Towards IP Traceback based Defense against DDoS Attacks

LAU Nga Sin

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Computer Science and Engineering

Abstract of thesis entitled:

Towards IP Traceback based Defense against DDoS Attacks

Submitted by LAU Nga Sin

for the degree of Master of Philosophy

at The Chinese University of Hong Kong in June 2004

In recent years, malicious attacks on the Internet have increased dramatically in frequency, severity and sophistication. Distributed Denial of Service (DDoS) attacks, in particular, have drawn great concern to the Internet community because they seek to consume the resources of target hosts or networks, thereby, denying or degrading the normal service provided by the victim to its legitimate users.

One effective countermeasure against DDoS attack is to quickly identify the attack sources and separate them from the victim's network. However, finding the attack sources could be difficult because

1. the attacker can arbitrarily spoof the source IP address; and
2. the Internet is stateless and hence the routers do not store any information about the packets forwarded by them.

Therefore, locating the attack sources usually involves a relatively lengthy process, often referred to as *IP traceback*, in which the paths of the relevant packets are traced.

In this thesis, we propose a novel and comprehensive IP traceback based DDoS defense scheme which exploits the attack source information from IP traceback to perform packet filtering at strategic positions. With the knowledge of attack paths and source routers, most of the malicious packets can be filtered.

The defense scheme is comprised of two major components, namely the *domain-based IP traceback scheme*, and the *route-based packet filtering scheme*. The underlying algorithms of the proposed defense scheme have been well tested through numerous simulation experiments. As shown by the experiment results, the defense scheme could remove over 90% of attack traffic and maintain a high percentage of normal packets reaching the victim during an on-going DDoS attack. In other words, the proposed solution significantly improves the overall throughput of the legitimate traffic, while reducing the collateral damage and bandwidth consumption by the attack traffic.

摘要：

論文題目：**分散式阻絕服務攻擊之防禦**
作者：**劉雅倩**
修讀學位：**哲學碩士**
香港中文大學計算機科學與工程學系
日期：二零零四年六月

近年來，網路攻擊事件的發生次數日漸頻密，其破壞力及複雜性亦大大提高。分散式阻絕服務攻擊(Distributed Denial of Service, DDoS attacks)更特別受廣大互聯網的用戶所關注，皆因此類攻擊能夠耗盡目標主機或網路的資源，從而阻止或拖延目標主機對客戶的服務，或使目標網段的交通壅塞。

對付此攻擊的最有效辦法，就是準確找出攻擊的來源，及盡快把它們從受害者的區域網路中分隔開。不過，找尋攻擊的真正來源並不容易，原因是：

一・攻擊者可隨意僞造封包(packet)的來源IP位址(source IP address)；
二・互聯網中的路由器(router)並不會記錄有關封包傳送的資料。

因此，確認攻擊的來源需要追蹤封包所經過的路徑，此項方法稱爲「位址追蹤」(IP Traceback)。

在這篇論文中，我們提出一個既創新又完善的DDoS防禦方案。利用已有的攻擊來源資料，我們可於適當位置安裝過濾程式(filtering scheme)，以丟棄接踵而來的封包。集合位址追蹤及封包過濾的技術，此防禦方案既可移除高達百分之九十的可疑封包(malicious packet)，亦同時能有效提高其他正常交通的流量，及減低頻寬(bandwidth)的使用量。

Acknowledgement

I would like to express my deepest gratitude to my supervisor, Prof. M. C. Lee for his generous guidance and patience in all aspects throughout the two years of my MPhil study. He introduced to me to research; polished my writing and presentation skills; gave me opportunities to attend international conferences, which benefit me for the rest of life.

I would like to extend my thanks to my thesis examiners, Prof. Ada Fu, Prof. Jeff Yan and Prof. Y. K. Chan, for their invaluable comments and time in marking my thesis. Without their efforts, I will not be able to strengthen and improve my research project and papers. I would also like to show my gratitude to the Department of Computer Science and Engineering, CUHK, for the provision of the best equipment and pleasant working environment required for high quality research.

Special thanks should be given to T. L. Kwan and C. H. Chan for their help and encouragement in the preparation of the thesis. I would also like to give thanks to my fellow colleagues, P. W. Chan, W. Hung, C. Y. Ip, Y. Lam, C. H. Law, K. Y. Lee, C. W. Leung, C. H. Ng, C. H. Ngai, W. M. Szeto, C. W. Wong, Y. T. Wu, Y. K. Yu, P. Yan, and J. Y. Zheng. They have given me a joyful and fruitful university life. Last but not least, I am deeply indebted to my family for their unconditional love and support over the years.

This work is dedicated to my family for their support and patience.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, *Denial of Service* (DoS) attacks have become a serious threat to the Internet and they have increased dramatically in frequency, severity and sophistication [38, 63]. Such attack normally involves a malicious party sending enormous volumes of traffic to a remote host or a network, thereby denying the victim from providing normal service to its legitimate users. *Distributed Denial of Service* (DDoS) attacks are a much more powerful form of DoS attacks because they are designed as a coordinated attack from many sources simultaneously against one or more targets [48].

In this paper, we propose a comprehensive DDoS defense scheme which combines the techniques of IP traceback and packet filtering to withstand the effect of DDoS attacks. The defense scheme is comprised of two major components, namely *domain-based IP traceback* and *route-based packet filtering.*

In our IP traceback scheme, a participating router would mark the packets with a low probability. After collecting sufficient packets, the victim would be able to reconstruct the attack graph incorporating attack paths and the source routers identified, with each node on the paths viewed as a domain.

The attack graph serves as a filtering signature, which would be sent to the

routers equipped with our packet filtering scheme. The routers would examine the markings embedded in each incoming packet and match them with the filtering signature. If the packet is found to be malicious, the routers would drop it with certain probability. By performing this preferential filtering, the majority of attack traffic would be discarded before they reach the destination.

1.1 Research Motivation

The Internet, designed and started as the *Advanced Research Projects Agency Network* (ARPANET) in 1970s [85], has now become almost an indispensable communication and electronic service provision platform for our various daily life activities. Unfortunately, as witnessed by the dramatic increase of attack incidents, every computer host connected to the Internet is liable to being attacked. Not only the majority of Internet users are annoyed by the failures of Internet service access, the commercial entities on the Internet are also suffering from financial loss and bad publicity.

Distributed Denial of Service, or DDoS attacks, in particular, have caused severe Internet service disruptions in recent years [32]. They are among the hardest security problems to address because they are easy to implement, difficult to prevent, and hard to trace [72]. Since DDoS attack is a much tougher problem than the single-site form DoS attack, we will take DDoS attack as our target problem in the sequel.

According to the CIAC (Computer Incident Advisory Capability), the first DDoS attack occurred in the summer of 1999 [22]. In February 2000, one of the first major DDoS attacks was waged against Yahoo.com. This attack kept Yahoo off the Internet for about two hours (see figure 1.1) and cost Yahoo a significant loss in advertising revenue [7].

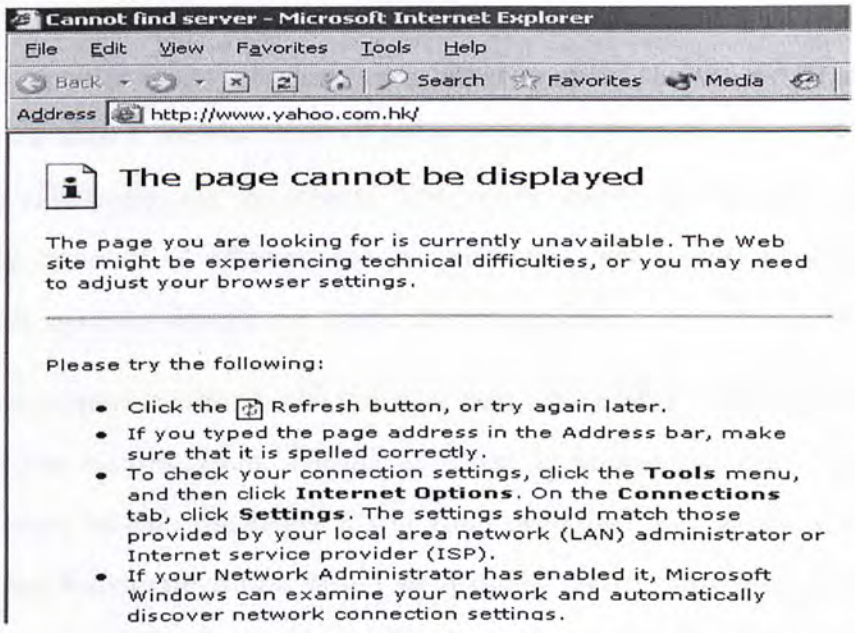

Figure 1.1: Denial of Service attack against Yahoo.com in Feb 2000

Another DDoS attack occurred in October 2002, against the 13 root servers that provide the Domain Name System (DNS) service to Internet users around the world [8]. These incidents have demonstrated the vulnerability of the Internet to DDoS attacks. Therefore, there is pressing need to carry out research in the field network security to devise an effective methodology for defending against DDoS attacks.

1.2 Problem Statement

Many researches have been carried out on defending against DDoS attacks, and different solutions have been proposed. However, there are two major limitations in the existing proposals:

1. The primary difficulty of dealing with DDoS attacks is *IP Spoofing* [33], which is a quite simple technique prevailing in DDoS attacks and other

network crimes due to the anonymous nature of Internet. When launching an attack, the attacker can arbitrarily spoof the source IP addresses in the attack packets to avoid being traced and blocked, so as to prolong the effectiveness of the attack. The source address being scattered across large amount of different spoofed addresses also makes some detection tools hard to identify the traffic anomalies [63].

2. Few proposals address the issues of *real time attack mitigation* and *false positive minimization.* During an on-going flooding attack, if no action is taken to stop the attack traffic from overwhelming the network, much of the legitimate traffic would be dropped by the upstream routers before reaching its destinations [58], or in other words, the legitimate traffic would suffer from *collateral damage.* The traffic is either harmed by the congestion of network, or is filtered by the deployed defense mechanism [19].

These are the challenges in defeating DDoS attacks, and we aim at developing a comprehensive defense scheme which can overcome these inherent problems. This kind of research could make a significant contribution to the advancement of network security.

1.3 Research Objectives

Our work is motivated by the increased frequency and sophistication of DDoS attacks, and we mainly focus on the flooding-based DDoS attacks, which could potentially cripple or disable essential Internet services in minutes. In order to design a robust and effective DDoS defense scheme, an intensive survey has been conducted on both the DDoS attack and its existing solutions. Various

DDoS countermeasures are compared and evaluated. Throughout the study, we discover that there are different proposals available for IP traceback, which aim at locating the potential attack sources. Nevertheless, they cannot be employed to defend against DDoS attacks.

Based on this finding, a new approach - a hybrid traceback-filtering defense scheme is proposed [50, 51]. In this scheme, traceback and filtering are the two major components in defending against DDoS attacks. The reasons for us to select and combine the two techniques are:

1. *IP traceback* is a major practice against IP spoofing problems, which involves tracing the paths of the anonymous packets back towards their genuine sources [10, 16]. Though IP traceback technique by itself has limited capability of sustaining the victim's service availability, the knowledge of the attack sources and paths is very useful in eliminating the attack traffic as well as improving the effectiveness of DDoS defense [72]. Such information is vital for restoring normal network functionality, preventing reoccurrences, and ultimately, holding the attackers accountable [54].

2. Accurate packet filtering could mitigate the effect of DDoS attacks rapidly by dropping the majority of attack traffic, thus significantly enhancing the throughput of legitimate traffic with minimum collateral damage and bandwidth consumption [19].

The main purpose of our work is to develop a DDoS defense scheme which can efficiently trace the attack paths, as well as make intelligent use of the traceback information for packet filtering purpose. Through numerous testing and evaluations, we aim at improving our scheme so as to:

- strengthen its defense power by minimizing false positives, processing overhead and collateral damage;

- make it more robust and effective even in the presence of multiple attacks;
- outperform existing packet marking and filtering schemes and overcome many of their inherent problems.

Our novel IP traceback based DDoS defense scheme exploits the attack source information from IP traceback to perform attack packets filtering at strategic positions. It is comprised of two schemes, as depicted in figure 1.2:

1. a domain-based IP traceback scheme, for tracing attack sources (see chapter 3);
2. a route-based packet filtering scheme, for dropping most of the attack traffic and preserving legitimate traffic (see chapter 4).

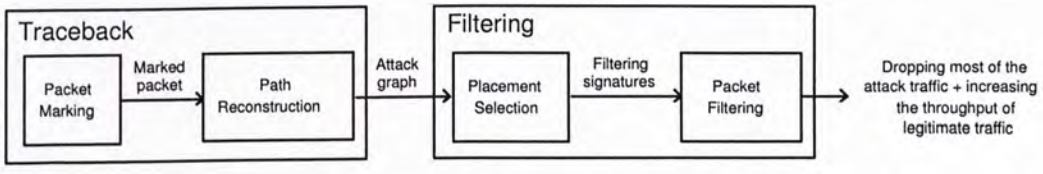

Figure 1.2: Architecture of the proposed DDoS defense scheme

The design of the defense scheme aims at maximizing the filtering of attack packets and minimizing the undesirable dropping of the legitimate packets, so that the DDoS attacks can be ultimately thwarted.

1.4 Structure of the Thesis

This thesis is organized as 6 chapters. Chapter 1 is an introduction of the thesis. In chapter 2, we present a background study of our work, including a survey of DDoS attack, a literature review on some of its countermeasures and their pros and cons. The core part of this thesis is our proposed DDoS defense

scheme, which would be presented in two parts: domain-based IP traceback scheme in chapter 3, and route-based packet filtering scheme in chapter 4. The simulation experiment results of our defense scheme are presented in chapter 5. Finally, we conclude this thesis and discuss the future work in chapter 6.

□ **End of chapter.**

Chapter 2

Background Study on DDoS Attacks

2.1 Distributed Denial of Service Attacks

Have you ever tried to make a telephone call but couldn't because all the telephone circuits were busy? This may happen on a major holiday. The reason you could not get through is because the telephone system is designed to handle a limited number of calls at a time. Imagine that an attacker wanted to make the telephone system unusable by customers. One way would be to make call after call in an attempt to make all circuits busy. This type of attack is called a *denial of service*, or *DoS*, attack.

Computer systems can also suffer *DoS* attacks. A malicious user exploits the connectivity of the Internet to cripple the services offered by a victim site, often simply by flooding a victim with many requests [39, 44]. A DoS attack can be either a single-source attack, originating at only one host, or a multi-source, where multiple hosts coordinate to flood the victim with a barrage of attack packets. The latter is called a *distributed denial-of-service*, or *DDoS*,

attack. It is a virulent, relatively new type of attack on the availability of Internet services and resources. DDoS attackers infiltrate large numbers of hosts by exploiting software vulnerabilities. These unwitting hosts are then invoked to wage a coordinated, large-scale attack against one or more victims.

To achieve our project objective - developing an effective DDoS solution, we first study some common techniques used in DDoS attacks and present a comprehensive scope in this section. We aim at defining DDoS attack architecture, classifying different DDoS techniques, and describing the characteristics of the software tools used in setting up a DDoS attack network. These taxonomies help us to understand the similarities and differences in DDoS attacks and tools, and the scope of the DDoS problem.

2.1.1 DDoS Attack Architecture

In a typical DDoS attack, many hosts are compromised to fire packets simultaneously against one or more targets. Such attack is large-scale and coordinated in a way to deny the services of a system or network resource. Moore et al. [63] use backscatter packets (the unsolicited responses that a DDoS victim sends to the spoofed IP address that it receives in the attack packet) to gauge the level of Internet DoS activity. Jung et al. [43] attempt to answer the question of how a site can differentiate between a DDoS attack and a simple high load condition by analyzing client request rates and file access patterns.

The target under attack is defined as *primary victim*, while the compromised hosts used to launch the attack are often called *secondary victims*. The use of secondary victims in performing a DDoS attack provides the attacker with the ability to wage a much larger and more disruptive attack, while making it more difficult to track down the original attack source.

Since a number of attacking hosts are involved, the attacker has to first set

up a DDoS attack network before launching an attack, as shown in figure 2.1. The *masters* are software packages located on computing systems throughout the Internet that the attacker uses to communicate indirectly with the *agents*. The agent software exists in compromised systems that will eventually carry out the attack on the victim system. A detailed description of the entire process of building a DDoS attack network is given in [32]. Some researchers have also analyzed other forms of DDoS network in [90, 89, 36]. In descriptions of DDoS tools, the terms *master* and *agents* are sometimes replaced with *handler* and *daemons* respectively. Also, the systems that have been violated to run the agent software are also defined as *secondary victims* or *zombies*.

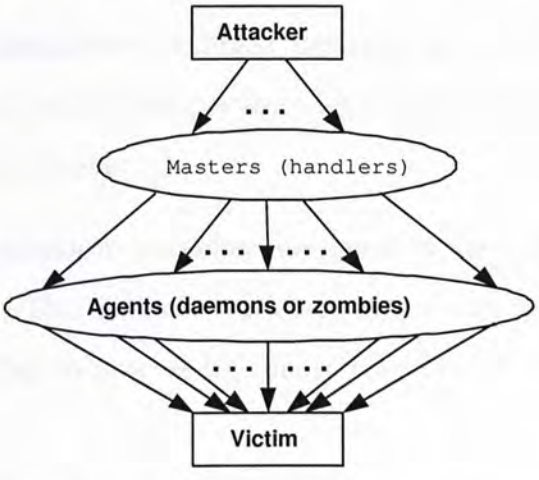

Figure 2.1: DDoS Attack Network

The attacker communicates with any number of masters to identify which agents are running, when to schedule attacks, or when to upgrade agents. Usually, attackers will try to place the master software on a compromised router or network server that handles large volumes of traffic. This makes it harder to identify messages between the master and agents. The users of the agent systems typically have no knowledge that their system has been compromised and will be taking part in a DDoS attack. When participating in

a DDoS attack, each agent program uses only a small amount of resources (both in memory and bandwidth), so that the users of these computers experience minimal change in performance [38]. Today's DDoS attack tools can launch attacks against multiple victims at the same time, and use various types of attack packets.

2.1.2 DDoS Attack Taxonomy

There are a wide variety of DDoS attack techniques. We present a taxonomy of the main DDoS attack methods in figure 2.2. There are two main classes of DDoS attacks:

1. **Bandwidth depletion attacks:** designed to flood the victim network with unwanted traffic that prevents legitimate traffic from reaching the primary victim system;

2. **Resource depletion attacks:** designed to tie up the resources of a victim system. This type of attack targets a server on the victim system, making it unable to process legitimate requests for service.

Bandwidth Depletion Attacks

There are two main classes of DDoS bandwidth depletion attacks. A *flood attack* involves the zombies sending large volumes of traffic to a victim system, to congest the victim system's bandwidth. An *amplification attack* involves either the attacker or the zombies sending messages to a broadcast IP address, using this to cause all systems in the subnet reached by the broadcast address to send a message to the victim system. This method amplifies malicious traffic that reduces the victim system's bandwidth.

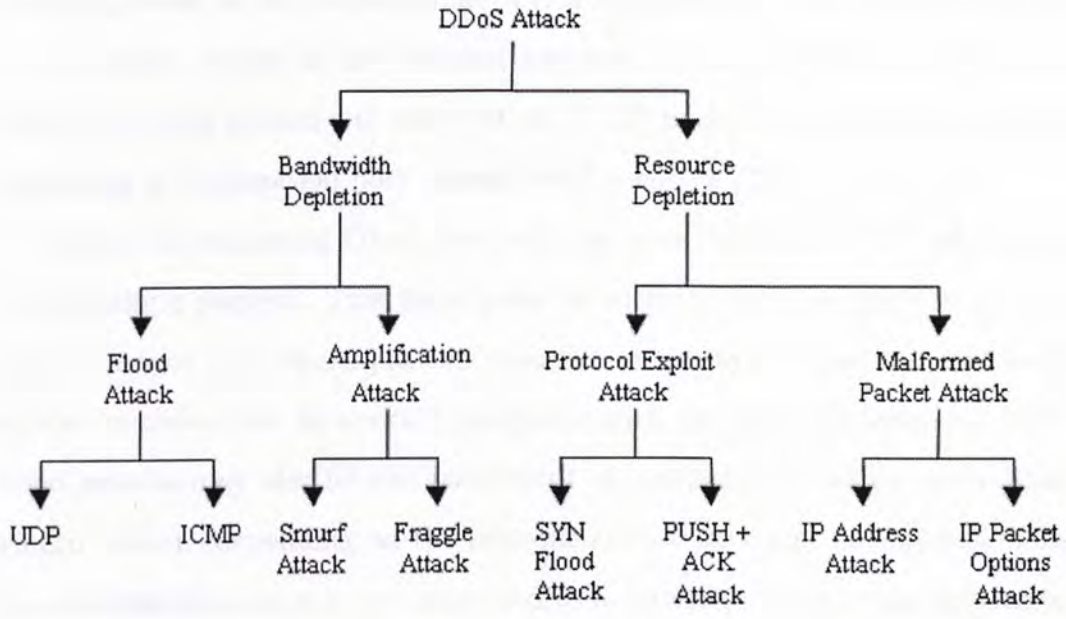

Figure 2.2: DDoS Attack Taxonomy

Flood Attacks. In a DDoS flood attack the zombies flood the victim system with IP traffic. The large volume of packets sent by the zombies to the victim system slows it down, crashes the system or saturates the network bandwidth. This prevents legitimate users from accessing the victim.

UDP Flood Attacks. User Datagram Protocol (UDP) is a connectionless protocol. When data packets are sent via UDP, there is no handshaking required between sender and receiver, and the receiving system will just receive packets it must process. A large number of UDP packets sent to a victim system can saturate the network, depleting the bandwidth available for legitimate service requests to the victim system.

In a DDoS UDP Flood attack, the UDP packets are sent to either random or specified ports on the victim system. Typically, UDP flood attacks are designed to attack random victim ports. This causes the victim system to process the

incoming data to try to determine which applications have requested data. If the victim system is not running any applications on the targeted port, then the victim system will send out an ICMP packet to the sending system indicating a "destination port unreachable" message [22].

Often, the attacking DDoS tool will also spoof the source IP address of the attacking packets. This helps hide the identity of the secondary victims and it insures that return packets from the victim system are not sent back to the zombies, but to another computer with the spoofed address. UDP flood attacks may also fill the bandwidth of connections located around the victim system (depending on the network architecture and line-speed). This can sometimes cause systems connected to a network near a victim system to experience problems with their connectivity.

ICMP Flood Attacks. Internet Control Message Protocol (ICMP) packets are designed for network management features such as locating network equipment and determining the number of hops or round-trip-time to get from the source location to the destination. For instance, ICMP ECHO REPLY packets ("ping") allow the user to send a request to a destination system and receive a response with the roundtrip time.

A DDoS ICMP flood attack occurs when the zombies send large volumes of ICMP ECHO REPLY packets to the victim system. These packets signal the victim system to reply and the combination of traffic saturates the bandwidth of the victim's network connection [22]. As for the UDP flood attack, the source IP address may be spoofed.

Amplification Attacks

A DDoS amplification attack is aimed at using the broadcast IP address feature found on most routers to amplify and reflect the attack (see figure 2.3). This feature allows a sending system to specify a broadcast IP address as the destination address rather than a specific address. This instructs the routers servicing the packets within the network to send them to all the IP addresses within the broadcast address range.

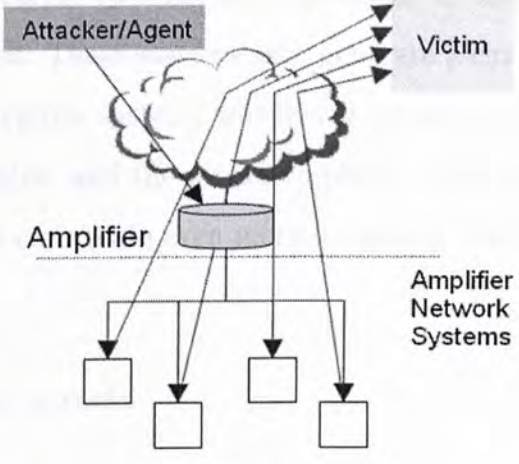

Figure 2.3: Amplification Attack

Smurf Attacks. In a DDoS Smurf attack, the attacker sends packets to a network amplifier (a system supporting broadcast addressing), with the return address spoofed to the victim's IP address. The attacking packets are typically ICMP ECHO REQUESTs, which are packets (similar to a "ping") that request the receiver to generate an ICMP ECHO REPLY packet. The amplifier sends the ICMP ECHO REQUEST packets to all of the systems within the broadcast address range, and each of these systems will return an ICMP ECHO REPLY to the target victim's IP address. This type of attack amplifies the original packet tens or hundreds of times.

Fraggle Attacks. A DDoS Fraggle attack is similar to a Smurf attack in that the attacker sends packets to a network amplifier. Fraggle is different from Smurf in that Fraggle uses UDP ECHO packets instead of ICMP ECHO packets. There is a variation of the Fraggle attack where the UDP ECHO packets are sent to the port that supports character generation (chargen, port 19 in Unix systems), with the return address spoofed to the victim's echo service (echo, port 7 in Unix systems) creating an infinite loop [60]. The UDP Fraggle packet will target the character generator in the systems reached by the broadcast address. These systems each generate a character to send to the echo service in the victim system, which will resend an echo packet back to the character generator, and the process repeats. This attack generates even more bad traffic and can create even more damaging effects than just a Smurf attack.

Resource Depletion Attacks

DDoS resource depletion attacks involve the attacker sending packets that misuse network protocol communications or sending malformed packets that tie up network resources so that none are left for legitimate users.

SYN Flood Attack. It is a TCP-based attack which exploits a leak in the implementation of the TCP/IP protocol. It involves sending a large number of spoofed TCP connection requests, which is never successfully established, to the server and therefore cramming the stack for storing these requests and tying the resources of the target machine. Thus, other legitimate connection requests are denied. Before we show how SYN Flood is launched, we should introduce TCP *three-way handshake* [67] first.

Figure 2.4 shows the structure of TCP header [47]. From the figure we can see that there are URG, SYN, ACK, RST, PSH and FIN six flags in a

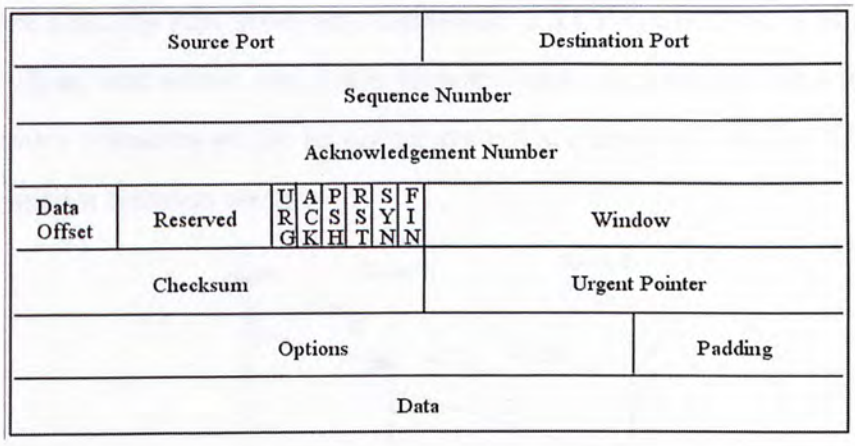

Figure 2.4: TCP Header

TCP header. The *three-way handshake* that is involved in a TCP connection establishment is illustrated in figure 2.5.

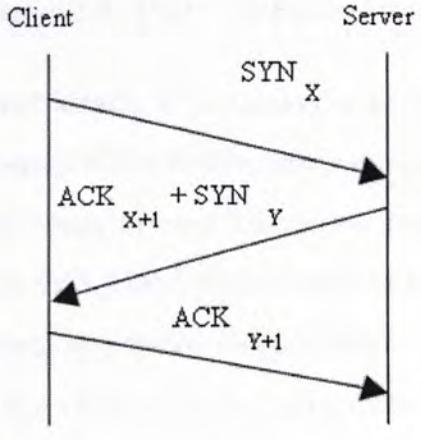

Figure 2.5: TCP Synchronization

At first, the initiating system sends a SYN (Synchronize) request. The receiving system sends an ACK (acknowledgement) with its own SYN request. The sending system then sends back its own ACK and communication can begin between the two systems. If the receiving system is sent a SYN_X packet but does not receive an ACK_{Y+1} to the SYN_Y it sends back to the sender, the receiver will resend a new $ACK + SYN_Y$ after some time has passed [20]

(see figure 2.6). By this three-way handshake a TCP connection is established and the client and server can begin transmit data each other. The processor and memory resources at the receiving system are reserved for this TCP SYN request until a timeout occurs.

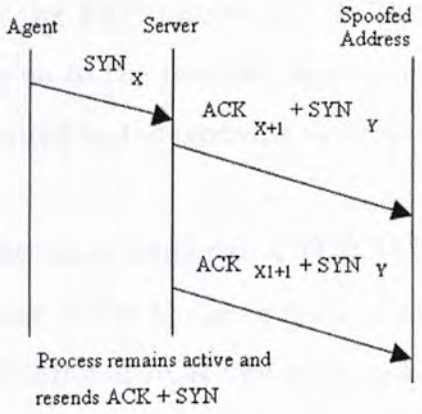

Figure 2.6: TCP SYN Flood Attack

In a DDoS SYN Flood attack, the attacker instructs the zombies to send such bogus TCP SYN requests to a victim server in order to tie up the server's processor resources, and hence prevent the server from responding to legitimate requests. The TCP SYN Flood attack exploits the three-way handshake between the sending system and the receiving system by sending large volumes of TCP SYN packets to the victim system with spoofed source IP addresses, so the victim system responds to a non-requesting system with the ACK+SYN. When a large volume of SYN requests are being processed by a server and none of the ACK+SYN responses are returned, the server begins to run out of processor and memory resources. Eventually, if the volume of TCP SYN attack requests is large and they continue over time, the victim system will run out of resources and be unable to respond to any legitimate users.

PUSH + ACK Attacks. In the TCP protocol, packets that are sent

to a destination are buffered within the TCP stack and when the stack is full, the packets get sent on to the receiving system. However, the sender can request the receiving system to unload the contents of the buffer before the buffer becomes full by sending a packet with the PUSH bit set to one. PUSH is a one-bit flag within the TCP header [4]. TCP stores incoming data in large blocks for passage on to the receiving system in order to minimize the processing overhead required by the receiving system each time it must unload a non-empty buffer.

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

Another type of TCP-based attack is to congest a victim's incoming link. Under these attacks, the victim usually responds with RST packets, except when the attack packets are also RST packets. ICMP messages (echo requests and timestamp requests) and UDP packets may also be used to achieve the same result. In these cases, the victim usually responds with the corresponding ICMP reply and error messages, and UDP packets.

Based on a backscatter analysis performed on the response packets sent back by attack victims, Moore et al. have provided insight into the prevalence of DoS activity on the Internet [63]. One notable observation reported is that most attacks used TCP packets (over 94 present), followed by UDP packets (2 present) and ICMP packets (2 present). The TCP-based attacks are observed mainly based on SYN-ACK packets, RST packets, and ICMP error messages

sent back by victims in response to attacks.

Malformed Packet Attack. It is an attack where the attacker instructs the zombies to send incorrectly formed IP packets to the victim system in order to crash the victim system. There are two types of malformed packet attacks. In an IP address attack, the packet contains the same source and destination IP addresses. This can confuse the operating system of the victim system and cause the victim system to crash. In an IP packet options attack, a malformed packet may randomize the optional fields within an IP packet and set all quality of service bits to one so that the victim system must use additional processing time to analyze the traffic. If this attack is multiplied using enough agents, it can shut down the processing ability of the victim system.

2.1.3 DDoS Tools

In 1998 the first known DDoS tool was developed. In rough chronological order, the DDoS tools [45] commonly seen today include:

Trinoo was the first known DDoS tool, starting to appear in June or July 1999. It is a kind of distributed SYN DoS attack [5, 27].

Tribe Flood Network (TFN) started to appear after Trinoo. TFN client and daemon programs implement a DDoS network capable of employing a number of attacks, such as ICMP flood, SYN flood, UDP flood, and SMURF style attacks. TFN is noticeably different than trinoo in that communication from the TFN client to daemons is accomplished via ICMP ECHO REPLY packets. The absence of TCP and UDP traffic sometimes makes these packets difficult to detect because many protocol

monitoring tools are not configured to capture and display the ICMP traffic [29].

Stachedraht (German for "barbed wire") is a DDoS tool that started to appear in the late summer of 1999 and combines features of trinoo and TFN. It also contains some advanced features, such as encrypted attacker master communication and automated agent updates [28].

Shaft is a DDoS tool which became available in November 1999. A Shaft network looks conceptually similar to a trinoo; it is a packet flooding attack and the client controls the size of the flooding packets and duration of the attack. One interesting signature of Shaft is that the sequence number for all TCP packets is 0x28374839 [26].

Tribe Flood Network 2000 (TFN2K) was released in December 1999. It is a complex variant of the original TFN with features designed specifically to make TFN2K traffic difficult to recognize and filter, remotely execute commands, hide the true source of the attack using IP address spoofing, and transport TFN2K traffic over multiple transport protocols including UDP, TCP, and ICMP. TFN2K attacks include flooding (as in TFN) and those designed to crash or introduce instabilities in systems by sending malformed or invalid packets, such as those found in the Teardrop and Land attacks [14].

Mstream was discovered in late April 2000 on a compromised Linux system at a major university. This system was identified to be flooding packets using forged source addresses, targeted at over a dozen IP addresses [30].

W32/MyDoom.B is reported on March 2004. It harvests email addresses from an infected system, generates network traffic and installs a backdoor.

It causes collateral denial-of-service conditions in networks where a significant number of systems are infected, large volumes of virus-related email are handled, or DDoS traffic is aggregated [2].

Attacks using these tools are based on the client/server model and can involve a large number of sites simultaneously.

2.1.4 DDoS Detection

The goal of attack detection is to detect every attempted DDoS attack as early as possible and to have a low degree of false positives. Upon attack detection, steps can be taken to characterize the packets belonging to the attack stream and provide this characterization to the response mechanism. Most of the detection works are performed by Intrusion Detection System (IDS) [49, 86, 25]. We classify different attack detection strategies as follows:

Pattern Detection

Mechanisms that deploy pattern detection store the signatures of known attacks in a database. Each communication is monitored and compared with database entries to discover occurrences of DDoS attacks. Occasionally, the database is updated with new attack signatures. As the event patterns are also called signatures, pattern detection is sometimes called "signature-based detection" [56].

The obvious drawback of this detection mechanism is that it can only detect known attacks, and it is usually helpless against new attacks or even slight variations of old attacks that cannot be matched to the stored signature. On the other hand, known attacks are easily and reliably detected, and no false positives are encountered. Systems like *Snort* [71], *USTAT* [40] and *Bro* [68]

are examples using pattern detection.

Anomaly Detection

Mechanisms that deploy anomaly detection have a model of normal system behavior, such as a model of normal traffic dynamics or expected system performance. The current state of the system is periodically compared with the models to detect anomalies. It relies on matching a user behavior against his/her profile of normal behavior, and raises an alarm when a deviation from normal behavior is detected [13]. Examples of anomaly detection systems are *IDES* [25], *Hyperview* [24] and *Wise and Sense* [84].

The advantage of anomaly detection over pattern detection is that unknown attacks can be discovered. However, anomaly-based detection has to address two issues:

1. *Threshold setting.* Anomalies are detected when the current system state differs from the model by a certain threshold. The setting of a low threshold leads to many false positives, while a high threshold reduces the sensitivity of the detection mechanism.

2. *Model update.* Systems and communication patterns evolve with time, and models need to be updated to reflect this change. Anomaly-based systems usually perform automatic model update using statistics gathered at a time when no attack was detected. This approach makes the detection mechanism vulnerable to attacks which are launched over a long period with small change of traffic rate, and thus be able to mislead the models to avoid attack detection.

2.2 DDoS Countermeasure: Attack Source Traceback

Several traceback approaches have been proposed. We will present four representatives out of these approaches.

2.2.1 Link Testing

The technique link testing is introduced where traceback starts from the router closest to the victim and interactively tests its upstream links until they determine which one is used to carry the attacker traffic. This procedure is repeated recursively on the upstream router until the source is reached [72]. This technique requires that an attack remains active until the completion of a trace. There are two variants of link testing schemes: input debugging and controlled flooding.

Input debugging [77] is one implementation of the link-testing approach. This method requires cooperation between system administrators of different ISPs to identify which router interface a particular packet was received. This may involve substantial management overhead in communicating and coordinating efforts across multiple network boundaries and ISPs. It requires time and personnel on both the victims' and ISPs' side. DDoS attacks compound this problem because attack traffic could originate from machines under the jurisdiction of many separate ISPs. Table 2.1 illustrates the advantages and disadvantages of input debugging.

Another technique that falls into the link-testing category is controlled flooding [18]. This technique works by selectively exhausting selected network resources and monitoring the traffic so as to detect the links that a DoS attack is traversing. But this technique is only applicable during on-going attacks. Moreover, the technique itself is a sort of DoS attack, which can disrupt le-

Table 2.1: Advantages and disadvantages of input debugging

Advantages	Disadvantages
Compatible with existing protocols	High overhead in terms of time and resources in organizations along the attack traffic path
Insignificant network traffic overhead	Communications and cooperation of ISPs along the attack path must be established
Supports incremental implementation	The attack must last long enough for a successful trace
Compatible with existing routers and network infrastructure	Less suitable for distributed denial-of-service attacks

Table 2.2: Advantages and disadvantages of controlled flooding

Advantages	Disadvantages
Compatible with existing protocols	Serves as a kind of denial-of-service attack
Support for incremental implementation	Requires accurate map of the network topology
Compatible with existing routers and network infrastructure	ISP cooperation might be required

gitimate traffic on the unsuspecting upstream routers and networks. This, of course, makes it unsuitable for widespread routine usage on the Internet. Table 2.2 illustrates the advantages and disadvantages of controlled flooding.

2.2.2 Logging

An obvious solution to establishing the true origin of offending Internet traffic is to log the packets at key routers throughout the Internet and then use data-mining techniques to extract information about the attack traffic's source. Although this solution seems obvious and allows accurate analysis of attack traffic even after the attack has stopped, its most significant drawbacks include the amount of processing and storage power needed to save the logs. Also, the need to save and share this information among ISPs poses logistical and legal

problems as well as privacy concerns. Given today's link speeds, packet logs can grow quickly to unmanageable sizes, even over short timeframes.

Although logging a probabilistic sampling of the packet stream and compression can reduce resource demands somewhat, those demands are still quite significant. Snoeren et al. [74] proposed a novel approach to logging and IP traceback called *SPIE* (Source Path Isolation Engine). Instead of storing the whole packet, they suggested storing only a hash digest of its relevant invariant portions in an efficient memory structure called a Bloom filter. To complete an IP traceback request, a network of data collection and analysis agents spanning the different networks could use this method to extract significant packet data and generate appropriate attack graphs, thus identifying the attack traffic's origin.

An alternate and innovative logging approach is proposed in [12], which entailed an overlay network built of sensors that could detect potential attack traffic, tracing agents that could log the attack packets on request, and managing agents that could coordinate the sensors and tracers and communicate with each other. This approach attempts to overcome traditional logging methods' limitations by selectively logging traffic-after an attack is recognized and logging only certain characteristics, rather than entire packets. The approach also allows for increased speed and requires less storage. Current logging-based traceback methods use a sliding time window for storing logged data to avoid excessive storage and analysis requirements in exchange for catching attacks while in progress or shortly thereafter (so that the required logging data is still available). Table 2.3 illustrates the advantages and disadvantages of logging.

Table 2.3: Advantages and disadvantages of logging

Advantages	Disadvantages
Insignificant network traffic overhead	Resource-intensive in terms of processing and storage requirements
Allows post-attack analysis	Sharing of the logging information among several ISPs leads to logistic and legal issues
Compatible with existing routers and network infrastructure	Less suitable for distributed denial-of-service attacks

2.2.3 ICMP-based traceback

Bellovin [17] proposed a new traceback scheme which relies on the IP protocol. He named his approach as *iTrace*, which adds a new type of ICMP message as the traceback message. For each packet received, routers would generate, with a small probability, an ICMP message to the destination address of the packet containing the IP address of the router. Network managers could piece together these messages to trace a packet's path back to its origin. To limit the additional traffic this method generates, a router would generate an ICMP traceback message for only one in 20,000 packets passing through it (0.005 percent). This low probability limits additional network traffic, but still lets the victim figure out the attack traffic's actual path.

One of the *iTrace* scheme's weaknesses becomes apparent in a DDoS attack in which each zombie contributes only a small amount of the total attack traffic. In such cases, the probability of choosing an attack packet is much smaller than the sampling rate used. The victim probably will get many ICMP traceback messages from the closest routers but very few originating near the zombies' machines.

To overcome this drawback, Mankin et al. [59] presented an improvement to this scheme, called *intension-driven ICMP traceback*. This technique sep-

Table 2.4: Advantages and disadvantages of ICMP-based traceback

Advantages	**Disadvantages**
Compatible with existing routers and network infrastructure	Generates additional network traffic
Supports incremental implementation	Unless there also is an encryption scheme with key distribution implemented, attackers could inject false ICMP traceback messages into the packet stream to mask the attack traffic's true origin
Allows post-attack analysis	ICMP traffic increasingly is filtered by organizations due to its use in several common attack scenarios
ISP cooperation is not required	Very few ICMP traceback messages from distant routers in the case of a distributed denial-of-service attack

arates the messaging function between the *decision module* and the *iTrace generation module*. A recipient network supplies specific information to the routing table to indicate it requests ICMP traceback message. On the basis of specific information provided in the routing table, the decision module would select which kind of packet to use next to generate an *iTrace* message. It also lets a recipient network signal whether it is interested in receiving *iTrace* packets, which increases the proportion of messages considered useful to the receiving network. This scenario also would be helpful if a given network suspects or detects that it is under attack: it could request *iTrace* packets from the upstream routers to identify the attack traffic's origin. Although this improvement reduces the overhead of ICMP traceback significantly, unless there is an encryption scheme with key distribution implemented, attackers could forge ICMP traceback messages to mask the attack traffic's true origin. Table 2.4 lists the advantages and disadvantages of ICMP-based traceback.

2.2.4 Packet marking

Packet marking methods are characterized by embedding traceback data within the IP header inside the packets to be traced. This approach lets the host machine use markings in the individual packets to deduce the path the traffic has taken. To be effective, packet marking should not increase the packets' size (to avoid additional downstream fragmentation, thus increasing network traffic). Furthermore, packet-marking methodologies must be secure enough to prevent attackers from generating false markings. Problems also arise when we try to work within the framework of existing IP specifications. The order and length of fields in an IP header are specified, so for the packet marking method to be effective, it must work with those settings and not alter them. Packet marking algorithms and associated routers must be fast enough to allow real-time packet marking.

The packet marking techniques can generally be grouped into two major categories – one based on tracing a single packet, and the other based on collecting a large number of packets for tracing back to the attackers.

Hash-based traceback [74], the representative of the former technique, digests and logs some particular information of every packet on the routers. The victim can query the routers whether a certain packet has passed through them. There are two obvious problems: it requires a large-scale database (bloom filter) on each router to store and manage the packets information. Furthermore, the queries must be done before the relevant records of database are updated. Yaar et al. [88] proposed a *Path Identification* (Pi) mechanism which allows the victim to identify the paths in a per packet fashion. Each packet travelling along the same path carries the same identifier, enabling a victim to identify packets traversing the same paths through the Internet on

a per packet basis.

The marking scheme proposed in this thesis belongs to the latter one. In the literature, different approaches, based on using a large number of packets, have been proposed for IP traceback.

Savage et al. [72] proposed algorithms for packet marking, ranging from simply appending the current router address to employing probabilistic traffic-sampling. He used a probability of 1/25 to avoid excessive overhead on the routers' packet marking, and proposed three kinds of probabilistic marking schemes – *node sampling*, *edge sampling* and *compressed edge fragment sampling*. Node sampling records only one IP address in the packet according to the marking probability; so it cannot trace multiple attacks. Edge sampling records the IP addresses of two adjacent routers and the distance from the further router to the victim; and it needs more than 70 bits of storage space for the markings, which is not available in a normal IP header.

Compressed edge fragment sampling, which is also known as *Fragment Marking Scheme* (FMS), divides the edges into fragments to overcome the storage problem: each router computes a uniform hash of its IP address once, at startup. This hash is interleaved with the original IP address. The resulting quantity known as edge-id is then broken into k fragments each labelled with its offset, which the router selects randomly when marking a packet. The next downstream router uses the offset to select the appropriate fragment to XOR - thereby encoding part of an edge. If enough packets are sent by the attacker, the victim will eventually receive all fragments.

However, there are two major drawbacks especially when the number of attack paths increases: one is the high computation overhead for examining various combinations of the edge fragments; the other one is the large number of false positives which could be generated as a result of the collisions of the

encoded values.

Some other researchers have also evaluated the effectiveness of Probabilistic Packet Marking. Park and Lee [55] and Alder [9] study the tradeoff between the number of IP header used and the number of packets required for reconstruction. A recent research [87] indicates that Groups Of Strongly SImilar Birthdays (GOSSIB) can be used to obtain effects similar to a successful birthday attack on this scheme.

Song and Perrig [75] improved the probabilistic marking scheme and proposed the advanced and authenticated marking scheme that copes with multiple attackers. By using an upstream routers map they made a significant improvement on the performance as measured by the number of packets needed to reconstruct each path, the reconstruction time, the number of false positives, and the ability to deal with distributed DoS attacks. Furthermore, their marking scheme provides authentication for the marking information through the use of hash chains, which was not implemented in other previous work. They also use a 5-bit distance field, but they do not fragment marking information. Instead, they include a (b-5)-bit *XOR* of hashed message authentication codes (HMACs) from each router and its downstream router. Nevertheless, there exist some false positives when the number of attack paths is quite large. In addition, the design of effective hash functions, which are used in their marking scheme, is not an easy task.

Dean et al. [23] proposed an algebraic approach to encode the upstream router path for IP traceback. Nodes mark packets with evaluations of the sample points of a polynomial over a finite field. The coefficients of the polynomial are the IP addresses of the routers in the attack path. Unfortunately, their schemes do not easily lend themselves to authentication, without requiring knowledge of universal tree. Moreover, the reconstruction algorithms are

Table 2.5: Advantages and disadvantages of packet marking

Advantages	Disadvantages
Can be deployed incrementally and appears to be low cost	Requires modification to the protocol
Works with existing routers and network infrastructure	Produces false-positive paths
Effective against distributed denial-of-service attacks	Victim must receive minimum number of packets
ISP cooperation not required	Cannot handle fragmentation
Allows post-attack analysis	Does not work with IPv6 and is not compatible with IPSec

complex and slow for large-scale Distributed Denial of Service attacks, and it introduces certain amount of "noise". Chen et al. [21] has enhanced the algorithm to perform IP traceback for reflector attacks. Table 2.5 depicts the advantages and disadvantages of packet marking.

2.2.5 Comparison of various IP Traceback Schemes

In this section, we evaluate the above mentioned traceback schemes against certain metrics, which can serve as the criterion of being a good and effective traceback scheme.

- **ISP Involvement.** ISPs and enterprise networks do not have incentives to monitor for attack packets. The lack of incentives comes from the fact that monitoring for such packets has no immediate benefit to the ISP itself or its subscribers. Furthermore, most ISPs are reluctant to disclose their internal topology. Consequently, IP traceback solutions should not assume complete cooperation of ISPs. It is desirable for the scheme to have low ISP involvement, which implies that it can be easily built or inserted with little infrastructure or operational change.

- **Scalability.** It is related to the additional configuration on other devices

needed to add a single device to the scheme. It also measures the ability of the scheme to perform as network size increases. An ideal scheme should be scalable, and configuration of the devices involved should be totally independent of each other.

- **Number of packets required for traceback.** A good traceback scheme should be able to determine the source of an attack based on as few packets as possible once the attack is detected. This will enable the scheme to successfully trace back more attacks.

- **Processing overhead.** It can be incurred for every packet and during traceback. Processing overhead on the ISP routers is especially undesirable since it may result in the need to upgrade or buy more equipment. Therefore, a scheme with less processing overhead incurred on the network will most likely be accepted by an ISP.

- **Bandwidth overhead.** It is considered as the additional traffic the the network has to carry for traceback. Large bandwidth overhead is undesirable since it may exhaust the capacity of links and routers.

- **Memory requirements.** An ideal scheme would have a limited amount of additional memory required at the dedicated server, and no additional memory requirements on network equipment.

- **Ease of evasion.** The scheme is said to be easy to evade if the attacker, who is aware of the scheme, can easily orchestrate an attack that will be untraceable. Clearly, this quality is not desirable in a traceback scheme, and the ease of evasion should be as low as possible.

Table 2.6: A comparison of four approaches for IP traceback

Metrics	Link testing	Logging	ICMP-based traceback	Packet marking
ISP involvement	None	High	Low	Low
Scalability	N/A	Poor	High	High
Number of packets required for traceback	Huge	1	Thousands	Thousands
Prior knowledge of topology and routing required	Yes	No	Yes	Yes
Processing overhead	Fair	High	Low	Fair
Bandwidth overhead	Huge	High	Low	None
Memory requirements	Low	Low	Low	None
Ease of evasion	N/A	Low	High	Low

2.3 DDoS Countermeasure: Packet Filtering

Packet filtering is an alternative solution against DDoS attacks. By classifying the malicious packets out the normal packets, most of the attack traffic could be dropped (rate-limiting is another possible action). The effectiveness of packet filtering refers to the throughput of normal service that can be maintained by the victim during a DDoS attack.

2.3.1 Ingress Filtering

One effective action that the Internet Service Providers (ISPs) can take against DoS/DDoS is to eliminate routing of spoofed packets by discarding any packet with IP address outside the range of a customer's network. Early methods focused on detecting the incoming points of attack traffic within a single network administration, such as deploying network *ingress filtering* to eliminate the ability to forge source addresses [35]. Any packet with IP address outside the range of a customer's network would be discarded. The success of this approach depends to some extent on if all ingress routers have sufficient processing power to inspect the source address of every packet, and sufficient knowledge to distinguish between legitimate address and illegitimate addresses,

Table 2.7: Advantages and disadvantages of ingress filtering

Advantages	**Disadvantages**
Compatible with existing routers and network infrastructure	High deployment difficulty
Supports incremental implementation	Fail to filter packets with valid addresses
Low computation complexity	

which may not be possible in practice. In addition, this method cannot handle flooding attacks originated from valid IP addresses, and may negatively affect mobile IP services [65].

A more recent and functional approach to ingress filtering is proposed by Li et al. in [57]. Their protocol, called SAVE, has routers construct tables of valid source addresses per incoming interface. A packet whose sources address is out of the proper range is easily identified and dropped. Table 2.7 depicts the advantages and disadvantages of ingress filtering.

2.3.2 Egress Filtering

SANS institute urged network administrators to adopt egress filtering which prevents one's network from being the source of forged communications used in DoS attacks [6]. An egress filter is designed for implementation in the routers at the edge of a network. These filters analyze all packets forwarded, looking for forged IP addresses. Since any particular network is assigned a specific subset of IP addresses, any packet containing an invalid IP address is assumed to be spoofed, and the filter drops such packets. This ensures that only IP packets with valid source IP addresses leave the network and thus protects the outside from spoofed packets.

This approach can effectively deter attackers from attacking others with one's own network. However, it has three severe shortcomings. Firstly, there

Table 2.8: Advantages and disadvantages of egress filtering

Advantages	Disadvantages
Effective to filter packets with spoofed addresses	Only restrict traffic from local network
Do not incur extra overhead	Not applicable to forward legitimate traffic that is not part of its address space
Compatible with current infrastructure	Require large portion of deployment

is little incentive for an ISP to provide egress filtering since it does not protect from the attack, it only keeps an attacker from using your network for a DDoS attack. If egress filtering is not employed by a significant number of networks, it will not be a viable solution to DDoS attacks. Secondly, egress filtering is difficult for many service providers who frequently need to forward legitimate traffic that is not part of its address space. Thirdly, egress filtering will not detect internally spoofed IP addresses. Table 2.8 depicts the advantages and disadvantages of ingress filtering.

2.3.3 Route-based Packet Filtering

The route-based packet filtering (RPF) technique shares some similarity with Ingress filtering method by installing packet filters at certain amount of autonomous systems distributed in the Internet. At an extreme, with all autonomous systems and their routers implementing RPF method is no much different from the effect of perfect ingress filtering, with no spoofed IP flows being able to escape.

The RPF approach [66] uses routing information to determine if a packet arriving at a border router at an autonomous system, is valid with respect to its claimed source/destination addresses, since a source and destination address pair should follow the reachability constraints imposed by routing and network

Table 2.9: Advantages and disadvantages of RPF

Advantages	Disadvantages
Incremental deployment is possible	Require BGP messages to carry source addresses
Effective under current Internet AS connectivity structure	High global deployment

topology.

The main merit of this method as shown by their performance evaluation is that, with a partial coverage on at 18% of such filters equipped autonomous systems, significant amount of spoofed packets can be dropped and the origins of those spoofed IP flows succeed reaching their targets can be localized to within a small, constant number of sites (less than 5 for Internet AS topologies). However, the cooperation of thousands of autonomous systems is still an unfavorable requirement, with every ingress/egress router of which has to install the filter. In addition, the effectiveness depends intimately on the connectivity structure of the underlying AS graph. Another problem with this method is that computing appropriate filtering tables alongside existing inter-domain routing protocols (e.g., BGP) is a nontrivial problem due to the destination-based structure of Internet routing protocols. Table 2.9 presents the advantages and disadvantages of RPF.

2.3.4 IP Traceback-based Packet Filtering

Minho Sung and Jun Xu proposed a novel technique [78] for defending against Internet DDoS Attacks based on knowledge of reconstructed attack paths from underlying IP traceback schemes. They use some packet marking IP traceback scheme to reconstruct the attack graph, and call the network edges on the attack graph "infected" edges. Consequently, packets marked with the "infected" edge markings will be preferentially filtered out, while packets from a

legitimate client, on the other hand, have higher probability to be filtered out, since typically most of the edges on the legitimate path to the victim are not infected. They modified the underlying traceback scheme a bit for effective filtering effect. The marking performed by each router is divided as two types: signaling is for traceback purpose as the underlying IP traceback scheme requires (around 5% of total marks); the other is for marking packets with edge information so as to serve as the filtering signatures (as proposed at 95%). A defense-line with implementation of filters around the victim will impose rate-limit on incoming traffic flows based on the inscribed marks in each packet. However, to make sure significant amount of traffic can be filtered based on the second type of marks, only 5% of the marks are signaling marks. This implies that the reconstruction of the whole attack graph will be 20 times slower than in the underlying IP traceback scheme, which is a considerable delay when considering the relatively short lifetime of most DDoS attacks. In addition, as the attackers become more distributed, the normal paths between legitimate clients and attack paths have high probability to overlap, which means an "infected" edge would also be traversed by a non-attack packet, especially when they near the victim.

2.3.5 Router-based Pushback

In *Pushback* mechanism [41], a congested router nearest to the victim uses statistics and pattern analysis to determine from which most adjacent upstream routers the unexpected traffic volume are coming, and then send signals to notify the traffic contributors to rate-limit the suspect traffic. The approach is then repeated at the upstream routers in a chain to identify and rate-limit the traffic contributors. This scheme therefore requires immediate action during the attack, and requires considerable coordination between net-

Table 2.10: Advantages and disadvantages of router-based pushback

Advantages	**Disadvantages**
Incremental deployment is possible	High storage space requirement for pushback daemon to analyze dropped packets
No need for a map of upstream routers	Lack of trust relationship among ISPs to respect pushback requests from others

work operators.

The main drawback with the hop-by-hop tracing method is that, in large-scale DDoS attacks, it has limited capabilities to separate the legitimate packets from attack packets in a pattern-based way. As a result, rate-limiting imposed on the detected aggregates usually drop attack packets as well as normal packets because both match the aggregates signature [58]. Table 2.10 demonstrates the pros and cons of the router-based pushback approach.

Table 2.11: Classification of DDoS countermeasures according to several viewpoints

Traceback	Probabilistic Packet Marking	[72, 75, 74, 23, 21, 66, 69, 88, 55, 87, 9, 37, 83, 34, 15, 52, 31, 73, 50]
	ICMP Traceback	[17, 59]
	Others	[12, 18, 77, 53, 10, 16]
Filtering	Packet Filtering	[35, 78, 6, 80, 79, 81, 70, 57]
	Rate Limiting	[91, 41, 58, 62, 61, 90]

□ **End of chapter.**

Chapter 3

Domain-based IP Traceback Scheme

Our proposed defense scheme exploits the attack path reconstructed by our proposed IP traceback scheme to perform attack packet filtering at strategic positions. So tracing the attack packets back to their sources is the first and essential step in making attackers accountable.

To locate the attack sources, we propose a *domain-based marking scheme* (DBMS) for IP traceback, which supports two types of packet marking. A participating router would perform deterministic intra-domain source router marking when a packet enters the network from an end-host, and probabilistic inter-domain edge marking when it receives a packet from another domain. After collecting sufficient packets, the victim would reconstruct the attack graph incorporating attack paths and the source routers identified, with each node on the paths viewed as a domain.

3.1 Overview of our IP Traceback Scheme

Our proposed IP traceback scheme can be split into two phases: packet marking executed by the routers, and attack path reconstruction operated by the victim, as depicted in figures 3.1 and 3.2. We use V, R, and A to denote the victim, router, and attack source respectively.

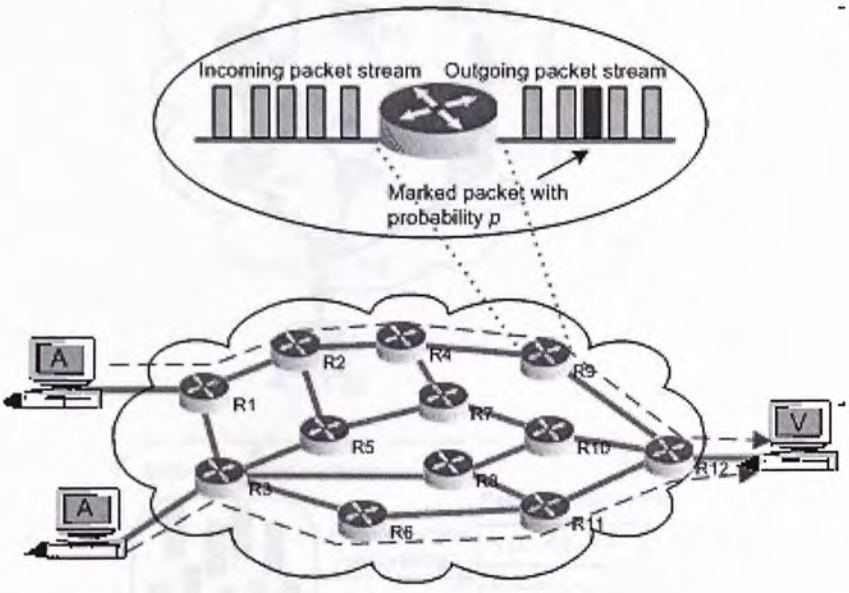

Figure 3.1: Packet Marking

During the packet marking phase, the routers marks the incoming packets probabilistically before forwarding it to the next router. The marked packets contain partial or complete information of the paths which they traversed. As shown in figure 3.1, there are two paths traversed by the packets: (R_1 - R_2 - R_4 - R_9 - R_{12}) and (R_3 - R_6 - R_{11} - R_{12}). In the case of DDoS attack, since the attacker normally floods the network with a huge volume of packets, most of these packets carry the markings associated to the two attack paths.

Upon receiving these marked packets, the victim can make use of the pieces of marking information to trace the path traversed by the packet, *hop by hop*

through a series of upstream router, and finally reach the source of the attack packets, which could be a source router in a particular domain.

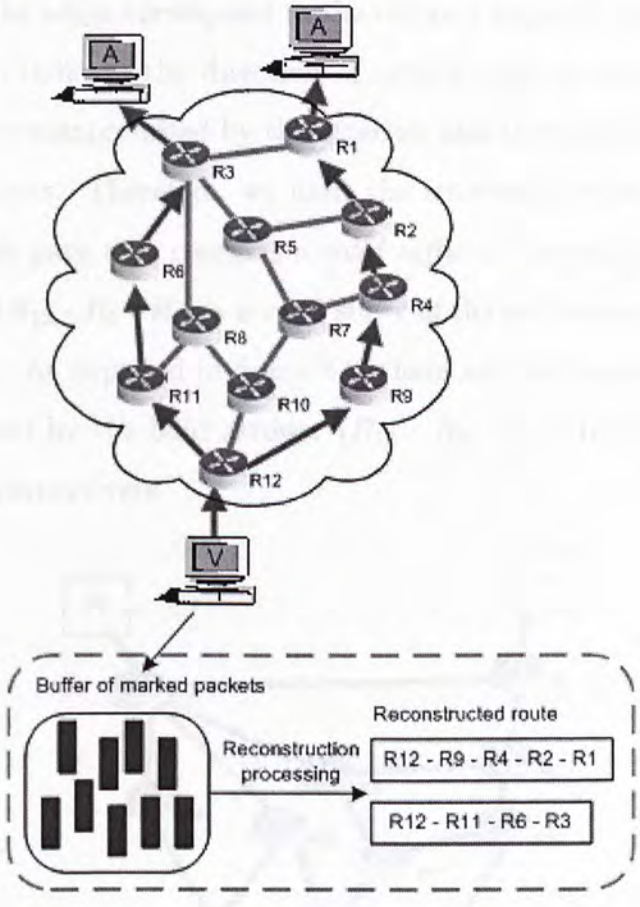

Figure 3.2: Path Reconstruction

The final product of the path reconstruction phase is an attack graph G reconstructed based on the markings from the packets and an *upstream router map*, which is a map describing the topology of the upstream routers from a single host and capturing their IP addresses. Figure 3.3 depicts an upstream router map from the view of the victim. Here *upstream* is used to describe routers viewed from the victim. For example, R_4 is the upstream router of R_7 and R_9.

With reference to figure 3.1, there are two attack paths – (R_1 - R_2 - R_4 - R_9

- R_{12}), and (R_3 - R_6 - R_{11} - R_{12}) respectively. In figure 3.2, as indicated by the bold arrows, we reconstruct the attack paths *edge by edge* starting from the victim; where the edges correspond to the relevant edges in the Internet map; the arrows also indicate the direction of attack path reconstruction. Some routers might be compromised by the attacker and they would forge information in the packets. Therefore, we limit the traceback problem to finding a candidate attack path that contains a *valid suffix* of the real attack path. For example, path (R_{12} - R_9 - R_4) is a valid suffix of the real attack path (R_{12} - R_9 - R_4 - R_2 - R_1). As depicted in figure 3.2, there are two reconstructed attack paths represented by the bold arrows: (R_{12} - R_9 - R_4 - R_2 - R_1) and (R_{12} - R_{11} - R_6 - R_3) respectively.

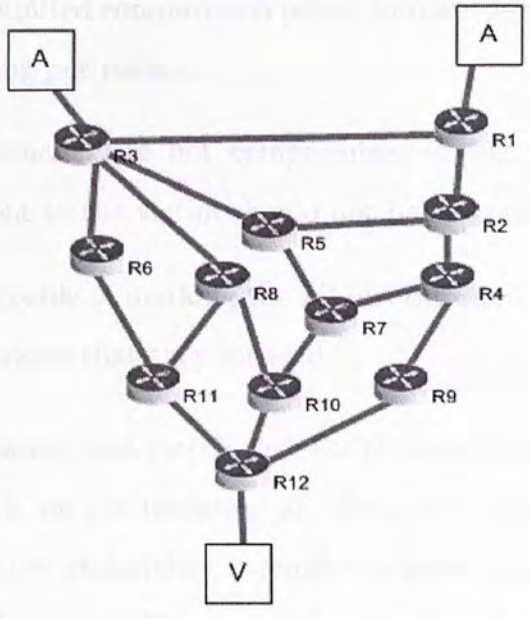

Figure 3.3: Upstream router map as viewed by the victim

3.2 Assumptions

To make our marking scheme more practical and effective, the following assumptions are made in the design of our algorithms:

1. Attackers can send a large number of packets.
2. Multiple attack paths may exist.
3. Attackers might be aware that they are being traced.
4. Packets may be reordered or lost.
5. Routes between the attacker and the victim are fairly stable.
6. Routers have limited computation power so that they cannot perform too much processing per packet.
7. Routers or domains are not compromised in big proportion and the routers adjacent to the victim should not be compromised.
8. Routers are capable of marking the *IP Identification* and *Type of Service* fields of all packets that they forward.

The first two assumptions match with the characteristics of flooding-based DDoS attacks which we are targeting at. Since our marking scheme marks packets with a very low probability, it requires a good number of marked packets for path reconstruction. This is usually not a problem since the attacker normally floods the network with a huge volume of packets.

The third one is a conservative evaluation of the abilities of the attackers. The sophisticated attackers should be aware that they are being traced and may send fake packets or packets with spoofed IP addresses to make the victim

confused. So the traceback method proposed must consider such a potential ability of the attackers.

With the knowledge of current network infrastructure, the forth, fifth and sixth assumptions are quite easy to understand. For the seventh assumption, if some routers are compromised, we might only trace the source back to the compromised router since it could tamper the information marked by its upstream routers. So we use valid suffix instead of the entire attack path to assess the robustness of the traceback technique.

Nonetheless, the *source router identification* procedure in our proposed marking scheme helps us to confirm indirectly if the last domain of a reconstructed attack path is a source domain or simply a compromised one. Note that the routers nearest to the victim should not be compromised; otherwise they could tamper any information marked by the upstream routers and the victim might reconstruct incorrect paths.

Regarding the last assumption, since the 16-bit *IP Identification field* and the 8-bit *Type of Service field* are little used in current network design [76, 11], overloading these two fields is considered feasible. Using these fields for packet marking is quite common in the literature, first proposed by Savage et al. [72] and later by others [23, 75, 59, 6, 88] in their proposed marking schemes.

3.3 Proposed Packet Marking Scheme

In this section, we introduce our domain-based marking scheme (DBMS) in detail. To locate the attack source(s), we propose a domain-based packet marking scheme for IP traceback, which supports two types of marking; A participating border router would perform deterministic *intra-domain source router marking* when a packet enters the network from an end-host, and prob-

abilistic *inter-domain edge marking* when it receives a packet from another domain. After collecting sufficient packets, the victim would reconstruct the attack graph incorporating attack paths and the source routers identified, with each node on the paths viewed as a domain.

3.3.1 IP Markings with Edge Sampling

Our proposed marking scheme exploits the idea of *probabilistic edge sampling* [72], to mark the packets with a low probability in order to reduce the marking overhead of the routers. *Probabilistic edge sampling* is a promising technique proposed by Savage et al. [72]. Its main idea is to let routers probabilistically mark packets with partial path information during packet forwarding. The path information includes the IP addresses of two adjacent routers in an Internet map, and the distance between the victim and the router last marked the packet.

Below are the marking and path reconstruction algorithms of *probabilistic edge sampling* proposed by Savage et al. [72].

Algorithm 1 Marking procedure at router R

```
for each packet w do
  let x be a random number from [0...1)
  if x < p then
    write R into w.start and 0 into w.distance
  else
    if w.distance = 0 then
      write R into w.end
    end if
    increment w.distance
  end if
end for
```

As shown in the marking algorithm, three finite fields are pre-allocated in the IP headers. These fields are {*start*, *end*, *distance*}. Each participating

Algorithm 2 Path reconstruction procedure at victim v

```
let G be a tree with root v
let edges in G be tuples (start, end, distance)
for each packet w from attacker do
  if w.distance = 0 then
    insert edge (w.start, v, 0) into G
  else
    insert edge (w.start, w.end, w.distance) into G
  end if
  remove any edge (x, y, d) with d ≠ distance from x to v in G
  extract path (R_i...R_j) by enumerating acyclic paths in G
end for
```

router marks the packets targeted to the victim site with a probability p. When the router decides to mark a packet, it writes its own IP address into the *start* field and zero into the *distance* field. Otherwise, if the *distance* field is already zero, which means this packet has been marked by the previous router, it records its IP address in the *end* field and increments the *distance* value by one. If the router does not mark the packet, it simply increments the *distance* value by one. This value indicates the number of *hops* between the victim and the router which last marked the packet. The mandatory increment of the *distance* value is crucial to minimize the spoofing of the markings by an attacker, so that a single attacker would be unable to forge an edge between itself and the victim [72]. Figure 3.4 illustrates the set of marked and unmarked packets as collected by the victim.

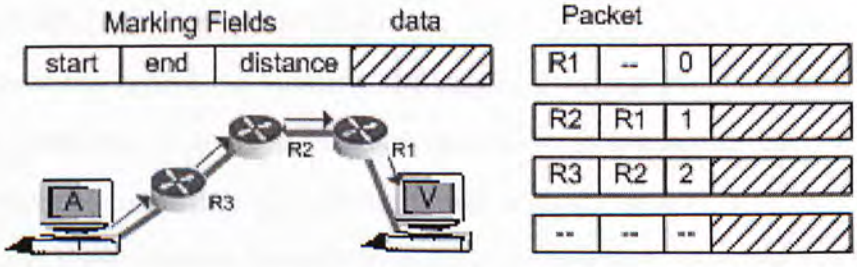

Figure 3.4: Set of marked and unmarked packets collected by the victim

3.3.2 Domain-based Design Motivation

Existing marking schemes normally reconstruct the full path or suffix of the path to the attacker in order to defend the victim. In practice, however, there could be much difficulty in doing so. The reasons behind are:

1. There is a trend that ISPs only use public addresses for interfaces to customers and other networks, and use private addressing within their own networks; in this situation, full address traceback may not be practical since we cannot identify the IP addresses of private address routers;

2. Even if public addressing is used within an ISP's network, ISPs are generally reluctant to disclose their topologies. Therefore, to avoid this inherent difficulty, our marking scheme involves only the border routers of a domain, and the markings can be based only on the prefix of the IP addresses of such routers.

As we are primarily interested in identifying the location of the attacker, we need to trace only the domains in the Internet map which the attack packets traversed, without tracing the individual routers within a domain. Therefore, instead of requiring all routers within each domain to participate in packet marking, we apply only domain-level packet marking which involves mainly the border routers of each domain.

Nonetheless, information about the routers closer to the attack sources would be most useful for locating the origin of attackers. Therefore, we also perform marking on *source routers*, which serve as a gateway between a local area network containing the attack nodes and the rest of the Internet [61]. Physically, these routers connect to both end users and other routers forwarding packets from another domain through different interfaces.

The design proposed in this thesis assumes that a source router is able to identify the interfaces through which it receives the incoming and outgoing traffic, so it can distinguish whether the packets are forwarded from an individual host (end-user) or from another domain. The source routers may not be adjacent to the attackers, but they must be within the domains containing the attack sources. Both the domain-level marking and source router marking are done probabilistically in order to reduce the packet's marking overhead of the routers concerned. Figure 3.5 depicts the packet marking participation of routers at different positions.

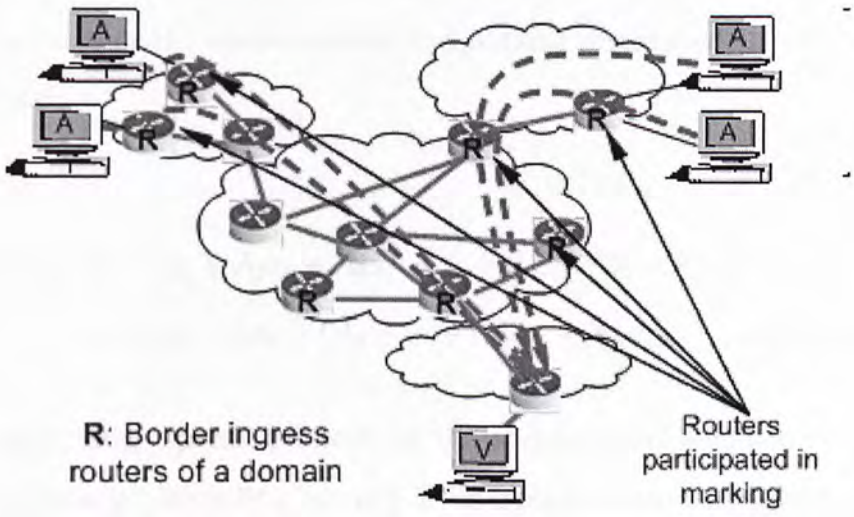

Figure 3.5: Participation of routers at different positions

3.3.3 Mathematical Principle

The main idea of our packet marking scheme relies on computing a *fullpath* value based on the matrix equation, shown below, as introduced in [23].

Figure 3.6 shows a matrix equation (or system of equations) with *Vandermonde* matrix coefficients. In *linear algebra*, there is a theorem stating that the above matrix equation, with A_i's unknown, has a unique solution if and only if

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{n-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_n & x_n^2 & \dots & x_n^{n-1} \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \\ \dots \\ A_n \end{bmatrix} = \begin{bmatrix} Fullpath_1 \\ Fullpath_2 \\ \dots \\ Fullpath_n \end{bmatrix}$$

Figure 3.6: Matrix equation with *Vandermonde* matrix coefficients

the x_i's are distinct [64]. By applying field theory to the above theorem, we can obtain a similar theorem over $GF(m)$, where GF denotes *Galios Field* and m is a prime number if the x_i's and $fullpath_i$'s are elements in $GF(m)$ [42]. Each $fullpath$ value in the above system of equations corresponds to the following polynomial:

$$\begin{aligned} fullpath_i &= (A_1 + A_2 x_i + A_3 x_i^2 + ... + A_{n-1} x_i^{n-2} + A_n x_i^{n-1}) \bmod m \\ &= (A_1 + (A_2 + (A_3 + ... + (A_{n-1} + A_n x_i) x_i) ...) x_i) x_i) \bmod m \end{aligned}$$

We apply such special property of the above matrix equation to a packet marking scenario. We split a router's IP address into four fragments; each with 8 bits (1 byte) long. Consider a series of A_i's in the matrix as the four fragments of a router's IP address. Upon receiving a packet, the router performs the marking probabilistically by assigning a distinct value x_i to the packet, and using it together with a particular fragment of the IP address, represented by A_i, to compute the corresponding $fullpath$ value. The $fullpath$ value would then be written into the packet header.

In general, an attack packet will pass through a number of routers before reaching the victim. The first router that decides to make a marking assigns an x to the packet, and computes a $fullpath$ value according to the above

polynomial. Then the next router computes its *fullpath* value by multiplying the *fullpath* value from the packet by x, and adding its IP address fragments. The following routers mark the packet similar to what the second one does. When the packet arrives at the victim, it records a *fullpath* value related to a certain number of routers. In fact, it is the value of the above polynomial with IP address fragments of two adjacent routers, represented by A_i's.

Note that there is no way for a router to know whether it is the "first" participating router on a particular path, so the packet would be re-marked under a low probability value. The router will generate a random number r, and if this number is smaller than our predefined marking probability p, the router will select an x for the marking of this packet and do the marking in the capacity as the first router; otherwise, it assumes that it is not the first router and simply follows the procedure presented above.

3.3.4 Marking Mechanism

Similar to all marking schemes, our marking scheme (DBMS) involves recording the markings in the packets' IP headers by the routers and reconstructing the attack paths by the victim. Our proposed marking scheme handles two types of marking: *domain edge* marking and *source router* marking. Domain edge marking is performed probabilistically by a border router for a domain whenever it receives a packet from another domain; whereas source router marking is carried out deterministically by a router when it detects that a packet enters the network from an end host. The algorithm of DBMS marking is presented below.

To reduce the size of the marking field, we only record the *prefix* (the first two bytes) of the IP address of the router which performs the marking; the prefix corresponds to the domain's identity. Only the border routers of

Algorithm 3 Domain based marking algorithm at border router R

```
for each packet pkt do
  if pkt is passed from an upstream domain then
    domainEdgeMarking(pkt)
  else if pkt enters the network from an end-host then
    sourceRouterMarking(pkt)
  end if
end for
```

every domain are involved in packet marking. As it is essential to get more information about the source of an attack, our marking algorithm also marks the suffix (the last two bytes) of the source router IP address; besides the source router, the suffix of the IP address of any other routers would be ignored.

The markings recorded in each marked packet include four integer values: *flag*, *x*, *distance* and *fullpath*:, as depicted in figure 3.7.

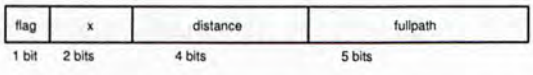

Figure 3.7: Marking fields inside a packet

- *flag* indicates the type of markings; it is set to 1 if a *source router* marking is written, otherwise it is set to 0 in the case of *domain edge* marking;
- x is a value assigned by the router which first took part in creating the markings carried by a packet to the victim;
- *distance* is the number of hops between the victim and the router which last took part in creating the markings in a packet;
- *fullpath* is a value computed based on the matrix equation presented in figure 3.6, for a domain edge (considered as a sub-path) comprising (1) the prefix of the IP address of a start router in one domain and the prefix of the IP address of an end router in an adjacent domain, or (2) only

the prefix of the router's IP address if the router is in the domain closest to the victim in the Internet map. In case of source router marking, the *fullpath* would be for a single router based on the full IP address.

Domain edge marking

The idea behind the proposed packet marking scheme is similar to edge sampling. The prefix of an IP address consists of 16 bits; we split it into two fragments in our marking scheme. Consider a packet being marked respectively by any two consecutive routers R_1 and R_2; that is, R_1 and R_2 would become the *start* router and *end* router of the edge respectively in the marking. If router R_1's IP address is 137.189.89.101, then the prefix of R_1's IP address would be 137.189, and the two fragments, $A_{1,1}$ and $A_{1,2}$, of the prefix would be 137 and 189 respectively. Router R_1 may compute the *fullpath* as follows:

$$fullpath = (A_{1,1} + A_{1,2}x) \bmod m \tag{3.1}$$

Similarly, we split the prefix of the IP address of R_2 into two fragments $A_{2,1}$ and $A_{2,2}$. Then R_2 would compute its new value of *fullpath* for the edge as follows:

$$fullpath = (A_{1,1} + A_{1,2}x + A_{2,1}x^2 + A_{2,2}x^3) \bmod m \tag{3.2}$$

where $A_{i,j}$'s (i,j = 1, 2) form the prefixes of the two adjacent routers' IP addresses, and m is the smallest prime number larger than 15 (2^4-1), i.e. 17. If the router is adjacent to the victim, the last two terms of *fullpath* should be omitted. The objective of letting the above formula modulo by m is to reduce

the value of $fullpath$ so that it would occupy fewer bits in the IP header. Algorithm 4 depicts the procedure of *domain edge marking*.

Algorithm 4 Domain edge marking procedure at border router R

```
generate a random number r in the range [0...1)
if r ≤ p then
  // p is the domain edge marking probability
  randomly select an integer x in the range [0...3]
  pkt.x = x
  pkt.fullpath = (A_{1,1} + A_{1,2}x) mod m
  pkt.flag = 0
  pkt.distance = 0
else
  if pkt.distance = 0 then
    // upstream border router has marked the packet
    x = pkt.x
    pkt.fullpath = (pkt.fullpath + A_{1,1}x^2 + A_{1,2}x^3) mod m
  end if
  increment pkt.distance by one
end if
```

As shown in the marking procedure, when border router R receives a packet pkt from its upstream domain, it first generates a random number r and performs marking depending on r and the *distance* recorded in the packet.

As an example, let the IP address of router R be 137.189.89.101 and the values of (x, *distance*, $fullpath$) from the packet being marked are (2, *distance*, 15). Router R would first generate a random number r. Then the marking algorithm would produce one of the 3 possible outcomes:

- **Case 1** ($r \leq p$): suppose the randomly selected x is 3. Then, $fullpath = (137 + 189 * 3) \bmod 17 = 7$, distance = 0.
- **Case 2** ($r > p$ and $distance = 0$): Assume *distance* from packet is 0. $fullpath = (15 + 137 * 2^2 + 189 * 2^3) \bmod 17 = 1$, $distance = 1$.
- **Case 3** ($r > p$ and $distance > 0$): Increment *distance* by 1.

Source router marking

In addition to domain-based marking, we perform source router marking to record the router's whole IP address when the packet is traversed from a host. The source router markings can be used during attack paths reconstruction to locate the attack source more accurately. The markings can also be used to confirm if the end of a reconstructed path corresponds to the source of an attack.

The *fullpath* computation for source router marking is similar to that for domain edge marking as mentioned above. However, source router marking involves node sampling, where the whole IP address of the source router is divided into four fragments. For example, if the IP address of the source router R is 137.189.90.101, we split it into 4 fragments $A_{1,1}$, $A_{1,2}$, $A_{1,3}$, and $A_{1,4}$, with the values 137, 189, 90, and 101 respectively; where $A_{1,1}$ and $A_{1,2}$ form the prefix of R's IP address; $A_{1,3}$ and $A_{1,4}$ form the suffix. Algorithm 5 depicts the procedure of *source router marking*.

Algorithm 5 Source router marking procedure at router R

generate a random number r in the range [0...1)
if $r \leq q$ **then**
  // q is the *source router* marking probability
  randomly select an integer x in the range [0...3]
  $pkt.x = x$
  $pkt.fullpath = (A_{1,1} + A_{1,2}x + A_{1,3}x^2 + A_{1,4}x^3) \bmod m$
  // $A_{1,1}$, $A_{1,2}$, $A_{1,3}$ and $A_{1,4}$ are the 4 fragments of R's IP address
  $pkt.flag = 1$
end if

When 4 (or 2) packets with distinct x's arrive at the victim, the victim can solve the relevant matrix equation in section 3.3.3 to obtain the IP addresses of two adjacent routers (or the nearest router to the victim) in the attack path. Therefore, we use a set of 4 distinct x's (0-3) to do the marking. Note that we

Table 3.1: Storage space for the marking fields in the IP header

Marking field	Bit size
flag	ln(2) = 1 bit
x	ln(4) = 2 bits
distance	ln(15) = 4 bits
fullpath	ln(17) = 5 bits
Total	12 bits

use two different marking probabilities in the two procedures (see algorithms 4 and 5): p is used for domain edge marking and q is for source router marking. Since source router marking is more useful in identifying the actual attack sources by our IP traceback scheme, the value of q is much higher than that of p. We will discuss the optimal values of the two marking probabilities in section 5.2.2.

3.3.5 Storage Space of the Marking Fields

This sub-section introduces the marking fields and the corresponding bits required for storage. In domain-based marking, the markings written in the IP header of each marked packet includes four fields: *flag*, x, *distance* and *fullpath*. The *flag* field occupies 1 bit. The values of x range between 0 and 3, so we need 2 bits for x. In general, a packet would reach its destination through no more than 32 hops [82], whereas the inter-domain path length would not exceed 15 hops, so allocating 4 bits for the *distance* field should be sufficient. Besides, we split a router's IP address into 4 fragments; its prefix, made up of the first two IP address fragments, requires 16 bits. The m parameter used in the *fullpath* computation formula can be set to 17, which is the smallest prime number larger than 15 (2^4-1). Thus the value of *fullpath* should not exceed 17, and it requires 5 bits for storage.

Table 3.1 shows the details of the marking fields and the required storage

space in terms of the total number of bits. Totally 12 bits are required to store all the fields in the packet header.

3.3.6 Packet Marking Integrity

One shortcoming of our marking scheme is that the packet markings are not authenticated. A compromised router on the attack path could forge the markings of upstream routers, such as the values of *distance* and *fullpath*. Moreover, it could forge the markings according to their probability distribution, thus preventing the victim from detecting and determining the compromised router by analyzing the markings distribution. To alleviate the seriousness of this problem, we need a mechanism to authenticate the packet markings.

Existing authentication methods often require sharing a secret key between two parties. This approach, however, may not be practical in a network environment since it is impractical to require each router to share a secret key with each potential victim. To avoid the sharing of secret keys problem, we can compute a checksum, based on the marking contents, to be included in each marked packet; then the integrity of a packet's markings can be verified by examining the checksum.

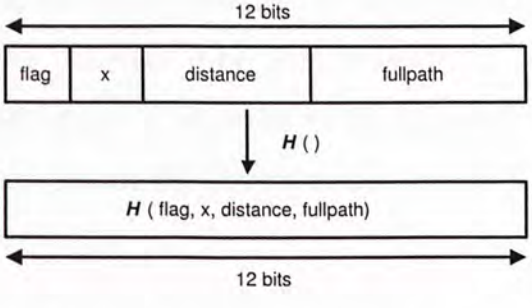

Figure 3.8: Checksum computation based on marking fields

As shown in figure 3.8, we use the entire packet marking fields for message

checksum computation. The packet marking fields include the values $flag$, x, *distance* and $fullpath$. We first concatenate the four values together to form a string. Then we encode the string using a uniform one-way hash function [46] to produce a *message checksum*. This checksum would be appended to the packet marking. In our marking scheme, two routers for two adjacent domains are involved in each edge sampling. To reduce the checksum computation overhead, only the routers which mark the *end* domain of a domain edge are required to compute a checksum as mentioned above.

3.3.7 Path Reconstruction

To identify the attack sources, we first construct an attack domain graph and locate the intermediate domains along each attack path. With the attack graph, we can identify the potential source domains, and then trace back any source routers by exploiting the IP addresses of the routers within any identified potential source domain in the Internet map and checking if there is any packet which carries markings based on any full IP address of those routers.

Note that we only attempt to identify the source routers within a potential source domain; we do not aim to identify the paths of the attack packets within a source domain since there could be much difficulty or overhead in doing so; for example, the ISPs may be reluctant to disclose their network topologies. Moreover, information about the attack source routers would be most useful for defense purpose.

Algorithm 6 depicts the path reconstruction procedure. It first employs the upstream Internet map M of the victim to reconstruct the domain based attack graph G. From the attack graph G, we can identify any potential source domains.

Then the victim attempts to identify the source router(s) within each potential source domain using the source router identification procedure depicted in algorithm 7. To verify that the markings are not forged by attackers, we check the integrity of a packet's markings by examining the checksum before inserting any node into the attack graph G.

□ **End of chapter.**

Algorithm 6 Inter-domain path reconstruction algorithm at victim site v

sort all marked packets by *distance*, then by values of x
let P denote the group of packets with flag 0, P_d denote the set of packets with distance $d (0 \leq d \leq maxd)$
let $P_{d,x}$ denote the subsets of packets with a certain value x in P_d
let M denote the upstream inter-domain Internet map
let G be reconstructed attack graph initialized with node v
for each direct upstream domain D of v in M **do**
  $count = 0$
  for $x = 0$ to 3 **do**
    $path = (A_{1,1} + A_{1,2}x) \bmod m$
    for each packet pkt in $P_{0,x}$ **do**
      if $path = pkt.fullpath$ **then**
        increment *count* by one
        exit the inner-most *for* loop
      end if
    end for
  end for
  if $count = 2$ **then**
    insert D next to v in G
  end if
end for
for $d = 1$ to $maxd$ **do**
  for each domain D inserted into G in the last *for* loop **do**
    for each upstream domain D_i of D in M **do**
      $count = 0$
      for $x = 0$ to 3 **do**
        $path = (A_{1,1} + A_{1,2}x + A_{2,1}x^2 + A_{2,2}x^3) \bmod m$
        for each packet pkt in $P_{d,x}$ **do**
          if $path = pkt.fullpath$ **then**
            increment *count* by one
            exit the inner-most *for* loop
          end if
        end for
      end for
      if $count = 4$ **then**
        insert D_i next to D in G
      end if
    end for
  end for
end for
output the reconstructed attack paths in G

Algorithm 7 Source router identification procedure at domain D

sort all marked packets by values of x
let Q denote the group of packets with flag 1, Q_x denote the subsets of packets with a certain value x in Q
let G be reconstructed attack graph initialized with node v
for the remote end domain D of each path in G **do**
  for each router R within D **do**
    $count = 0$
    for $x = 0$ to 3 **do**
      $path = (A_{1,1} + A_{1,2}x + A_{1,3}x^2 + A_{1,4}x^3) \bmod m$
      for each packet pkt in Q_x **do**
        if $path = pkt.fullpath$ **then**
          increment $count$ by one
          exit the inner-most *for* loop
        end if
      end for
    end for
    if $count = 4$ **then**
      add R as a source router of D in G
    end if
  end for
end for
output the set of source routers contained by each remote end domain D

Chapter 4

Route-based Packet Filtering Scheme

In addition to trace and locate the attack sources, our proposal seeks to mitigate the effect caused by the DDoS attacks by exploiting packet filtering. Facilitated by the domain-based IP traceback scheme presented in previous chapter, our proposed packet filtering scheme supplies the gap of IP traceback's incapability of defending against on-going DDoS attacks with the practice of removing the attack packets from the network.

The attack graph reconstructed from the IP traceback scheme is encoded as signature bitmaps, consisting of marking of detected source routers or domains on attack paths. The signature bitmaps would then be sent to the upstream routers, which would examine the markings embedded in each incoming packet and match them with the filtering signature. If the packet is found to be malicious, the routers would drop it with certain probability. By this preferential filtering, the majority of attack packets would be discarded before they reach the destination.

4.1 Placement of Filters

Filtering attack traffic from multiple sources normally requires the cooperation of routers throughout the whole network. However, this incurs high deployment and processing overhead to the network. With the prior knowledge of attack paths and source routers, it would be easier to distinguish between normal packets and malicious packets, as well as to place filters at some strategic positions. One feasible option is to place filters at the ends of each attack path, so that most of the attack traffic is restrained at the path end, and thus the congestion of the entire network is minimized.

Filtering mechanism can be deployed at the firewall or gateway in front of the victim, since these devices possess functionality to monitor the incoming traffic as well as control the passing rate. Alternatively, filtering can also be carried out at the last hop of the attack paths, preferably at the border routers from the identified *source domain* (where the source router may reside), so that attack traffic can be removed near the sources before it gets to consume the victim's network bandwidth.

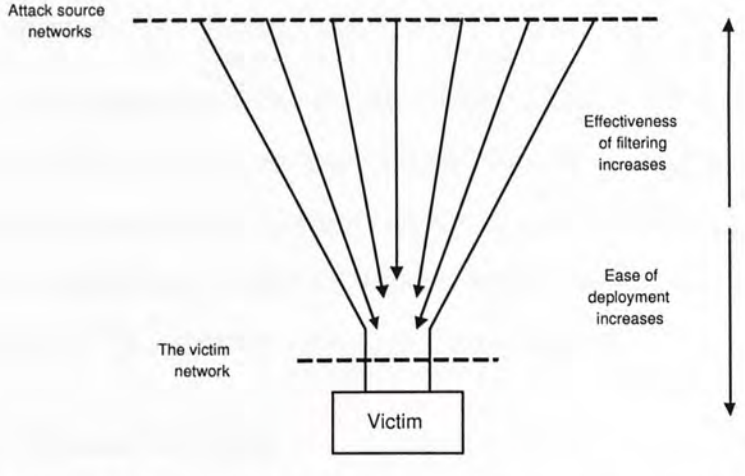

Figure 4.1: Possible locations to perform packet filtering

In general, there are two options concerning where to perform packet filtering: source-end network and victim-end network. As depicted in figure 4.1, a DDoS attack resembles a funnel in which attack packets are generated in a dispersed area. The effectiveness of packet filtering declines rapidly as attack packets are dropped closer to the victim, because more normal packets would also be dropped. However, deployment of filters near the victim's network is relatively easier, due to the shorter distance between the victim and filters.

4.1.1 At Sources' Networks

There are several advantages to place packet filters close to the sources of the attack over placing them further downstream:

1. Restraining attack streams near the source preserves Internet resources that are usually overwhelmed by the attack traffic.
2. Overall congestion can be reduced and more resources are available to legitimate users.
3. Filtering closer to the sources adversely reduces the range of legitimate traffic affected.

However, this approach relies on the victim sending filtering signatures to the corresponding remote routers, which requires a reliable, secure and authenticated communication between the victim and the source router. Since security and authentication might be difficult to achieve across different ISPs, the performance of this approach may not be guaranteed.

4.1.2 At Victim's Network

Placing filters near the victim-side can directly protect the victim from congested traffic. In a DDoS attack where the victim is flooded by a huge volume

of traffic, the filters can block all the incoming packets before they reach the victim. Obviously, it is undesirable to drop all traffic traversed towards the victim, since there may exist some packets coming from legitimate hosts. That is the reason why we need a wise and effective filtering mechanism to preserve the normal traffic.

In addition, victim-end deployment does not require the help of other ISPs, so that the amount of communication across different ISPs is reduced.

However, the drawbacks of a victim-end filtering mechanism are:

1. Routers resided at the victim side are less accurate in differentiating the normal packets and the attack packets, since their traversed paths are highly integrated. Both attack packets and legitimate packets can traverse along an overlapped sub-path. It is thus possible to filter legitimate packets falsely, causing *collateral damage* and decrease in throughput.

2. The overlapping is getting more serious when the filters are placed closer to the victim, since most of the packets traversed towards the victim tend to go through the same paths. As a result, it is more difficult to distinguish between normal packets and attack packets.

3. Another practical issue is that during an on-going attack, if an incoming link is jammed by attack packets, the victim may not be able to do anything but shut down its network and ask the upstream ISP to filter the packets involved.

4.2 Proposed Packet Filtering Scheme

Our proposed scheme improves the throughput of legitimate traffic during a DDoS attack, by preferentially filtering the traffic that is more likely to

come from an attacker than a legitimate host. the filtering is either performed at the sources' network or at the victim side. With the knowledge of the source routers and attack paths obtained from our IP traceback scheme, packet filtering can be performed in a more accurate manner.

4.2.1 Classification of Packets

Basically, if a packet is found to be malicious, with our filtering scheme, the router would choose a filtering probability to drop the packet according to its "marking type". The markings contained in the packets are generally classified into three types:

1. Marking corresponds to an edge inside the attack graph G.
2. Marking corresponds to a "clean" edge (edge not included in the attack graph).
3. Empty marking when no router on the path inscribes a mark on the packet due to the probabilistic nature of the marking algorithm.

Packets of type 1 are most reliable to be attack ones among all types of packets. In contrast, packets of type 2 are most likely to be legitimate ones. Unmarked packets (type 3) can either be transmitted from an attack source, or from a normal host. To limit the congested attack traffic rate, we would also drop a little proportion of packets with no markings.

Our packet filtering scheme aims at removing the packets of type 1, while at the same time preserving the packets of type 2. With our IP traceback scheme, a marked packet (type 1) can contain either one of two types of markings: *domain edge* marking, or *source router* marking. Since packets with *source router* marking are most likely to come from an attack source, these packets would

be dropped at a higher probability than those with *domain edge* marking. Based on these marking types, the following three filtering probability values are used:

- f_d: the probability of dropping packets with *domain edge* markings
- f_r: the probability of dropping packets with *source router* markings
- f_b: the probability of dropping packets with no marking (blank)

4.2.2 Filtering Mechanism

The routers would check the marking concerned against one of two bitmaps: either the one for *domain edge* markings or the one for *source router* markings, which encode the attack graph, as follows. Let U be the bitmap with each entry indexed by the *domain edge* marking and containing one bit with binary values: it would be set to 1 (`INFECTED`) if its marking corresponds to any identified domain on the attack paths, and set to 0 (`CLEAN`) otherwise. In the same manner, bitmap V encodes the set of *source router* markings corresponding to the identified source router(s). To better enhance the accuracy and efficiency of filtering, we deploy two different filtering mechanisms at either source-end or victim-end, as depicted in section 4.1.1 and 4.1.2.

Packet Filtering at Source-end

Within the attack source domains identified by our IP traceback scheme, most out-going packets either carry the *source router* markings or no marking at all. Some packets may contain a *domain edge* markings, but they are most probably coming from other domains. Therefore, our source-end filtering scheme mainly filters the packets with *source router* markings or no marking, as presented in algorithm 8.

Algorithm 8 Packet filtering algorithm at source-end router R

```
let V represent the bitmap for source router markings
let f_r and f_b be filtering probabilities for packets with source router markings or
no marking respectively
for each packet pkt destined to the victim do
  if pkt contains a source router marking i then
    if V[i] = INFECTED then
      drop pkt with probability f_r
    end if
  else
    drop pkt with probability f_b
  end if
end for
```

Packet Filtering at Victim-end

Near the victim-end, the paths contain traffic coming from different hosts within the entire network. Our victim-end filtering scheme mainly filters the packets with *domain edge* markings and *source router* markings, as presented in algorithm 9.

Algorithm 9 Packet filtering algorithm at victim-end router R

```
let U represent the bitmap for domain edge markings
let V represent the bitmap for source router markings
let f_d and f_r be filtering probabilities for packets with domain edge markings or
source router markings respectively
for each packet pkt destined to the victim do
  if pkt contains a domain edge marking i then
    if U[i] = INFECTED then
      drop pkt with probability f_d
    end if
  else if pkt contains a source router marking j then
    if V[j] = INFECTED then
      drop pkt with probability f_r
    end if
  end if
end for
```

Packets whose markings match with our bitmaps of attack graph would be dropped with probabilities f_d and f_r respectively. Since packets with *source*

router markings most likely came from attack sources, the value of f_r would be set higher than that of f_d. The results with varying filtering probability f_d would be demonstrated in the performance evaluation section.

Note that the path of legitimate packets and an attack path may have one part overlapped. Therefore both attack packets and legitimate packets can traverse along such an overlapped sub-path. It is thus possible to filter packets with *domain edge* markings falsely (we define it as *false drops*). One possible solution to reduce the number of false drops is to suppress packet filtering with *domain edge* markings from nearby routers.

We notice that markings from nearby routers would incur false drops more likely based on the following two observations. First, as the probability of receiving a packet marked d hops away by the victim is $p(1-p)^{d-1}$ (with marking probability p), it is more likely that the packet would carry a marking from routers near the victim. Second, the attack graph could be generally viewed as a tree rooted from the victim, and the normal paths and attack paths overlap more near the victim, leading to more false drops. Therefore, we let pass the packets with markings from routers within certain distance to the victim. We introduce a parameter r to denote this radial distance between the victim and the nearby routers for deploying filters, and adjust the value of r ranging from 1 to 3.

□ **End of chapter.**

Chapter 5

Performance Evaluation

We have performed a good number of simulation experiments to examine the feasibility and to evaluate the performance of our proposed DDoS defense scheme. The overall performance of the defense scheme clearly depends on the performance of the two constituent parts - domain-based IP traceback scheme and route-based packet filtering scheme. In this chapter, we present how the simulation experiments are carried out and depict the corresponding experimental results. Various satisfactory experimental results indicate that our proposed scheme has very good performance and can be effectively put into practice.

5.1 Simulation Setup

Extensive experiments have been conducted to examine the feasibility and to evaluate the performance of the proposed DDoS defense scheme.

In preparation for the simulation experiments, we prepared an upstream router map, which is comprised of around 50,000 routers and 10,000 domains. The routers were then assigned some real IP addresses obtained from the

traceroute dataset of Caida's Skitter Internet mapping project [1]. Each domain is connected with 1 to 5 other domains. The topology of the map consists of several routes from a single origin to multiple hosts on the Internet. In our simulations, we assume that this origin is the victim and the attackers and legitimate users are randomly distributed among the destination hosts in the map. We fix the number of legitimate users to 250, and the number of attackers vary as other parameters vary.

The attack paths are randomly chosen from the paths in the map. The paths are not totally isolated; there are around 40% overlapped edges in the map. A number of simulated packets are generated at some destination hosts and transmitted along each of these paths. For performance evaluation purposes, we assume that the packet sending rate at all attack sources is the same and the transmission rate along each path is constant throughout the simulations. Moreover, there will not be any abnormal or high bandwidth traffic congesting the network except the flooding packets generated from the attack sources.

The simulation results are obtained through numerous experiments using Network Simulator 2 [3] and C programs in Unix platform. They will be presented as several plots and evaluated in the later sections. Each data point in the plots corresponds to an average value of around 1,000 experiment runs. Since the simulated metric of our scheme is independent on the actual values of the parameters concerning bandwidth and data rates, we will use relative ratios for characterizing these parameters instead of the actual rates.

Table 5.1: Control parameters and performance metrics used in our traceback scheme

Performance metrics	P_{min}: Minimum number of packets for attack path reconstruction
	False positives: Number of routers added to the reconstructed attack graph by the traceback mechanism, but are not included in the original attack path
	$P_{success}$: Successful reconstruction probability
	T_{recon}: Reconstruction time (seconds)
Control Parameters	l: Path length (hops)
	a: Number of attack sources
	p: Probability for domain edge marking
	q: Probability for source router marking

5.2 Experiments on IP Traceback Scheme

The primary objective of the following experiments is to investigate the robustness of our proposed IP traceback scheme. Each router on the attack paths simulates marking the packets as defined in the marking algorithm. With the pool of marked packets collected, the victim simulates applying the proposed reconstruction algorithm to reconstruct all the attack paths.

5.2.1 Performance Metrics

The experiments examine a number of parameters having an impact on the performance of our IP traceback scheme, which include the minimum number of packets needed for attack path reconstruction, number of false positives, the reconstruction time, number of attack sources, attack path lengths and marking probabilities. Table 5.1 lists the control parameters and performance metrics used in our IP traceback scheme.

An effective IP traceback scheme should use as few packets as possible to reconstruct an attack path; otherwise, the amount of time required by the IP traceback scheme to collect sufficient packets would be longer. The time for path reconstruction should also be reasonably short, so that the victim can complete the reconstruction of attack graph rapidly, and make use of

the attack graph for defending against the attack and restoring the service availability within a short time. It is also desirable for a IP traceback scheme to have low false positives, so that legitimate hosts would not be included in the reconstructed attack graph.

We have compared our results with the two recently proposed packet marking schemes: fragment marking scheme (FMS) [72] and the advanced marking scheme (AMS) [75]. Recall from section 2.2.4, FMS records the IP addresses of two routers constituting of an edge. But to overcome the storage problem, the marking is broken into k fragments each labelled with its offset. The next downstream router uses the offset to select the appropriate fragment to XOR- thereby encoding part of an edge. If enough packets are sent by the attacker, the victim will eventually receive all fragments.

AMS does not fragment marking information. Instead, they include a (b-5)-bit *XOR* of hashed message authentication codes (HMACs) from each router and its downstream router. Nevertheless, there exist some false positives when the number of attack paths is quite large.

5.2.2 Choice of Marking Probabilities

Fixed Probability

To guarantee a sufficiently short attack path reconstruction time, the marking probability p for domain edge marking should not be too small. According to the *coupon collecting problem* [64], for each attack path with d routers (excluding the victim), the expected number of packets needed to reconstruct the attack path is $N(d) = \frac{ln(d)}{p(1-p)^{d-1}}$. We observe that $N(d)$ is minimized when $p = \frac{1}{d}$. This is the best result we can achieve for marking algorithm with a fixed probability. Suppose the maximum distance d for an inter-domain path

is 15 hops, we can get the value of marking probability p to be around 6%.

Since the markings from source routers are more valuable to our defense scheme, particularly in the attack source identification and packet filtering processes, the marking probability q used in the source routers should be slightly higher than p. Higher marking probability, however, implies more marking overhead to the routers. Therefore, a large value of q is undesirable and so we set q to be 20%.

Distance-adjustable Probability

In the original probabilistic edge sampling scheme [72], a fixed marking probability is used by all the routers. Let $P_m(d)$ denote the probability that a packet is marked by a router d hops from the victim and that it arrives at the victim without being re-marked. If the two fixed marking probabilities p and q are applied in the marking scheme, we have:

$$P_m(d) = q(1-p)^{maxd-1} + \sum_{d=1}^{maxd} p(1-p)^{d-1}$$

With the predefined values of $p = 6\%$ and $q = 20\%$, $P_m(d)$ would be around 66%. The above relation shows that $P_m(d)$ will be smaller for a larger value of d. This implies that the probability of a packet being marked by a router far away from the victim and not subsequently re-marked by routers close to the victim will be small. In another word, markings in packets with a large distance value will have a greater chance of being overwritten [69]. As a result, there will be more packets marked by routers close to the victim than those marked by routers further away.

From this we can deduce that, by using a fixed marking probability, the

victim will receive too many packets marked by nearby routers, and too few packets marked by remote routers for path reconstruction. To avoid this problem, the marking probability can be made adjustable; it should be *proportional* to the *distance* between the router concerned and the victim. Through this adjustment, the marking probability employed in the marking process would gradually decrease when the packets are transmitted along the attack paths towards the victim.

Regarding the parameter setting in the experiments, the *domain edge* marking probability p is set to be 6%. After adjusting it based on the *distance*, the range of p is between 3% and 6%. For attack source identification purpose, the value of *source router* marking probability q is larger, which is set to be 20%.

5.2.3 Experimental Results

Various experimental results are presented in figures 5.1 to 5.5. From the plots, it can be observed that the proposed marking scheme is feasible and the overall performance is quite promising.

Minimum number of packets for path reconstruction

Figure 5.1 shows the minimum number of packets, required for reconstruction, sent by an attacker along any single path with marking probabilities $p = 6\%$ and $q = 20\%$. In the simulations, the probabilities for a successful path reconstruction are at least 95%. Although our scheme is domain-based, to have a fair comparison with other marking schemes, the path length presented here is of router-level. The plot compares the minimum number of packets required by our proposed marking scheme with those presented in FMS [72] and AMS [75].

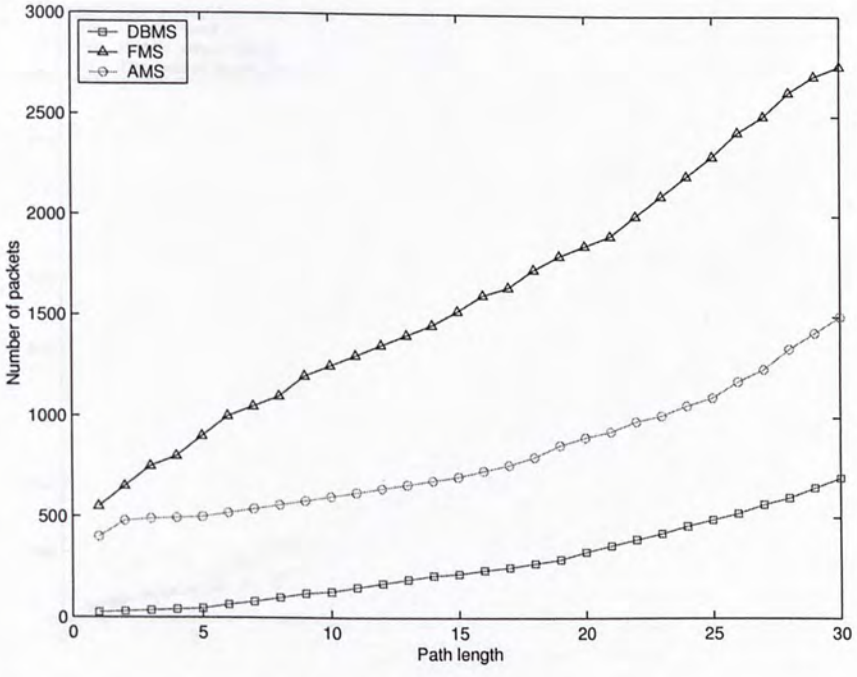

Figure 5.1: Minimum number of packets for path reconstruction against different path lengths

From figure 5.1 we can observe that our marking scheme requires significantly fewer packets for attack path reconstruction. One main reason is that in our proposed marking scheme, the markings are formed based on the prefix of a router's IP address, which corresponds to a domain identity, and so in theory we need only four packets from the same router (representing the domain participating in the marking) for path reconstruction. This number is less than that of FMS, which involves splitting an IP address into a number of fragments during packets marking and combining the fragments from different packets during path reconstruction.

In general, each of the marking schemes requires more packets for a larger value of path length. However, the underlying rate of increase of packets with path length of our proposed marking scheme is smaller than those of other marking schemes.

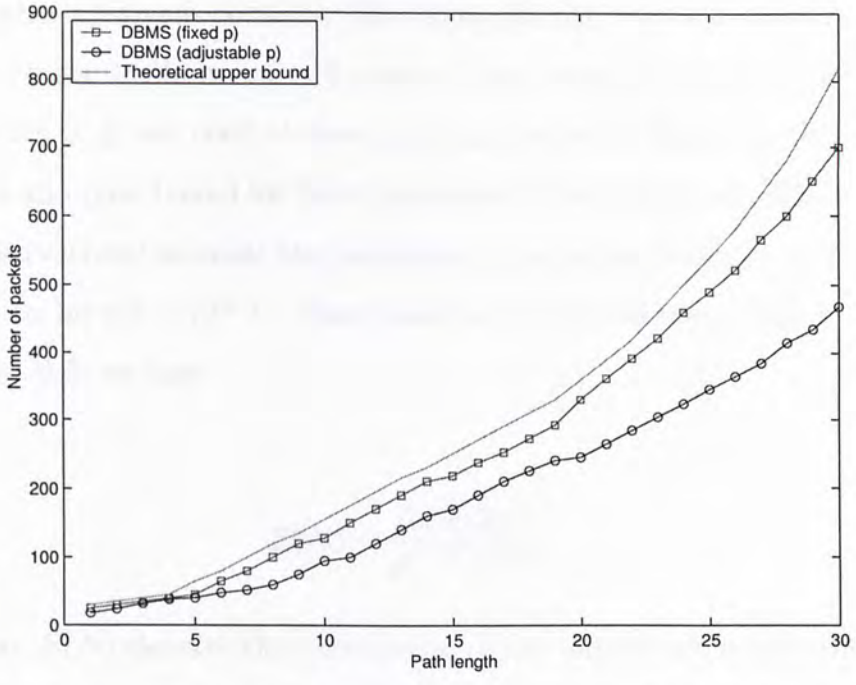

Figure 5.2: Minimum number of packets for reconstructing paths with different lengths using DBMS with fixed and distance-adjustable marking probabilities

Figure 5.2 compares results based on a fixed marking probability equal to 6% from the proposed marking scheme with those based on an adjustable marking probability of around 3% to 6%. Using a fixed marking probability could lead to the problem of having more packets marked by routers close to the victim and less packets from remote routers and as a result we need more packets from the attackers for attack path reconstruction. As shown in the plot, using a distance-adjustable probability requires fewer packets for path reconstruction.

Figure 5.2 also presents an estimated theoretical upper bound for the number of packets needed for attack path reconstruction in our marking scheme. Suppose we split an IP address into c identical chunks and the distance from the attacker to the victim is d. In our marking scheme, we need c packets to identify each domain adjacent to the victim and $2c$ packets to identify each

of the other upstream domains. The victim should receive at least $2c$ packets marked by the furthest router for attack path reconstruction. If the marking probability is p, we need at least $\frac{2c}{p(1-p)^{d-1}}$ packets. Based on this, we can evaluate an upper bound for the expectation of the number of packets needed. We conservatively estimate the probability of a packet marked with a distance $d_1 < d$ to be $p(1-p)^{d-1}$. Then based on the well-known *coupon collector problem* [64], we have

$$E(N) < \frac{2c \ln(2cd)}{p(1-p)^{d-1}}$$

where $E(N)$ denotes the expectation of the number of packets needed for reconstruction. Since the value of c in our traceback scheme is fixed, the value of $E(N)$ directly depends on the marking probability p. This value has also been plotted in figure 5.2, so as to assess if our simulation results are reasonable.

Multiple attack sources

Our attack path reconstruction algorithm does not need to discern the packets by the paths through they traversed to the victim. Since *edge sampling* is used, we can unambiguously identify two adjacent routers of each path by using the proposed reconstruction algorithm. Therefore, our traceback scheme is effective at tracing multiple attacks.

Figure 5.3 further compares the use of fixed and adjustable marking probability in our proposed marking scheme under different number of attack sources. By examining the results, we have three observations: (1) more attack sources require more packets for reconstruction, and the number of

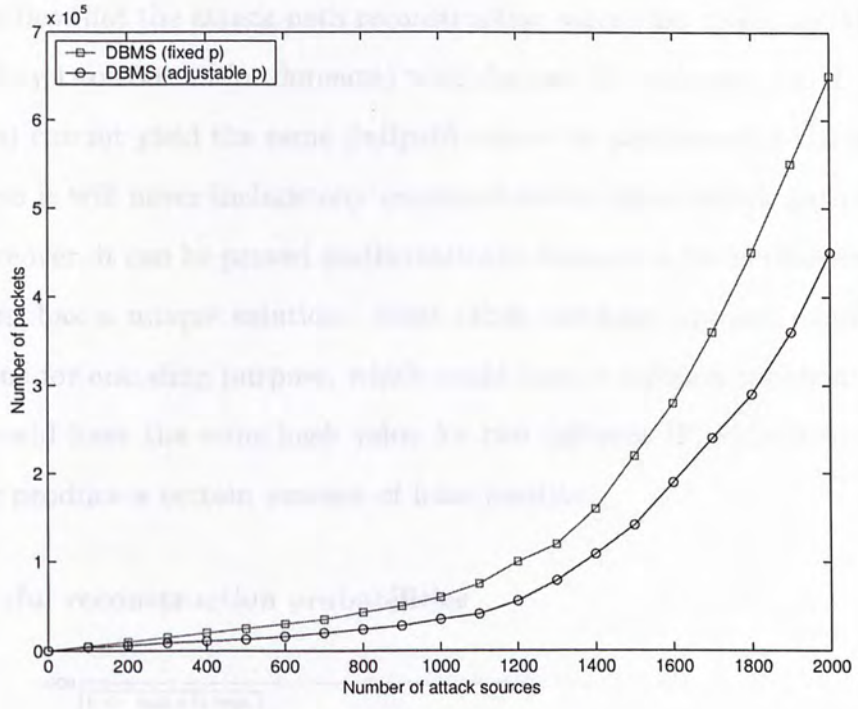

Figure 5.3: Minimum number of packets for reconstructing paths with multiple attack sources using fixed and distance-adjustable marking probabilities

packets increase exponentially with the number of attack sources; (2) when compared with the adjustable marking probability, the fixed marking probability requires additional packets for reconstruction; and (3) the additional packets mentioned in (2) would become more pronounced when the number of attack sources becomes large. In conclusion, the results confirm the proposed adjustable marking probability could be quite significant since it would enable the victim to trace the attack sources with relatively fewer packets sent by the attacker(s).

False positives

The most prominent advantage of our marking scheme is that no false positives are generated in our millions of path reconstruction experiments. This

also verifies that the attack path reconstruction algorithm yields no false positives. Any two routers (or domains) with distinct IP addresses (or IP address prefixes) cannot yield the same $fullpath$ values for packets with the same set of x's, so it will never include any unrelated router in an attack path.

Moreover, it can be proved mathematically because a *Vandermonde* matrix equation has a unique solution. Most other marking schemes employ hash functions for encoding purpose, which could have a collision problem; that is, they would have the same hash value for two different IP addresses. So they usually produce a certain amount of false positives.

Successful reconstruction probabilities

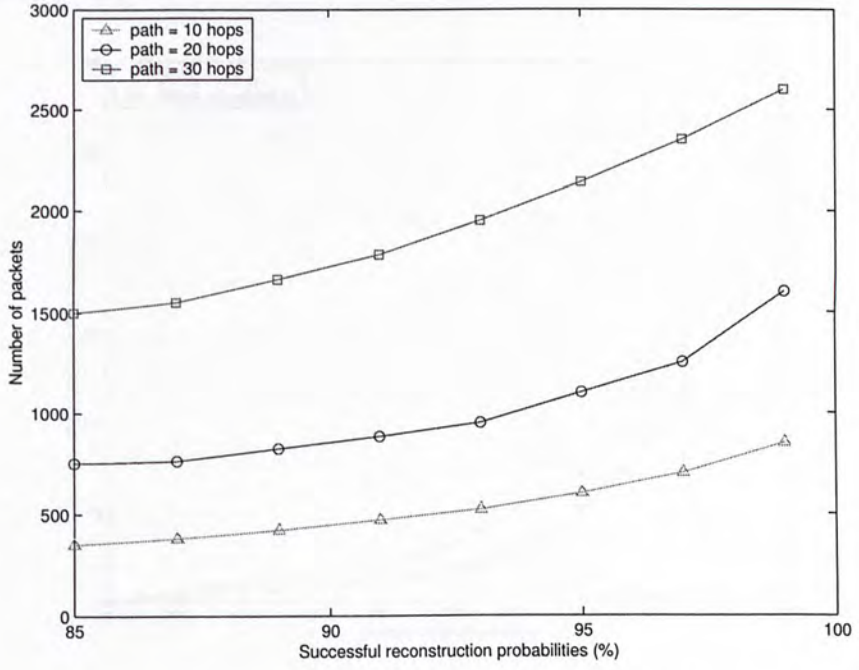

Figure 5.4: Number of packets needed to reconstruct paths with successful reconstruction probabilities between 85% and 99%

We have also performed experiments to investigate how the number of packets needed for reconstruction varies with different successful reconstruction

probabilities. Figure 5.4 shows the results based on a marking probability of 6%, and reconstructed paths with length of 10 hops, 20 hops and 30 hops respectively. In the plot, the probability to have a successful reconstruction ranges from 85% to 99%.

From figure 5.4, we observe that for a given path length, the number of packets for reconstruction increases geometrically as the success probability increases. The figure also shows that the number of packets for reconstruction increases non-linearly with the success probabilities. The increase in the number of packets will be more acute as the successful reconstruction probability approaches 100%.

Reconstruction time

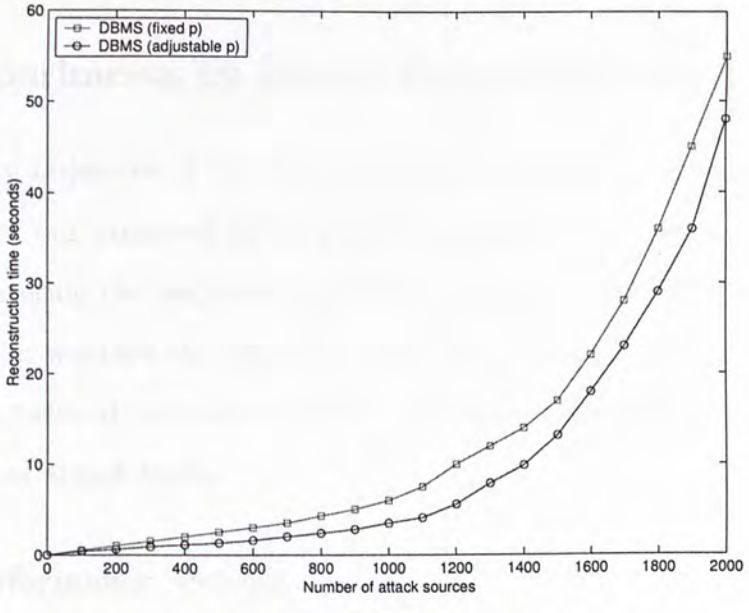

Figure 5.5: Path reconstruction time under attack with multiple sources

Figure 5.5 presents the reconstruction time of the proposed marking scheme in handling multiple attacks. It can be observed that our reconstruction algorithm can reconstruct 2,000 distributed attack paths (with a path length of

20) within just one minute on a 1GHz Pentium IV Linux workstation, which is considered an acceptable time duration even for real-time IP traceback applications. It is also much faster than FMS [72], which requires a high computation overhead in examining various combinations of the edge fragments.

A good portion of the reconstruction time has been spent on sorting and grouping the packets. When the number of received packets becomes very large, say, more than 300,000, the proposed reconstruction algorithm might take more time than does the advanced marking scheme [75]. However, in practice, the victim can simply use a subset of received packets for reconstruction if the reconstruction time is crucial; moreover, if necessary, the overhead on grouping the packets could be much reduced by using sophisticated sorting algorithms and implementation techniques.

5.3 Experiments on Packet Filtering Scheme

The primary objective of the following experiments is to investigate the effectiveness of our proposed filtering scheme. Each router on the attack paths simulates filtering the malicious packets as defined in the filtering algorithm. Depending on whether the filtering is deployed at source-end and victim-end, the filtering ratio of the routers would vary so as to contribute to an efficient elimination of attack traffic.

5.3.1 Performance Metrics

To estimate the accuracy of our packet filtering scheme, we measure both the normal and attack packets drop ratio at the points of filtering routers. Since the markings embedded in the packets are not unique within the network, false positives may exist in case a packet carrying the same checksum as the

Table 5.2: Control parameters and performance metrics used in our filtering scheme

Performance metrics	N_{drop} (False Drop Ratio): Percentage of the normal packets dropped (false positives)
	A_{drop} (Attack Traffic Drop Ratio): Percentage of the attack packets dropped
	N_{surv} (Normal Traffic Survival Ratio): Percentage of normal packets finally arrived at the victim
Control Parameters	p: Probability for domain edge marking
	q: Probability for source router marking
	a: Number of attack sources
	g: Percentage of normal packets out of all packets generated
	r: Perimeter radius centered at the victim
	f_d: Filtering probability for packets with domain edge markings
	f_r: Filtering probability for packets with source router markings
	f_b: Filtering probability for packets with no marking (blank)

attack ones is passing through the filtering agent. Packet filtering at the victim provides the fast relief for the victim, whereas the complementary source-end filtering can accurately remove the majority of attack traffic near the sources. Table 5.2 lists the control parameters and performance metrics used in our filtering scheme.

False Drop Ratio (N_{drop}) represents the percentage of the normal traffic being dropped. It is obtained by the number of normal packets dropped out of the total number of normal packets. This ratio is also referred as false positives, in which a normal packet is incorrectly classified as an attack one. Attack Traffic Drop Ratio (A_{drop}), on the other hand, represents the percentage of the attack packets being dropped. It is obtained by the number of attack packets dropped over the total number of attack packets. This ratio indicates the proportion of attack traffic that our filtering scheme can successfully eliminate.

The Normal traffic Survival Ratio (N_{surv}) is the metric we would like to improve, which is the percentage of normal packets finally arrived at the victim. It is calculated by the number of normal packets remained out of the total number of packets received by the victim.

The control parameter r is the radial distance from the victim. It is used in

the victim-end filtering, which indicates the number of hops between the victim and the filtering routers. As mentioned in 4.2.2, we let pass the packets with markings from routers within certain distance to the victim so as to minimize the number of false drops; and we introduce r to denote this distance. In the experiments, we choose r to be within the range of 1 to 3 hops. The parameter g represents the percentage of legitimate traffic generated over the total traffic. In the following plots, we use three different normal traffic percentages (5%, 10% and 20%) to test our filtering scheme.

5.3.2 Choices of Filtering Probabilities

In our scheme, there are three filtering probabilities, f_d, f_r and f_b. In setting the values for these probabilities, we have the following considerations:

Source-end Filtering

There are two tuning parameters for dropping malicious packets in our *source-end* filtering scheme, namely f_r and f_b. At sources' networks, since most marked packets with *source router* markings are likely to be coming from the attack source, we set f_r to be 1.0, so as to filter all of this kind of packets. However, source router markings are inscribed under a probability q, thus some of the packets coming from the source may remain unmarked. Therefore, in our filtering scheme, we also drop a certain number of blank packets, so as to avoid the attack packets passing further downstream. This approach aims at rate-limiting the attack traffic rather than identifying the attack packets. Figure 5.6 depicts how the Attack Traffic Drop Ratio (A_drop) changes with different values of f_b. The plot shows that A_drop increases linearly with f_b.

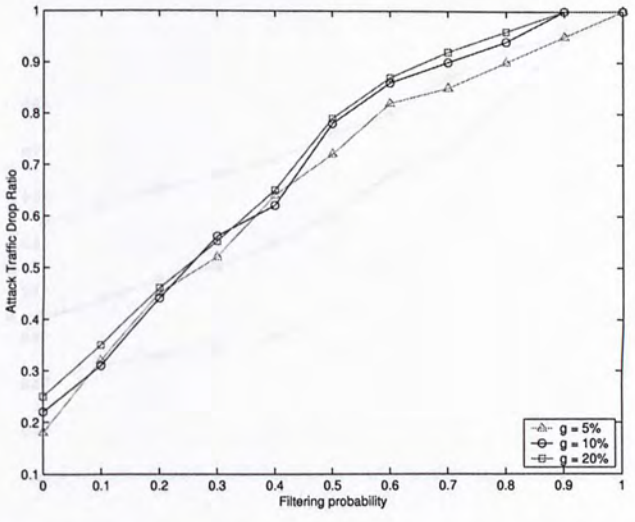

Figure 5.6: Attack Traffic Drop Ratio in source-end filtering, with varying f_b and $f_r = 1.0$

Victim-end Filtering

Similarly, in *victim-end* filtering scheme, we pass the packets with two filtering probabilities: f_d and f_r. Since most marked packets with source router markings are likely to come from the attack source, we set f_r to be a high probability, which is 0.8. We do not set f_r to be 1.0 here, since at the victim side, some packets carrying a *source router* marking may also come from legitimate hosts. Figure 5.7 depicts how the normal traffic survival ratio changes with different values of f_d.

5.3.3 Experimental Results

The different experiment results are as presented in figures 5.8 to 5.15. The metrics for measuring the effectiveness of the packet filtering scheme include False Drop Ratio, Attack Traffic Drop Ratio, and Normal Traffic Survival Ratio.

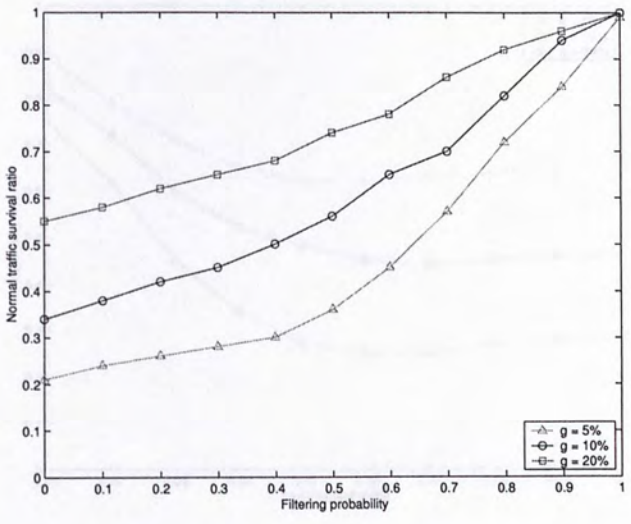

Figure 5.7: Normal Traffic Drop Ratio in victim-end filtering, with varying f_d and $f_r = 0.8$

False Drop Ratio

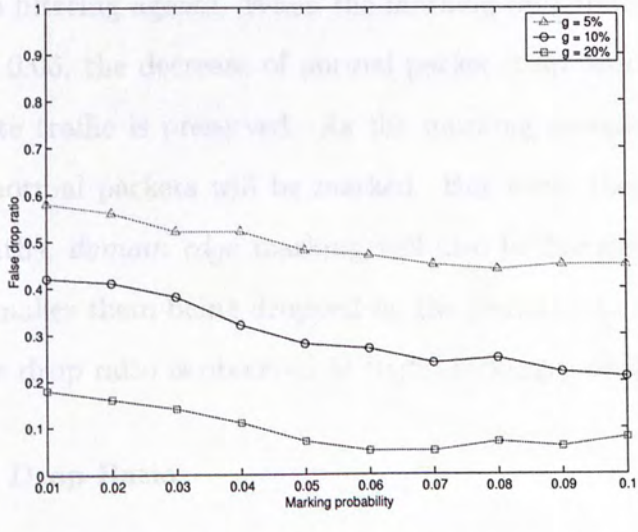

Figure 5.8: False Drop Ratio with different marking probabilities (at source-end)

Figures 5.8 and 5.9 show respectively the normal packet drop ratio at source-end and victim-end under varying marking probability. There are three curves in each figure, corresponding to three different normal traffic percentages (5%, 10% and 20%). We can see from figures 5.8 and 5.9 that under a

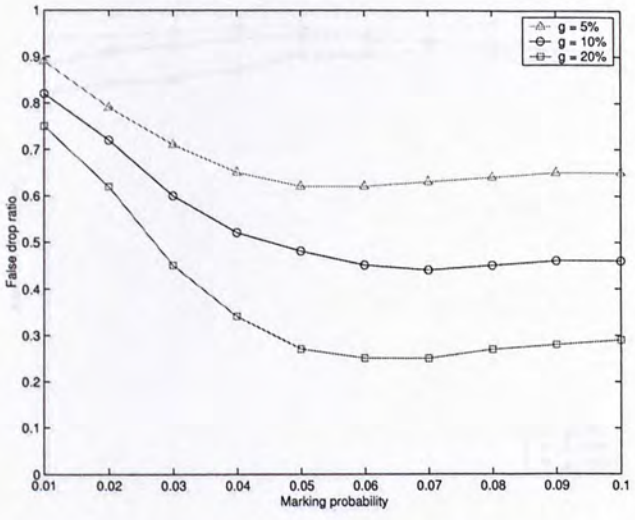

Figure 5.9: False Drop Ratio with different marking probabilities (at victim-end)

relatively low marking probability (around 0.01 to 0.03), the normal packet drop ratio is quite high, since many packets are unmarked and thus they are dropped by the filtering agents. When the marking probability increases, say around 0.04 to 0.06, the decrease of normal packet drop ratio indicates most of the legitimate traffic is preserved. As the marking probability further increases, more normal packets will be marked. But when they traverse along some attack paths, *domain edge* marking will also be inscribed into their IP header, which makes them being dropped by the routers. As a result, a slight increase in false drop ratio is observed at high marking probability.

Attack Traffic Drop Ratio

Figures 5.10 and 5.11 show respectively the attack traffic drop ratio under varying marking probabilities. We can see that when the marking probability is around 0.01 to 0.03, the attack traffic drop ratio is not so promising. The relatively low marking probability causes the packets transmitted along the attack paths remain unmarked. Therefore, the packets would not be dropped

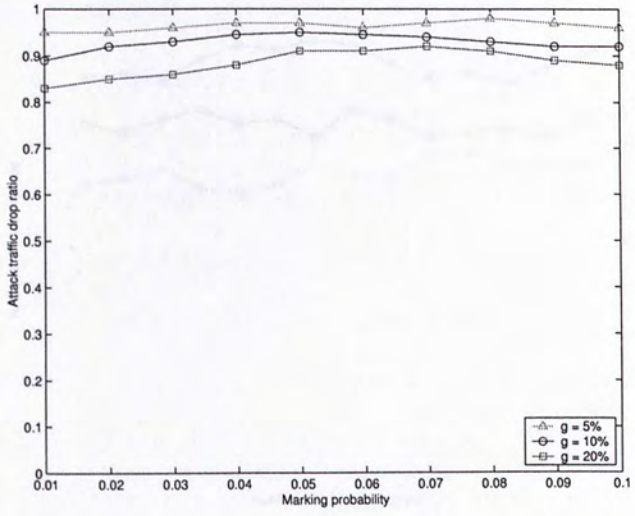

Figure 5.10: Attack Traffic Drop Ratio with different marking probabilities (at source-end)

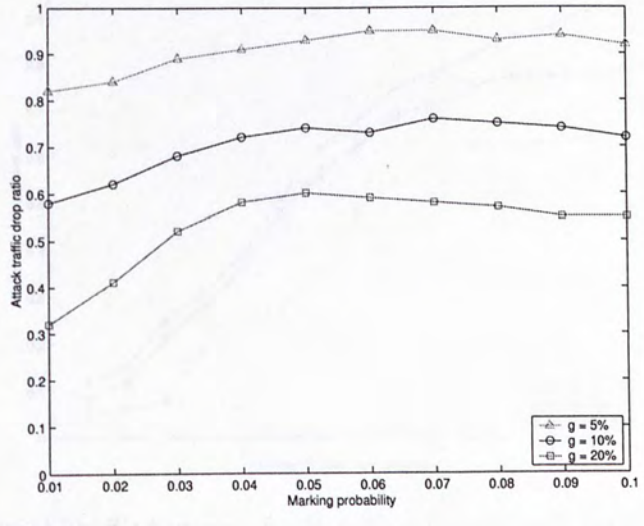

Figure 5.11: Attack Traffic Drop Ratio with different marking probabilities (at victim-end)

by the routers according to our filtering mechanism. Consequently, when the marking probability gradually increases, the number of packets being filtered increases.

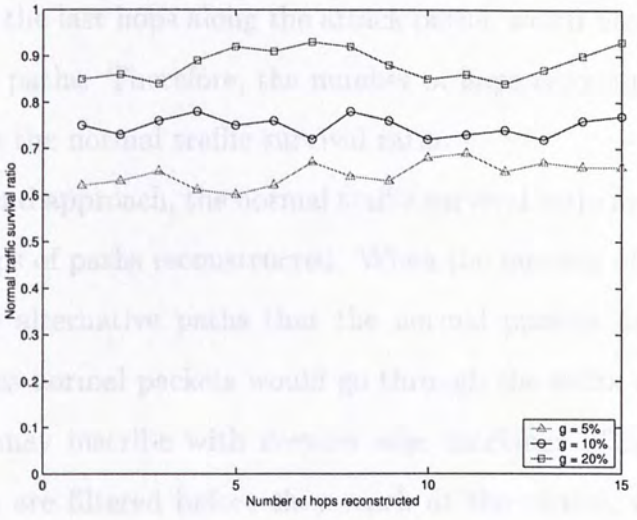

Figure 5.12: Normal Traffic Survival Ratio with different path length (at source-end)

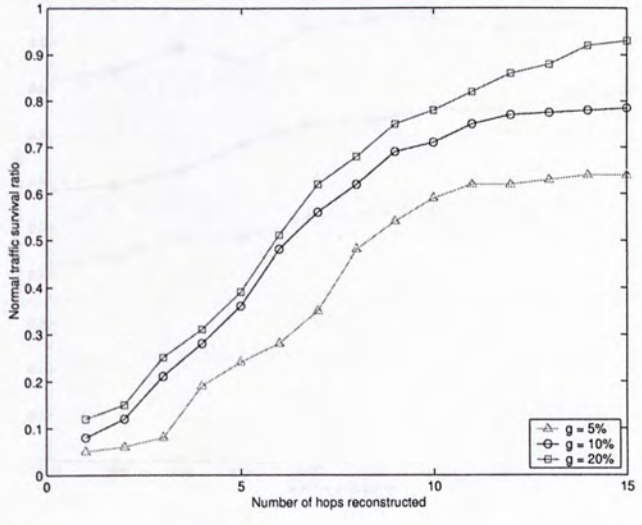

Figure 5.13: Normal Traffic Survival Ratio with different path length (at victim-end)

Normal Traffic Survival Ratio

Figures 5.12 and 5.13 show respectively the normal traffic survival ratio of source-end and victim-end approach with different number of hops reconstructed. From the plots we observe that in source-end approach, the ratio does not change with the number of hops, since the curves inside figure 5.12 are almost flat. This can be explained by the fact that packet filtering is per-

formed once at the last hops along the attack paths, which have nothing to do with the entire paths. Therefore, the number of hops reconstructed has little impact towards the normal traffic survival ratio.

For victim-end approach, the normal traffic survival ratio increases with the incremental hops of paths reconstructed. When the number of hops increases, there are more alternative paths that the normal packets can traverse. In other words, less normal packets would go through the suffix of attack paths, in which they may inscribe with *domain edge* markings. Consequently, less normal packets are filtered before they reach at the victim, which results in the increase of normal traffic survival ratio.

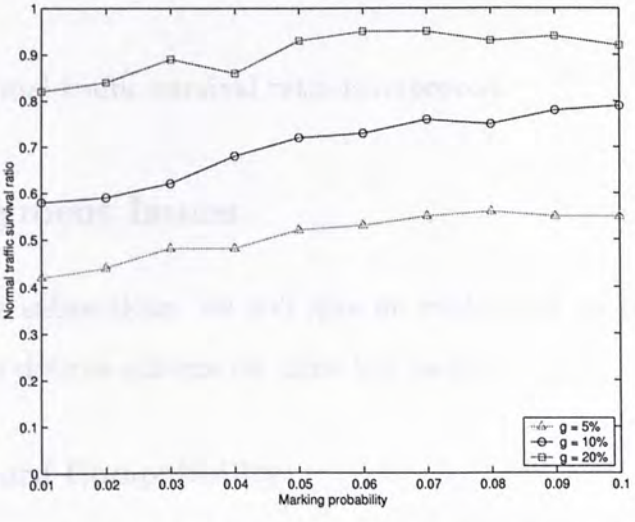

Figure 5.14: Normal Traffic Survival Ratio with different marking probabilities (at source-end)

Figures 5.14 and 5.15 show respectively the normal traffic survival ratio of source-end and victim-end approach under different marking probabilities. With low marking probabilities, the ratio is small, since there are more unmarked packets, causing the filtering routers not be able to differentiate the normal packets from the malicious one. Thus, more legitimate traffic is dropped during packet filtering. But as the marking probability further in-

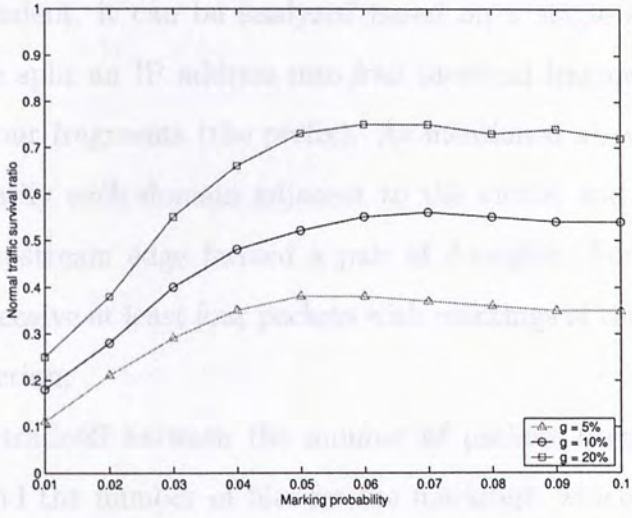

Figure 5.15: Normal Traffic Survival Ratio with different marking probabilities (at victim-end)

creases, the normal traffic survival ratio is improved.

5.4 Deployment Issues

In the following subsections, we will give an evaluation on our proposed IP traceback based defense scheme on some key issues.

5.4.1 Backward Compatibility

Backward compatibility is an important issue concerning whether the proposed method can be put into practice. As our marking scheme involves writing some information to the IP header of a packet, we should find out the maximum number of bits available in an IP header that can be used to store the markings. In our traceback scheme, the total number of bits required to store the markings is 12 bits. After appending the checksum for marking fields' authentication, 24 bits are required from the IP header.

As the minimum number of packets required to reconstruct an attack path

is path independent, it can be analyzed based on a single attack path. In our scheme, we split an IP address into four identical fragments and records only the first four fragments (the prefix). As mentioned above, we need four packets to identify each domain adjacent to the victim and four packets to identify each upstream edge formed a pair of domains. For each edge, the victim should receive at least four packets with markings of the edge for attack path reconstruction.

There is a tradeoff between the number of packets needed for path reconstruction and the number of bits for the markings, which depends partly on the number of IP fragments. In our scheme, we split an IP address into four identical fragments. A smaller number of IP fragments for an IP address implies:

1. Fewer packets and a shorter time would be required for attack paths reconstruction since the number of IP fragments would be smaller;

2. More bits would be needed since the value of each IP fragment would be larger. Though the range of distinct values for x would be smaller, the total number of bits needed would be larger.

For packet marking purpose, our traceback scheme would overload certain bit space inside the IP header to store the markings. The fields being overloaded includes the 16-bit *Identification field* and 8-bit *Type of Service field.* Figure 5.16 shows the structure of the IPv4 header.

The 8-bit type of service field is used to allow hosts way to give hints to routers as to what kind of route is important for particular packets [11], and it has been little used in current network design. The 16-bit Identification field enables the destination host to determine which datagram a newly arrived packet fragment belongs to. Since Stoica and Zhang [76] pointed out that

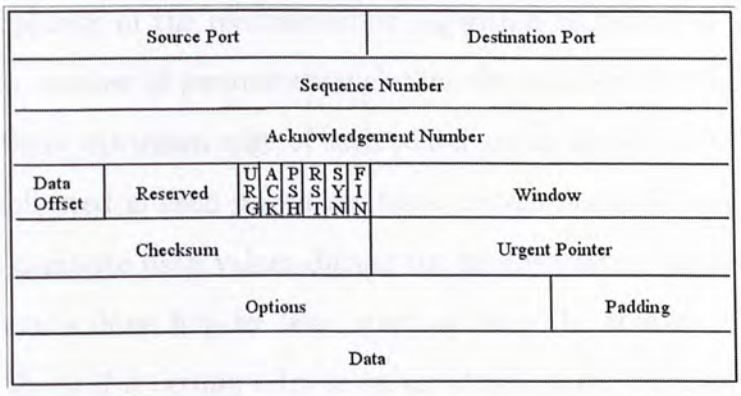

Figure 5.16: Structure of IPv4 Header

less than 0.25% of the entire network traffic is fragments, we consider that overloading the Identification field is appropriate. These 24 bits could be assigned to the marking fields. Therefore, the proposed marking scheme is backward compatible with current version of IP protocol and can be effectively put into practice.

Nonetheless, the proposed marking scheme could not be applied directly to IPv6, where the IP header does not have the Identification field and the IP address is 128 bits. However, it is possible that there could be similar space available in the IP header of IPv6; if the space available is not sufficient, we need to partition the IP address into more fragments.

5.4.2 Processing Overheads to the Routers and Network

The packet marking algorithm as shown in section 3.3.4 takes only a constant time to execute. Each router marks the packets with a low marking probability. When marking a packet, it computes a *fullpath* value for a single router or for an edge involving two adjacent routers. To reduce the overhead on the computation of such *fullpath* values, we can keep possible *fullpath* values in a table for each router. Thus, the marking overhead would become very small.

The complexity of the reconstruction algorithm as shown in section 3.3.7 depends on a number of parameters including the number of attack paths, the number of direct upstream edge of each router on an attack path, the number of packets collected in each packet set for a certain distance from the victim, the time to compute path values during the reconstruction process, etc. The reconstruction is done hop by hop, starting from the routers closest to the victim. To check if a certain edge is on an attack path, we need to compute four path values; overall, it is quite fast.

When compared to the probabilistic marking scheme of Savage et al. [72], checking each direct edge (from the upstream routers map) of a router already found to be on a reconstructed path is much more efficient than checking all possible combinations of IP fragments. While our traceback scheme has a computation complexity of around $O(dn^2)$, the method of Savage et al. has a complexity of around $O(dn^8)$, where d is the maximum path length and n is the number of attacking hosts. Since our domain based marking scheme involves a smaller distance d, its complexity is relatively small.

At the baseline, our proposed marking scheme requires only a random number generator, normal ALU operations, and compares. These could easily be accomplished using combinational logic in an ASIC [92] or custom chip. Therefore, the computation involved in packet marking is very efficient for contemporary routers.

In addition, with the aid of a upstream router map, we need not solve the polynomials of *Vandermonde* matrix as proposed in the algebraic approach by Dean et al. [23]. Thus, the processing time for path reconstruction can be much reduced. We can further speed up the reconstruction process by storing in a table the path values based on different values of x for each router. Then, instead of computing the path values, the reconstruction algorithm can

search from the table the path values for any upstream router being examined; so much computation time could be reduced. Overall, the proposed path reconstruction algorithm is quite efficient.

5.5 Evaluations

Through numerous simulation experiments in evaluating the feasibility of the proposed DDoS defense scheme, we have the following observations:

1. The scheme requires a much smaller number of packets for attack path reconstruction than other methods such as FMS [72] and AMS [75] as presented in the experiment results, section 5.2.3;
2. It can handle multiple attack sources effectively and can reconstruct essentially all attack paths;
3. No false positives are produced during attack path reconstruction. This can be proved by strict mathematical theorem that the matrix equation with *Vandermonde* matrix coefficient has unique solution.
4. It performs attack path reconstruction quite rapidly, and takes only one minute to reconstruct as many as 2,000 attack sources;
5. With the introduction of the proposed packet filtering scheme, around 90% of the attack packets can be successfully removed, and the normal traffic survival ratio could be enhanced from 20% (without filtering) to around 90%. Thus the service availability could be much improved.

□ **End of chapter.**

Chapter 6

Conclusion

This thesis investigates the knowledge of DDoS attack defense mechanisms. After noticing the importance of the network security in nowadays Internet world and reviewing the state-of-the-arts technologies of DDoS countermeasures, we present a practical approach comprised of IP traceback and packet filtering techniques to efficiently defend against DDoS attacks.

6.1 Contributions

Our DDoS defense scheme exploits an innovative IP traceback scheme to identify the attack paths. Then by preferentially dropping the packets coming from attack paths, our dynamic packet filtering scheme effectively mitigates the attack while protecting the legitimate traffic from collateral damage. In conclusion, our research work has the following contributions to the advancement of network security:

- While many proposals focus on IP traceback which aims at identifying potential attack sources, they cannot be employed to defend against DDoS attacks. Motivated by this, we propose a hybrid traceback - filter-

ing defense scheme which can both identify the attack paths, as well as preferentially drop the packets coming from the attack paths.

- What makes our approach more appealing is that making intelligent use of the traceback information can enhance the effectiveness of packet filtering. In the literature, there are very few proposals suggesting to use traceback information as filtering signatures. From our experiment results, it can be observed that the effectiveness of this approach is quite high, in terms of high accuracy in dropping attack packets, as well as high improvement on the throughput of legitimate traffic.

- Our proposed marking scheme is domain based, meaning that only the border routers of each domain are involved, without requiring the universal employment of all routers. The idea of encoding just inter-domain attack path instead of the whole path is innovative, in which the involvement and overhead incurred to the routers are much reduced. So the proposed method would be relatively easy for practical implementation.

- Most existing marking schemes do not address the problem of packet marking authentication. Consequently, a compromised router on the attack path could forge the markings of upstream routers. To solve this problem, we propose a light-weight authentication mechanism which enables the victim to check the integrity of the markings, so as to avoid them being forged by the attackers.

- We have proposed a novel IP traceback scheme, which features a number of advantages:

  - The markings are formed based on the prefix parts of a pair of adjacent domain border router IP addresses instead of the full IP

addresses, so fewer bits are encoded; only 12 bits are required for recording the markings.

- Since only domain border routers are considered, when measured in terms of the number of domains, the number of hops reconstructed is reduced. So the number of packets required to identify each attack path can be kept to a minimum. Further, the proposed use of an adjustable marking probability facilitates the use of even less packets for path reconstruction, since fewer packet samples are required from remote routers.

- Since fewer packets can be used to reconstruct the paths, the marking overhead as well as reconstruction time would be relatively small when compared with other marking schemes.

- To enable tracing more accurately the sources of the attack packets, it also performs the source router marking, which can be considered a kind of probabilistic node sampling. The source router markings can be used to confirm if the end domain of a reconstructed path corresponds to the source of an attack. If it is not the source of an attack, the last domain in a reconstructed path could be a compromised one.

- Attack path reconstruction is carried out quite rapidly by our marking scheme, and thus it could be used to locate attack sources in real time, which is one of the critical steps in defending against DDoS attacks.

- The path reconstruction process generates no false positives. This can be proved by strict mathematical theorem that the matrix equa-

tion with *Vandermonde* matrix coefficient has unique solution.

- In addition to the IP traceback scheme, we have proposed an effective packet filtering scheme, which has a promising performance in various aspects:
  - The packet filtering scheme can be deployed dynamically at various strategic positions, which enables the victim to quickly discard the majority of attack packets and restore the service availability.
  - When compared to other filtering methods which involve detecting anomaly traffic rate from the network monitored, the filtering signatures used in our scheme are formed based on the actual attack source information. Thus the chance of dropping legitimate traffic by the scheme can be minimized.
  - With our filtering scheme, about 90% of the attack packets can be successfully removed, and the normal traffic survival ratio could be enhanced from 20% (without filtering) to around 90%. Thus the service availability could be much improved.

Based on the above outstanding features, we believe that the proposed IP traceback based DDoS defense scheme can act as a light-weight, flexible and powerful DDoS deterrent to date, and show a significant potential in reducing the DDoS threat.

6.2 Discussions and future work

For any IP traceback solution to be effective, it would need to be deployed across corporate and administrative boundaries in a substantial portion of the

Internet infrastructure. However, it is somehow difficult to achieve in such a large-scale network which consists of millions of domains. To effectively put our scheme into practice, it should be enhanced to support better incremental deployment, which involves as few routers as possible, and could be implemented with negligible overhead.

Another limitation of our approach is that its current design could have difficulty to be applied to IPv6, where the IP header does not have the Identification field and the IP address is 128 bits (note that the IP address of IPv4 is only 32 bits). It is also not compatible with *IPSec*. These are the inherited problems of any marking-based traceback technique. A feasible solution should be worked out before IPv6 would become widely employed.

As complimentary to the filtering signatures generated based on the reconstructed attack graph and filtering probabilities, more parameters could be taken into consideration in limiting the attack traffic rate, such as the bandwidth at the sources' and the victim's network, and some statistical and historical data logged inside the routers.

With all the ongoing research into IP traceback and packet filtering methods, the final challenge is convincing the disparate entities now controlling the Internet to work together, share information about traffic flowing through their networks, and equip with proper defense mechanisms, so as to mitigate the vulnerability towards DDoS attacks.

□ **End of chapter.**

Bibliography

[1] Caida's Skitter Internet Mapping Project - Dataset. http://www.caida.org/tools/measurement/skitter/results.xml.

[2] Technical Cyber Security Alerts, W32/MyDoom.B. http://www.us-cert.gov/cas/techalerts/TA04-028A.html.

[3] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[4] Transmission Control Protocol DARPA Internet Program Protocol Specification. Internet Eng. Task Force RFC 793, Sept. 1981.

[5] Distributed Denial of Service Tools. Computer Emergency Response Team, July 1999. http://www.cert.org/incident_notes/.

[6] Egress Filtering v 0.2. Global Incident Analysis Center Special Notice, SANS Institute Resources, Feb. 2000.

[7] Yahoo on Trail of Site Hackers. Wired.com, Feb. 2000. http://www.wired.com/news/business/0,1367,34221,00.html.

[8] Powerful Attack Cripples Internet. Associated Press for Fox News, Oct. 2002. http://www.foxnews.com/story/0,2933,66438,00.html.

[9] M. Alder. Tradeoffs in Probabilistic Packet Marking for IP Traceback. In *Proceedings of the 34th ACM Symposium of Theory of Computing*, pages 407–418, 2002.

[10] H. Aljifri. IP Traceback: A New Denial-of-Service Deterrent? *IEEE Security and Privacy*, pages 24–31, 2003.

[11] P. Almquist. Type of service in the internet protocol suite. RFC 1349, Network Working Group, IETF, July 1992. http://rfc.sunsite.dk/rfc/rfc1349.html.

[12] T. Baba and S. Matsuda. Tracing Network Attacks to Their Source. *IEEE Internet Computing*, 6(3):20–26, 2002.

[13] R. Bace and P. Mell. Special Publication on Intrusion Detection System. Tech. Report SP 800.31, National Institute of Standards and Technology, Nov. 2001.

[14] J. Barlow and W. Thrower. TFN2K - An Analysis. Information Security Bulletin, Mar. 2000.

[15] A. Belenky and N. Ansari. IP Traceback with Deterministic Packet Marking. *IEEE Communications Letters*, 7(4), Apr. 2003.

[16] A. Belenky and N. Ansari. On IP Traceback. *IEEE Communications Magazine*, pages 142–153, July 2003.

[17] S. Bellovin, M. Leech, and T. Taylor. ICMP Traceback Messages. Internet Draft, Internet Eng. Task Force, Oct. 2001. work in progress.

[18] H. Burch and B. Cheswick. Tracing anonymous packets to their approximate source. In *Proceedings of 2000 USENIX LISA Conference*, pages 319–327, Dec. 2000.

[19] R. K. Chang. Defending against Flooding Based Distributed Denial of Service Attacks: A Tutorial. *IEEE Communications Magazine*, 40(10):42–51, Oct. 2002.

[20] Y. W. Chen. Study on the Prevention of SYN Flooding by Using Traffic Policing. In *Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 593–604, Apr. 2000.

[21] Z. L. Chen and M. C. Lee. An IP Traceback Technique against Denial-of-Service Attacks. In *Proceedings of 19th Annual Computer Security Applications Conference (ACSAC 2003)*, pages 96–105, Dec. 2003.

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

[23] D. Dean, M. Franklin, and A. Stubblefield. An Algebraic Approach to IP Traceback. *ACM Transactions on Information and System Security*, 5(2):119–137, May 2002.

[24] H. Debar, M. Becker, and D. Siboni. A neural network component for an intrusion detection system. In *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, pages 240–250, 1992.

[25] D. E. Denning. An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*, 13(2):222–232, 1987.

[26] S. Dietrich, N. Long, and D. Dittrich. An analysis of the "Shaft" distributed denial of service tool. Information Security Bulletin, May 2000. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt.

[27] D. Dittrich. The DoS Project's 'Trinoo' Distributed Denial of Service Attack Tool, Oct. 1999. http://staff.washington.edu/dittrich/misc/trinoo.analysis.

[28] D. Dittrich. The "stacheldraht" distributed denial of service attack tool, Dec. 1999. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt.

[29] D. Dittrich. The "Tribe Flood Network" distributed denial of service attack tool, Oct. 1999. http://staff.washington.edu/dittrich/misc/tfn.analysis.txt.

[30] D. Dittrich, G. Weaver, S. Dietrich, and N. Long. The "mstream" distributed denial of service attack tool, May 2000. http://staff.washington.edu/dittrich/misc/mstream.analysis.txt.

[31] T. W. Doeppner, P. N. Klein, and A. Koyfman. Using Router Stamping to identify the Source of IP packets. In *Proceedings of the 7th ACM*

conference on Computer and Communications Security (CCS'00), pages 184–189, Nov. 2000.

[32] C. Douligeris and A. Mitrokotsa. DDoS attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 44(5):643–666, Apr. 2004.

[33] R. Farrow. Spoofing Source Addresses. Information Network Security. http://www.spirit.com/Network/net0300.html.

[34] W. C. Feng, D. D. Kandlur, D. Saha, and K. G. Shin. Adaptive Packet Marking for Maintaining End-to-End Throughput in a Differentiated-Services Internet. *IEEE/ACM Transactions on Networking*, 7(5):685–697, Oct. 1999.

[35] P. Ferguson and D. Senie. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. Internet Eng. Task Force RFC 2827, May 2000. http://www.ietf.org/rfc/rfc2827.txt.

[36] T. M. Gil and M. Poletto. MULTOPS: A Data-Structure for bandwidth attack detection. In *Proceedings of USENIX Security Symposium*, pages 23–38, July 2001.

[37] M. T. Goodrich. Efficient Packet Marking for Large-Scale IP Traceback. In *Proceedings of 9th ACM conference on Computer and Communications Security*, pages 117–126, Nov. 2002.

[38] K. J. Houle and G. M. Weaver. Trends in Denial of Service Attack Technology. Technical report from CERT Coordination Center, Oct. 2001.

[39] A. Hussain, J. Heidemann, and C. Papadopoulos. A Framework for Classifying Denial of Service Attacks. In *Proceedings of ACM SIGCOMM 2003*, pages 99–110, Aug. 2003.

[40] K. Ilgun, R. A. Kemmerer, and P. A. Porras. State Transition Analysis: A Rule-Based Intrusion Detection Approach. *IEEE Transactions on Software Engineering*, 21(3):181–199, 1995.

[41] J. Ioannidis and S. M. Bellovin. Implementing Pushback: Router-Based Defense Against DDoS Attacks. In *Proceedings of Network and Distributed System Security Symposium*, Feb. 2002.

[42] T. W. Judson. Abstract algebra: theory and applications. Boston, MA: PWS Pub. Co., 1994.

[43] J. Jung, B. Krishnamurthy, and M. Rabinovich. Flash crowds and Denial of Service attacks: Characterization and Implications for CDNs and Web Sites. In *Proceedings of the Eleventh International World Wide Web Conference (WWW 11)*, May 2002.

[44] F. Kargl, J. Maier, and M. Weber. Protecting web servers from distributed denial of service attacks. In *Proceedings of World Wide Web*, pages 514–524, May 2001.

[45] G. C. Kessler. Defenses Against Distributed Denial of Service Attacks. SANS Institute, Nov. 2000.

[46] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. Internet RFC 2104, Feb. 1997.

[47] J. F. Kurose and K. W. Ross. *Computer Networking: A top-Down Approach Featuring the Internet.* Addison Wesley Longman, Inc., 2001.

[48] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic. Distributed denial of service attacks. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2275–2280, Jan. 2000.

[49] N. S. Lau and M. C. Lee. Intrusion Detection System Models. In *Proceedings of the 2003 International Conference on Security and Management (SAM'03)*, pages 359–364, June 2003.

[50] N. S. Lau and M. C. Lee. An Efficient Domain Based Marking Scheme for IP Traceback. In *Proceedings of the 7th IEEE International Conference on High Speed Networks and Multimedia Communications (HSNMC'04)*, pages 1080–1091, June 2004.

[51] N. S. Lau and M. C. Lee. Towards a IP Traceback based Defense Scheme against DDoS Attacks. Submitted to Journal of the American Society for Information Science and Technology (JASIST), Aug. 2004.

[52] K. T. Law, J. C. S. Lui, and D. K. Y. Yau. You can run, but you can't hide: An Effective Methodology to Traceback DDoS Attackers. In *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS'02)*, June 2002.

[53] M. C. Lee and C.-K. Fung. A Public-Key Based Authentication and Key Establishment Protocol Coupled with a Client Puzzle. *Journal of the American Society for Information Science and Technology (JASIST)*, 54(9):810–823, 2003.

[54] S. C. Lee and C. Shields. Tracing the Source of Network Attack: A Technical, Legal and Societal Problem. In *Proceedings of 2001 IEEE Workshop on Information Assurance and Security*, pages 239–246, 2001.

[55] W. Lee and K. Park. On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack. In *Proceedings of IEEE INFOCOM 2001*, pages 338–347, 2001.

[56] W. Lee and S. J. Stolfo. A Framework for Constructing Features and Models for Intrusion Detection Systems. *ACM Transactions on Information and System Security*, 3(4):227–261, Nov. 2000.

[57] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source Address Validity Enforcement Protocol. In *Proceedings of IEEE INFOCOM 2002*, June 2002.

[58] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. In *Proceedings of ACM SIGCOMM 2001*, 2001.

[59] A. Mankin, D. Massey, C. Wu, S. Wu, and L. Zhang. On Design and Evaluation of 'IntentionDriven' ICMP Traceback. In *Proceedings of IEEE*

International Conference on Computer Communications and Networks, pages 159–165, 2001.

[60] M. J. Martin. Router Expert: Smurf/Fraggle Attack Defense Using SACLS. Networking Tips and Newsletters, Oct. 2002. http://searchnetworking.techtarget.com/tip/1,289483,sid7_gci856112,00.html.

[61] J. Mirkovic, G. Prier, and P. Reiher. Source-End DDoS Defense. In *Proceedings of the 2nd IEEE International Symposium on Network Computing and Applications (NCA'03)*, Apr. 2003.

[62] J. Mirkovic, P. Reiher, and G. Prier. A Source Router Approach to DDoS Defense. UCLA CSD Technical Report no. 010042, 2002.

[63] D. Moore, G. Voelker, and S. Savage. Inferring Internet Denial-of-Service Activity. In *Proceedings of 10th USENIX Security Symposium*, Aug. 2001.

[64] Motwani and Rajeev. Randomized algorithms. Cambridge, New York, N.Y.: Cambridge University Press, 1995.

[65] A. Noureldien. Protecting Web Servers from DoS/DDoS Flooding Attacks. In *Proceedings of International Conference on Web-Management for International Organisations*, Oct. 2002.

[66] K. Park and H. Lee. On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack. In *Proceedings of IEEE INFOCOM 2001*, pages 338–347, Apr. 2001.

[67] Z. W. Park, J. H. Lee, and M. K. Kim. Design of an extended TCP for preventing DoS attacks. In *Proceedings of the 7th Korea-Russia International Symposium*, pages 385–389, June 2003.

[68] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. In *Proceedings of the 7th Annual USENIX Security Symposium*, Jan. 1998.

[69] T. Peng, C. Leckie, and K. Ramamohanarao. Adjusted Probabilistic Packet Marking for IP traceback. In *Proceedings of the Second International IFIP-TC6 Networking Conference*, May 2002.

[70] T. Peng, C. Leckie, and K. Ramamohanarao. Proactively Detecting Distributed Denial of Service Attacks Using Source IP Address Monitoring. In *Proceedings of Third International IFIP-TC6 Networking Conference*, pages 771–782, May 2004.

[71] M. Roesch. Snort - lightweight intrusion detection for networks. Addison Wesley Longman, Inc. http://www.snort.org.

[72] S. Savage, D.Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *Proceedings of ACM SIGCOMM 2000*, Aug. 2000.

[73] Y. Sawai, M. Oe, K. Iida, and Y. Kadobayashi. Performance evaluation of inter-domain IP traceback. In *Proceedings of 10th International Conference on Telecommunications*, pages 583–588, Feb. 2003.

[74] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer. Hash-Based IP Traceback. In *Proceedings of ACM SIGCOMM 2001*, Aug. 2001.

[75] D. Song and A. Perrig. Advanced and Authenticated Marking Schemes for IP Traceback. In *Proceedings of IEEE INFOCOM 2001*, 2001.

[76] I. Stoica and H. Zhang. Providing guaranteed services without per flow management. In *Proceedings of ACM SIGCOMM 1999*, pages 81–94, Dec. 1999.

[77] R. Stone. CenterTrack: An IP overlay network for tracking DoS floods. In *Proceedings of 2000 USENIX Security Symposium*, pages 199–212, July 2000.

[78] M. Sung and J. Xu. IP Traceback-Based Intelligent Packet Filtering: A Novel Technique for Defending against Internet DDoS Attacks. *IEEE Transactions on Parallel and Distributed Systems*, 14(9):861–872, Sept. 2003.

[79] K. R. T. Peng, C. Leckie. Protection from Distributed Denial of Service attack using history-based IP filtering. In *Proceedings of IEEE International Conference on Communications (ICC 2003)*, May 2003.

[80] K. R. Tao Peng, Christopher Leckie. Detecting Distributed Denial of Service Attacks Using Source IP Address Monitoring. Technical Report, Department of Electrical and Electronic Engineering, The University of Melbourne, Australia, Nov. 2002.

[81] K. R. Tao Peng, Christopher Leckie. Detecting Distributed Denial of Service Attacks by Sharing Distributed Beliefs. In *Proceedings of Information Security and Privacy, 8th Australasian Conference (ACISP 2003)*, pages 214–225, July 2003.

[82] W. Theilmann and K. Rothermel. Dynamics distance maps of the internet. In *Proceedings of IEEE INFOCOM 2000*, pages 275–284, Mar. 2000.

[83] U. K. Tupakula and V. Varadharajan. A Practical Method to Counteract Denial of Service Attacks. In *Proceedings of the Twenty-Sixth Australasian Computer Science Conference (ACSC2003)*, pages 275–284, Feb. 2003.

[84] H. S. Vaccaro and G. E. Liepins. Detection of anomalous computer session activity. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 280–289, May 1989.

[85] I. Valentine. Open systems and OSI - the issues of convergence. *IEE Colloquium on Message Handling-Past, Present and Future*, pages 2/1–2/6, Nov. 1991.

[86] T. Verwoerd and R. Hunt. Intrusion detection techniques and approaches. *Computer Communications*, 25(15):1356–1365, 2002.

[87] M. Waldvogel. GOSSIB vs. IP Traceback Rumors. In *Proceedings of the 18th Annual Computer Security Applications Conference (ACSAC 2002)*, pages 5–13, 2002.

[88] A. Yaar, A. Perrig, and D. Song. Pi: A Path Identification Mechanism to Defend against DDoS Attacks. In *Proceedings of 2003 IEEE Symposium on Security and Privacy (SP03)*, 2003.

[89] J. Yan. Denial of Service: Another Example. In *Proceedings of the 17th International Conference on Information Security (IFIP/SEC 2002)*, pages 161–170, May 2002.

[90] J. Yan, S. Early, and R. Anderson. The XenoService - A Distributed Defeat for Distributed Denial of Service. In *Proceedings of the CERT Information Survivability Workshop (ISW 2000)*, Oct. 2000.

[91] D. K. Y. Yau, J. C. S. Lui, and F. Liang. Defending Against Distributed Denial-of-service Attacks with Max-min Fair Server-centric Router Throttles. In *Proceedings of the 10th IEEE International Workshop on Quality of Service (IWQoS02)*, pages 35–44, May 2002.

[92] P. Zarkesh-Ha, P. Wright, S. Lakshminarayanan, C. C. Cheng, W. Loh, and W. Lynch. Backend Process Optimization for 90nm High-Density ASIC Chips. In *Proceedings of 2003 IEEE International Interconnect Technology Conference (IITC)*, pages 123–125, June 2003.