets faster than the normal traffic, its excessive packets will be dropped. If the amount of dropped packets exceeds a threshold, this IP address is put into the blacklist and all the packets from it will be dropped by the perimeter routers before reaching the firewall. If an attacker keeps sending low volume traffic, but the total number of packets exceeds the amount a normal user can send at most, it will be given a much smaller bandwidth limit. However,

it is still allowed to use excessive bandwidth if there is any. The bandwidth given to the non-legitimate users need to be set appropriately, so that the new users can get their first packets through.

This system promises the availability of services to legitimate users even when an attack occurs. Only the user who initiates a connection during an attack will be delayed. The firewall needs to perform network address translation (NAT) on the pseudo IP address and port number to get the actual ones when sends and receives a packet. The upstream routers also need to check the legitimacy of MAC key within each packet. The communication between the firewall and perimeter routers needs to be protected.

3.3.10 Proactive Server Roaming

Khattab et al. [32] proposed a proactive server roaming scheme, in which the active server changes its location within several distributed homogeneous servers proactively to prevent the occurrence of DoS attack or mitigate its effect. Quality-of-service (QoS) technique is used to distinguish and protect legitimate users when attack happens and ensure only legitimate users can follow the server when it roams. A secure roaming algorithm determines when to roam and where the new active server is, which reduces the possibility of being predicted or followed by attackers.

Each legitimate user has a QoS contract with this service. Only the legitimate users will know the server's roaming time and the address of new server. The roaming algorithm is based on backward hash chains and uses the generated number as the key to decide the location of active server and the period it keeps active. In offline registration, a shared

secret key is generated for each user to setup a secure channel with the system, through which the roaming update messages are sent. The user's information, such as the IP address, port number for receiving roaming update messages, the shared secret key, and how long it can follow the server, is encrypted and accessible to all servers. The users store periodic state checkpoints of the server and do the state recovery at the new server instead of periodic server-to-server updates. State recovery messages are sent using the secure channels to avoid the forged messages from attackers. Only the current active server accepts incoming connections. All the attacking connections are dropped when the server roams, so the legitimate users can get services at least in the beginning of each roaming epoch before the attacker finds out the active server again. If a legitimate user keeps violating its QoS contract, it will lose its legitimate statue.

When undetected attack happens, the proactive roaming scheme ensures the server to provide service to legitimate users. If a user is compromised by the attacker and sends attack packets which violate its QoS contract, it can be easily detected, so that the server is protected by refusing service to it. This scheme can be integrated into other DoS defense systems. The packets received by inactive server, especially the last active server, can be considered as the attack traffic. Without mixing with legitimate traffic, the source of attack can be found more efficiently. However, the expense of keeping and maintaining multiple resources is high. Server roaming and state recovery bring heavy overhead. Moreover, if the attack traffic is aggregate enough, it can send a large volume of packets to all servers in this system, then the active server will inevitably be attacked.

3.4 Classification of Defense Mechanisms

All DoS defense mechanisms can be classified using the following categories: *detection*, *reaction*, and *mitigation*. The detection deals with both anomaly and pattern detection. The reaction involves filtering, reconfiguration, rate limiting and identification. The mitigation is concerned with resource accounting and multiplication. Figure 3.1 shows the classification of mentioned defense mechanisms (the identification mechanisms are introduced in Chapter 4).

3.5 Summary of Defense Mechanisms

A summary of these defense mechanisms is given in Table 3.1.

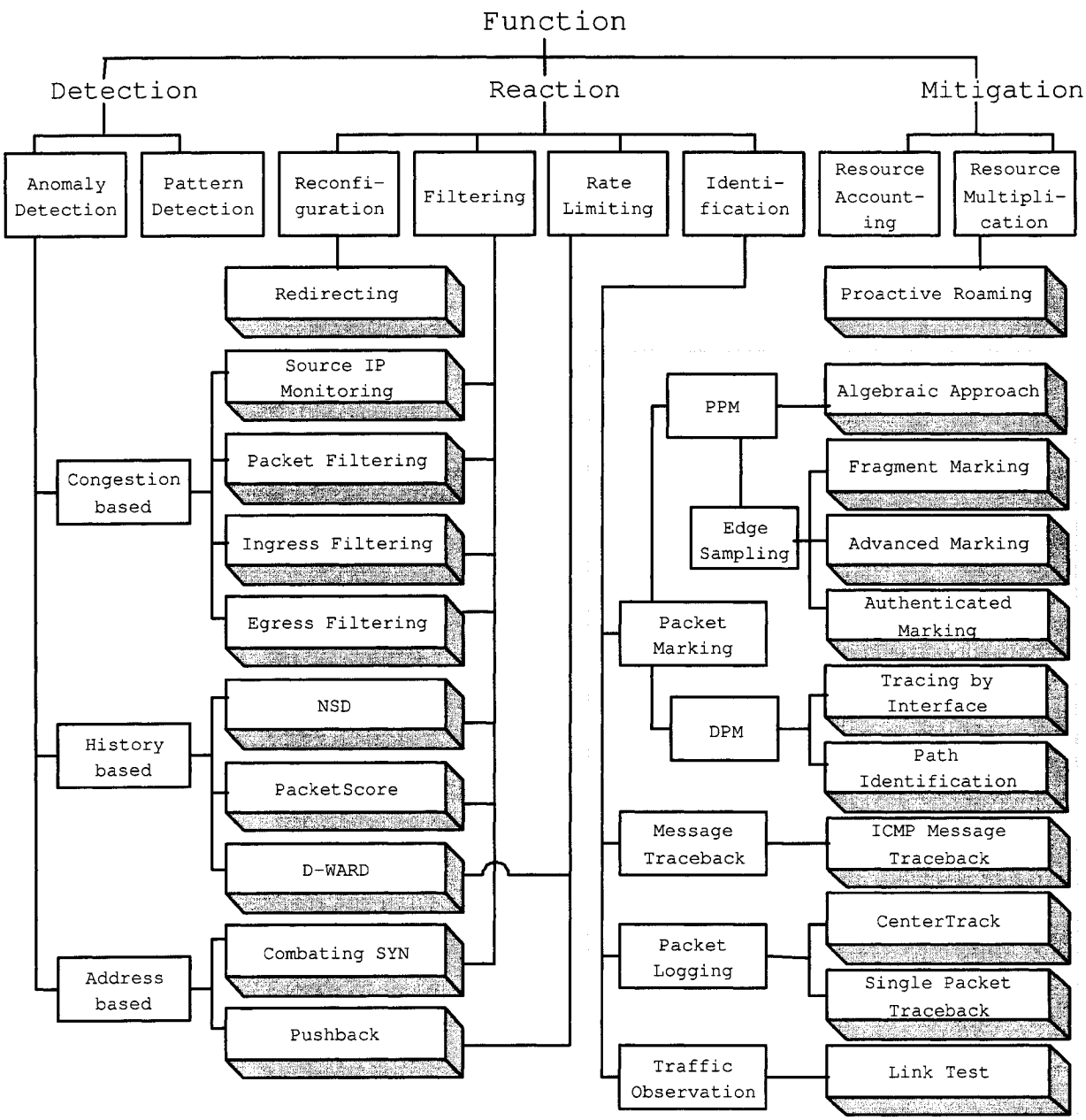

Figure 3.1: Classification of Defense Mechanisms

Table 3.1: Summary of Defense Mechanisms

Schemes	Disadvantages	Advantages
Ingress/ Egress filtering	1. Need to keep a list of legitimate users 2. Not applicable to attacks using spoofed source addresses belong to legitimate users 3. Need wide deployment	Can reduce address spoofing
D-WARD	1. Need longer time to detect the attacks which increase their sending rates slowly 2. Difficult for new users to setup connection 3. Could be utilized by attacker to DoS legitimate users 4. Precision of the normal model affects the detection 5. Delay in transmission and storage overhead	Can effectively detect many common DDoS attacks
PacketScore	1. Cannot detect attacks having low traffic volume 2. Precision of the normal model affects the detection 3. Delay in transmission, computation and storage overhead	Scalable and easy to be hardware implemented
Source IP Monitoring	1. Not effective to the attack which uses only a few new addresses 2. Cost of storage and information sharing, delay in attack confirmation	Can detect the attack with both explosive and constant rate
Combating SYN	1. Only apply to TCP SYN attack 2. Difficult for new users to setup connection 3. Legitimate packet could be dropped 4. Storage and administration cost	Protect the server from exhausting all its connection queue

Table 3.1 (continued): Summary of Defense Mechanisms

Schemes	Disadvantages	Advantages
Route-Based Packet filtering	1. Need route information 2. Only applicable to attacks using spoofed source IP addresses 3. Attackers can still spoof address within a small scope 4. Precise source information is still not available 5. Need wide deployment 6. Deployment location and network topology influence the effect 7. Difficult to detect spoofed packets when multiple paths exist 8. Delay in transmission	1. Can proactively filter out a large fraction of spoofed packets 2. Can reactively find the origins of the unfiltered spoofed packets within a small scope
Neighbor Stranger Discrimination	1. Need wide deployment 2. The number and location of deployment influence the effect 3. Learning phase affects the precision of detection 4. Normal packet could be dropped 5. Delay in transmission and storage overhead	Packet with spoofed source address cannot pass
Pushback	1. Need cooperation of upstream routers 2. The effectiveness lies on a precise congestion signature 3. Very possible to have false positive and false negative 4. Legitimate packets could be dropped 5. Pushback request could be spoofed 6. Storage and computation overhead, authentication cost	1. Can reduce the attack's threaten to victim 2. Can be complementary to other DoS defense schemes

Table 3.1 (continued): Summary of Defense Mechanisms

Schemes	Disadvantages	Advantages
Redirecting	1. Difficult for new users to setup connection 2. Firewall needs to do NAT 3. Need cooperation and communication between upstream routers and the firewall , and the communication need to be protected 4. Delay in transmission	Sustain web-service for legitimate clients even during a DDoS attack
Roaming	1. Management and investment cost of multiple servers 2. Server roaming and state recovery overhead 3. The attack could be aggregate enough to congest all servers	1. Ensure the service to legitimate users even under undetected attacks 2. Can help to find the source of attack 3. Can be complementary to other DoS defense systems

Chapter 4

Identifying the Attacker

In DoS attacks, attackers usually use spoofed packets to hide their actual locations, and the attack traffic is often indistinguishable from normal traffic. It is therefore important to identify the attack source and bring the culprit to justice. In this chapter, we focus on the solutions that deal with the identification of the attacker.

Traceback is a way to identify the actual source of attack traffic through the path the attack packets traverse. The existing solutions fall into one of the following four groups:

1. **Packet Marking:** The routers on the path encode their identities into packets. The victim can reconstruct the attack path after knowing the routers' addresses and their order in the path.

2. **Message Traceback:** The routers send traceback messages in separate packets to the destination. When enough messages from the routers along the path have been received by the victim, the attack path can be determined.

3. **Logging Packet Information:** Log packet information in routers and manage the information together to determine the path.

4. **Traffic Observation:** When the victim is under an attack, it observes the rate change of attack packets to determine the attack path.

4.1 Packet Marking

The packet marking can be either probabilistic or deterministic, and there are varieties of solutions for each category which will be discussed in this section.

4.1.1 Probabilistic Packet Marking (PPM)

Probabilistic Packet Marking means putting the path information into packets with certain probability, such that the victim can reconstruct the attack path using these markings. Usually a large number of packets are sent in a DDoS attack, so even markings are made with low probability, the victim can receive enough information about the attack path. To indicate the routers and their order in the path, two different methods have been proposed: *Edge Sampling* and an *Algebraic Approach.*

4.1.1.1 Edge Sampling

Edge Sampling [42] records two adjacent routers' addresses, which form the start and end point of an edge, into packets passing through and uses edges to express the path a packet takes. Each router writes its own address to the start field of a packet with some probability p and writes 0 to the distance field, which indicates the distance between the edge and the victim. When a router chooses to mark a packet and there is already a mark of another router in the start field, it overwrites the previous mark. If a router finds the distance field of a packet is 0, which means the packet is marked by its previous router,

it adds its address to the end field. This is how an edge between two adjacent routers is formed. For other packets, including the packets whose start and end fields have been marked, the distance field increases by 1 after each hop. Since an edge information is composed of two routers' addresses and the distance field, which requires more than 64 bits to store, efficient encoding schemes are needed to encode it into packets.

Segment Marking Scheme

Savage et al. [42] proposed to combine the following three techniques together to encode the edges: bit-interleaving IP address with its hash value, separating the interleaved value into segments, and exclusive-or (XOR) the segments of the start and end routers. The marking is stored in the 16-bit Identification field in IP header and composed of 3 parts: 3-bit offset to indicate which segment in this packet, 5-bit distance, and 8-bit edge fragment.

Each router calculates a hash value $h(a)$ of its IP address a and interleaves a with $h(a)$ bit by bit, in order to distinguish the edges at the same distance on multiple paths. Then the bit-interleaved address is divided into several segments to fit into each IP packet (as shown in Figure 4.1). When a router chooses to mark a packet with certain probability, it selects a segment a_k randomly to write and sets the distance field to 0. It also indicates which segment this packet records.

When the next router finds a packet with the distance field set to 0, it finds out which segment is written in the packet by the previous router and takes out the same segment b_k of its bit-interleaved address. To reduce the size of each marking further, it takes the marking from the packet, calculates the exclusive-or of the marking and its segment, then

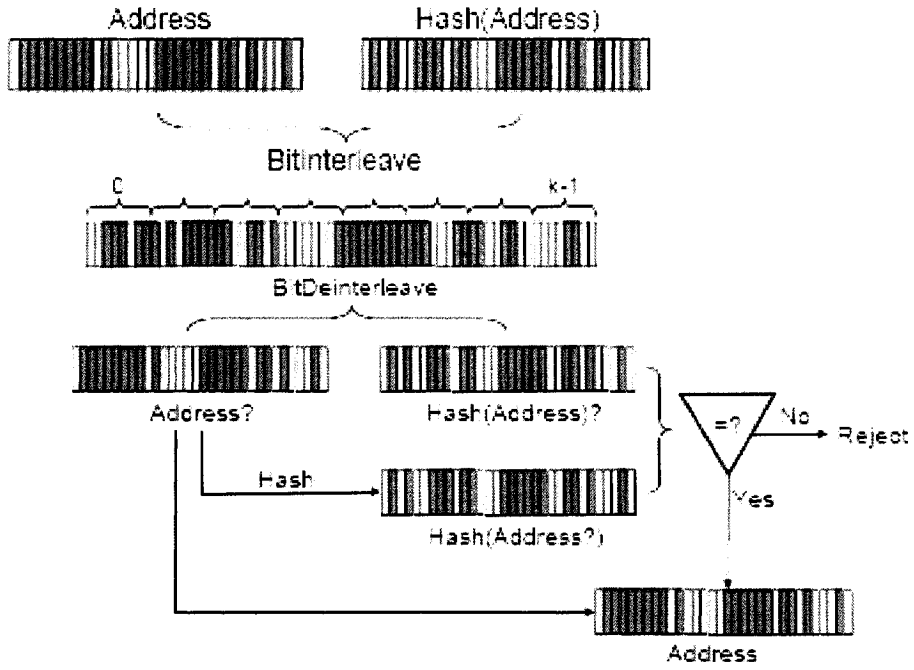

Figure 4.1: Bit-Interleaving IP Address with Hash Value [42]

puts the value $a_k \oplus b_k$ back. This process is performed by every router in the path, except for the routers only one hop away from the victim which have no following routers to XOR their markings n_k (as shown in the left part of Figure 4.2).

Since an edge between the routers at the last two hops to the victim is marked by $(n-1)_k \oplus n_k$, which is the XOR of the segment information of these two routers, XOR this value again with the segment information of the last router, the second last router's corresponding segment can be derived: $((n-1)_k \oplus n_k) \oplus n_k = (n-1)_k$. Doing this operation repeatedly, the segments of all routers can be calculated (as shown in the right part of Figure 4.2). After combining the segments and getting all of the possible bit-interleaved addresses, the original IP addresses and hash values are separated. By applying the hash function of each router again to its IP address and comparing with the hash values, the

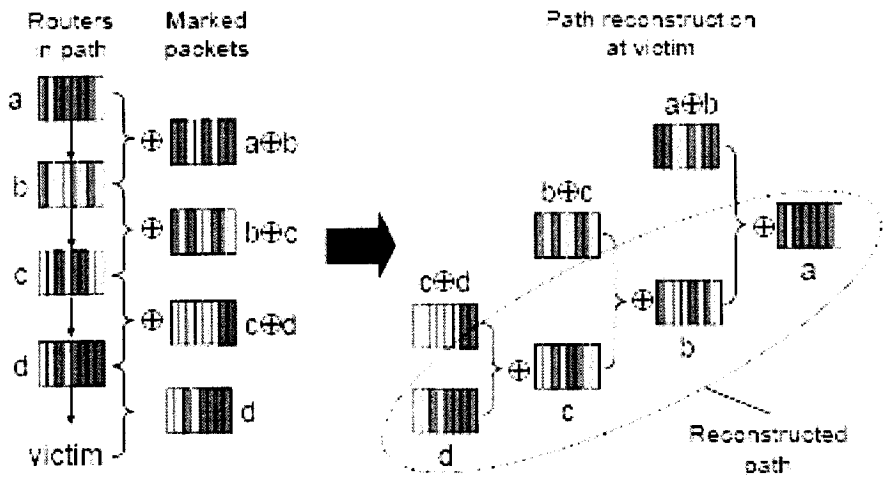

Figure 4.2: XOR Segments of Two Adjacent Addresses [42]

correct combinations are derived (as shown in the lower part of Figure 4.1). The IP addresses of the routers can construct the paths correctly according to their distances from the victim, no matter how many paths there are.

The attack paths can only be constructed after receiving all partial information of the routers. The routers which are far away from the victim can only mark relatively few packets, so a large number of packets are needed, but it is easily achieved in a DDoS attack. There are heavy overheads in XOR computation and the path reconstruction, especially when multiple paths exist. Though it can find out multiple attack paths, since the edges can be spoofed by attackers, the paths farther away than the nearest attacker are not believable. This traceback scheme can be performed after attack finishes.

Advanced Marking Scheme

Song and Perrig [45] proposed a scheme basing on a preestablished map of upstream routers, in which the address of each router is encoded by certain hash function to a

hash value. To differentiate the order of two routers on the edge, two sets of independent hash functions h and h' are used. The marking is stored in the ID field of a packet and composed of 5-bit distance field, w-bit edge information and (11-w)-bit flag field to indicate which hash function is used to mark this packet. When a router decides to mark a packet with certain probability, it uses a hash function in the set h to encode its IP address, puts it into the edge information field of the packet and indicates the number of the hash function it uses in the flag field. Its following router XORs its hash value, encoded by the same number of hash function in the set h' of itself, with the original value in the packet and puts the XOR result back.

Since a large number of packets are usually sent by attackers in a DDoS attack, it is quite possible for each router to use many of its hash functions to mark different packets. A router is considered to be on the attack path only when the number of its hash functions, whose values are matched to the markings in attack packets, exceeds a threshold. To the packet having 0 in the distance field, the edge information in its marking must be the hash value of one of the routers only one hop away from the victim. Therefore, comparing the markings with $h(x)$ (the hashed values of all hash functions in the set h) of all routers which are one hop away from the victim in the map of network, the routers which have matches exceeding the threshold are on the attack path and nearest to the victim. Then, the routers which are one hop away from the nearest routers, that are two hops away from the victim, can be found out in the map and the XOR values of their $h(x)$ with $h'(x)$ of the nearest routers can be computed. Comparing these values with the markings of the packets having 1 in their distance fields, the routers which have more same values than

the threshold are on the attack path and two hops away from the victim. Using the same method, all the routers in the attack paths can be deduced.

In the advanced marking scheme, a compromised router can forge the markings of upstream routers and is not easily detected. The map of upstream routers is necessary in path reconstruction.

Authenticated Marking Scheme

This Authenticated Marking Scheme [45] uses a similar marking method with the Advanced Marking Scheme, but the markings are encoded by each router with MAC (Message Authentication Code) functions which have secret keys, so that they can be verified.

Each router uses a one-way function to generate a series of keys, changes the current key periodically, and publishes it afterward. According to the property of the one-way function, the previous keys can be computed using a latter key. So the victim can compute the key used for a packet by knowing the latest key published by that router and the time the packet being marked. Since the secrete key changes in different time intervals, the flag field used in the Advance Marking Scheme is not necessary here. The reconstruction of attack paths is similar.

This scheme provides authentication to ensure the markings are not spoofed by attackers, but the authentication is complex to realize. It also relies on a preestablished map of upstream routers.

4.1.1.2 Algebraic Approach

Dean et al. [19] proposed an algebraic approach based on reconstructing a polynomial in a prime field to track packets. The principle is that, for any polynomial $f(x)$ with degree d in the prime field $GF(p)$, all the factors in $f(x)$ can be calculated if $f(x)$ is evaluated at $(d+1)$ different nodes. The marking of each packet is composed of an Accumulator and a Random Data. To simplify the path reconstruction process, it is assumed that there is only one path from a source to a destination.

In the scheme, a router considers itself as the first marking router of a packet with certain probability and puts the value of its IP address A_1 (32-bit number) mod p as the accumulator with a random number x_i in the packet, where p is the smallest prime larger than $2^{32}-1$ and x_i is different in each packet. If it is not the first router, the router multiplies the accumulator with the random number x_i and adds to its own IP address A_j, then calculates the sum *mod* p and puts the result into the accumulator. When the packet arrives at its destination, the accumulator will be $Acc_i = (A_1 x_i^{n-1} + A_2 x_i^{n-2} + \cdots + A_{n-1}x_i + A_n)$ *mod* p. After receiving enough packets, the victim can get the IP addresses of each router along the path by solving a matrix equation over $GF(p)$:

$$\begin{pmatrix} x_1^{n-1} & x_1^{n-2} & \cdots & x_1 & 1 \\ x_2^{n-1} & x_2^{n-2} & \cdots & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^{n-1} & x_n^{n-2} & \cdots & x_n & 1 \end{pmatrix} \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{pmatrix} = \begin{pmatrix} Acc_1 \\ Acc_2 \\ \vdots \\ Acc_n \end{pmatrix}$$

To make the marking small enough to fit in the ID field of IP header, the IP addresses can be separated into segments, then a router can mark a packet several times to put its whole address in before the packet leaves. However, more packets are needed to reconstruct the paths using this method. To reduce the required packets amount, the number of routers which will mark a packet can be restricted.

In this scheme, the router has a low overhead and the path reconstruction time is less than the Segment Marking Scheme. However, when there are multiple attackers, the packets needed for the paths reconstruction are quadratic to the number of attackers and, same as the Segment Marking Scheme, the markings made by routers further away than the nearest attacker are not trustable.

4.1.1.3 Adjusted PPM

Peng et al. [40] proposed to reduce the number of packets needed for the reconstruction of attack path in PPM by dynamically changing the marking probability of a router according to its location in the path. In forenamed PPM schemes, each router marks packets with a fixed probability, so that the victim needs to wait for the packets marked by the routers farther away from it, which are relatively fewer, to reconstruct the path. If the victim can receive packets marked by each router with equal probability, the reconstruction can be done with least number of packets. Therefore, the farther a router is to the victim, the higher the marking probability should be.

A router's location in the path can be known by recording the number of hops a packet has traversed from the its source or from the last router it is marked in the IP packet.

When a router receives a packet, it decides its marking probability using the location information and marks the packet with that probability. However, the attacker can spoof the hops account field to make a packet as coming from a farther source, then the following routers will mark this packet with lower probability and more packets are needed by the victim to reconstruct the path, which still includes a false segment introduced by the attacker intentionally.

Another method to indicate location can use the distance from the router to the destination of a packet. Current Internet protocol is destination-based, so it is easy to get the number of hops to the destination and the distance cannot be spoofed by attacker. Therefore, this is a safe scheme and can improve the efficiency of attack path reconstruction.

4.1.1.4 Analysis of PPM

The PPM schemes mark packets with low probabilities, which bring negligible overheads to routers. Due to the basic characteristic of DDoS attack, it is easy for the victim to receive enough path information even the mark probability is low. An improvement is also proposed to reduce the number of packets needed for the reconstruction of attack paths. However, complex computation is always needed in the reconstruction process.

Since the marking probability is low, it is possible for the attacker to insert packets with spoofed markings which can reach the victim without being overwritten. An authentication scheme has been brought forward to resolve this problem.

4.1.2 Deterministic Packet Marking (DPM)

Deterministic Packet Marking is to mark every packet.

4.1.2.1 Tracing by Interface

Belenky and Ansari [4][5] proposed a deterministic marking approach in which each interface of routers is considered as a unit of traceback. It uses the address of the first ingress interface a packet enters, instead of the full path the packet passes (as used in PPM), to recognize its source address and the mark remains unchanged until it arrives on the destination. Since packets passing through a same router from different directions enter different interfaces and each interface has a separate IP address, using the interface as the unit of traceback is able to identify the packets more clearly. DPM uses the 16-bit Identification field and 1-bit reserved Flag field in the IP header to mark packets.

In the marking scheme [5], an IP address is separated into k segments. Each segment is combined with the unique hash value of the whole IP address, which helps to reconstruct the segments, and segment identifier. The hash value is same for all packets marked by the same interface, but different for interfaces with different IP addresses. The length of segment identifier is $log_2(k)$, where k is the number of segments an address is separated into. Appropriate padding is added if the 32-bit address cannot be divided exactly by k. Since the mark is 17-bit, a 2^{17} bit reconstruction table is maintained by the victim to uniquely express every possible marking. After receiving a marking, the hash value, whose location in the marking is determined, is taken out and put into the table according to the value. The values of the address bits in the packets having the same hash value

are put into corresponding segments in that area. After getting the address bits of all segments in a hash value area, the original hash function is applied to the address to check whether the result is same with the hash value. If same, then the address is the right ingress interface address. When there are more than one value for an address segment, through the comparison with the hash value, the right address can be gotten. If the number of attackers is less than 2^d, where d is the bits of hash value, the right ingress interface address can always be reconstructed. If d is larger, the number of segments also increases because of the limited marking space, then more packets are needed to get the ingress interface address.

Because the route from the same source to a destination may change, it will be difficult for the victim to trace the attacker path using full-path traceback techniques. The nearest ingress interface of a router for a source must be same. Therefore, using the interface as the unit of traceback is a better solution. However, this scheme assumes that all the routers along the path belong to a single administrative organization. Otherwise, a packet will be marked multiple times when it enters different ISPs since there is an ingress interface in each of them, so that the packet cannot be traced to its origin. This scheme can trace multiple attackers, but the precision will be weakened if the number of attackers is too large, because the hash value will not be unique. This scheme still needs certain number of packets and has heavy storage and computation overhead to reconstruct the ingress address.

4.1.2.2 Path Identification

Yaar et al. [51] introduced a Path Identification mechanism which is deterministic and can identify attack packets coming from a same path after knowing the first one. The identification is basing on the marking, Path Identifier (Pi), of each packet, which is same to all packets passing through the same path. Though the marking does not include the information of entire path an attack packet has traversed, as other traceback mechanisms do, it can help the victim to distinguish attack packets basing on the markings they have.

In this scheme, the ID field of each packet is divided into $\lfloor 16/n \rfloor$ parts, each of which stores the marking of a router. To avoid the iteration of markings, a router computes the MD5 hash value of its IP address, which makes the value of its last n bits address distributed, and marks the last n bits hashed address into packets. Furthermore, to the packets coming from different upstream routers, a router uses different markings to distinguish them. Since the space is limited, it is not possible to put the markings of all routers along the path into it. A marking's location in the ID field is determined by the current TTL value of the packet, using TTL $mod \lfloor 16/n \rfloor$, and the new marking will replace the oldest one. To keep the markings of different routers in a packet as many as possible, n can be 1 or 2 and the internal nodes of autonomous systems can omit marking, because the internal routes can be known from the local administration department.

After the victim receives an attack packet, he can block all the following packets with the same marking to avoid the attack. However, attackers can easily escape by changing the initial TTL values of their packets, so that the markings of a same router will be in

different locations of the ID field. A method to deal with this problem is TTL unwrapping, in which the victim rotates the marking in each packet and puts the oldest one in a fixed location, then the ID field will be same to the packets passing the same path. When the number of attackers is large, lots of markings will be blocked, then it is quite possible for legitimate packets to have the same markings. Therefore, to decrease the false positive, a threshold can be applied so that a packet will be dropped only when the percentage of attack packets having the same marking exceeds the threshold.

This path identification mechanism is different with other traceback methods. The victim cannot use the marking to find the source of attack, but can use it to separate attack packets. Moreover, the victim do not need to wait until enough packets have been received to reconstruct the path, he can use only one packet to identify the attacker. Therefore, this mechanism has low overhead to both the routers and the victim. This mechanism can also be combined with pushback to get the attack signature, or used to detect spoofed packets, by storing the IP addresses and their corresponding markings in peace time. However, it needs an external source to know the markings of attack packets before filtering the following ones. Though a threshold is used in packet filtering, the false positive and false negative are inevitable and determined by the threshold value. If the value is low, many legitimate packets will be dropped; otherwise, lots of attack packets will come through, yet higher threshold is more suitable when the number of attackers is large. Also, if the attacker inserts random information into the marking field of packets and the markings are not overwritten by other routers, the packets will have different markings and cannot be identified.

4.2 Message Traceback

Bellovin [6] proposed to send separate ICMP traceback messages by routers with a low probability to identify the source of attack. The victim can reconstruct the entire path after receiving enough packets.

In this scheme, each router generates special ICMP packets for some of its received packets, which include the information about their links with previous and next routers and certain authentication, and sends them to the destinations of these received packets. After receiving enough ICMP packets, the victim combines the links together and uses the TTL differences of these ICMP packets to construct the attack path. To ensure each ICMP packet is not spoofed by attackers, an authentication and a validation field are included in the ICMP packet. The authentication field is a key or signature of the router; the validation field has recent, but not current, authentication keys, which can be used to validate a previous ICMP packet sent by the router. Therefore, the victim needs to receive at least two ICMP packets from each router to validate the former packet.

Since the router sends ICMP messages with low probability, the farther the router is from the victim, less ICMP messages can be received. Markin et al. [35] brought forward the "Intention-Driven" ICMP traceback, in which the value of an ICMP message is considered. The packets from the farther routers are more useful for the victim. Also, if the router can create the ICMP message soon after the attack begins and the victim has not received many of the same messages, this message is valuable. Routers may create ICMP messages for the non-attack packets or for the destinations which do not wish to

receive them, so these messages have no value. By considering these factors, the routers will create more valuable ICMP messages to help the victim performing traceback.

The special ICMP message can be filtered because of its difference with other traffic, especially during an attack, and be spoofed by attackers. The attack path is difficult to reconstruct if there are multiple sources or this scheme is not deployed widely.

4.3 Logging

4.3.1 CenterTrack

Stone [48] proposed an overlay network, CenterTrack, to reroute suspicious flow for the tracking purpose. The overlay network connects all edge routers of an ISP network with special tracking routers using IP tunnels. A copy of each packet is logged at the tracking router for detailed analysis.

When an attack is detected, all the packets destined to the victim are dynamically rerouted through the overlay network. Input debugging is performed at the tracking router closest to the victim hop-by-hop to find out the router sending the packet into the ISP. In the network, all the edge routers are connected to the tracking routers using IP tunnels, so the repetition of tracking finishes within 2 or 3 hops. The tracking routers are also connected with each other by IP tunnels, but physically connected to backbone routers. All the packets rerouted by the overlay network are required to be encapsulated.

Though the full packet recording provides enough information and evidence to investigate the attack, the tracking routers need to have the ability to store these infomation. Also, when there are multiple attacking paths in DDoS attacks, the tracking process

is much more difficult. So this logging technique is not scalable with DDoS attacks. Moreover, the overlay network increases the complexity of the network and brings extra administration overhead. The encapsulation of data increases the size of each packet, so the congestion of the network could be more serious when attack happens.

4.3.2 Single Packet Traceback

Snoeren et al. [43, 44] described a Source Path Isolation Engine (SPIE) to traceback relying on a single packet delivered recently.

The SPIE system consists of three components: the Data Generation Agent (DGA) located at each router, the SPIE Collection and Reduction (SCAR) Agent responsible for a region of the network, and the SPIE Traceback Manager (STM) which controls the whole system. In the DGA, a Bloom filter computes several distinct digests for the content of each packet using independent hash functions. A 24-byte invariant part of a packet, including the 16-byte IP header, except for Type of Service, TTL, and Checksum, and the first 8 bytes of the payload, are used in the digest computation. The values of a packet's digests are set in the corresponding bits of a digest array. The digest arrays formed in a time interval constitute a digest table. A DGA stores a number of digest tables, each of which is marked with the time interval and hash functions used during this time period. Due to the limited resources at a router, the digest tables kept by a DGA are reset after a period of time and the old tables are transferred to the SCAR.

When the STM receives a traceback request, it transfers the request to the SCARs, with the packet to traceback and its receipt time, and the SCARs queries their DGAs.

The DGA computes the digests for the packet according to the time interval and compares them with the records in the digest tables. Due to the possible delay in transmission, the packet may be transferred in an earlier time than indicated, so the DGA may need to check multiple digest tables. If corresponding records is found, it is considered that the packet passed this router in that time interval. The SCARs collect the results from DGAs in their regions to form their partial graphs and submit them to the STM to assemble the complete attack graph. Therefore, a trackback must be performed before the useful digest tables are removed from the DGAs. However, once each DGA's digest tables have been looked up for the suspicious packet, they can be stored in the SCARs for further analysis without time restriction.

In the SPIE system, traceback must be initialized in time and the system must have enough ability to perform traceback when the network is congested during an attack. The accuracy of trackback is influenced by the deployment scope of SPIE and the storage capability of each DGA. Since the digests of different packets may happen to be same, some routers in the attack graph are not actually passed by the packet, which creates the false positives. The false positives can also be caused by the poor time synchronization which makes the DGA cannot find the suitable digest tables to check.

4.4 Traffic Observation

Burch and Cheswick [7] developed a link test technique to traceback, basing on the knowledge of the Internet topology, to determine which link the attack traffic comes from by observing the change of attack traffic.

During an attack, the victim floods each incoming link with excessively large numbers of packets. Since the bandwidth is limited, if the attack traffic is coming from the link being tested, some of the attack packets will be dropped because of the excessive packets. If the rate of attack traffic is reduced, the victim can know part of the path that the attack traffic traverses. By performing the link test recursively, the victim can finally find the source of attack.

The scheme does not need the cooperation of intermediate networks except for the time when the link test is done. The ability to generate large numbers of packets may be not available to all networks. This scheme must be performed when an attack is occurring and the topology of Internet must be known in advance. It is not suitable for attacks using varying attack rates. It is a slow process because it needs to test upstream links recursively until the attacker is reached and is not appropriate for DDoS attacks which have multiple attackers. Also, the attacker who is aware of the test being performed can escape detection. Furthermore, the large numbers of packets sent in the test can become a DoS attack by themselves.

4.5 Summary of Attack Identification Schemes

Four different solutions have been proposed to trace the source of attack with various implementation methods.

- Packet marking encodes the path information into packets and reconstructs the attack paths using these markings. This kind of schemes use the available spaces in IP headers, often the ID field, to avoid increasing the size of packets. They

bring routers low overhead, but the reconstruction of paths always requires complex computation.

- Message traceback copies some of the packets into another special packets and sends them together with the original packets to the victim. The victim uses these messages to organize the path. Due to the special type of these messages and the congestion of victim under attack, they could be dropped before reaching the destinations. Therefore, their usefulness is limited.

- Packet logging records packets in the routers along the path when they pass through so that the victim can inquire these routers to see whether the attack packet was coming from there. However, the storage space and keeping time are two main problems.

- Traffic observation flushes links with large numbers of packets and observes the change of attack traffic. This method is only feasible during the constant rate attack, and the packets sent by the tester may cause damage by themselves.

Most of these solutions require the cooperation of intermediate networks, which need to mark packet, send special message, or log packet to help the victim know the path the attack packet has traversed. A summary of the attack identification schemes is given in Table 4.1 and their classification is shown in Figure 3.1.

Table 4.1: Summary of Attack Identification Schemes

Solution	Disadvantages	Advantages
Packet Marking	1. Need large numbers of packets and complex computation to reconstruct the attack path 2. Some schemes need to know network topology 3. Need the participation of upstream routers	1. Can trace multiple attackers 2. Can be performed after attack
Message Traceback	1. ICMP messages are different with other packets, so that could be filtered 2. Need large numbers of packets to reconstruct the attack path 3. Need the participation of upstream routers	1. Do not need the knowledge of network topology 2. Can be performed after attack
Packet Logging	1. Storage and processing overhead for routers 2. Not scalable with the DDoS attack 3. Need the participation of upstream routers	1. Can use single attack packet to traceback 2. Can be performed after attack 3. Do not need the knowledge of network topology
Traffic Observation	1. Only useful when an attack is occurring 2. Not suitable for attacks using varying rates 3. Cannot deal with the DDoS attack 4. Need to know network topology 5. Slow 6. The test process could become an attack by itself 7. Attacker could escape detection 8. Need the ability to generate enough packets	Do not need the cooperation of intermediate networks except for the link test time

Chapter 5

A Novel Scheme for Preventing Denial of Service Attacks

As discussed in Chapter 2, the success of most Denial of Service attacks depends on the use of spoofed IP addresses in the attack packets. If we can distinguish the packets which have spoofed IP addresses, we can selectively filter out them. In this chapter, we present a new Marking-based DDoS Attack Detection and Filtering (MDADF) scheme to identify spoofed IP packets. We use the packet marking method, which was first proposed by Savage et al. in [42], to determine the routes that the packets have traversed and use these markings to recognize attack packets. Savage et al. used packet marking to record path information by encoding the IP addresses of routers that a packet passes through, and this information was used to find out the source of the attack.

In today's DDoS attacks, attackers always exploit thousands of compromised hosts to launch attacks, so even if we can trace some of the attack sources, we can only stop these particular machines from sending more attack packets. Usually we cannot discover the genuine attacker hiding behind those compromised machines. In this case, finding out the sources of attack packets does not help us too much. A better way of defending the

attack is just stopping these attack packets from reaching the victim. The main objective of a DDoS defence scheme should be to identify the attack flows, stop them, or provide information to help stopping the attack.

To achieve this goal, we propose the MDADF scheme to differentiate between packets by marking them, so that a firewall can check these markings to identify spoofed packets and consequently drop them. We cannot expect every router in the Internet to participate in the marking scheme and mark all packets passing through it. So, our scheme is designed to work well even if a small percentage of the routers cooperate with this scheme.

5.1 Marking Scheme

To indicate the path a packet traverses, the simplest way is to add the IP addresses of all the routers into the packet. The number of hops that a packet passes through in the Internet is about 15 on average and mostly less than 31 [16][17] (as shown in Figure 5.1). Since the length of a path is uncertain, it is difficult to reserve enough spaces in a packet to put all the addresses in, and the packet size expands as the length of the path increases.

In order to avoid the increase in packet size, a possible method is to put all information into a fixed space. A router puts its IP address into the marking space of each packet it receives; if there is already a number in that space, it calculates the exclusive-or (XOR) of its address with the previous value in the marking space and puts the new value back. This method ensures that the marking does not change its length when a packet travels over the Internet, so the packet size remains constant.

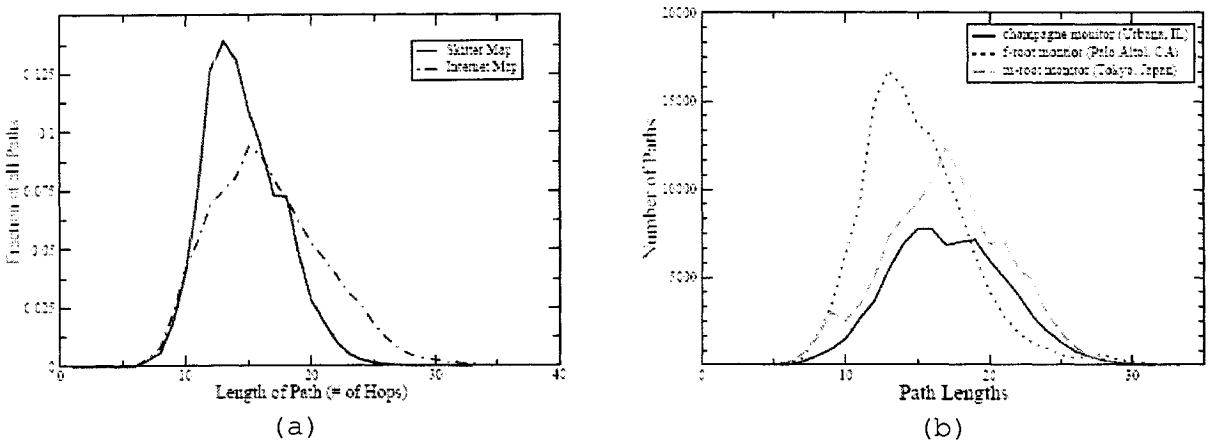

Figure 5.1: Internet Path Lengths Distribution (a) Skitter Map and Internet Map [51], (b) Three Skitter Monitors [52]

However, to make the marking scheme faster and easier to implement, we had better use the space available in the IP packet. The 16-bit Identification field in the IP header, currently used to identify the fragments, is suitable to use as the marking space, since only less than 0.25% of the packets in the Internet are fragmented [47]. Moreover, the vulnerability of fragmented packets has been exploited by attackers in DoS attacks, such as the Teardrop and other fragmentation attacks, so that some networks reject fragmented packets. Therefore, employment of identification field as the marking space will not affect normal packet transmission much. In fact, many other schemes ([4] [5] [19] [40] [42] [45] [51]) have already proposed to employ the identification field as the marking space.

5.1.1 Computing the Packet Marking

To fit the 32-bit IP address A into the identification field, we adopt a hash function h, which converts the IP address into a 16-bit value. We use the CRC-16 hash function which is easy to compute and has low collision rate.

It is easy for an attacker to know the IP addresses of the routers along any route on the Internet. So, the attacker may be able to compute the final marking for a packet that traverses a particular route. This means that the attacker can spoof the marking on a packet. If a packet with such a spoofed marking arrives at the firewall without being marked by other routers, the firewall would be unable to identify this packet as an attack packet. To avoid such spoofing of the marking, each router R selects a random 16-bit number as the key K_R. Its marking is calculated as M_R = h(A) XOR K_R, where A is the IP address of the router. After receiving a packet, the router computes the marking $M = M_R \oplus M_{old}$, if an old marking M_{old} exists in that packet, and replaces M_{old} with M.

5.1.2 Inserting Order information

One possible drawback with the scheme mentioned above is that the marking on a packet depends only on the routers through which it passes but not on the order in which it passes these routers.

To further reduce the collision of markings, each router performs a Cyclic Shift Left(CSL) operation on the old marking M_{old} and computes the new marking as $M = CSL(M_{old}) \oplus M_R$. In this way, the order of routers influences the final marking on a packet received by the firewall.

The marking scheme is shown in Figure 5.2, and the pseudo code is described below:

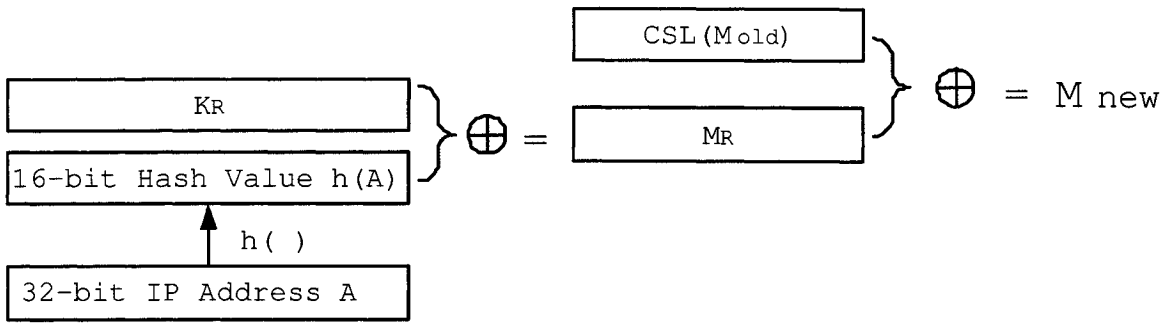

Figure 5.2: The Marking Scheme

```
Marking procedure at router R (having IP address A):

    let k be a random 16-bit number
    M(R) <- k XOR h(A)
    For each packet w
    {
        If W.ID = 0 Then
            w.ID <- M(R)
        Else
        {
            M_old <- w.ID
            M_new <- M(R) XOR CSL(M_old)
            w.ID <- M_new
        }
    }
```

5.2 Filtering Scheme

Using the marking scheme we provide in the above section, the path information is encoded into each packet. Instead of using the markings to trace to the sources of packets (as in [4][5][19][40][42][43][45]), we intend to use them to differentiate packets with spoofed IP addresses. A filter deployed at the firewall will filter the spoofed packets out. The general process is shown in Figure 5.3.

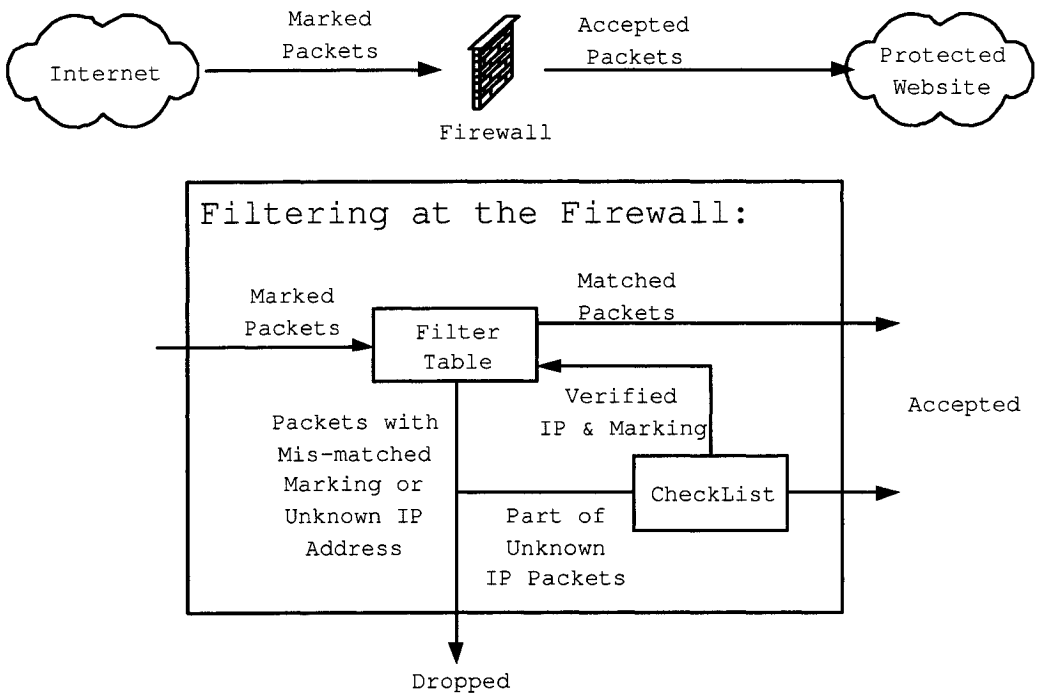

Figure 5.3: The System Structure

5.2.1 Basic Filtering Scheme

In most DDoS attacks, attackers use spoofed IP addresses to hide their identities as well as the identities of the network of compromised machines. By employing our proposed marking scheme, the routers on the route of a packet would mark the packet according to the prescribed method. So, when the packet arrives at its destination, its marking will depend only on the path it traversed. If the source IP address of a packet is spoofed, this packet must have a marking that is different from that of a genuine packet. The spoofed packets can be easily identified and dropped by filter, while the legitimate packets containing the correct markings are accepted.

To implement this scheme, the firewall needs to have a record of the genuine marking for an IP address, so that it can differentiate the spoofed ones. During normal time, when

no attacks are happening, the firewall can learn about the right markings of packets. The (IP-address, Marking) pairs are stored in a Filter Table, which are later used to verify each incoming packet and distinguish spoofed packets. The learning phase continues for a sufficient time to allow most of the filter table to be filled up. After the learning phase ends, the firewall begins to perform its normal filtering operations.

In the normal filtering procedure, to the packet from an IP address recorded in the Filter Table, it is accepted if it has a consistent marking; otherwise, it is dropped. For the packet from a new IP address, we accept it with probability ρ and put the (IP-address, Marking) pair to a Check List, so that the marking can be verified. The value of ρ is set to high (close to 1) initially. When an attack is detected, most of the victim's resources will be occupied in handling the attack flow, so the value of ρ is decreased according to the packet arrival rate and the victim's capability for handling the incoming traffic.

To verify the markings in the Check List, a random echo message is sent periodically to the source of every each (IP-address, Marking) pair, and a counter is used to record the number of echo messages have been sent for it. To avoid the reply being imitated by the attacker, the content of the echo message is recorded in the Check List and is compared with the content of reply message. On receiving an echo reply, if it proves that the marking is correct, the (IP-address, Marking) pair is moved to the Filter Table; otherwise, it indicates the previously received packet is spoofed, then this pair is deleted from the Check List. If the counter in the Check List shows the echo message has been sent more than 10 times for an IP address A, then the entry for this IP address is removed from the Check List and the pair (A,$\emptyset$) is added to the filter table, where $\emptyset$ is a special

symbol, which denotes that all packets having source IP address A should be discarded. Since in this situation, this source IP address must be non-existing, inactive, or all the genuine packets from this source cannot arrive at the victim due to network congestion, the packets received are all from the attacker and need to be rejected.

The size of the Filter Table is limited; therefore, we can keep a record of the markings only for those IP addresses from which a packet has been received recently. The time when the last packet was received from an IP address is logged. When the Filter Table is full, any new entry to be added to the Filter Table replaces the oldest entry.

To detect the start of a DDoS attack, we use a counter called Total-Mismatches-Counter (TMC), which counts the number of packets whose marking cannot be matched at the firewall. It counts packets with incorrect markings as well as packets from unknown source addresses that are not recorded in the Filter Table. When the TMC value becomes greater than a threshold θ, it is considered as a signal of DoS/DDoS attack and TMC is reset to zero. The value of TMC is also reset after fixed intervals to ensure that the cumulative result over a long duration is not considered as the indication of attack by mistake.

5.2.2 Route Change Consideration

Though the routes on the Internet are relatively stable, they are not invariable. Once the route between two hosts has changed, the packet received by the destination will have a different marking with the one stored in the Filter Table, so that it will be dropped according to our basic filtering scheme.

Taking route changes into consideration, we introduce another counter called Specific-IP Mismatches Counter (SMC). When the firewall receives a packet with mismatching marking from an IP address, the SMC value of that address is incremented. When the value of SMC reaches a threshold δ, the (IP-address, Marking) pair is copied to the Check List to test whether the route from this source has changed and the SMC is reset to 0. If the new marking is verified by the Check List verification process, the marking for this IP address is updated in the Filter Table. Otherwise, the original marking is preserved for packet verification. Unless the route change has been verified, the original marking is still used to filter packets.

The structures of the Filter Table and Check List are shown in Figure 5.4.

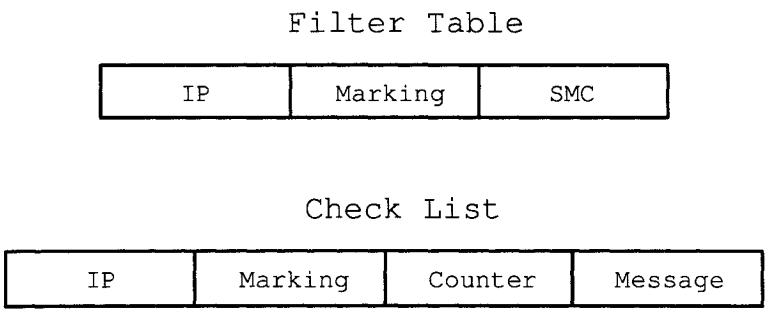

Figure 5.4: The structure of the Filter Table and Check List

5.2.3 Complete Filtering Scheme

Using the techniques and criteria introduced above, a complete filtering procedure is described below. Any packet received by the firewall is checked by the filter according to the following rules:

1. If the (IP-address, Marking) pair is same with one of the records in the Filter Table, the packet is received.

2. If the source IP address of the packet exists in the Filter Table, but the marking does not match, this packet is considered to be a spoofed packet and is dropped. TMC is incremented in this case.

3. If the source IP address does not appear in the Filter Table, then this packet is accepted with a probability ρ. TMC is incremented also.

4. If the TMC value exceeds a threshold, an attack is signaled.

5. All echo reply messages that are received as responses to the firewall's requests are handled by the Check List verification process and they are not passed through the filter.

In general, the filtering scheme has the following functions:

- Distinguish and filter out spoofed packets by checking the marking of each packet using the Filter Table.

- Detect the occurrence of DDoS attacks, so that appropriate defensive measures can be taken before serious damage is caused.

- Ensure that not many legitimate packets are dropped mistakenly, due to route changes on the Internet.

Pseudo code for the filtering scheme is shown below:

```
Parameters:
 T_L: Duration of Learning Phase
 CL: the Check List
 FT: the Filter Table
 \theta: the number of total mis-matching packets which indicates the
        beginning of an attack
 \delta: the number of mis-matching packets for a specific IP address
        which indicates the marking needs to be verified due to
        path change
 T: the time interval to reset TMC
 S: the time interval to send echo messages
 p: the probability to accept the packet from an unknown IP address


Learning Phase at the firewall:

    While ((current-time - start_time) < T_L )
    {
        For each packet w
        {
            If w.sourceIP is not in Filter Table Then
                add (w.sourceIP, w.ID) to FT
            Accept w
        }
    }



Filtering procedure at the firewall:

    If (current_time - start_time) mod T = 0 Then
       TMC <- 0

    For each packet w
    {
        If w is a valid echo reply Then
            send w to Check List Verification Process

        Else
        {
```

```
            If (w.sourceIP, w.ID) exists in Filter Table Then
                accept w
            Else
            {
                TMC <- TMC + 1
                If TMC > \theta Then
                    attack warning

                If there exist entry f in Filter Table, f[IP] = w.sourceIP Then
                {
                    drop w
                    f[SMC] <- f[SMC] + 1
                    If f[SMC] > \delta Then
                    {
                        add (f[IP], f[Marking]) to CL
                        f[SMC] <- 0
                    }
                }
                Else
                {
                    let x be a random number from [0..1)
                    If x < p Then
                    {
                        If w.sourceIP is not in Check List Then
                            add (w.sourceIP, w.ID) to CL
                        accept w
                    }
                    Else
                        drop w
                }
            }
        }
    }
```

Verification of the Check List at the firewall: (every S time)

```
    For each entry c in Check List
    {
        If c[Counter] < 10 Then
        {
            If c[Counter] = 0 Then
```

```
        {
            let y be a random number
            c[Message] <- y
        }
        send echo(c[Message]) to c[IP]
        c[Counter] <- c[Counter] + 1
    }
    Else
    {
        If c[IP] is not present in Filter Table Then
            add (c[IP], @) to FT
        remove c
    }
}


For each echo reply w received
{
    If (w.sourceIP,*,*, w.Content) exists in Check List Then
    {
        If there exists entry f in Filter Table, f[IP] = w.sourceIP Then
            replace f with (w.sourceIP, w.ID, 0)
        Else
            add (w.sourceIP, w.ID, 0) to FT
        drop w
        remove c from Check List where c[IP]=w.sourceIP
    }
}
```

5.3 Pushback Implementation

By employing our filtering scheme, the firewall can protect the victim website by filtering out attack packets. However, sometimes the attack flow may be too large and the firewall may not have enough resources to handle it. In that case, we may employ the method of pushback [31] (see Section 3.3.8).

In the Pushback method, the victim of a DDoS attack sends the signatures of attack to upstream routers and ask them to help filtering out these packets. Since one IP address can be used in the attack packets from many different sources, if we use the markings of spoofed packets as the attack signatures, large numbers of comparison need be done by the upstream routers. Instead, we create a list of IP addresses with their corresponding markings from the Filter Table and send this list (called the Pushback List) to the upstream routers.

Whenever the firewall adds new entries or updates old entries in the Filter Table, these entries are sent as updates to the upstream routers, so that the Pushback List can be updated. The upstream routers compare each packet with the Pushback List after marking it and discard spoofed packets. Most of the attack packets are filtered before arriving at the victim, so that the victim website can continue with its normal operations.

In some instances, the upstream routers of the victim still cannot deal with the attack flow, then they need to pushback further. To perform this function, each router R transforms all original markings $M_i(i = 0...n)$ in the Pushback List by computing $M_i' = CSR(M_i \oplus M_R)$, where CSR (Cyclic Shift Right) is the inverse of the CSL operation. The router then sends the new generated markings $M_i'(i = 0...n)$ to its upstream routers. This process can be performed recursively until the attack flow is controlled.

The structure of the Pushback List is shown in Figure 5.5.

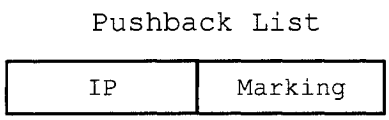

Figure 5.5: The structure of the Pushback List

Chapter 6

Analysis and Evaluation of the Proposed Scheme

In this chapter, we analyze the performance of our MDADF scheme, based on results obtained by implementing and testing the scheme under simulated DDoS attacks.

6.1 Practical Implementation

In this section, we discuss the implementation details and the selection of parameters for the simulation.

6.1.1 Internet Dataset

In order to make our simulation realistic, we use the actual topological data obtained from the Internet Mapping Project [17] of Lumeta Corporation in the experiments.

The data was generated by using *traceroute* to probe the paths in Internet from a single host (netmapper.research.lumeta.com, 65.198.68.56). The database records IP addresses of destinations and the routers in each path, and other relevant information. If a destination is targeted by the probe, the full path is shown as the IP addresses of the routers from the source to it. Otherwise, only the reached routers in the path are shown.

Due to the growth and change of Internet, these routes may change over time and the topology can also vary. The database we use in our experiment was generated on July 4, 2005, which contains 113,827 traceroute records.

In our simulation, we use the single source in the dataset as the victim of the DDoS attacks, the reached hosts as the legitimate users, and the unreachable ones as the attackers. Since all the paths in the database congregate at the single node, this node is quite suitable to act as the victim in the simulations of DDoS attacks. The reached hosts are known to be connected with the source node and can actually send packets to it, so these hosts play the role of legitimate users in our simulation. The unreachable IP addresses, which possibly do not exist at all or are temporarily not in use, are used to simulate the attackers.

6.1.2 Simulation of Normal Internet traffic

We use a packet generator process, which periodically sends a packet from a randomly selected legitimate user, to simulate the normal Internet traffic. The route that this packet takes to the destination is looked up, then a packet marking process is simulated, in which this packet is marked by each router on the route. Finally, the marked packet is inserted into a packet-queue of the firewall at the victim. The rate, at which packets are added to the packet-queue, mimics the normal traffic flow for a typical web-server on the Internet.

6.1.3 Simulation of DDoS Attack traffic

Attackers usually use forged source IP addresses in their packets to disguise their actual locations. They have two methods: spoofing another's IP address, which kind of

attack is called spoofed attack, and putting a random number into the source address field, called randomized attack. We simulate these two kinds of attack separately. In both of them, an attack packet is generated from each of the attackers periodically to form the high volume of attack traffic, but at a various rate influenced by the attackers amount.

To simulate the spoofed attack, an IP address is randomly selected from the legitimate users and put into the source IP address field of the attack packet. To conceal the attacker further, random value is put into the marking field, so that even the first router on the path does not know the attacker is only one hop away from it. A same marking process with normal traffic is then simulated, in which the routers on the path from the attacker to the destination mark this packet. The marked attack packets are also added to the packet-queue.

Similar process is performed in the random attack simulation, except that the attackers use random values as their source IP addresses.

6.1.4 Packet Filtering

The packet filtering is implemented at the firewall which reads each packet from the packet-queue and decides whether to accept or discard it. The firewall process has three phases: the learning phase, the normal filtering phase, and the attack-time filtering phase. We describe these phases below.

Learning Phase

The learning phase is used to populate the Filter Table. During this learning period, every packet is accepted, and the new (IP-address, Marking) pair is put into the Filter Table. We use a fixed-time learning phase, after which the normal filtering phase begins.

Normal Filtering Phase

After the learning phase, markings for most of the legitimate users should have been recorded in the Filter Table. During the filtering phase, for every incoming packet, the marking is checked and compared with the records stored in the Filter Table.

If the IP address of a packet exists in the Filter Table, it should have a same marking with the one in record; otherwise, it is quite likely from an attacker. Therefore, packets with same markings are accepted and those with mis-matching markings are dropped.

If a packet comes from a new IP address that is not present in the Filter Table, it will be either accepted or rejected with certain probability p. The value of p should be set according to the capability of the firewall and how fast it can handle the incoming packets. The (IP-address, Marking) pairs of accepted packets are added into the Check-List to verify the markings.

When a packet with incorrect marking arrives, it may be due to two reasons: either the IP address on this packet was spoofed, or the route from this IP address to the victim has changed. In any case, the SMC counter for this IP address is incremented. When the SMC of an IP address becomes larger than a threshold δ, this IP address is added into the Check-List so that its marking can be verified again.

The system is designed to signal the occurrence of DDoS attack when the value of TMC becomes larger than a threshold θ. On detecting an attack, the normal filtering phase switches to the *attack-time filtering phase.*

Attack-time Filtering Phase

During the attack time, lots of system resources are consumed by attack traffic. In order to sustain the service to existing users, the probability p to accept packets from new IP addresses is kept much lower than before. The same filtering steps are performed as in the normal filtering phase, except that the firewall no longer checks the TMC value for attack detection.

6.1.5 Check List Verification Process

For each IP address in the Check List, an echo message is send at regular intervals. If the IP address belongs to a real host on the Internet, it will receive this echo message and reply. The reply message will be marked, then the firewall will receive the genuine marking. If the source IP address is already in the Filter Table, the recorded marking is compared with the new one and is replaced if they are different; otherwise, the IP address and the received marking are added to the Filter Table.

If the attackers know about our filtering scheme, they can spoof the echo reply messages to make the firewall validate an incorrect marking. To solve this problem, a simple method is to put a special content to each echo request and record it before sending the message out. Since the echo reply would contain the same message as the echo request, the firewall can verify whether the echo reply is genuine or fake.

If echo messages have been sent out several times for an IP address without receiving a reply, this record is removed from the Check List. This can happen in two cases – either this IP address does not actually exist but only appears in spoofed attack packets, or this IP address belongs to a legitimate user, but the path between the user and the victim is jammed due to the denial of service attack, thus no packets can be received from this host. In any case, it is unnecessary to send any more echo messages specially in the second case, because sending more packets can only exacerbate the congestion.

6.1.6 Parameter Selection

In this section, we explain the choice of parameters for our simulation.

6.1.6.1 Filter Table Size

In MDADF system, the IP addresses of legitimate users and their markings are stored in the Filter Table. Those not recorded in the Filter Table are considered new to the system and their packets are only accepted with certain probability. Therefore, the size of Filter Table should be big enough, so that the markings for most of the legitimate hosts can be stored. In our simulation dataset, there are 10,000 legitimate hosts in total. We have tested our scheme using different Filter Table sizes, ranging from 5,000 to 10,000 (which are capable of storing the records of 50% to 100% of the hosts.)

6.1.6.2 Length of Learning Phase

In the learning phase, the system learns the IP addresses and corresponding markings for the legitimate users. As the system learns more such (IP-address, Marking) pairs, more easily packets from the legitimate users be recognized in the future communication.

Thus, the learning phase needs to be long enough. We have used a learning phase of 10 minutes, which in most cases allows the firewall to learn the markings of nearly 80% of legitimate users.

6.1.6.3 Participation Rate of Routers

To make our filtering scheme work, we need the cooperation of routers in the Internet. The routers mark each packet, then the filter can differentiate good and spoofed packets. The number of participating routers affect the performance of MDADF scheme. We vary the percentage of the participation from 100% to 0 to test the effect on packet filtering.

6.1.6.4 Magnitude of Attack

During a DDoS attack, all the attackers send packets to the victim simultaneously, creating a huge attack traffic and causing congestion. We test the performance of MDADF system under the attacks of different magnitudes by varying the number of attackers in the simulation. We have tested the system under smaller attacks with 100 attackers to massive attacks by 5000 attackers. For most of the other tests, we have used a default value of 500 attackers.

6.1.6.5 Probability p for Accepting Packets from Unknown IP Addresses

During the normal filtering phase, the value of p is kept high (0.75). After the attack is detected, in order to deal with large traffic flow of attack packets, the value of p is decreased to very low (0.1). We have also tested the scheme with other values of p (0.9, 0.75, 0.5 for before attack detection and 0.5, 0.25, 0.1 for after attack detection) to assess the effects of varying the value of the probability.

6.1.6.6 Threshold for Signaling Attack

One of the important functions of the MDADF system is attack detection, which is realized by comparing TMC with a threshold θ. Every received packet with mismatched marking or unknown IP address makes TMC increase. It is reset to zero after fixed intervals to ensure that the cumulative result over a long duration is not considered as the indication of attack by mistake. Therefore, the threshold θ needs to be big enough so that no attack is signaled in normal time, but the attacks can still be detected as soon as possible. In our simulation, TMC is reset every two minutes and θ is set to 2700.

6.2 Results

The results of the simulation are presented in this section.

6.2.1 Performance under Spoofed Attack

The number of attackers in an attack affects the level of congestion at the victim and disruption of normal service to legitimate users. We have tested the performance of our scheme under spoofed attacks, with the number of attackers ranging from 100 to 5000. Figure 6.1 shows the ratio of packets that are accepted at the firewall under different magnitudes of attacks.

Our MDADF scheme needs the participation of routers on the Internet to help marking the packets. We cannot expect all the routers to be willing to cooperate; therefore, we have tested the effect of different participation rates with our scheme. The simulation results for various participation rates ranging from zero to 100% are shown in Figure 6.2.

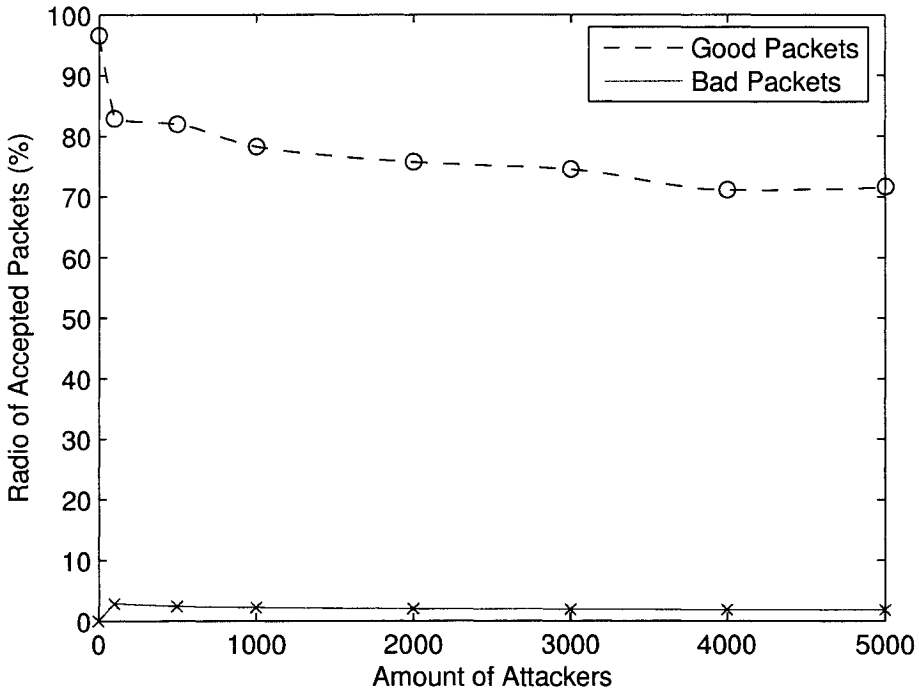

Figure 6.1: Ratio of Accepted Packets vs. Different Amount of Attackers under Spoofed Attack

We show the difference between acceptance ratio of good and bad packets, which is called "Acceptance Ratio Gap".

The Filter Table is a critical part of MDADF system, which stores the (IP-address, Marking) pairs and provides information for packets filtering. Due to the limited resources, the firewall employing our scheme may have no enough space to store the information for every legitimate user. Thus, we have tested the performance of the scheme with different sizes of Filter Tables and the results are shown in Figure 6.3.

The packets coming from unknown IP addresses are accepted with certain probability p after the learning phase. Table 6.1 and Table 6.2 show how changing the value of probability influences the acceptance ratio of packets in normal and attack time, respectively.

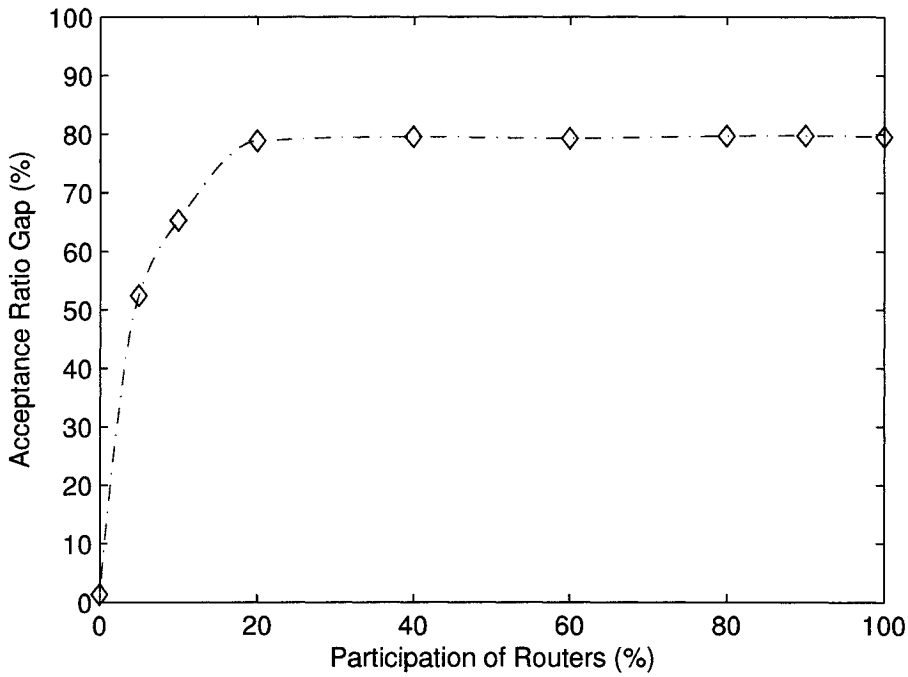

Figure 6.2: Acceptance Ratio Gap vs. Different Participation Percentage under Spoofed Attack

p(before attack detection)	Good_Packet_Accepted
0.9	98.73%
0.75	96.58%
0.5	92.13%

Table 6.1: Accepting Ratio of Packets with Different Probability in Normal Time

6.2.2 Performance under Random Attack

We have also tested MDADF scheme's performance under randomized attack when attackers use randomly generated IP addresses. The effect of the number of attackers on accepting good and bad packets under different participation ratio are shown in Figure 6.4 and Figure 6.5, respectively.

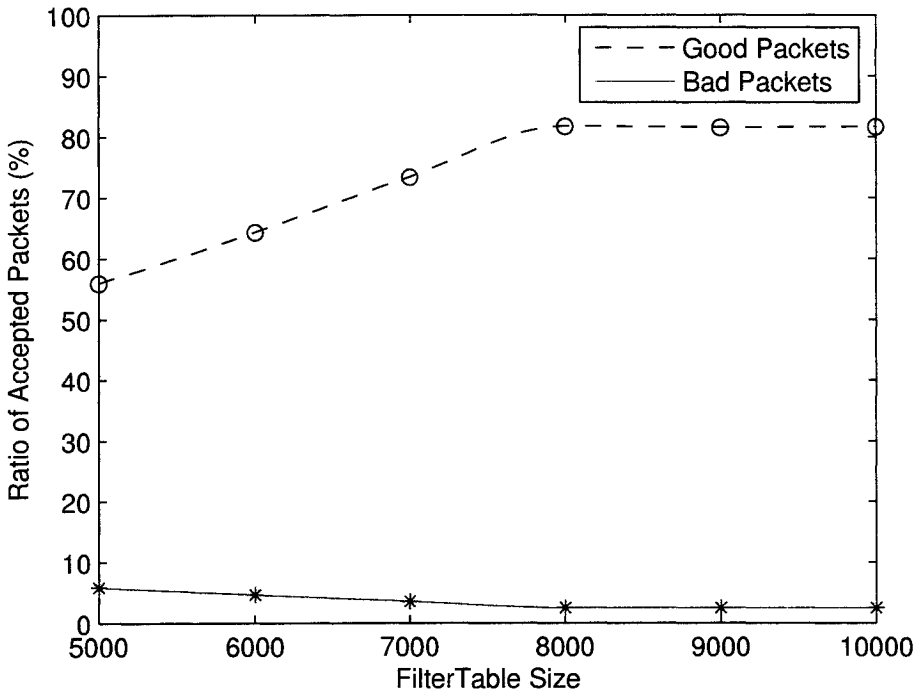

Figure 6.3: Ratio of Accepted Packets vs. Different Filter Table Size

6.2.3 Attack Detection Time

Under both types of attacks, our scheme can detect the occurrence of attack in 3 - 4 seconds under most simulated conditions. However, the number of attackers and their participation rate can affect the detection time as shown in Table 6.3 and Table 6.4.

p(before attack detection)	p(after)	Good_Packet_Accepted	Bad_Packet_Accepted
0.9	0.1	82.11%	2.52%
0.75	0.1	82.00%	2.41%
0.5	0.1	81.69%	2.29%
0.75	0.1	82.00%	2.41%
0.75	0.25	86.59%	4.86%
0.75	0.5	92.28%	8.04%

Table 6.2: Accepting Ratio of Packets with Different Probability in Attack Time

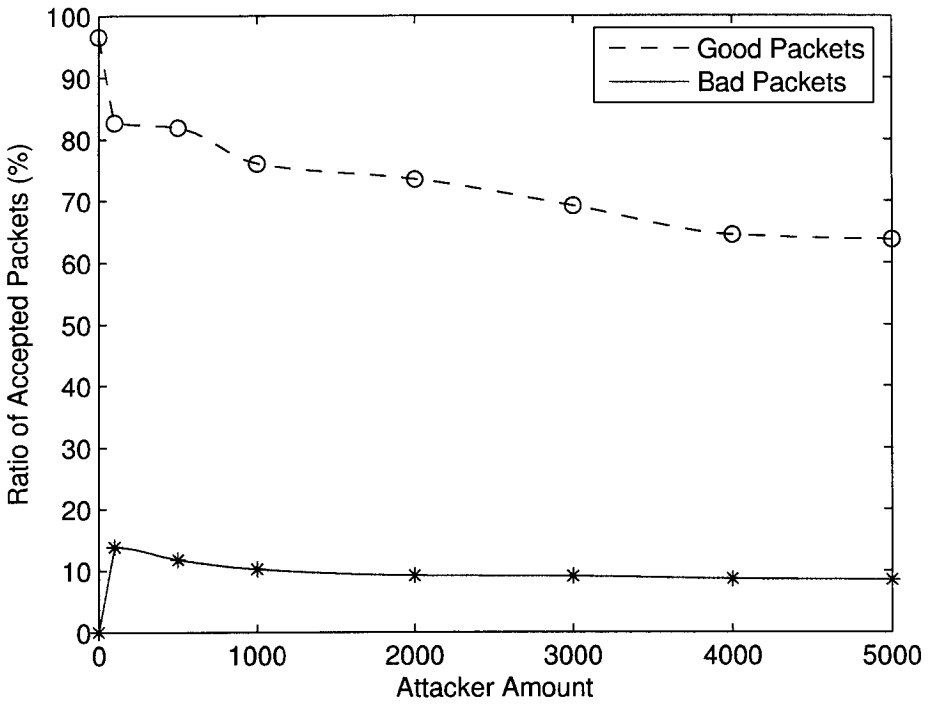

Figure 6.4: Ratio of Accepted Packets vs. Different Number of Attackers under Random Attack

6.3 Analysis

Using the results of simulation, we can analyze the overall performance of MDADF scheme. All the results are based on one of the critical settings – a ten-minutes learning phase and the start of attack immediately after it.

Figure 6.1 shows that more than 70% good packets are accepted even in the most severe condition under spoofed attack, in which the attack traffic is almost 10 times of the normal traffic.

Though the good packets acceptance ratio decreases a little with the increase of attackers, the bad packets accepting ratio stays at a low level – even less than 3%. Some packets

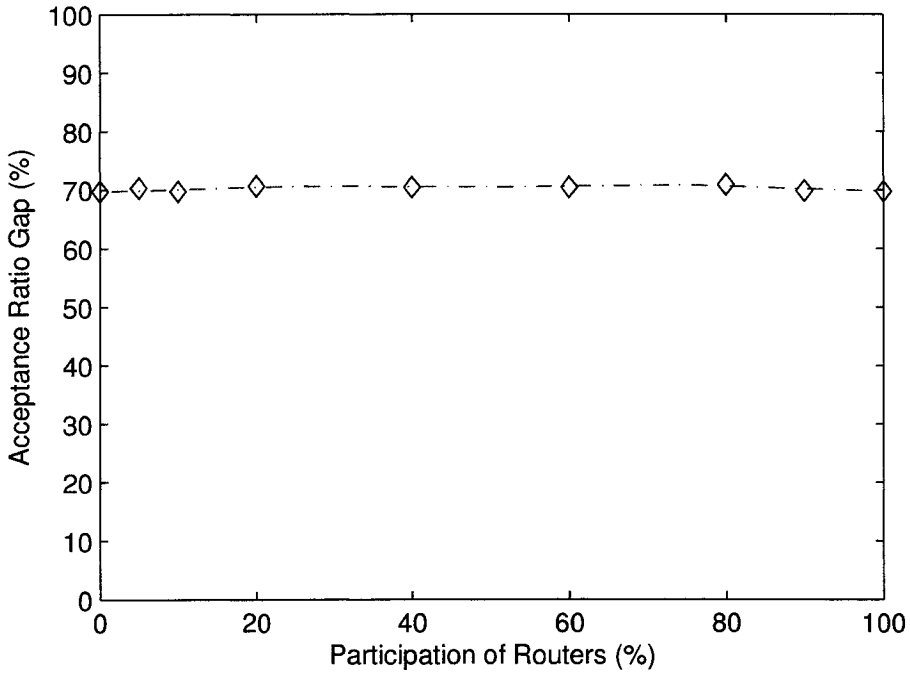

Figure 6.5: Acceptance Ratio Gap vs. Different Participation Percentage under Random Attack

are dropped due to the heavy congestion. This problem could be handled in our simulation by increasing the size of packet-queue, and it is actually decided by the bandwidth, processing speed, buffer size, or other resources of the website in real implementation.

The participation of routers is important to MDADF scheme, since their markings are the criteria for differentiating good and bad packets. Figure 6.2 shows that many more good packets are accepted than the bad ones even when the participation rate is only 20%. This means that the MDADF scheme can efficiently distinguish between good and bad packets when just 20% of routers in the Internet deploy our marking scheme.

The size of the Filter Table has a considerable effect on the performance. Undoubtedly, more records are stored in the Filter Table, less good packets will be dropped mistakenly.

Number of Attackers	Attack Detection Time (sec)	
	Spoofed Attack	Randomized Attack
100	6.99	7.02
500	3.19	3.29
1000	3.10	3.12
2000	3.06	3.09
3000	3.03	3.08
4000	3.03	3.05
5000	3.05	3.05

Table 6.3: Attack Detection Time for Different Number of Attackers

Participation Rate	Attack Detection Time (sec)	
	Spoofed Attack	Randomized Attack
100%	3.21	3.25
90%	3.22	3.37
80%	3.22	3.18
60%	3.30	3.22
40%	3.18	3.20
20%	3.22	3.25
10%	3.76	3.27
5%	4.29	3.21
0%	12.74	3.24

Table 6.4: Attack Detection Time by Different Participation Rate

Due to the limitation of resources in real implementation, the Filter Table can possibly record the information of only a fraction of the users. Figure 6.3 shows that keeping 80% of the legitimate user records is sufficient to filter packets, and even 70% is good enough.

For those packets whose IP addresses are not recorded in the Filter Table, their accepting ratio is absolutely decided by the probability value. Table 6.2 shows the effect of the value of p (both before and after attack detection) on the accepting ratio of good and bad packets. If the Filter Table is big enough and learning phase is sufficiently long, not too many legitimate users are new to the system, so the value of p will not affect the

good packets accepting ratio very much. Similarly, if the markings of most of the new IP addresses have been verified before attack happens, then p(after) has little effect on the acceptance of good packets. In an actual implementation, the probability value can be decided based on the processing ability of the firewall.

In the tests for random attack, we can see that MDADF scheme still works well under aggregate attacks (Figure 6.4). Though it is not as good as that under spoofed attack, the good packet accepting radio is more than 60% and about 10% bad packets are accepted. As to the participation rate, Figure 6.5 shows that the participation does not change the acceptance ratio gap. As the attackers use random IP addresses, instead of spoofing legitimate users' IP addresses, we cannot recognize the attack packets basing on the markings. The possibility that the random address happens to be same with one of the legitimate users is quite low, so every attack packet possibly has a new IP address and the possibility to accept it is totally decided by the value of p.

It can be seen from these results that the MDADF scheme can really protect a website or a network under DDoS attack, by accepting most of the good packets and dropping the majority of attack flow. Also, it detects the occurrence of attack quickly (as shown in Table 6.3 and Table 6.4), so that the victim has time to employ suitable countermeasures.

In a real world implementation, the results could be better than those obtained in the simulation. Normally a web-server on the Internet gets more packets from hosts which have sent packets to it recently than from new hosts. However, in our simulation, every host has equal probability to send a packet. That is why we believe that the good packet accepting ratio will be better in the real-world scenario.

Chapter 7

Conclusion & Future Work

7.1 Summary

In this thesis, we have analyzed DoS attack and some typical defense mechanisms. Aiming at improving the practical implementation issues existing in current proposed solutions, we have provided a low-cost and efficient scheme, named Marking-based DDoS Attack Detection and Filtering (MDADF) scheme.

The MDADF scheme is composed of two parts: marking process and filtering process. The marking process requires the participation of routers in the Internet to mark packets so that the path information can be encoded into them. After learning the markings of legitimate users, the spoofed attack packets will be filtered out. For those packets with unknown IP addresses, they are accepted with certain probability, and the markings in the accepted packets will be verified by sending echo request messages to their sources. When the total number of packets with mis-matching markings and with unknown IP addresses exceeds a threshold, an attack is signaled.

We have simulated the Internet traffic to test MDADF scheme's performance. The results show that our scheme can effectively and efficiently differentiate between good and

bad packets under spoofed attack when the routers' participation rate is as low as 20%. Also, most good packets are accepted even under the most severe attack, whose traffic is about 10 times of normal traffic. At the same time, the bad packet accepting ratio is maintained at a quite low level.

Some settings, such as the length of learning phase, the size of Filter Table, and the probability of accepting new packets, have direct influence on the final results. The first two parameters affect the overall packet acceptance rate, while the last one impacts those packets not recorded in the Filter Table. As long as the Filter Table can store 80% of the legitimate users' information, MDADF scheme will have an excellent performance.

Though the performance of MDADF scheme under random attack is not as good as that under spoofed attack, the good packet accepting ratio is still high. Because the attackers do not spoof legitimate users' IP addresses in their packets, we cannot use Filter Table to identify the attack packets so that bad packet accepting ratio is slightly higher.

Under both spoofed and random DDoS attacks, our scheme can detect the occurrence of attack precisely within 3 - 4 seconds. The quick detection is valuable to the victim so that a suitable defense mechanism can be deployed to alleviate the damage caused by DDoS attack to a great extent.

Therefore, the deployment cost of the MDADF scheme is low – only 20% of the Internet routers' cooperation is needed. It ensures that most legitimate users can get services from the victim after DDoS attack happens. Moreover, attacks can be detected precisely and quickly.

7.2 Future Work

In this thesis, we have presented a novel scheme for detecting DDoS attack and have tested its performance under various operating conditions. The results show that it is a low-cost and efficient scheme with certain limitations. However, it is desirable to have a thorough test of the scheme to fully understand how the limitations can be handled. Our future work will focus on performing a complete analysis of all the parameters to find out the suitable configurations under different network conditions. Other desirable tasks are outlined below.

- Simulate the packet generation using different Internet dataset. Except for the Internet Mapping Project of Lumeta Corporation, whose database is used in this simulation, the Internet topology is also researched by the Cooperative Association for Internet Data Analysis (CAIDA) [16]. We can use these two databases together to evaluate the performance of our MDADF scheme.

- We have used fixed value for certain parameters in the simulation. These values are carefully chosen through tests, but may have certain influences on the overall performance. We plan to examine the effect of these parameters, such as the size of packet-queue, the length of normal time before attack happens, the interval to send echo messages, the time to reset TMC, the value of θ and δ, and so on.

- We have tested the effect of different sizes of Filter Tables and different probability values on the packets accepting ratio. Actually, these values can be changed dynamically basing on the users volume and resource occupancy. For example, we

can increase the size of Filter Table when there are too many verified (IP-address, Marking) pairs, or decrease the p(after) when the resources are almost exhausted. We will research on the suitable conditions for changing these values.

- The results show that MDADF scheme is not quite effective against random attacks. We will investigate criteria that can identify the packets with random source IP addresses.

- We have only introduced the concept of pushback implementation, but it is not tested or simulated. We plan to incorporate it as part of the test in the future.

Bibliography

[1] Najwa Aaraj, Sleiman Itani, and Darine Abdelahad. "Neighbor Stranger Discrimination (NSD)- A New Defense Mechanism against DDoS Attacks". *Proceedings of the 3rd FEA Student Conference*, May 2004.

[2] Jason Anderson. "An Analysis of Fragmentation Attacks". March 2001.

http://www-src.lip6.fr/homepages/Fabrice.Legond-Aubry/www.ouah.org/fragma.html

[3] Jason Barlow and Woody Thrower. "TFN2K – An Analysis".

http://www2.axent.com/swat/News/TFN2k_Analysis.htm

[4] Andrey Belenky and Nirwan Ansari. "IP Traceback With Deterministic Packet Marking". *IEEE Communications Letters*, vol. 7, no. 4, pp. 162-164, April 2003.

[5] Andrey Belenky and Nirwan Ansari. "Tracing Multiple Attackers with Deterministic Packet Marking (DPM)". *2003 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM 03)*, pp. 49-52, August 2003.

[6] Steve Bellovin. "ICMP Traceback Messages". *Internet draft, work in progress*, March 2000.

[7] Hal Burch and Bill Cheswick. "Tracing Anonymous Packets to Their Approximate Source". *Proceedings of the 14th Systems Administration Conference (LISA 2000)*, pp. 319-327, December 2000.

[8] CERT® Advisory CA-1996-01. "UDP Port Denial-of-Service Attack". http://www.cert.org/advisories/CA-1996-01.html

[9] CERT® Advisory CA-1996-21. "TCP SYN Flooding and IP Spoofing Attacks". http://www.cert.org/advisories/CA-1996-21.html

[10] CERT® Advisory CA-1997-28. "IP Denial-of-Service Attacks". http://www.cert.org/advisories/CA-1997-28.html

[11] CERT® Advisory CA-1998-13. "Vulnerability in Certain TCP/IP Implementations". http://www.cert.org/advisories/CA -1998-13.html

[12] CERT® Advisory CA-2000-21. "Denial-of-Service Vulnerabilities in TCP/IP Stacks". December 2000. http://www.cert.org/advisories/CA-2000-21.html.

[13] CERT® Coordination Center. "Denial of Service Attacks". http://www.cert.org/tech_tIPs/denial_of_service.html

[14] CERT® Coordination Center. "CERT/CC Statistics 1988-2005". http://www.cert.org/stats/cert_stats.html#incidents

[15] CERT® Incident Note IN-2000-05. " 'Mstream' Distributed Denial of Service Tool". May 2000. http://www.cert.org/incident_notes/IN-2000-05.html

[16] Cooperative Association for Internet Data Analysis. Skitter.

http://www.caida.org/tools/measurement/skitter/

[17] Bill Cheswick and Hal Burch. Internet Mapping Project.

http://research.lumeta.com/ches/map/

[18] Brandi Copans. "NAPTHA: A New Type of Denial of Service Attack". December 2000. http://madchat.org/reseau/ddos/naptha2.htm

[19] Drew Dean, Matt Franklin, and Adam Stubblefield. "An Algebraic Approach to IP Trackback". *Proceedings of the 2001 Network and Distributed System Security Symposioum*, pp. 3-12, February 2001.

[20] Sven Dietrich, Neil Long, and David Dittrich. "An Analysis of the 'Shaft' Distributed Denial of Service Tool". March 2000.

http://home.adelphi.edu/ spock/shaft_analysis.txt

[21] Sven Dietrich, Neil Long, and David Dittrich. "Analyzing Distributed Denial of Service Tools: The Shaft Case ". *Proceedings of the 14th Systems Administration Conference (LISA 2000)*, pp. 329-339, December 2000.

[22] David Dittrich. 'The DoS Project's 'Trinoo' Distributed Denial of Service Attack Tool". October 1999. http://staff.washington.edu/dittrich/misc/trinoo.analysis.txt.

[23] David Dittrich. 'The 'Stacheldraht' Distributed Denial of Service Attack Tool". December 1999. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis

[24] David Dittrich. "The 'Tribe Flood Network' Distributed Denial of Service Attack Tool". December 1999. http://staff.washington.edu/dittrich/misc/tfn.analysis

[25] David Dittrich, George Weaver, Sven Dietrich, and Neil Long. "The 'Mstream' Distributed Denial of Service Attack Tool". May 2000.

http://staff.washington.edu/dittrich/misc/mstream.analysis.txt

[26] Anastasia Gemona, Ishbel Duncan, Colin Allison, and Alan Miller. "A Measurement Approach To Combating SYN Floods". *5th International Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, pp. 134-139, 2004.

[27] Kevin J. Houle, George M. Weaver, Neil Long, and Rob Thomas. "Trends in Denial of Service Attack Technology". October 2001.

[28] Alefiya Hussain, John Heidemann, and Christos Papadompoulos. "A Framework for Classifying Denial of Service Attacks". *Proceedings of ACM SIGCOMM 2003*, August 2003.

[29] Internet System Consortium. "ISC Domain Survey: Number of Internet Hosts".

http://www.isc.org/index.pl?/ops/ds/host-count-history.php

[30] Internet World Stats. "Internet User Statistics – The Big Picture: World Internet Users and Population Stats". http://www.internetworldstats.com/stats.htm

[31] John Ioannidis and Steven M. Bellovin. "Implementing Pushback: Router-Based Defense Against DDoS Attacks". *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, pp.6-8, February 2002.

[32] Sherif M. Khattab, Chatree Sangpachatanaruk, Rami Melhem, Daniel Mosse, and Taieb Znati. "Proactive Server Roaming for Mitigating Denial-of-Service Attacks". *Proceedings of the 1st International Conference on International Technology: Research and Education (ITRE 03)*, pp. 500-504, August 2003.

[33] Yoohwan Kim, Wing C. Lau, Mooi C. Chuah, and H. Jonathan Chao. "PacketScore: Statistics-based Overload Control against Distributed Denial-of-Service Attacks". *Proceedings of IEEE INFOCOM*, pp. 2594-2604, March 2004.

[34] Malachi Kenney. "Ping of Death".

http://www.insecure.org/sploits/ping-o-death.html

[35] Allison Mankin, Dan Massey, Chien-Lung Wu, S. Felix Wu, Lixia Zhang. "On Design and Evaluation of "Intention-Driven" ICMP Traceback". *IEEE International Conference on Computer Communication and Networks (ICCCN'01)*, pp. 159-165, October 2001.

[36] Michael Marchesseau. " 'Trinity' - Distributed Denial-of-Service Attack Tool". September 2000.

http://www.giac.org/certified_professionals/practicals/gsec/0123.php

[37] Jelena Mirkovic, Gregory Prier, and Peter Reiher. "Attacking DDoS at the Source". *Proceedings of the IEEE International Conference on Network Protocols 2002*, pp. 312-321, November 2002.

[38] Jelena Mirkovic and Peter Reiher. "A Taxonomy of DDoS Attack and DDoS Defense Mechanisms". *ACM Computer Communications Review*, vol. 34, no. 2, pp. 39-54, April 2004.

[39] Kihong Park and Heejo Lee. "On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets". *Proceedings of ACM SIGCOMM 2001*, August 2001.

[40] Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. "Adjusted Probabilistic Packet Marking for IP Traceback". *Networking 2002*, pp. 697-708, May 2002.

[41] Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. "Detecting Distributed Denial of Service Attacks by Sharing distributed Beliefs".*Proceedings of the 8th Australasian Conference on Information Security and Privacy 2003*, pp. 214-225, July 2003.

[42] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson. "Practical Network Support for IP Traceback". *Proceedings of ACM SIGCOMM 2000*, August 2000.

[43] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer. "Hash-Based IP Traceback". *Proceedings of ACM SIGCOMM 2001*, pp. 3-14, August 2001.

[44] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer. "Single-Packet IP Traceback". *IEEE/ACM Transactions on Networking (ToN)*, 10(6), December 2002.

[45] Dawn Xiaodong Song and Adrian Perrig. "Advanced and Authenticated Marking Schemes for IP Traceback". *Proceedings of IEEE INFOCOM*, pp.878-886, April 2001.

[46] Stephen Specht and Ruby Lee. "Taxonomies of Distributed Denial of Service Networks, Attacks, Tools, and Countermeasures". Technical Report CE-L2003-03. May 2003.

[47] Ion Stoica and Hui Zhang. "Providing Guaranteed Services Without Per Flow Management". *Proceedings of ACM SIGCOMM 1999*, pp. 81-94, April 1999.

[48] Robert Stone. "CenterTrack: An IP Overlay Network for Tracking DoS Floods". *Proceedings of USENIX Security Symposium '00*, pp.199-212, August 2000.

[49] Steven J. Templeton, and Karl E. Levitt. "Detecting Spoofed Packets". *Proceedings of 3rd IEEE DARPA Information Survivability Conference and Exposition (DISCEX)*, pp.164-175, 2003.

[50] Jun Xu and Wooyong Lee. "Sustaining Availability of Web Services under Distributed Denial of Service Attacks". *IEEE Transactions on Computers*, vol. 52, no. 2, February 2003.

[51] Abraham Yaar, Adrian Perrig, and Dawn Song. "Pi: A Path Identification Mechanism to Defend against DDoS Attacks". *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 93-109, May 2003.

[52] Abraham Yaar, Adrian Perrig, and Dawn Song. "SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks". *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 130-146, May 2004.