2005

Mitigation and traceback countermeasures for DDoS [ie DoS] attacks

Basheer Nayef Al-Duwairi
Iowa State University

Mitigation and traceback countermeasures for DDoS attacks

by

Basheer Nayef Al-Duwairi

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Program of Study Committee:
Manimaran Govindarasu, Co-major Professor
Thomas E. Daniels, Co-major Professor
Ahmed E. Kamal
Douglas W. Jacobson
Wallapak Tavanapong

Iowa State University

Ames, Iowa

2005

Graduate College
Iowa State University

This is to certify that the doctoral dissertation of

Basheer Nayef Al-Duwairi

has met the dissertation requirements of Iowa State University

Signature was redacted for privacy.

Co-major Professor

Signature was redacted for privacy.

Co-major Professor

Signature was redacted for privacy.

For the Major Program

DEDICATION

I dedicate this dissertation with love and admiration to my parents. All that I am and will ever be are due to their constant love and support which has sustained me throughout my life.

ACKNOWLEDGEMENTS

All praise and thanks are due to ALLAH, the most merciful, the most compassionate for his countless blessings on me. There are many people who through their generosity and knowledge have made important contributions to this dissertation. It would be impossible, to list everyone who has contributed or to adequately list the extent of the contributions of those who are mentioned.

First of all, I express my deepest gratitude to my advisors, Dr. Maniamaran Govindarasu and Dr. Thomas E. Daniels, for giving me the opportunity to do my thesis work under their guidance. Dr. Manimaran's unlimited enthusiasm for research and his many side interests have made it a pleasure to work with him. I thank him for his positive attitude, kind patience and trust in this work during my entire Ph.D study. Also, I would like to thank him for teaching me the art of writing scientific articles, proposals, reports and, of course, this dissertation. His constant persistence to reach the highest standards in all aspects of his work has inspired me to work very hard. Dr. Daniels's knowledge and experience in the area of network security had clarified many of the doubts that I had at the early stages of my Ph.D. research. I wish to thank him for taking interest in my work and for many ideas and suggestions he made with respect to the research constituting this dissertation.

I wish to express my cordial thanks to Dr. Ahmed E. Kamal. I am always amazed by his knowledge, preciseness, calmness, and truthfulness. I will never forget his generosity in helping me whenever I needed. I am very pleased to have him on my PoS committee. Also, I would like to express my sincere thanks to Dr. Douglas Jacobson and Dr. Wallapak Tavanapong for dedicating their time to participate in my PoS Committee. Their feedback and comments were very helpful in improving the work presented in this dissertation.

My graduate study has been made possible through the support of the Jordan University of Science and Technology (JUST), the National Science Foundation (NSF), the Litton Professorship, and the Advanced Research and Development Activity (ARDA).

The members of the Real-time Computing and Networking Laboratory at ISU were very helpful and supportive. They always provided constructive suggestions, creative solutions, and lively discussions. I would like to especially thank Mohammad Fraiwan and Muthusrinivasan Muthupras for patiently reading initial drafts of my dissertation and papers, finding mistakes and making corrections. Also, I would like to thank Dr. Anirban Chakrabarti, Suzhen Lin, Anil Kumar, and Durga Kocherlakota for actively participating in many of my lengthy and probably boring presentation rehearsals.

I am proud to have many friends from the Arab and Muslim community of Ames. I am grateful to all of them for making me feel at home, and for providing help and advice.

Finally, I owe my warm gratitude to my parents, brothers, and sisters for their love, support, and practical help throughout my endless years of study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

THESIS SUMMARY

This dissertation provides a comprehensive understanding of the problem of DoS attacks and the various attack mechanisms. It also classifies the known DoS countermeasures into three main categories, namely, prevention, mitigation, and traceback. In this context, the dissertation addresses two challenging problems: (1) the problem of distinguishing attack packets from legitimate packets for the purpose of DoS mitigation and (2) the problem of identifying the paths traversed by attack packets (DoS traceback). Specifically, the dissertation first develops efficient DoS mitigation schemes that can be performed at the perimeter of an ISP network by using suitable packet classification techniques. Then, it develops efficient traceback schemes based on a novel integration of packet marking and packet logging concepts. Finally, the dissertation identifies several future research issues to counter current and emerging DoS attacks effectively.

ABSTRACT

Denial of service (DoS) attacks impose an imminent threat to the availability of Internet services. The alarming increase of such attacks coupled with the emergence of sophisticated DoS attack techniques, call for efficient defense mechanisms to counter these attacks. Although there has been ongoing research in this area – focusing on DoS prevention, mitigation, and traceback – the existing countermeasures lack in several qualitative and quantitative metrics. In this context, this dissertation makes two key contributions in the design and analysis of efficient, scalable schemes for DoS mitigation and traceback.

First, efficient perimeter mitigation schemes based on novel concepts, such as "protocol-determinism" and "victim-assistance" are proposed. The proposed schemes enable ISP edge routers to perform timely mitigation of both end-host and network exhaustion attacks. The proposed mitigation schemes have been evaluated through analytical studies for classical and advanced attacks quantifying security metrics, such as false positive and false negative rates, and performance metrics, such as effective attack rate and connection establishment latency increase. Our analysis shows that the proposed schemes offer very low false positive and false negative rates, and reduce attacker's effective attack rate significantly with an acceptable increase in connection establishment latency.

Second, hybrid IP traceback schemes that integrate the concepts of packet marking and packet logging in a novel manner are proposed. The goal is to achieve a drastic reduction in the number of attack packets required to conduct the traceback process. The proposed traceback schemes have been evaluated through a combination of analytical and simulation studies quantifying performance metrics, such as number of attack packets, storage overhead, and attack localization distance. Our studies show that the proposed traceback schemes are

superior in comparison to the well known PPM scheme.

This dissertation opens up several directions for future research which includes (1) designing efficient mitigation schemes in the context of inter-domain network, (2) designing efficient mitigation schemes that employ traceback, and (3) designing a comprehensive DoS defense mechanism that integrates DoS prevention, mitigation, and traceback in an efficient manner.

CHAPTER 1. DENIAL OF SERVICE ATTACKS

The phenomenal growth of the Internet, and its entry into many aspects of daily life, has led to the dependency on its services very often. Unfortunately, it has been shown [1]-[5] that it is easy to disturb the functionality of the Internet by attacking its infrastructure taking advantage of the vulnerable Internet elements and protocols. Denial of Service (DoS[1]) attacks are among the top threats to the Internet infrastructure [6]. These attacks can quickly incapacitate a targeted business, costing victims thousands, if not millions, of dollars in lost revenue and productivity. The objective of this dissertation is to develop efficient countermeasures for these attacks. The dissertation outline is as follows. In this chapter, we devise a classification of DoS attacks. In Chapter 2, we survey the main research efforts to counter DoS attacks. In Chapter 3, we present the proposed DoS mitigation schemes. In Chapter 4, we present the proposed IP traceback schemes. Finally, we present the conclusion and the future work in Chapter 5.

In this chapter, we discuss the problem of DoS attacks focusing on their original causes, and on their various mechanisms. Also, we discuss the research challenges associated with this problem. Finally, we describe the organization of this dissertation.

1.1 The Increasing Threat of DoS Attacks

DoS attacks are malicious means of denying Internet services to legitimate users or processes. These attacks continue to pose a significant threat for today's Internet as they are growing in number and sophistication. The recent attack incidents confirm the devastating effects of these attacks. For example, in October 2002 eight out of the thirteen root servers

[1]Throughout this dissertation, the term DoS attacks is used to refer to multiple source attacks (i.e., distributed denial of service) as well as single source attacks.

were significantly affected by a massive DoS attack [7]. Many other attacks were reported subsequently in 2003 and 2004 (e.g., [8, 9]). In a recent study by D. Moore et. al., [10], the DoS activity in the Internet was estimated using a method called "backscatter". The study found that nearly 4,000 DoS attacks are launched each week. Some systems studied were attacked as often as once per minute, usually with attacks of up to 1,000 packets per second, though some attacks ran as much as 600,000 packets per second.

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

Although any system connected to the Internet is considered to be a potential target, most of the reported DoS attacks are against high profile sites that include online gaming sites, Web hosting and to some extent against financial services. The outages caused by these attacks usually lead to significant financial losses. The motive behind launching DoS attacks varies across a spectrum of reasons including economical, political, and personal. Despite the research efforts done to counter DoS attacks, the fear that future attacks could do even worse damage still exists. Therefore, it is evident that a lot of issues need to be addressed, and a more effective countermeasures need to be developed. In this context, the research presented in this dissertation provides effective countermeasures for a class of DoS attacks that represent an imminent threat.

1.2 Roots of the DoS Problem

Looking back to the evolution of the Internet, one can explain how the Internet has become ubiquitous. Most of the R&D efforts in the past had been on improving the performance

and scalability of Internet protocols. However, little attention was paid to securing these protocols. This resulted in several vulnerabilities that were exploited to adversary's advantage. Adversaries have always abused the following characteristics of Internet protocols to perform DoS attacks.

- *Destination oriented routing*: The routing protocols were designed to be destination oriented. Packet forwarding is the main task of Internet routers. A router places a packet on the interface that takes it to the next hop on its way to the destination, without bothering about where it came from.

- *Stateless nature of the Internet*: Routers do not maintain any state information about forwarded packets. Because of this, accountability and computer forensics are difficult to conduct.

- *Lack of authenticity over the Internet*: Without authentication, malicious Internet users can claim the identities of other users without being easily detected or located.

- *Deterministic nature of Internet protocols*: This is not a design flaw, but is often necessary to the proper operation of Internet protocols (e.g., TCP connection establishment procedure and TCP congestion control mechanisms).

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

1.3 Classification of DoS Attacks

DoS attacks are emerging in various forms and becoming more sophisticated. The scope of these attacks spread over wide range of Internet protocols to prevent users from gaining access to a spectrum of Internet services. Generally, the various attack types share a common nature in the sense that they rely on the deterministic behavior of most Internet protocols and the lack of authentication over the Internet. To launch a powerful DoS attack, an attacker has to secure enough resources to achieve the desired damage to the victim. A single attacker could mobilize thousands of compromised computers (known also as zombies or slaves) from unsuspected users. Compromising this many computers is done in a phase, known as the recruitment phase, that precedes the actual DoS attack. The systematic way of attack recruitment phase includes gaining remote unauthorized access to large number of computers. The basic steps of the recruitment phase are outlined as follows.

- The attacker performs extensive scanning of remote machines searching for vulnerabilities and security holes.
- The discovered vulnerabilities are exploited to break into the scanned systems. At this point, the attacker gets access to these systems, which are then called zombies or slaves.
- The attacker installs the attack tool on the compromised computers. At this point, the compromised computers become ready to participate in the attack, or even to be used in the recruitment of other computers.

It is also possible to perform attack recruitment via spreading of viruses and Trojan horses. In this chapter, we assume that attackers have the ability to compromise and recruit sufficient number of end systems to perform an attack. However, the actual methods used to scan remote machines, exploit their vulnerabilities, and compromise them are not discussed. We refer the reader to [11] for a detailed discussion about this topic. Our focus will be mainly on DoS attack characterization and classification. We follow a systematic approach to devise an attribute-based characterization of DoS attacks. Also, we provide two alternative classifications of DoS

attacks in order to reflect how these attacks can be viewed from different angles. The first classification is based on the means of DoS attacks, while the second classification is based on the impact of DoS attacks.

1.3.1 Attribute-based Characterization of DoS Attacks

Before launching a DoS attack, an attacker should configure the attack tool in such a way as to achieve the desired damage to the victim. This involves the specification of several attack attributes that shape the overall nature of the attack. The term "attribute" here refers to certain aspect of an attack. In this subsection, we specify the following attributes that apply for a variety of DoS attacks: (1) header spoofing, (2) attack indirection, (3) attack amplification factor, (4) attack rate dynamics, (5) number of attackers vs. number of victims, and (6) attacker's reaction to the victim's defense. These attributes provide a systematic way for characterizing DoS attacks. Under this approach, an instance of DoS attack (DoS-inst) would be characterized using the corresponding values of the attributes of that instance as follows: DoS-inst$(A_1, A_2, ...A_n)$. Where, A_i is the i-th attribute of the attack. Figure 1.1 shows a summary of DoS attack attributes and their classifications. It is to be noted that these attributes are common to most of the DoS attacks.

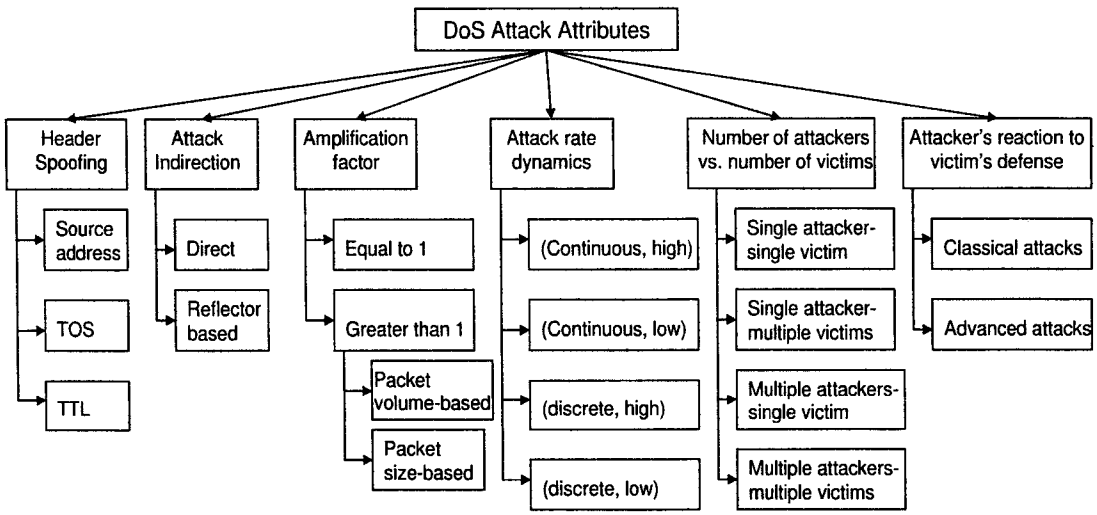


Figure 1.1 Summary of DoS attack attributes.

1.3.1.1 Attribute 1: Header Spoofing

Spoofing the content of TCP/IP header is a common practice in most DoS attacks. While this practice is essential in certain attacks (e.g., reflector-based DoS attacks [12]), it is usually employed by attackers to gain additional advantages such as confusing the victim in a way that reduces its ability to characterize attack traffic. Also, forensic investigation becomes much harder when attackers employ header spoofing. In this context, many fields in the TCP/IP header can be spoofed. This include the source address field, Type of Service (TOS) field, and the Time To Live (TTL) filed. While spoofing of the source address filed is the most common, spoofing of other fields (i.e., TOS and TTL) is useful in certain attack scenarios. For example, TOS field spoofing can be used in stealing network resources in a DiffServ domain [13]. Spoofing a packet's TTL field means that the attacker will set the packet's initial TTL to a value different than the default TTL value used by the operating system of the attack machine[2]. This has the impact of disguising the actual distance between the source of the attack and the destination (i.e., the victim). TTL spoofing makes DoS defense much harder especially if the victim uses TTL field to characterize incoming attack traffic, or when performing packet filtering based on the distance traveled by packets as in the hop count filtering scheme [15].

1.3.1.2 Attribute 2: Attack Indirection

Attack indirection refers to the flow direction of DoS attack traffic from attack machines (the zombies) to their victim. There are two major types of DoS attack indirection. The first type is *direct*, where attackers employ a simple concept: sending bogus packets directly from a remote node, under attackers control, to a given target. Attackers usually gain access to as many as thousands of vulnerable zombie computers in order to magnify and direct their full-scale assault against a single victim from all directions. Fig. 1.2 illustrates the basic steps to generate a direct DoS attack.

The second type, called *reflector-based*, is more complicated as it introduces one more level in the attack structure. Instead of flooding the target directly, the compromised machines

[2]According to [14], most modern operating systems use only a few selected initial TTL values, 30, 32, 60, 64, 128, and 255.

(zombies) are instructed to continuously send request packets to a set of Internet reflectors (an Internet reflector is an IP host that will reply to any request packet). The source address of each of these request packets is spoofed to be the address of the targeted machine. As a result, the reflectors send their replies to the given target address causing packet flooding at that machine. It is to be noted that source address spoofing is essential in this type of attack indirection. Fig. 1.3 shows the basic steps to conduct a reflector-based DoS attack.

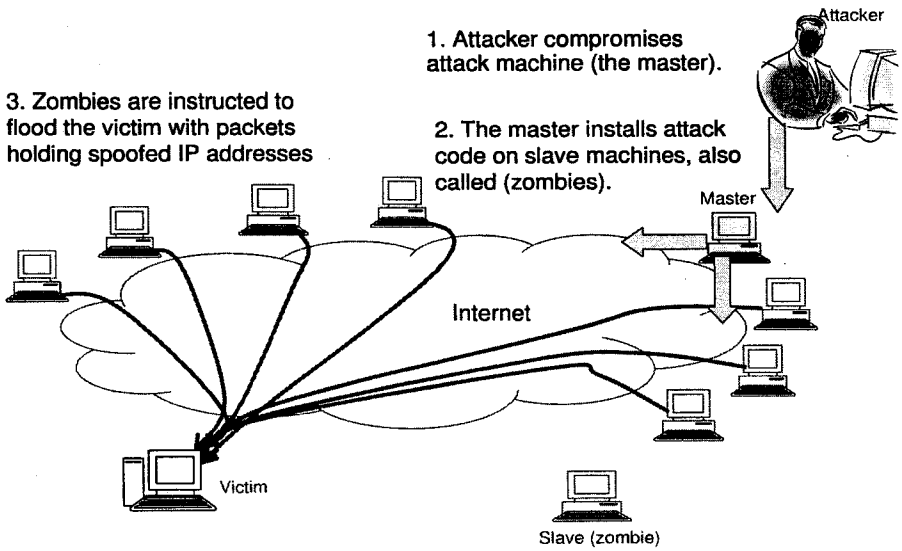


Figure 1.2 Direct DoS attack scenario.

1.3.1.3 Attribute 3: Attack Amplification Factor

Attack amplification refers to the amount of gain in resource (e.g., bandwidth) an attacker achieves for each emitted attack packet. If the attacker emits an attack packet of size x, for which the victim receives an amount of traffic of size y, then we say that the amplification factor for this attack is $f = \frac{y}{x}$. Most of direct DoS attacks have an amplification factor of 1. This is because each attack packet reaches the victim without any increase in its size. However, in reflector-based DoS attacks, an amplification factor of more than one is usually noticeable. In this context, there are two main types of attack amplification. The first is *number-based* amplification. The second is *packet size-based* amplification. Smurf attack [16] is a typical example of number-based amplification reflector DoS attack. The attacker sends an ICMP

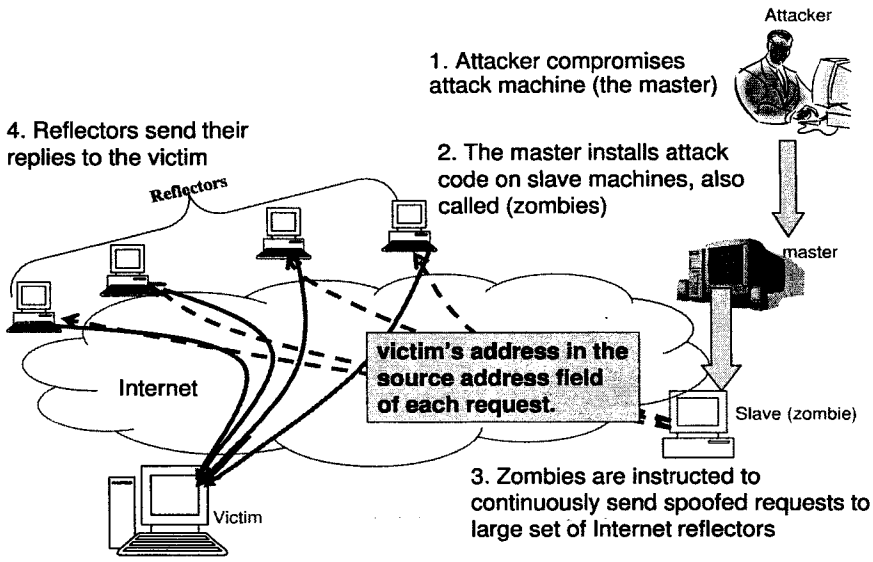


Figure 1.3 Reflector-based DoS attack scenario.

echo request (ping) to a broadcast address. The source address of the echo request is the IP address of the victim (uses the IP address of the victim as the return address). After receiving the echo request, all the machines in the broadcast domain send echo replies (responses) to the victims IP. Victim will crash or freeze when receiving larger-sized packet flood from many machines.

DNS-based reflector DoS attack [17] is a typical example of packet size-based amplification. In this attack, a large number of UDP-based DNS requests are sent to a name server using a spoofed source IP address that is identical to the victim's IP address. Any name server response is sent back to the spoofed IP address as the destination. Because name server responses can be significantly larger than DNS requests, there is a potential for bandwidth amplification. In other words, the responses may consume more bandwidth than the requests. It is obvious, that the attack amplification is due to the increase of the packet size rather than due to the increase in the number of packets volume. Hence, the name packet size-based amplification is used to describe such attacks.

1.3.1.4 Attribute 4: Attack Rate Dynamics

An important attribute of DoS attacks is the rate of arrival of attack packets at the victim. Setting a pattern and a value for the attack rate is a basic step in configuring attack tools. Generally, attack rate can be described using the ordered pair (continuity, intensity). Continuity specifies whether the attack is continuous or discrete, while intensity specifies the severity of the attack (e.g., high, low, etc.). Generally, the amount of damage at the victim is proportional to the attack rate. Higher the attack rate, higher the damage. However, there is an inherent tradeoff from the perspective of an attacker between the attack rate and the ability to elude attack detection. For example, continuous high rate attack has severe impact on the victim, but, it is very easy to detect. While continuous low rate attack has less impact on the victim, but, it is more difficult to detect. For this reason, attackers tend to manipulate attack rate dynamics in a manner that causes significant damage at the victim, while making attack detection much harder.

1.3.1.5 Attribute 5: Number of Attackers vs. Number of Victims

DoS attacks can be classified as follows based on the number of attackers and the number of victims. C1: Single_attacker Single_victim, C2: Single_attacker Multiple_victims, C3: Multiple_attackers Single_victim, and C4: Multiple_attackers Multiple_victims. Among these, C3 is the most common and the worst one from the perspective of an individual victim because the intensity of the attack is expected to be very high. Usually, victims tend to tolerate DoS attacks with low intensity as in C1 and C2 because the overhead associated with mitigating such attacks is very high compared to the benefit of mitigation. However, mitigating high intensity attacks, such as in C3 and C4 is extremely important for each victim.

1.3.1.6 Attribute 6: Attacker's Reaction to Victim's Defense

Current DoS attack tools are automated and pre-configured by attackers ahead of time before an attack onset. In most cases, the attacker does not react to defenses employed by the victim during attack period. We refer to such attacks as *classical attacks*. However, it is

expected to see more sophisticated attacks, in which the attacker monitors victim's reaction and acts accordingly. We call such attacks *advanced attacks*. In fact, advanced attacks impose a real threat, because traditional countermeasures can be bypassed by attackers. A recent study [18] showed that advanced attacks have a significant impact on statistical packet filters. We believe that attacker's awareness is going to be an important attribute in future DoS attack tools.

1.3.2 Means-based Classification of DoS Attacks

In this subsection, we devise a classification of DoS attacks from attacker's viewpoint. This classification takes into consideration the the means of performing a DoS attack. In this context, we classify DoS attacks into two main categories: (1) Brute force-based attacks, and (2) Protocol exploitation-based attacks. Attacks in the first category adopt the idea of brute force resource exhaustion to achieve their goal. Attacks in the second category adopt the idea of exploiting the deterministic nature of certain Internet protocols to significantly degrade their throughput without injecting a lot of traffic in the Internet. A classification of DoS attacks in each category is shown in Figure 1.4. In the following discussion, we provide detailed description of these attacks.

1.3.2.1 Brute Force-based DoS Attacks

The main objective of brute force-based DoS attacks is to overpower the victim while concealing attacker's identity. This goal is achieved by overwhelming the victim by an amount of traffic, requests, or computational tasks, that are much larger than what can be handled by the victim. The premise here is that overwhelming the victim by an intensive load will render it unavailable for legitimate users. The targeted resource could be victim's buffer space, bandwidth, CPU cycles, or a combination of them. These attacks are further classified based on the location of the targeted resource into the following types.

- *End system brute force-based DoS attacks:* The aim of these attacks is to occupy a disproportional amount of victim's resources for maximum amount of time. Attacks of this type

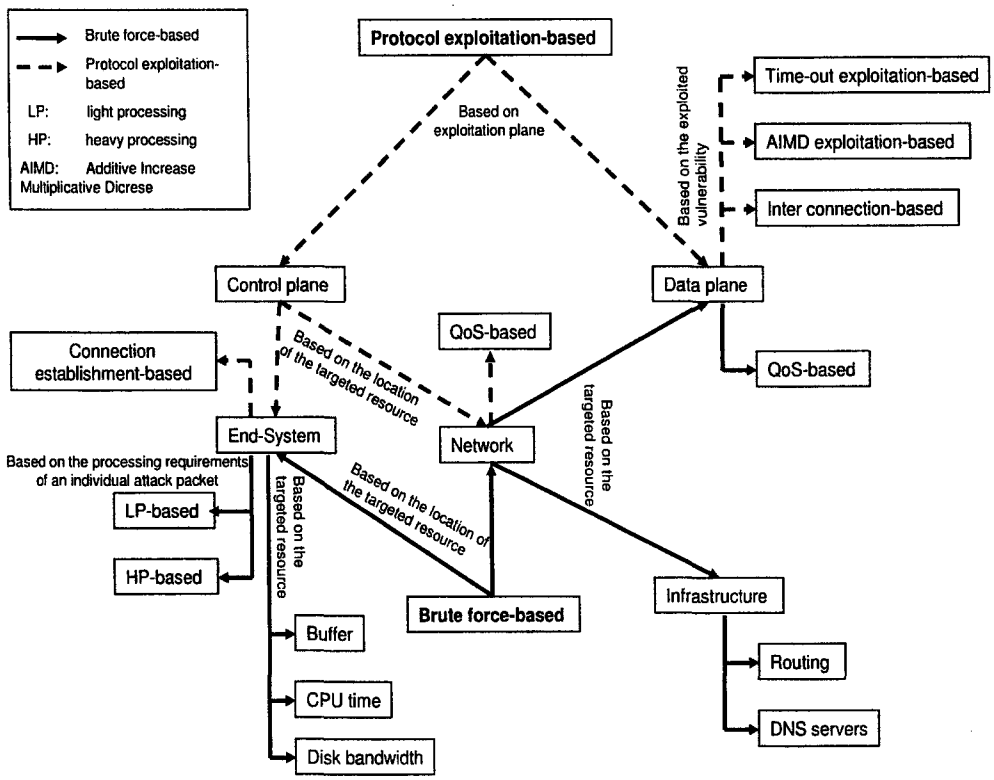


Figure 1.4 Means-based classification of DoS attacks.

are classified into *light processing-based* (LP-based) attacks, and *heavy processing-based* (HP-based) attacks. This classification is based on the load imposed on the system by each attack packet (or request) *individually*. LP-based attacks are usually characterized by a very intensive attack rate that brings the total load beyond the victim's capacity. The targeted resources in this attack include victim's bandwidth, buffers, memory, and CPU cycles. Flooding based attacks, such as SYN flooding [19], and smurf [16] which is an ICMP based flooding attack, belong to this category. On the other hand, HP-based attacks are usually characterized by submitting a large number of heavy processing tasks to the victim. A heavy processing task is basically a computationally intensive task. For example, it can be in the form of an authentication process, or it can be in the form of downloading huge files from a Web or FTP server in overwhelming numbers. Such attacks target higher layer resources such as CPU cycles, disk bandwidth, and database bandwidth. According to [20], these attacks profile the victim and mimic legitimate Web

browsing behavior of a large number of clients. These attacks are hard to defend against as the malicious requests differ from the legitimate ones in intent but not in content.

- *Network brute force-based DoS attacks:* In addition to targeting critical Internet *infrastructure elements*, such as DNS root servers and BGP routers, the primary target of network brute force-based DoS attacks are applications with sensitive timing constraints, such as multimedia applications and real-time communication services, in general. There have been several proposals to provide real-time support for such applications through network resource reservation. Differentiated services (DiffServ) architecture [21] is a typical example. Network brute force-based DoS attacks that target a DiffServ domain are also known as *Quality of Service (QoS)-based attacks* [13; 22, 23]. These attacks aim at stealing or exhausting DiffServ domain resources such as bandwidth and router queues in order to degrade the service perceived by subscribed multimedia applications. For example, an adversary within the same stub network as that of a multimedia server, or at a neighboring network that is connected with the same ISP (Internet Service Provider), can passively monitor the ongoing traffic toward ISP edge routers. The adversary can then impersonate as legitimate sources by flooding spoofed data packets that have the same IP header as valid data packets (including Type of Service field). The packet filters that work based on packet header information only, cannot screen out these spoofed packets. As a result of traffic shaping at the network egress, legitimate packets maybe dropped blindly, leading to significant service degradation.

It is worth mentioning that the majority of the reported DoS attacks are based on a brute-force approach. These attacks are proven to be very effective and powerful. However, most of them can be easily detected, which is viewed as a weakness from the attacker's perspective, because attack detection is usually followed by activation of a certain countermeasure whenever possible. This issue has led researchers to believe that attackers will soon follow more efficient approaches in performing DoS attacks without being easily detected. The following subsection, describes several approaches that are more likely to be used by attackers in the near future.

1.3.2.2 Protocol Exploitation-based DoS Attacks

The deterministic nature of Internet protocols opens the doors for attackers to tune their attack traffic dynamics and parameters in a manner that abuses Internet protocols and causes significant throughput degradation, if not complete denial of service, to legitimate users. Protocol exploitation-based DoS attacks are mostly feasible in the TCP and the 802.11 MAC protocols as well as in QoS networks. The protocol exploitation and abuse in these protocols can take place either in the *control* plane, or in the *data* plane. The following discussion provides a classification of DoS attacks in both planes.

Control plane protocol exploitation-based DoS attacks: Lack of authenticity of control messages in many Internet protocols combined with the predictable reaction of communicating parties, enable attackers to take over valuable end system and network resources with minimum effort. Therefore, denying service to legitimate users. In this context, we discuss the following two control plane-based DoS attacks.

- *TCP's connection establishment procedure-based DoS attack:* TCP uses a three-way handshaking procedure to set-up a connection, it works as follows.

  1. A connection is set up by the initiating side sending a segment with the SYN flag set and the proposed initial sequence number in the sequence number field (ISN = x).
  2. The receiver then returns a segment with both the SYN and ACK flags set with the sequence number field set to its own assigned value for the reverse direction (ISN =y) and acknowledge field of $x + 1$ (ack = $x + 1$).
  3. On receipt of this, the initiating side records y and returns a segment with just the ACK flag set and an acknowledgment field of $y + 1$.

  A TCP listen port has a finite number of slots in its listen queue and normally this number is relatively small. When an attacker sends enough faked SYN packets, the listen queue can be fully occupied and subsequently deny any legitimate SYN packet from

entering into the listen queue. Therefore, until the connection establishment process times out, a disproportional amount of system resources are occupied: a slot in the attacked port's listen queue, memory to maintain connection information, and CPU and network bandwidth to retransmit the SYN-ACK packet. Such attacks are also known as SYN flooding attacks [19].

- *Resource reservation procedure-based DoS attack:* To provide QoS guarantees, the Resource Reservation Protocol (RSVP) [24] has been proposed. This protocol is basically a signaling technique used to reserve bandwidth on each router for a particular flow along a given data path. The routers along the path must support RSVP functionality. The lack of authenticity in RSVP makes it a tempting target of DoS attacks. In such an environment, an intruder can steal network bandwidth by initiating bogus reservation messages. Therefore, leaving little resources for genuine reservation requests. Such attacks are also known as QoS-based attacks.

Data plane protocol exploitation-based DoS attacks: Both of TCP and 802.11 MAC protocols were designed to adapt to abnormal network conditions such as congestion and collisions. This adaptation behavior is always necessary to achieve a stable and an efficient utilization of network bandwidth given that the Internet hosts adhere to the rules of the adaptation mechanism. Recent research studies, in wired networks (e.g., [25, 26, 27]) and in wireless networks (e.g., [28, 29, 30]), have shown that protocol adaptation mechanisms can be exploited in several ways so as to create service outages and a significant throughput degradation. The main objective of protocol exploitation-based DoS attacks is to achieve a severe throughput degradation without using the conventional brute force packet flooding approach. Therefore, avoiding detection by affected systems. The following discussion is mainly focused on attacks that are based on exploiting the TCP protocol[3].

TCP protocol is characterized by its ability to adapt to network conditions in a manner that leads to congestion avoidance. This is achieved by using congestion window at the sender

[3]To the best of our knowledge, none of these TCP exploitation-based approaches have been used yet systemically by attackers in today's Internet.

side. At any given time, the congestion window indicates the maximum amount of data that can be sent out on a connection without being acknowledged. Starting by a congestion window of size one, TCP enters *slow start* where congestion window is incremented by one for each acknowledgment of a new data. TCP remains in the slow start phase as long as no congestion is detected. TCP detects congestion when it fails to receive an acknowledgment for a packet within the estimated timeout, or upon the receipt of three or more duplicate ACKs indicating a data segment is lost. If packet loss occurs and three duplicate ACKs are received, then TCP enters fast recovery. However, if packet loss occurs and less than three duplicate ACKs (DACKs) are received, TCP, waits for a period of retransmission timeout to expire, sets the slow start threshold to half of the current congestion window, reduces its congestion window to one packet and resends the packet, performs slow start to the new threshold and then enters *congestion avoidance* phase, where additive increase multiplicative decrease (AIMD) is performed.

Recent studies showed that the TCP congestion control mechanism is vulnerable to several types of attacks. These attacks exploit different stages of the congestion control mechanism. The following discussion focuses on attacks that are based on exploiting the TCP's time out mechanism, the AIMD behavior in the congestion avoidance phase, and the rate of bandwidth capture at different stages of the TCP's congestion control operation, referred to as inter connection-based DoS attack.

- *Time out exploitation-based DoS attacks:* It has been shown in [25] that an attacker can force TCP connections passing through a given link or router to time out repeatedly, Thereby degrading their throughput. To achieve this goal, an attacker targets that link (or router) using short duration traffic bursts having RTT-scale burst length, and repeating the burst periodically at slower RTO (Retransmission Time Out) timescales. Such timely traffic bursts forces the targeted TCP connections to time out. Most of the modern TCP implementations have the base RTO = 1 sec, which means that all affected TCP connections will initiate retransmission after 1 second since the first attack burst is observed. The attacker waits approximately for 1 second before injecting the second

burst to ensure that retransmitted packets will be dropped as well. The same process repeats in a periodic fashion. The targeted TCP connections will be throttled to near-zero throughput while the attacker will have low average attack rate, making it difficult to detect.

- *AIMD exploitation-based DoS attacks:* This attack aims at forcing the targeted TCP connection (at the sender side) to frequently enter the fast recovery state. Different than the time out-based attack, an AIMD-based attack induces traffic bursts in such a way that some packet loss in the targeted TCP connection occurs, but a sufficient number of DACKs can still be received by the sender. This condition is to ensure that the connection will enter fast recovery state rather than entering time out. This attack becomes eventually similar to the time out-based attack if the congestion window drops below a certain level, because in such case there may not be enough DACKs to trigger fast recovery mechanism. The work presented in [26] describes two versions of this attack, one is synchronous in the sense that attack epochs always coincide with a fixed set of congestion window values. The second is asynchronous in the sense that attack pulses are generated with a fixed period, and they do not necessarily coincide with a pre-specified congestion window values.

- *Inter connection-based DoS attacks:* The fact that TCP's congestion window evolves exponentially in the slow start phase, while it evolves linearly in the congestion avoidance phase indicates that TCP flows in slow start rapidly secure greater proportion of bandwidth as compared to TCP flows who are in the congestion avoidance phase. Since short-lived TCP flows spend their lifetime in slow start phase, while long-lived TCP flows spend most of their lifetime in the congestion avoidance phase, it is possible to use short-lived TCP flows to attack long-lived TCP flows. The work presented in [27] was the first to show scenarios in which short-lived flows attack long-lived flows so as to drastically impact their performance. The authors in [27] used a combination of simulations, analysis, and experiments to study the dependence of the severity of impact on ling-lived flows on key parameters of short-lived flows.

1.3.3 Impact-based Classification of DoS Attacks

In this subsection, we devise an impact-based classification of DoS attacks. This classification takes into consideration the attack impact on the victim. The impact of an attack refers to the aftermath of an attack on the targeted resource. In this context, DoS attacks can be classified into two main categories: (1) Resource exhaustion-based attacks. (2) Resource stealing-based attacks. This classification is inline with the primary objective of any DoS attack- that is to render the targeted resource unavailable. Our intention here is to put the attacks described in subsection 1.3.2 in a new skeleton that reflects this view. Therefore, the discussion provided about these attacks will be brief. Figure 1.5 summarizes the attacks in each category.

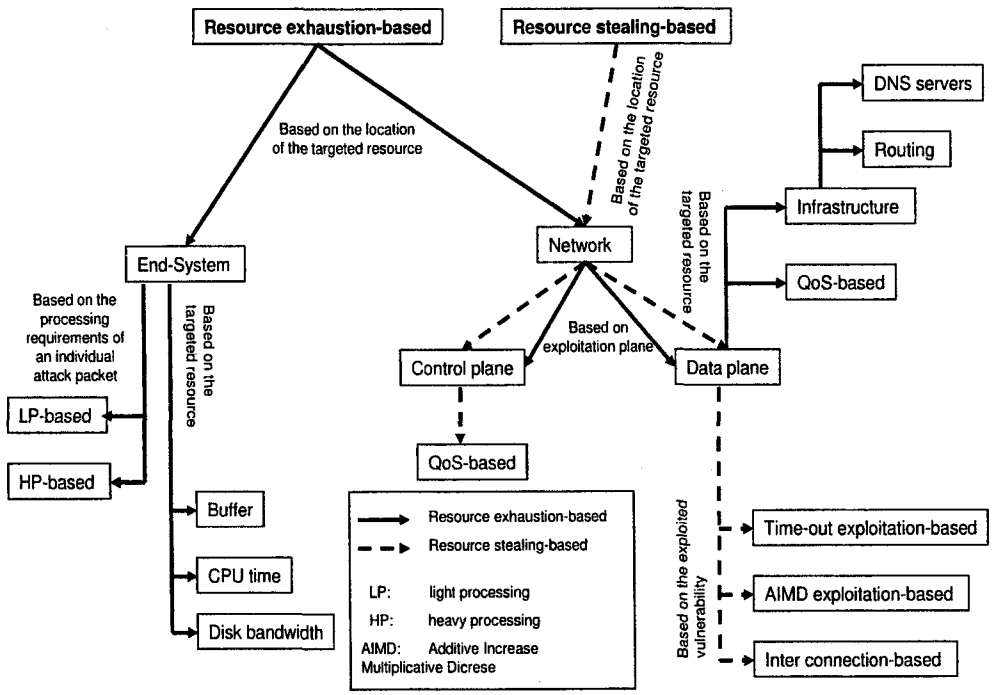


Figure 1.5 Impact-based classification of DoS attacks.

- *Resource exhaustion-based DoS attacks:* Attacks in this category target end system resources as well as network resources. End system resource exhaustion-based attacks include LP-based attacks, HP-based attacks, TCP's connection establishment exploitation-

based attack, with the targeted resources are mainly buffers, memory, CPU cycles, and disk bandwidth. On the other hand, network resource exhaustion-based attacks include attacks against infrastructure elements such as DNS servers and routing protocols. Also, it includes data plane QoS-based attacks, where attackers exhaust the resources available for genuine users.

- *Resource stealing-based DoS attacks:* Attacks in this category target network resources mainly, aiming at stealing network resources from legitimate users either by capturing these resources or by creating the illusion to legitimate users that the requested resources are not available. Resource capturing-based attacks include the control plane QoS-based attacks, in which an attacker uses bogus reservation messages to block significant amount of network resources from valid users, and the inter connection-based DoS attacks, in which an attacker uses short lived-TCP flows to capture more network resources as compared to long-lived TCP flows. Resource unavailable-based attacks include the time out-based attacks, and AIMD-based attacks. These attacks force legitimate TCP clients to back off repeatedly thinking that the available network resources are not sufficient to accommodate them.

1.4 Research Challenges in Countering DoS Attacks

Dealing with DoS attacks has proven to be extremely difficult. The following are the main challenges that face researchers in this field.

- *Distinction between attack traffic and legitimate traffic before reaching the target:* Malicious packets differ from the legitimate ones in intent not in content. This is due to the fact that attackers use the same Internet protocols used by legitimate users and generate packets that appears to be genuine. Therefore, attack traffic characterization is a very challenging problem. DoS countermeasures should perform accurate traffic characterization. Otherwise, legitimate traffic can be affected considerably.

- *Deployment of DoS countermeasures:* Coming up with DoS countermeasures[4] that are transparent to existing Internet protocols represents a major challenge. Most of the known DoS countermeasures require either a network support, an end-system support, or a protocol modification. Such requirements may stand against deploying these countermeasures, because it is very difficult to enforce deployment of DoS countermeasures in an uncontrolled domain such as the Internet.

- *Handling huge volume of attack traffic at high Internet speeds:* As discussed in Section 1.3, most of the reported DoS attacks are based on brute force approach, where overwhelming number of packets overload the victim. On the other hand, DoS countermeasures usually involve some kind of packet processing such as marking, validation, storage, filtering, etc. Given the fact that DoS attacks are characterized by huge packet volumes, performing these tasks at very high Internet speeds is a major challenge.

- *Detection and handling of sophisticated DoS attacks:* New generation of sophisticated DoS attacks are emerging in different forms as explained in Section 1.3. These attacks can lead to degradation of service rather than complete blocking of service. Moreover, these attacks use much lower amount of traffic as compared to the flooding-based attacks. Therefore, conventional DoS detection methods will not be able to identify the existence of an attack. Developing fast and accurate detection schemes is an important issue in this context. Also, it is equally important to develop effective prevention and mitigation schemes to handle these attacks.

- *Determining the number of attackers:* Since attackers employ source address spoofing widely, it is difficult to know if the attack traffic is originating from a single source or from multiple sources, and if it is originating from multiple sources, then what is the actual number of these sources. Knowing the number of attack sources may lead to strategic defense decisions in certain cases.

[4]DoS countermeasures are discussed in Chapter 2.

1.5 Dissertation Organization

The rest of this dissertation is organized as follows.

- In Chapter 2, we provide an overview of the state of the art research in countering DoS attacks. The chapter's focus will be on prevention, mitigation, and traceback as the main countermeasures of DoS attacks. Also, we motivate for the work proposed in this dissertation, and we list its main contributions.

- In Chapter 3, novel concepts for DoS mitigation are proposed namely, victim-assistance and protocol-determinism, respectively. The proposed concepts are used in the context of a perimeter based DoS mitigation architecture as complementary methods for edge routers to perform online attack packet identification. The proposed mitigation schemes have been evaluated through analytical studies for classical and advanced attacks quantifying security metrics, such as false positive and false negative rates, and performance metrics, such as effective attack rate and connection establishment latency increase.

- In Chapter 4, hybrid IP traceback schemes that integrate the concepts of packet marking and packet logging in a novel manner are proposed. The goal is to achieve a drastic reduction in the number of attack packets required to conduct the traceback process. The proposed traceback schemes have been evaluated through a combination of analytical and simulation studies quantifying performance metrics, such as number of attack packets, storage overhead, and attack localization distance.

- In Chapter 5, some concluding remarks are made and potential future research directions are identified.

CHAPTER 2. STATE OF THE ART IN DoS COUNTERMEASURES

The alarming increase in the number of DoS attacks against e-commerce companies and other organizations, the emergence of newly sophisticated attacks, and the growing fear from potential powerful coordinated attacks have led to a significant amount of research in recent years aiming at countering these attacks on all fronts. However, by considering the different ways by which attacks can be conducted (as explained in Chapter 1), it can be seen that winning the battle against attackers is not easy at all. In fact, the research done so far in this field provides partial solutions to the problem, or in some cases, solutions to specific instances of DoS attacks rather than providing a comprehensive solution.

This chapter presents the state of the art research in countering DoS attacks, focusing mainly on the major defense mechanisms, their pros and cons, and their applicability to known DoS attacks. Before going into the details of DoS countermeasures, we provide the following discussion about the main criteria that are typically used for qualitative evaluation of these countermeasures.

- *Proactiveness:* DoS countermeasures can be either proactive or reactive in nature. Proactive countermeasures are those who work all the time. While reactive countermeasures are those who are activated only during an attack period. Obviously, there are several implications regarding a scheme being proactive or reactive. Generally, proactive schemes are more effective than reactive schemes because they start defense at an early stage of an attack. However, this comes at an additional cost represented by an additional overhead and a need for a wide scale deployment.

- *Effectiveness:* The performance of a given DoS countermeasure is usually evaluated through several metrics (e.g., false positive rate, false negative rate, processing over-

head, storage overhead, communication overhead, etc.). The effectiveness of a scheme is determined by taking all performance metrics into account.

- *Generality:* A challenging issue in countering DoS attacks is that these attacks vary and change from time to time. Typically, DoS countermeasures should be *generic* in the sense that they should be applicable to a wide range of attacks. Although this feature is preferred whenever possible, it is not always necessary because each attack instance involves limited number of protocols, and because generic mitigation schemes are usually expensive and difficult to implement.

- *Deployability:* Deployment of DoS countermeasures is an important issue that must be considered. A practical DoS countermeasure should be easy to deploy in the sense that it minimally interferes with existing Internet protocols and settings. Also, the scale of deployment should be reasonable. Countermeasures that require *local* deployment are usually preferred over those which require *global* deployment. However, if a DoS countermeasure requires global deployment, then this deployment should be incrementally feasible.

- *Static versus dynamic:* As discussed in Chapter 1, based on attacker's awareness of the defense mechanism, a DoS attack can be classical or it can be advanced. DoS countermeasures should take this issue into account. Static DoS countermeasures are expected to be effective against classical attacks. However, they are expected to fall short in the face of advanced attacks. DoS countermeasures should be dynamic in the sense that they should be able to detect attacker's awareness of the defense scheme in use, and to act accordingly in a dynamic manner.

It is important to mention that the above criteria should be considered collectively when evaluating DoS countermeasures. Obviously, a typical countermeasure would be reactive, effective, generic, locally deployable, and dynamic. In practice, it is acceptable to have countermeasures that meet some of these criteria rather than meeting them all. We employ this evaluation methodology in the following discussion which provides a broad classification of

DoS countermeasures. Based on their primary objectives, DoS countermeasures are classified into three major categories, namely, prevention, mitigation, and traceback. Figure 2.1 shows a classification of DoS countermeasures and summarizes the main approaches in each category.

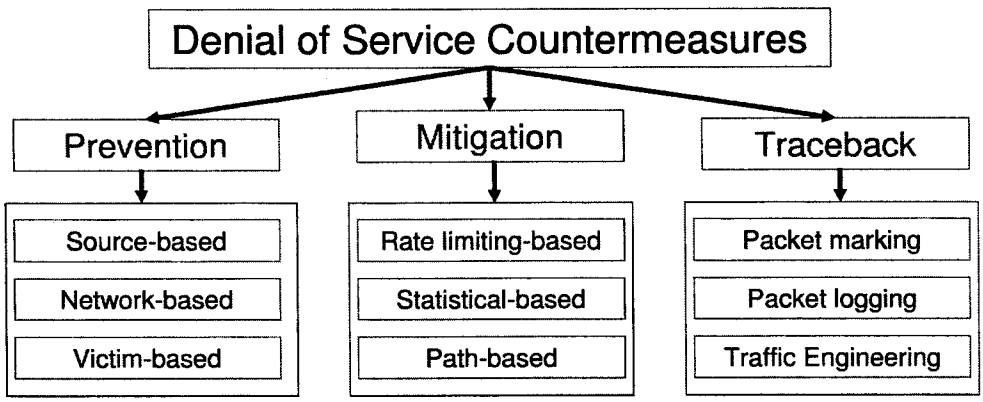


Figure 2.1 Taxonomy of DoS countermeasures.

- *DoS Prevention:* Preventing DoS attacks aims at avoiding the occurrence of these attacks in the first place. Since DoS attack mechanisms vary, attack prevention techniques vary as well. In fact, there are few and not very effective schemes for preventing various types of DoS attacks. Most of these schemes (e.g., [31, 32, 33, 34]) are especially useful for preventing attacks that employ source address spoofing. Therefore, aiming at stopping attack packets from reaching their target by performing some sort of source address validation. Other schemes, however, apply for attacks that do not necessarily rely on source address spoofing. For example, lightweight authentication is used for preventing QoS-based attacks [13], while TCP's time-out randomization is used for preventing low-rate TCP attacks [25]. Prevention schemes are usually proactive in nature, generally require global deployment in order to be effective, and they are usually static in nature. In section 2.1, we further classify DoS prevention schemes based on the location of their deployment as: (i) source-based, (ii) network-based, and (iii) victim-based.

- *DoS Mitigation:* Mitigation of DoS attacks is a reactive countermeasure that is usually initiated by the victim after detecting an attack. The main challenge in DoS mitigation is to accurately identify attack packets and filter them without causing collateral damage to legitimate traffic destined to the victim. There has been three major approaches to perform DoS mitigation, namely, rate limiting-based (e.g., [35, 36, 37, 38]), statistical-based (e.g., [39, 40, 41]), and path-based (e.g., [15, 42, 43, 44]). *Rate limiting-based schemes* deal with DoS attacks either as a congestion control problem or as a resource management problem. In both cases, the main idea is to characterize attack traffic and rate limit it. This approach is reactive, does not require wide scale deployment in most cases, generic in the sense that it applies to flooding attacks of any protocol type. However, it is not effective since it does not avoid collateral damage. *Statistical-based schemes* are based on the fact that attack traffic is more likely to hold a combination of attributes that were rarely seen by the victim. Therefore, incoming packets that hold attributes that do not match the attributes of an up-to-date traffic profile maintained at the victim are filtered directly. This approach is not effective as it introduces high false positive rates. *Path-based schemes* perform attack traffic filtering based on the identification of the paths traversed by attack packets (e.g., [42, 43]), the length of the paths followed by attack packets (e.g., [15]), or even by controlling the paths followed by legitimate packets (e.g., [44]). In section 3, we discuss DoS mitigation approaches in more detail.

- *DoS Traceback:* IP traceback, defined as the process of identifying the actual sources of IP packets, is an important countermeasure for DoS attacks that employ source address spoofing. In this context, IP traceback schemes usually rely on Internet routers in performing certain functions that eventually aid in locating attackers. IP traceback has the benefit of holding attackers accountable for abusing the Internet[1]. Also, it helps in mitigating DoS attacks either by isolating the identified attack sources, or by filtering attack packets coming from these sources far away from the victim. Using IP traceback

[1]Personal identification of attackers can be done by further investigation and analysis of the compromised systems discovered by the IP traceback process.

for DoS mitigation was considered in [43]. The IP traceback problem has received considerable research attention since the early days of DoS attacks [51]-[68]. This research has resulted in many IP traceback schemes that are broadly classified into three main categories, namely, packet marking-based schemes, packet logging-based schemes, and traffic engineering-based schemes. In section 2.3, we survey the main research efforts in this field focusing on the main ideas of the existing schemes and highlight their major deficiencies.

In the following sections we discuss the state of the art research in each DoS countermeasures category. Also, we motivate the work proposed in this dissertation, and highlight the main contributions.

2.1 DoS Prevention Schemes

In this section, we classify DoS prevention schemes based on the location of their deployment as: (i) source-based, (ii) network-based, and (iii) victim-based. The following subsections discuss the main schemes in each category.

2.1.1 Source-based DoS Prevention Schemes

In the source-based DoS prevention approach, the verification of the legitimacy of IP packets is performed at the periphery of the Internet, typically at the link between a customer network (i.e., the source of the packets to be validated) and an ISP network. This approach has been adopted by schemes such as ingress filtering [31], DWARD [46, 47], and IP Easy-pass [13]. Ingress filtering is mainly intended for filtering IP packets with spoofed source addresses. This filtering is done by configuring the edge router that connects the customer network to the ISP network so as to block packets that arrive with source addresses having prefixes that do not match the customer's network prefix. DWARD performs a proactive identification and filtering of suspicious flows originating from a customer network. This is achieved by monitoring the nominal per destination type traffic arrival and departure rate of TCP, UDP, ICMP packets, as well as any abnormal asymmetrical behavior of the two-way traffic at the edge

router connecting to a stub-network. IP Easy-pass, on the other hand, adopts a lightweight authentication scheme in order to prevent DoS attacks that target real-time applications within a DiffServ architecture [21].

Typically, preventing DoS attacks at their sources is the best solution because it protects the Internet from unwanted traffic at the earliest possible location. Therefore, preventing malicious flows from consuming network and end system resources. However, the major problem of this approach is that the deploying networks (i.e., those who adopt source-based prevention) should perform proactive detection and prevention of attacks originating from them. This leads to an extra overhead that discourage network administrators from adopting such approach, especially that the benefits of source-based prevention cannot be felt directly by the deploying network. Also, it may violate some existing setups and protocols such as Mobile IP and multi-homing (as in the case of ingress filtering). In addition, the success of this approach (especially in the case of ingress filtering and DWARD) depends directly on its wide scale deployment. Moreover, each scheme has its own limitations. For example, the scope of IP-Easy pass is limited only to environments similar to the DiffServ domain, and DWARD cannot prevent colluding attacks in which attackers generate traffic in both directions (i.e., inbound and outbound).

2.1.2 Network-based DoS Prevention Schemes

As a proactive solution to DoS attacks, several filtering schemes, which must execute on Internet routers, have been proposed to prevent spoofed IP packets from reaching intended victims. In this approach, if the source address of a given packet does not map to an expected router's interface, then the packet is dropped. The main challenge in this approach is to establish a valid mapping between a router's interface and a source address space. This approach has been adopted by the distributed packet filtering architecture (DPF) [32], in which it has been shown that an edge router of an AS (autonomous system) can perform proactive packet filtering by utilizing the routing information to determine whether an incoming packet is valid with respect to the packet's inscribed source and destination IP addresses. Also, this approach has been adopted by the SAVE protocol [33], which provides Internet routers with information

to build incoming tables that map source address spaces to router's interfaces.

Generally, the main problem with this approach is that IP packets with enroute spoofed source address cannot be filtered [11], because an attacker would spoof the address of a node that is located along the path between the attack node and the intended victim. Also, this approach requires wide scale deployment to be effective. Moreover, due to frequent route changes, legitimate traffic may get filtered even in the absence of an attack. In addition, each scheme has its own problems. For example, the major challenge in the DPF architecture is to make source-based routing information available to routers such that they can perform spoofed packet filtering. This issue was left as an open problem. SAVE protocol, on the other hand, is still subject to several problems that include incoming table's poisoning due to attacker's ability to inject false update messages, and incoming table's inconsistency due to frequent route changes, leading to false packet filtering.

2.1.3 Victim-based DoS Prevention Schemes

Deploying DoS prevention schemes at potential targets is more incentive as compared to their deployment at the source of an attack or inside the network. This is because the benefit of DoS prevention is felt directly by the deploying system or network. TCP's time-out randomization [25, 48], and spoofing prevention method (SPM) [34] have adopted this approach. TCP's time-out randomization requires a slight modification to the TCP stack such that TCP clients can avoid the effects of low rate TCP attacks [25]. Generally, TCP's time-out randomization has limited benefit because an attacker can still degrade the performance of the targeted TCP connections. SPM, on the other hand, requires installation of packet filters at the edge of a network to filter IP packets with spoofed source addresses. In order to achieve this functionality, the scheme requires each IP packet leaving a source network towards a certain destination to be tagged by a key that is associated with that (source network, destination network) pair. The dependency of SPM on authentication makes it similar to IP Easy-pass [13]. The main difference is that IP Easy-pass performs lightweight authentication at a single domain scale, while SPM performs lightweight authentication at an Internet scale. Although

such scheme is more incentive to deploy as compared to ingress filtering, mutual cooperation among large number of ISP networks is still required. Moreover, SPM imposes additional overhead at the source network (due to key tagging) as well as at the destination network (due to key verification).

2.2 DoS Mitigation Schemes

DoS mitigation is a strategic defense mechanism that is usually activated after detecting an attack. A core issue in DoS mitigation is the ability to make a distinction between attack packets and legitimate packets. There have been three major approaches in the literature in order to achieve this distinction. The following subsections are mainly devoted to discuss these approaches.

2.2.1 Rate Limiting-based DoS Mitigation Schemes

DoS attacks are usually characterized by huge packet volumes that lead to a network congestion and to an end system overloading. Therefore, it is logical to deal with this problem as a congestion control problem as in the Pushback scheme [35, 36, 37, 38], or as a resource management problem as in the max-min fair server-centric router throttles [49]. In both cases, the main objective is to throttle attack flows, such that legitimate flows obtain fair share of the available resources. This goal can be achieved by applying rate limiting on incoming flows selectively. Pushback adds a functionality to each router to detect and preferentially drop packets that probably belong to an attack flow. In the max-min fair server-centric router throttles, the routers along forwarding paths to a server under DoS attack are asked to regulate the contributing packet rates to more moderate levels such that overall packet arrival rate at the server falls within acceptable range.

The deployment requirement of rate-limiting based DoS mitigation varies from one scheme to another. For example, Pushback scheme requires wide scale deployment in order to be effective. While the max-min fair server-centric router throttles requires local deployment. In terms of effectiveness, rate-limiting approach affects legitimate traffic significantly. Also, this

approach requires Internet routers to characterize malicious traffic precisely before being able to throttle it. This seems to be difficult to achieve given the dynamic nature of various types of flooding attacks. In fact, the characterization process is left as an open problem as pointed out in [35]. Moreover, attackers who are aware of the existing defense mechanisms can change their traffic dynamics in such a way as to avoid the filtering rules.

2.2.2 Statistical-based DoS Mitigation Schemes

Attack packets are more likely to hold attributes that were rarely seen by the victim. This observation is mainly due to the fact that attackers usually employ random spoofing of source addresses, TTL values, TOS field, and other packet attributes. By keeping an up-to-date profile of legitimate traffic, a system under attack can use this profile to perform per-packet filtering. Packets with attributes that are rarely seen in the traffic profile are dropped during attack time. History-based filtering [39] and Packetscore [40, 41] have adopted this approach. During an attack, history-based filtering drops packets from sources that rarely communicate with the targeted system in the absence of an attack, while allowing packets with source addresses that regularly visit the network. Packetscore enables each edge router of an ISP network to compute a "score" for each suspicious packet and ranks the likelihood of the packet being an attacking packet, given the attribute values it carries, using a Bayesian-theoretic approach. Statistical-based DoS mitigation has several drawbacks. For example, the traffic profiles maintained at the victim or at the edge routers can be poisoned before launching an attack, and sophisticated attackers can generate packets with attributes that match these traffic profiles.

2.2.3 Path-based DoS Mitigation Schemes

Since attackers, in most cases, do not have control over the actual paths traveled by attack packets, it is possible to filter these packets either by identifying the paths followed by them (e.g., path identifier [42], and traceback-based intelligent packet filtering [43]), by inspecting certain attributes of these paths (e.g., hop-count filtering [15]), or even by controlling the paths followed by legitimate packets (e.g., Secure Overlay Services [44, 45]). Pi [42] and

IP traceback-based intelligent packet filtering [43] are based on the fact that attack packets originating from the same source are more likely to traverse the same path (assuming routing stability). Therefore, by identifying a single attack packet, it is possible to filter all subsequent packets coming along the same path. Hop count-based filtering [15] is based on the fact that the distance (i.e., number of hops) traveled by a packet that holds a spoofed source address is more likely to be different than that traveled by a packet emitted from the spoofed source itself. SOS architecture [44]is basically an overlay network, composed of nodes that communicate with one another atop the underlying network substrate. This architecture performs access verification at several nodes distributed throughout the Internet, then authorized traffic is forwarded through the overlay to a secret node. A perimeter that surrounds the protected target allows traffic coming from the secret node only while filtering any other traffic.

The main advantage of the path-based DoS mitigation approach is that it is generic in the sense that it is protocol independent (i.e., it can be applied to any type of attack traffic). However, its major drawback is that it requires a network support (e.g., [42, 43, 44]) or an up-to-date hop count profiles (e.g., [15]) in order to perform accurate distinction between attack packets and legitimate packets, which makes its deployment less incentive. In terms of the scale of deployment, some path-based schemes (e.g., [42, 43]) require global deployment, some schemes require an overlay network support (e.g., [44]), while other schemes require local deployment (e.g., [15]). In terms of effectiveness, although the collateral damage in [42, 43] is low in general, it still exists, especially when the paths followed by legitimate packets share large portions with the paths followed by attack packets because, in this case, both types of packets are more likely to have the same path identifier by the time they reach the victim. The effectiveness of hop-count filtering [15] is determined by the accuracy of the hop-count profile maintained at the victim. Although the SOS architecture can theoretically filter all attack traffic[2], it introduces a significant increase in the the latency across the communication path which estimated to be in the order of 10 times larger than in the direct communications case.

[2]It is important to emphasize that SOS architecture is applicable only for mission-critical systems where authentication is a prerequisite for using these systems, and it is not intended for mitigating DoS attacks against public systems such as Web servers.

2.3 IP Traceback Schemes

The focus of this section will be mainly on the major IP traceback approaches, namely, packet marking, packet logging, and traffic engineering.

2.3.1 Packet Marking-based IP Traceback Schemes

Packet marking-based IP traceback schemes employ a simple concept. As IP packets traverse their paths, Internet routers proactively augment them with traceback information. Once an attack is detected, the victim extracts the traceback information from the received attack packets and uses it to reconstruct the path followed by attack packets. Therefore, packet marking-based IP traceback schemes usually consist of two algorithms. One is the *packet marking algorithm* that is running at the Internet routers. The other is the *traceback algorithm* that is initiated by the victim to trace the attackers using the marking information found in the received attack packets. Packet marking schemes are usually constrained by the limited space available in the IP packet header that can be used to accommodate traceback information. Based on the fact that packet fragmentation is very infrequent in today's Internet [50], the focus has been always on utilizing the 16-bits ID fragmentation field and few other unused bits in the IP packet header for this purpose.

Packet marking approach has been adopted originally by the *probabilistic packet marking scheme* (PPM) [51]. In this scheme, routers perform probabilistic marking on forwarded packets. For this purpose, three marking fields are reserved in each packet: start, end and distance. The scheme is designed such that at any node along a packet's path, the distance field would represent the number of hops between the edge defined by the pair (start, end) and that node. In other words, if a packet reaches its destination with the marking fields (start, end, distance) set to (x, y, d), respectively, the destination infers that the packet had been forwarded through the edge (x, y) which is d hops away. The actual marking procedure is performed by Internet routers as follows: If a router decides to mark a packet according to certain probability, q, it writes its IP address in the start field and initializes the distance field to zero. Otherwise, if the distance field is zero it writes its IP address to the end field and increments the distance

field.

Once an attack is detected by the victim, it has to collect sufficient number of attack packets and extract marking information (i.e., edges and their distances) from them. It then combines them together to reconstruct the attack path (or attack tree if multiple attackers are involved). This scheme requires 64 bits for the start and end fields and 8 bits for the distance field. Therefore, a total of 72 bits would be allocated in each packet for marking purposes. To overcome this practical limitation, the authors in [51] proposed an encoding scheme that reduces the marking requirements to 16-bits. In addition to the fact that PPM itself is not immune to the spoofed marking field problem [52], there are a lot of problems associated with the marking information encoding which includes the requirement of huge number of attack packets to conduct the path (or tree) reconstruction process, the high overhead imposed on the victim in enumerating and combining a significant number of IP addresses' portions in the tree reconstruction process, and the high false positive rates in the case of multiple attackers.

Several other schemes (e.g., [53, 54, 55]) have been proposed subsequently in order to enhance the performance and the functionality of PPM. The focus of these schemes was mainly on issues related to traceback authenticity and robustness, especially when there is a very large number of attackers. Although, many improvements have been made, these schemes still suffer from problems that were inherited originally from the PPM scheme, such as the requirement of a very large number of packets, and the spoofed marking field problem. Another approach for IP traceback is to locate attack nodes without identifying the paths followed by attack packets. This approach was adopted by the deterministic packet marking (DPM) scheme [56, 57]. In this approach, packet marking is performed only at the interface closest to the source of the packet on an edge ingress router. This scheme inherits some of the problems of ingress filtering [31] especially from the deployment point of view.

As an alternative for inscribing the marking information within the IP packets themselves, S. M. Bellovin [58] proposed to inscribe the marking information within separate ICMP messages (called ICMP traceback). In this scheme, each router decides with some probability, q, to send an additional ICMP message about a given forwarded packet to the destination

rather than writing the marking information into the packet itself. A major problem with this scheme is that it introduces additional communication overhead in the sense that it increases the network traffic even without having a DoS attack. To address this issue, intention driven ICMP traceback (iTrace) was proposed in [59], where ICMP tracebak messages are emitted only toward destinations that already indicated their interest in receiving such messages. To achieve this functionality, iTrace scheme suggests utilizing the BGP routing information exchange as the vehicle to distribute system's interest in obtaining ICMP traceback messages. A similar scheme, called path information caching and aggregation (PICA), was proposed in [60]. The main difference is that message generation is router-driven rather than destination-driven. The scheme has the advantage of triggering traceback messages only when the aggregate count of packets going to the same subnet exceeds certain threshold during a specified interval of time. The main problem with PICA is that in large scale DoS attacks, the specified threshold may never be exceeded, especially at routers far away from the victim.

2.3.2 Packet Logging-based IP Traceback Schemes

Packet logging is the counterpart of packet marking in the sense that instead of writing router's information into IP packets, packet's information (digests, signature, or even the packet itself) is written into router's memory. Once an attack is detected, victim's upstream routers are queried to check if their memories contain attack packet's information or not. If attack packet's information is found in a given router's memory, then that router is assumed to be part of an attack path. Obviously, the major challenges in packet logging are the storage requirements at intermediate routers, maintaining privacy, and collecting packet's information from Internet routers. This approach was adopted by [61], wherein routers log information about traversing packets and then use the logged data to trace each packet from its final destination back to its source hop by hop. The major problems of this traceback method are the high storage demand, especially at high Internet speeds, and the high false positive rate.

The fact that today's Internet link speeds could reach multiple Gbps suggests that classical packet logging at Internet routers would require huge amount of storage resources. A. Snoeren

et.al., [62] came up with an innovative approach to reduce the storage requirement significantly. Their basic idea was to store packet digests at a given router using a storage efficient data structure known as Bloom filter [69]. This approach provides a significant reduction in the storage requirements at each router. Also, it maintains privacy because packet's digests stored at a router does not reveal any of its content. A hardware design was proposed in [63] to support the implementation of this scheme at a very high Internet speeds. The major drawbacks of Hash-based traceback is that it incurs relatively large processing and storage overhead due to its deterministic nature. Also, the method employed to download packet information from network routers is inefficient and requires special resources. Moreover, a major concern in Hash-based traceback is the small window of time through which packets can be successfully traced.

2.3.3 Traffic Engineering-based IP Traceback Schemes

Traffic engineering-based traceback schemes involve changing the network traffic in a controlled manner during the attack period. This approach has been adopted mainly by CenterTrack [64] and link-testing [65]. Centertrack is a reactive IP traceback scheme that aims at locating the link through which attack flows enter an ISP network, by selective rerouting of packets that have common attack signature through an overlay network consisting of IP tunnels that connect ISP's edge routers to special tracking routers. The major problems of this scheme is that it incurs a lot of management of overhead. Also, it is applicable only to large flow attacks. In link-testing, a short burst of traffic, generated using the UDP chargen service [70], is applied to a link under test to see whether it is part of the attack path or not. The major problems with this scheme is that it is not useful in multiple attackers case, or in variable rate attacks. Also, it includes multiple branch points and communication overhead due to message exchange. In addition, it represents a denial-of-service attack by itself.

On the positive side, these schemes are reactive in the sense that they are initiated only when there is an attack, which means that the overhead associated with them is limited to the attack period rather than all the time. In addition, they are usually easier to deploy as

compared to packet marking and packet logging-based approaches. However, on the negative side, they have limited effectiveness, in the sense that they are not applicable for tracing DoS attacks that involve many attackers, also in the sense that the whole traceback process should be performed while the attack is going on, which imposes timing constraints on network administrators. From legal point of view, these schemes cannot be used to hold attackers accountable, because they do not collect evidence about attack packets and their sources.

2.4 Motivations

Considering the three major DoS countermeasures discussed in this chapter, DoS prevention is the most desirable solution. However, complete DoS prevention is not feasible in today's Internet due to: (1) the fact that DoS attacks are emerging in various forms and becoming more complicated, and (2) the proactive nature of prevention schemes makes their deployment less incentive, especially because DoS attacks are the exception rather than the norm. On the other hand, DoS mitigation is a more practical solution as compared to prevention for two main reasons: (1) at any given time, portion of the Internet rather than the whole Internet can be under attack, and (2) DoS mitigation schemes can be activated reactively based on the type of ongoing attack. It is implicitly assumed that only the targeted system or network may elect to mitigate an ongoing attack. Given that neither DoS prevention nor DoS mitigation can provide a clue about attacker's location, IP traceback remains a very important requirement for forensic investigations.

Based on the previous discussion, the work presented in this dissertation focuses mainly on DoS mitigation and IP traceback. Despite the considerable amount of research done in these two areas, the existing mitigation and traceback schemes lack in several qualitative and quantitative metrics, which forms the motivation for the work presented in this dissertation. In the following subsections, we motivate the need for efficient DoS mitigation and IP traceback approaches. In particular, we briefly discuss the major drawbacks of existing schemes in both areas, and lay the ground for the work presented in this dissertation.

2.4.1 The Need for Efficient DoS Mitigation Approaches

Table 2.1 provides a qualitative comparison between major DoS mitigation schemes in terms of practical aspects such as deployability, and in terms of performance aspects such as collateral damage, network exhaustion, and victim exhaustion. It can be seen that existing schemes have several practical and performance limitations. With respect to practicality, some schemes (e.g., Pushback, Pi, and Traceback-based filtering) rely on direct network support for their operation. Other schemes (e.g., PacketScore, History-based, and Hop-count) require up-to-date time variant traffic profiles or path length records to be maintained at the victim side. With such requirements, the practicality of these schemes becomes limited.

With respect to performance, collateral damage is not avoidable under rate limiting approaches. In certain scenarios, false positive rates will be high in the case of statistical-based approaches. A typical DoS mitigation scheme should provide protection from both end system exhaustion attacks and network exhaustion attacks. It is clear that this feature is not available in many of the schemes listed in table 2.1. Based on the above discussion, it is obvious that none of the existing DoS mitigation schemes combines good features in terms of practicality and performance. These shortcomings establish the need for more efficient and practical DoS mitigation schemes. Different than earlier work, the proposed research presented in Chapter 3 of this dissertation promotes a new paradigm for DoS mitigation. The spirit of the proposed work is in devising new concepts that aid in performing online (i.e., without network support and without traffic profiling) DoS mitigation far away from the victim (e.g., at an ISP's perimeter).

2.4.2 The Need for Efficient IP Traceback Approaches

Among the three major IP-traceback approaches discussed in Section 2.3, packet marking and packet logging approaches have dominated the area of IP traceback. At the same time, less attention has been paid to the traffic-engineering based approach due to its limited benefit and little efficiency. However, despite their popularity and despite their promising features, packet marking and packet logging-based traceback approaches still lack in several aspects. Table

Table 2.1 Qualitative comparison of existing DoS mitigation schemes

Approach	Scheme	Deployability	Collateral Damage	Network Exhaustion	End System Exhaustion
Rate Limiting Based	Pushback [35]	network support reqd.	high	low	low
	Max-Min fair router throttles [49]	perimeter-based	high	low	low
Statistical Based	PacketScore [40]	traffic profiling reqd.	low/medium	low	low
	History based [39]	traffic profiling reqd.	low/medium	high	low
Path Based	Pi [42]	network support reqd.	low	low	low
	IP traceback based [43]	network support reqd.	low	low	low
	Hop count-based [15]	traffic profiling reqd.	low	high	low

2.2 provides a qualitative comparison between these two approaches. The comparison takes into account the basic method employed in each approach, the number of packets required to conduct the traceback process, router's processing and storage overhead, packet's overhead, and the issue of collecting path information from Internet routers.

It can be seen that neither packet marking nor packet logging performs well in all aspects of comparison. Specifically, these approaches require either a large number of attack packets to be collected by the victim to infer the paths (packet marking), or a significant amount of resources to be reserved at intermediate routers (packet logging). This lacking in performance motivates the need for more efficient IP traceback approaches that offer better performance in all aspects. In this context, the proposed research presented in Chapter 4 adopts a hybrid traceback approach in which packet marking and packet logging are integrated in a novel manner to achieve the best of both worlds, that is, to achieve small number of attack packets to conduct the traceback process and small amount of resources to be allocated at intermediate routers for packet logging purposes.

2.5 Dissertation Contributions

The contributions of this dissertation span the areas of DoS mitigation and IP traceback. With respect to DoS mitigation, we promote the concept of perimeter-based *online* DoS mitigation, wherein edge routers of an ISP network take advantage of the inherent features of the ongoing attack to perform per-packet classification and filtering. In this context, we advocate

Table 2.2 Qualitative comparison between packet marking and packet logging-based IP traceback approaches

Traceback approach	Packet Marking	Packet logging
Basic method	routers write their IDs (IP addresses) in the forwarded packets (deterministic/probabilistic)	packet information (digests or signatures) is written into router's buffer
Number of attack packets needed to infer an attack path	a large number of attack packets (probabilistic); single attack packet (deterministic)	Same as packet marking
Overhead	no buffer overhead at routers; but high packet overhead; router CPU overhead for marking	high buffer overhead at routers; but no packet overhead; router CPU overhead for logging
Collecting path information	not a big issue, i.e., can be done using the attack packets	coordination among routers required

two new concepts to supplement ISP's edge routers by special mechanisms in order to perform online attack mitigation. In particular we make the following contributions.

- *Advocating the concept of protocol determinism-based online DoS mitigation:* In this dissertation, we utilize the deterministic nature of Internet protocols and the inherent features of DoS attacks in performing online DoS mitigation. We apply this concept within a novel scheme, called intentional dropping-based filtering, for mitigating SYN flooding attacks.

- *Advocating the concept of victim-assisted based online DoS mitigation:* In this dissertation, we show that the victim of a DoS attack can play a direct role in mitigating the attack. Specifically, we show how a victim can guide its ISP's edge routers to filter packets that do not comply with a pre-specified rules enforced by the victim itself during an attack period. Two DoS mitigation schemes are proposed based on this concept, namely, the SYN number based filtering and the distributed packet pairing based filtering.

- *Performance analysis of the proposed DoS mitigation schemes:* The proposed DoS mitigation schemes are evaluated through a probabilistic analysis under different scenarios that include classical attacks as well as advanced attacks.

With respect to IP traceback, we design, analyze, and study the performance of two novel hybrid IP traceback schemes employing packet marking and packet logging. In particular, we

make the following contributions.

- *Design and analysis of Distributed Link-List Traceback (DLLT):* In this dissertation, we employ the concept of a "distributed link list" in the context of a novel IP traceback scheme, called DLLT. The basic idea of DLLT is to establish a link between routers that perform probabilistic marking for a given packet, such that the identity of these routers can be retrieved by following the pointer of that list.

- *Design and analysis of Probabilistic Pipelined Packet Marking (PPPM):* In this dissertation, we employ the concept of a "pipeline" in the context of a novel IP traceback scheme called PPPM. The basic idea of PPPM is to identify the identities of routers that perform probabilistic packet marking for a given packet by propagating the marking information from one marking router to another, within packets destined to the same destination, so that it eventually reaches the destination.

- *Performance evaluation of the proposed IP traceback schemes:* We evaluate the proposed IP traceback schemes through a combination of simulation and analytical studies. In particular, the proposed schemes are evaluated in terms of the number of packets required to construct full attack path and in terms of their storage requirement. In addition, new performance metrics are defined to evaluate the proposed schemes.

Fig. 2.2 shows a summary of the novel contributions of this dissertation.

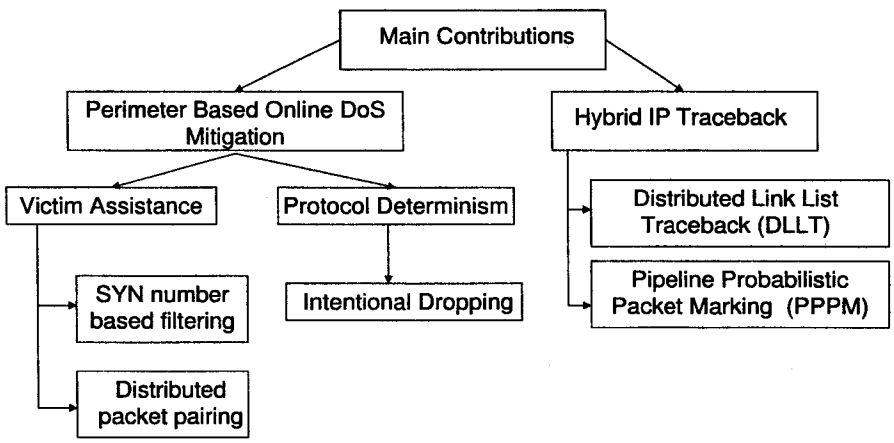


Figure 2.2 Summary of the work presented in this dissertation.

CHAPTER 3. PERIMETER BASED ONLINE DOS MITIGATION

In this chapter, we advocate two novel concepts, namely, victim-assistance and protocol-determinism, for DoS mitigation. These concepts are mainly useful for supplementing edge routers of an ISP network with special mechanisms that enables them to distinguish attack traffic from legitimate traffic. In this context, the novelty of the proposed concepts is due to their benefit in performing *online* DoS mitigation at an ISP's perimeter. A feature that is not available in current perimeter-based DoS mitigation schemes.

The concept of victim-assistance aims at making effective use of the information available at the victim about an ongoing attack. The spirit of this concept is envisioned via the cooperation between the victim of an attack and the edge routers of the ISP network where the victim is located. This cooperation is performed in a manner that enables edge routers to identify and filter attack traffic before reaching its target. For the sake of illustration, two DoS mitigation schemes are developed based on this concept, namely, packet pairing-based filtering, and SYN number-based filtering.

The concept of protocol-determinism, on the other hand, aims at mitigating DoS attacks in a transparent manner. Without any communication (except for attack detection) between the victim and the ISP's edge routers, each edge router can perform packet classification and filtering independently by taking advantage of the proposed concept. Protocol-determinism concept takes advantage of the deterministic nature of Internet protocols in order to identify attack packets. The premise here is that while attackers usually exploit the deterministic nature of Internet protocols to perform DoS attacks (e.g., SYN flooding, low-rate TCP, etc.), they do not fully adhere to the rules of these protocols. Hence, the main objective of the proposed concept is to force attackers to fully adhere to the protocol rules, which results in a

significant reduction of attack traffic. Intentional dropping-based filtering is developed around this concept to mitigate SYN-flooding attacks.

The rest of this chapter is organized as follows. In Section 3.1, we discuss the possible deployment options for DoS attack detection and mitigation and we motivate the need for a perimeter-based architecture. In Sections 3.2 and 3.3, we describe the proposed concepts and show how they can be utilized for DoS mitigation. Finally, summary is presented in Section 3.5.

3.1 Perimeter-based DoS Mitigation - Why?

It is well known that a DoS attack against an end-system has a direct impact not only on that system, but also on the network in which the targeted system is located. This is due to attack traffic aggregation near the victim, which leads to network resource exhaustion. Therefore, it is of primary importance to filter attack traffic as far from the victim as possible. In this context, the location of DoS detection and mitigation schemes involves several tradeoffs between efficiency and practicality that need to be considered. At one extreme, these schemes can be placed at the victim itself. At another extreme, these schemes can be placed at the ISP's perimeter. Between the two extremes, the location could be along an optimal boundary defined by the victim. The following discussion explains the tradeoffs introduced in each case.

- *At the victim:* With respect to DoS detection, the victim is the best location for fast and accurate attack detection in most attack scenarios. With respect to DoS mitigation, the victim is also the best location from deployment point of view, because the DoS mitigation module would be installed at a single system (i.e., at victim), which has complete knowledge about the attack, and, in most cases, it can easily classify incoming traffic as attack or legitimate. Unfortunately, this ability does not alleviate the effect of attack packets as they consume significant network and end system resources by the time they are identified.

- *At the ISP's perimeter:* With respect to DoS detection, edge routers along an ISP's perimeter cannot precisely detect the existence of an attack because the amount of at-

tack traffic seen by each edge router individually is very small as compared to that seen by the victim. This is especially valid for flooding based attacks. However, with respect to DoS mitigation, installing the DoS mitigation schemes far away from the victim (e.g., perimeter) provides protection for the network as well as for the targeted system. Therefore, it is beneficial to install the DoS mitigation schemes at the ISP's edge routers to form a line of defense at the ISP's perimeter, because (1) any traffic destined to the victim has to pass through one of the ISP's edge routers, (2) edge routers are usually computationally capable and can perform the task of packet classification and filtering. However, the major problem with this approach is that edge routers cannot perform accurate traffic classification by themselves due to the lack of knowledge about the ongoing attack and about how to make a distinction between attack packets and legitimate packets.

- *At an optimal boundary:* An optimal boundary is a conceptual term that refers to the periphery around the victim where it is possible to protect against end system exhaustion as well as bandwidth exhaustion. Such periphery is located somewhere between the ISP's perimeter and the victim itself. The DoS detection/mitigation schemes installed along that periphery can (1) detect attacks that are hard to detect at the ISP's perimeter, (2) perform more efficient DoS mitigation as compared to mitigation at the victim itself. Although such placement policy achieves the best of both worlds (i.e., protection from bandwidth exhaustion with accurate detection and efficient mitigation), it has several practical limitations that prevent its deployment. For example, the conceptual optimal boundary is victim-oriented. Moreover, it requires modification at core routers. Therefore, it raises several deployment concerns.

Fig. 3.1 depicts the tradeoffs introduced by the placement of the DoS detection/mitigation schemes. Obviously, performing DoS mitigation at the ISP's perimeter has several benefits that can be summarized as follows.

- *Early filtering*: Attack traffic is filtered before it enters the ISP network that contains the victim. Therefore, providing protection from bandwidth exhaustion as well as end

system exhaustion.

- *Fruitful deployment*: The direct benefit of deploying an attack mitigation scheme is felt by the deploying network, which convinces network administrators to adopt this architecture.

- *Ease of deployment*: Perimeter-based DoS mitigation is easy to deploy, manage, and operate due to its intrinsic feature of being under a single administrative domain such as an ISP network.

In this dissertation, we adopt the perimeter-based DoS mitigation architecture. However, different than earlier work that adopted the same architecture (e.g., [71, 72]), we introduce the concept of *online* perimeter-based DoS mitigation wherein edge routers of the ISP network take advantage of the inherent features of the ongoing attack to perform per-packet classification and filtering. In this context, ISP's edge routers are supplemented by special mechanisms in order to perform online DoS mitigation. Two new concepts, namely, victim-assistance and protocol-determinism, are proposed in the following sections, respectively, in order to realize this objective.

3.2 Victim-Assisted Schemes for DoS Mitigation

Victim-assistance refers to the direct role of the victim in identifying attack traffic before reaching its target. This definition implies that the victim is required to cooperate with an ISP's edge routers in a manner that leads to accurate classification of incoming traffic. The basic idea of this concept is to involve the victim in guiding an ISP's edge routers to filter packets that do not comply with a pre-specified rules enforced by the victim during the attack period. In this section, we illustrate the use of this concept within two schemes that are designed to mitigate a class of attacks known as reflector-based DoS (RDoS) attacks. The first scheme is generic in the sense that it is applicable to any type of RDoS attacks. However, the second scheme is specific in the sense that it is mainly applicable for TCP-based RDoS attacks.

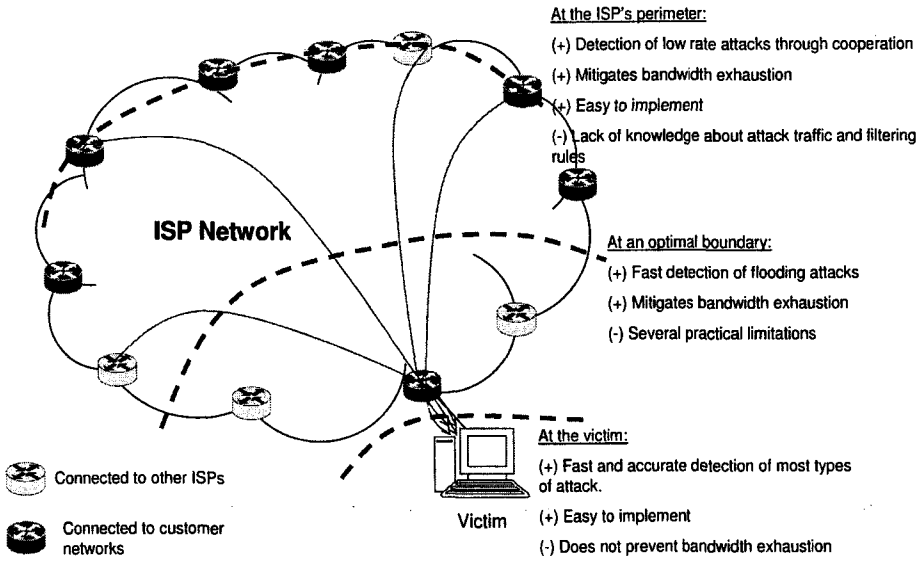


Figure 3.1 Tradeoffs introduced in the placement of DoS detection and mitigation schemes within an ISP network.

3.2.1 An Overview of RDoS Attacks

Recently, RDoS attacks are being used by attackers frequently to affect the availability of high profile servers [12, 73]. In these attacks, a large number of compromised computers under attackers' control (the slaves) are instructed to continuously send request packets to a set of Internet reflectors (an Internet reflector is an IP host that will reply to any request packet). The source address of each of these request packets is spoofed to be the same as the address of the targeted site. As a result, the reflectors send their replies to the system specified by the given address. Therefore, flooding it by significant amount of traffic. Using Internet reflectors complicates the problem of DoS attacks. Researchers are more concerned about these attacks because (1) attack packets (reply packets originated from the reflectors themselves) carry legitimate IP source addresses making it difficult to trace original attack sources, (2) these attacks are usually characterized by an amplification factor that increases their intensity.

The analysis presented in [12] shows that RDoS attacks are feasible in variety of re-

quest/reply based protocols, including TCP, UDP, ICMP, and DNS. For example, in Smurf attacks [16], the attacker sends ICMP echo requests (pings) to the broadcast address of a network, so the victim is hit by many more packets. The Fraggle (UDP packet magnification) attack uses UDP echo packets in the same fashion as the ICMP echo packets. In TCP-based RDoS attacks [73], attackers take advantage of the availability and connectivity of large number of Internet reflectors. This can be done by abusing the TCP protocol in the following way: an attacker, A, selects a set of Internet reflectors R=$\{R_1,R_2, ...,R_n\}$. It then sends low rate faked SYN packets to each of these reflectors with a spoofed source address equals to that of the final target, V. For each received SYN packet, the reflectors reply with a SYN-ACK packet to the given address, V. Therefore overwhelming the victim, V, by high aggregate rate of SYN-ACK packets. The following subsection highlights the importance of mitigating this type of attacks in particular.

3.2.1.1 The Need for Protection from TCP-based RDoS Attacks

In TCP-based RDoS attacks, the target can be a web server that accepts connections from clients throughout the Internet. In most cases, a web server does not initiate TCP connections by itself. Therefore, TCP-based RDoS attacks against such systems can be mitigated by filtering any incoming SYN-ACK packet. However, there exist several scenarios where the targeted system requires to initiate TCP connections with other systems across the Internet. Protection of such systems from reflected attack SYN-ACK packets is not trivial because it is difficult to distinguish these packets from legitimate SYN-ACK packets that are necessary for connection establishment. Here, we list few examples of servers that initiate TCP connections. These servers are potential *targets* of TCP-based RDoS attacks.

- *FTP Server [74]:* In active mode FTP, the client connects from a random unprivileged port (Port Number $PN > 1024$) to the FTP server's command port, port 21. Then, the client starts listening to port $PN + 1$ and sends the FTP command "PORT $PN + 1$" to the FTP server. The server will then connect back to the client's specified data port from its local data port, which is port 20.

- *Proxy Server [75]:* A proxy server receives a request for an Internet service (such as a Web page request) from a user. If it passes filtering requirements, the proxy server looks in its local cache of previously downloaded Web pages. If it finds the page, it returns the page to the user without needing to forward the request to the Internet. If the page is not in the cache, the proxy server, acting as a client on behalf of the user, uses its own IP address to request the page from the server out on the Internet. When the page is returned, the proxy server relates it to the original request and forwards it on to the user. Although using a proxy server in some cases is optional and can be avoided if it is under attack, some ISP's make all their users use a proxy server to block sites with unsuitable content. Therefore, a lot of management overhead would be incurred if ISP customers were asked to reconfigure their web browsers to avoid connecting to the Internet via a proxy server that is under attack.

- *SOCKS Server [76]:* SOCKS server is a function that is used to manage the connections between clients/servers on a secure internal network and clients/servers on an untrusted network such as the Internet. The SOCKS server sits between the trusted and untrusted systems. The server regulates which connections are allowed, logs information regarding the connections, and hides the internal network information (such as internal IP addresses). Hosts outside the secured network perceive the SOCKS server as the source of the communication. The server resends requests and responses between the trusted and untrusted systems.

In addition to their potential damaging effects on FTP, Proxy, and SOCKS servers, TCP-based RDoS attacks have gained a special importance due to the following reasons.

- TCP carries 95% of today's Internet traffic and 80% of the total number of flows in the Internet [77]. Therefore, attackers prefer using TCP based traffic in order to avoid detection.

- Any general purpose TCP connection-accepting Internet server could be used as a packet reflection server. This provides attackers with a large pool of servers to be used as

reflectors.

- Several incidents of TCP-based reflector attacks have been reported (e.g., in addition to the reported attacks against GRC.com [73], many TCP-based reflector attacks were captured at Los Nettos ISP network [78]).

It is worth mentioning that the special importance of TCP-based RDoS attacks does not imply that other types of RDoS attacks do not pose a threat. In fact, mitigating these attacks is equally important. In the following subsections, we propose two schemes for mitigating TCP-based RDoS attacks[1]. These schemes are based on the proposed concept of victim-assistance.

3.2.2 Scheme I: Packet Pairing-based Filtering (PF)

Based on the concept of victim-assistance, we propose a scheme for RDoS attack mitigation. The proposed scheme, called pairing based filtering (PF), works by informing edge routers of an ISP network about TCP connection establishment requests initiated by the victim such that incoming reply packets can be paired with them, while replies that represent attack packets can be filtered directly. It is important to mention that the basic idea of request/reply pairing is not new as it has been used earlier for DoS attack detection and mitigation (e.g., D-WARD project [47], SYN-dog [79], and TCP flooding blocking [80]). However, different than earlier work, request/reply pairing in our scheme is performed at the perimeter that is composed of an ISP's edge routers. This is more challenging as compared to performing the request/reply pairing at the single edge router of a customer network as in [47, 79, 80]. The benefit of this approach is that attack packets can be filtered before they enter the ISP network. In this subsection, we discuss the design and implementation of the PF scheme and characterize its behavior. Also, we perform statistical analysis to study the effects of traffic unpredictability (represented by routing instability and routing asymmetry) and other design parameters on the performance of the PF scheme.

[1]It is to be noted that the PF scheme, proposed in subsection 3.2.2, is applicable to other types of RDoS attacks as well.

3.2.2.1 Assumptions

Our methodology in mitigating the effect of TCP-based RDoS attacks is based on inherent features of the attack itself and the deterministic nature of the TCP protocol. We observe that an attacker has some constraints that shape the overall attack. Among these constraints is the total number of reflectors that an attacker can use, and more importantly, the mapping between theses reflectors and the edge routers of the ISP network which contains the targeted system. By this mapping we mean that each subset of reflectors used by the attacker have to forward their attack packets through the same edge router during the attack period.

This conclusion is based on the assumption of *routing stability*[2], referred to as the situation of having the same ingress router for all packets sent from a given source to a destination located in the ISP network. It is important to distinguish routing stability from *routing symmetry* which refers to the situation of having identical egress and ingress edge router for a SYN packet and its corresponding SYN-ACK packet. Although having routing symmetry facilitates the pairing of SYN packets generated by the victim and their corresponding SYN-ACK packets (this is the basis of our proposed scheme), we do not necessarily assume routing symmetry. Instead, we assume that a mechanism such as loose source routing [75] can be used to force a SYN packet generated by the victim to pass through the edge router at which the corresponding SYN-ACK packet is expected to arrive.

3.2.2.2 An Overview of the PF Scheme

The PF scheme is a SYN-ACK packet classification and filtering scheme to be implemented at the edge routers of an ISP network. The scheme is assumed to be reactive in nature, which means that it should be activated only after detecting a TCP-based RDoS attack. The main idea of the proposed PF scheme is as follows: since an attacker keeps using the same set of reflectors continuously during an attack, subsequent SYN-ACK packets from the same reflectors can be filtered at their ingress points to the ISP network unless they correspond to previously sent SYN packets by the victim.

[2]This assumption is relaxed in subsection 3.2.2.4.

The basic architecture of PF scheme can be viewed as a two level filter. Generally, SYN-ACK packets destined to the victim can be classified as legitimate, attack, or suspicious. The filter is designed such that legitimate packets are passed directly, attack packets are dropped directly, and suspicious packets are marked and passed. Initially, all packets are assumed to be suspicious. However, by taking advantage of the nature of the RDoS attack and the deterministic behavior of the TCP protocol itself, subsequent packets can be classified with high accuracy. The following data structures are required at each edge router to implement the classification and filtering functions of the PF scheme.

- *Legitimate Packet List (LPL):* LPL contains a list of SYN-ACK packets that are expected to arrive at the edge router. We use a combination of source IP address and ACK-number to uniquely identify each packet. LPL is updated frequently by forcing SYN packets generated by the victim for a given destination to exit the ISP perimeter at the same edge router at which the corresponding SYN-ACK packet is expected to arrive[3] (this is achievable using loose source routing [75]). The knowledge about expected SYN-ACK ingress point is obtained by keeping an up-to-date table at the victim, in which the ingress point of each received suspicious SYN-ACK packet is recorded. This can be achieved by marking these packets (at their ingress points) by a code that uniquely specifies the edge router through which the packet has been forwarded[4] (i.e., the ingress point).

- *Source Filtering List (SFL):* The SFL contains a list of the reflectors that are being continuously used by an attacker. It is built gradually while passing suspicious packets deterministically. When a suspicious packet is passed, its source address is inserted into the SFL, such that subsequent packets from the same source are filtered unless they are found to be in the LPL.

The PF scheme is activated by the victim upon detection of a TCP-based RDoS attack. All ISP's edge routers are informed (through an authentic multicast message) to activate the PF

[3]If the victim does not have knowledge about the expected ingress point of the corresponding SYN-ACK packet, then the SYN packet is sent normally.

[4]This issue is discussed in subsection 3.2.2.3.

scheme. It is important to mention that the same procedure is performed at all edge routers of a given ISP, and this procedure is applied only to SYN-ACK packets destined to a given victim.

3.2.2.3 The PF Algorithm

In practice, any data structure could be used to implement the LPL and SFL. However, for efficient packet processing and storage, we adopt using Bloom filters [69] to implement LPL and SFL. We refer the readers to Appendix B for a detailed description of Bloom filters. For the purpose of discussion, we use Bloom filters that are characterized by k hash functions and false positive rate of P_f, unless otherwise specified. Fig. 3.2 shows the PF algorithm. In this algorithm, an incoming SYN-ACK packet, P, is inspected for legitimacy (step 1). This is done by inspecting the LPL bits indexed by $H_1(P.\text{source}, P.\text{ack-number})$, $H_2(P.\text{source}, P.\text{ack-number})$, ..., $H_k(P.\text{source}, P.\text{ack-number})$. This combination (i.e., P.source and P.ack-number) should be the same as the combination of the destination's IP address and the (SYN number+1) of the corresponding SYN packet that is supposed to be already inserted in the LPL. If the packet is found to be legitimate, then it is passed (step 1.a), and its entries are erased from the LPL table (step 1.b).

The packet is dropped if its legitimacy cannot be established and its address is found to be in the SFL (step 2). SFL membership test (i.e., to check if certain source is in the SFL) is done by inspecting the bits indexed by $H_1(P.\text{source})$, $H_2(P.\text{source})$, ..., $H_k(P.\text{source})$. Otherwise, the packet is classified as suspicious (step 3) and its address is inserted in the SFL by setting the SFL bits indexed by the same values used for membership test such that future SYN-ACK packets from the same source are classified as attack packets if they fail the legitimacy test. The packet is also marked by a mark that is composed of a 1-bit flag that indicates that the packet is suspicious, and a code that uniquely represents the edge router through which the packet is passed. For an ISP network with N edge routers, $\lceil \log_2 N \rceil$ bits would be required to represent each edge router. Table 3.1 shows the number of edge routers in different ISP networks. It can be seen that the number of bits required for edge router encoding is ISP

dependent, and in most cases, it is less than 13 bits. Such code can be written in the 16 bit ID field of the IP packet header.

Algorithm name: Packet pairing-based filtering (PF)
Input: SYN-ACK packet, P, destined to victim V

---

1. *if* (Packet P is in the LPL)
   (a) pass P
   (b) erase P from the LPL
2. *else if* (P.source is in the SFL) drop P
3. *else*
   (a) insert P.source in the SFL
   (b) mark P with edge router ID
   (c) pass P

Figure 3.2 The PF algorithm. This algorithm is to be performed at each edge router of an ISP network.

The purpose of marking suspicious packets is two fold. (1) By receiving a suspicious packet, SP, the victim knows that the source address of the given packet has been inserted in the SFL of the edge router represented by the router ID obtained in the marked packet. If the packet is found to be legitimate, the victim should send an erase request to that edge router to remove the source address of the packet from its SFL. This is done by resetting the SFL bits indexed by $H_1(SP.\text{source})$, $H_2(SP.\text{source})$, ..., $H_k(SP.\text{source})$. Failing to do so may result in setting all SFL bits as a result of continuous insertions of newly seen sources, which leads to complete blocking of legitimate packets.

(2) The marking at an edge router helps the victim in updating its information about the mapping between different packet sources and ISP edge routers. This information is used to update LPLs whenever a TCP connection is established by the victim. Whenever an edge router receives a SYN packet from the victim itself to establish a connection with certain system outside the ISP perimeter, the combination of the packet destination and (SYN number + 1)

ISP	Number of edge routers (N)	$\lceil \log_2 N \rceil$
AT&T (US)	8044	13
Ebone (Europe)	108	7
Exodus (US)	49	6
Level 3 (US)	875	10
Sprintlink (US)	5990	12
Telstra (Australia)	2249	12
Tiscali (Europe)	182	8
Verio (US)	2846	12
VSNL (India)	10	4

Table 3.1 Number of edge routers in various ISP networks. This information was extracted from the Rocketfuel [81] ISP topology raw traces

is inserted in the LPL by setting the bits indexed by $H_1(P.\text{destination}, P.\text{SYN-number} + 1)$, $H_2(P.\text{destination}, P.\text{SYN-number} + 1)$, ..., $H_k(P.\text{destination}, P.\text{SYN-number} + 1)$.

3.2.2.4 Theoretical Analysis of the PF Scheme

Ideally, the PF scheme should filter all attack packets without causing any collateral damage to legitimate packets. However, due to design and implementation factors of the scheme, and due to Internet traffic unpredictability, ideal operation of the scheme cannot be achieved. The following properties characterize the behavior of the PF scheme.

- PF scheme does not provide instant filtering of attack packets at the moment the scheme is activated. This means that the edge routers will experience some delay before being able to classify incoming SYN-ACK packets with high accuracy. This delay is due to the gradual process of building the SFL at each edge router.

- Ideally, all legitimate SYN-ACK packets destined to the victim are allowed to pass the ISP perimeter. The validity of this property depends on the knowledge of the victim about the expected ingress point for each SYN-ACK packet, and on the *symmetry* of egress/ingress of request/reply packets from/to the victim. This is because failing to

inform the correct edge router about an incoming legitimate SYN-ACK packet may lead to filtering of that packet.

- Ideally, only one packet from each attack source is allowed to pass the ISP perimeter. This is because when the source address of the first attack packet from a given source is inserted in the SFL, all subsequent packets from the same source are going to be filtered. However, this property does not hold always due to *routing instability*, which changes the ingress point (to the ISP network) of incoming packets originating from the same source. If an attack packet from certain source enters the ISP perimeter at an edge router different than the one at which previous packets from the same source arrived, then such packet would be allowed to pass since its source address is not in the SFL of the current edge router[5].

Our aim is to evaluate the PF scheme in terms of the the following performance metrics which map to the previous properties.

- *Transient defenseless period (TDP):* the period since the scheme is activated until all distinct attack sources are identified.
- P_l: probability of filtering a given legitimate SYN-ACK packet.
- P_a: probability of allowing more than one packet from a given attack source (i.e., from a reflector) to pass the ISP perimeter.

TDP Analysis: It is imperative to mitigate the effect of a DoS attack within short period of time. Complete mitigation[6] of an attack is achieved only by identifying all distinct attack sources (i.e., reflectors being used by the attacker) because attack packets will be subject to deterministic dropping if their sources are found to be in the SFL. There is a tradeoff from the perspective of an attacker, between available resources (e. g., available bandwidth), detection avoidance by the reflectors themselves (e. g., high rate SYN packets from certain source may

[5]An exception to this is when a false positive occurs to the packet's source address. In this case the packet would be filtered.

[6]This is in ideal situation.

be considered as an indication of ongoing reflector attack), and the amount of damage desired at the targeted system.

Assuming that a given edge router receives attack packets from n different reflectors with a Poisson rate of λ packets/second from each reflector (i.e., an aggregate rate of λn packets/second), then we need to find the average number of attack packets, M, required to insert all attack reflector addresses in the SFL of a given edge router, ER_x. This problem is an instant of the well known *coupon collector problem.* It has been shown that $M = n(1 + ... + \frac{1}{n-1} + \frac{1}{n})$ [82]. Therefore, the average time to collect all distinct attack reflector addresses can be expressed as $\frac{M}{\lambda n}$. This value represents the average transient defenseless period at router ER_x. In practice, the average time would be larger due to the ramp up behavior observed in DoS attacks in general [78].

P_l **Analysis:** The probability of dropping a legitimate SYN-ACK packet depends on whether the packet source is *new* to the victim (i.e., the first time destined to the victim since the scheme is activated), or it is *old* (i.e., it has been destined to the victim earlier since the scheme was activated). The following analysis focuses on estimating the percentage of legitimate traffic that may get dropped. This analysis applies for packets that arrive after the transient defenseless period.

The importance of an address being new or old reflects the knowledge of the mapping between the source address and the ingress point to the ISP network that contains the victim. Equations (3.1) and (3.2) show the probability of blocking a legitimate packet originating from a new source, P_{bnew}, and the probability of blocking a legitimate packet originating from an old source, P_{bold} respectively.

$$P_{bnew} = P_{asymm} P_f \tag{3.1}$$

$$P_{bold} = P_{inst} P_f \tag{3.2}$$

where P_{asymm} represents the probability of egress/ingress asymmetry (i.e., the probability that a given SYN packet exits the ISP perimeter at certain edge router, while the corresponding SYN-ACK packets enters the ISP perimeter at different edge router), P_{inst} represents the

probability of routing instability, and P_f represents the false positive rate of the Bloom filter that represents the SFL.

The value of P_f depends on different design parameters including m: the size of the SFL, k: the number of hash functions used, and n: the number of source addresses inserted in the SFL. Obviously, the number of reflectors that are being used by the attacker which map to certain edge router depends on the actual network topology, routing protocol, and attacker's choice. Therefore, parameter n is expected to be different for different edge routers. We can, however, make some simplifying assumptions in order to derive an upper bound on the false positive rate of the SFL at each edge router by assuming that up to n_{max} reflectors out of those used by an attacker map to any edge router. For the purpose of discussion, SFL's parameters m, k, and n_{max} were set to $128k$ bits, 4, and 5000, respectively, which corresponds to a P_f of 0.0004, unless otherwise specified.

In order to obtain an expression for the probability of blocking a legitimate SYN-ACK packet, P_l, we define P_{new} to be the probability that the source of a legitimate SYN-ACK packet is new to the victim, and we combine Equations (3.1) and (3.2) to obtain:

$$P_l = P_{new}P_{asymm}P_f + (1 - P_{new})P_{inst}P_f \tag{3.3}$$

It is important to realize that P_{new} is not fixed. Initially, all legitimate SYN-ACK sources are new to the victim. However, as time proceeds, most of the legitimate SYN-ACK sources become old. To study the effect of routing instability and routing asymmetry separately, we fix one of them and vary the other along the possible values of P_{old} (the same as $1 - P_{new}$). Fig. 3.3 shows the effect of routing instability on P_l. It can be seen that the blocking probability of legitimate packets increases when the routing instability increases, which can be explained by recalling that a legitimate SYN-ACK packet coming from an old source cannot be paired with the corresponding SYN packet if it arrives at an edge router different than the expected one.

Fig. 3.4 shows the effect of routing asymmetry on P_l. It can be seen that P_l increases when the routing asymmetry increases, which can be explained by recalling that a legitimate SYN-ACK packet coming from a new source cannot be paired with the corresponding SYN

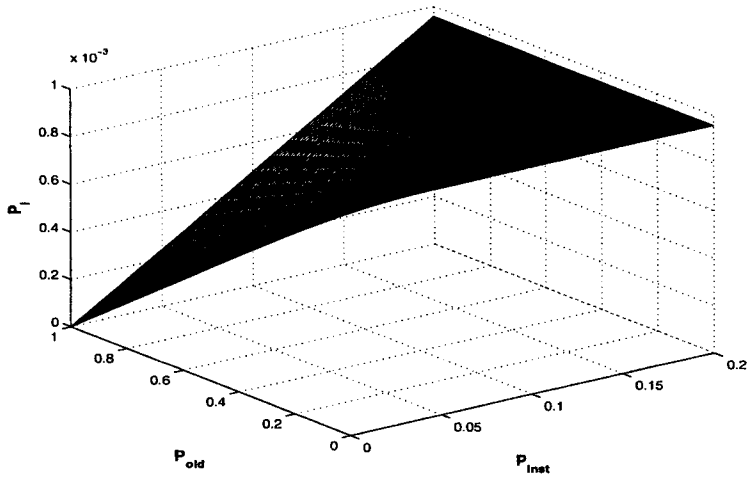


Figure 3.3 Probability of filtering a legitimate SYN-ACK packet (P_l). P_{asymm} and P_f were fixed to 0.2 and 0.0004, respectively.

packet if it arrives at an edge router different than the one at which the SYN packet departed the ISP perimeter. Overall, the small values of P_l, which are in the range of 10^{-3}, indicate that collateral damage under PF scheme is minimal.

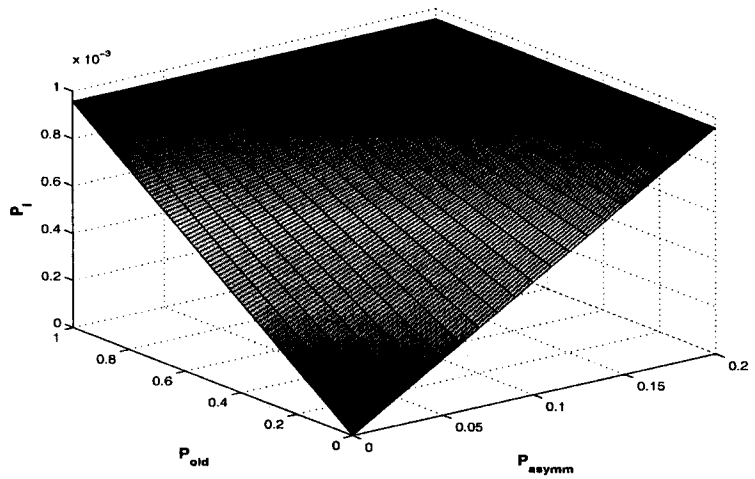


Figure 3.4 Probability of filtering a legitimate SYN-ACK packet (P_l). P_{inst} and P_f were fixed to 0.2 and 0.04, respectively.

P_a **Analysis:** The probability of allowing more than one packet from the same attack source can be expressed as:

$$P_a = P_{inst}(1 - P_f) \tag{3.4}$$

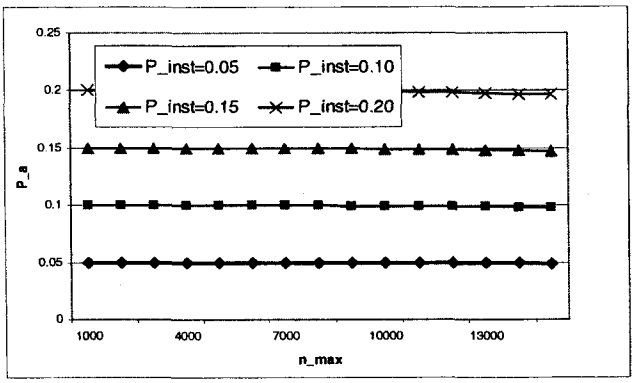


Figure 3.5 Probability of allowing more than one packet from the same attack source (i.e., the same reflector).

For the purpose of discussion, we will consider using a SFL of size $128K$ bits with four hash functions. We assume different values for P_{inst} ranging from 0.05 to 0.2. Fig. 3.5 shows the value of P_a as a function of n_{max}. It is clear that stable routing (i.e., low values for P_{inst}) reduces the chances for more than one attack packet per source to pass the ISP perimeter. We also observe that P_a decreases slightly by increasing n_{max}. In fact, increasing n_{max} beyond the range shown in the figure leads to a sharp decrease in P_a. This can be interpreted by recalling that increasing n_{max} corresponds to higher values of P_f. This represents one of the constraints imposed on the attacker by the PF scheme, which prevents the use of very large number of reflectors, because this would lead to higher P_f and consequently lower P_a. The effect of n_{max} on P_l can be seen in Fig. 3.6 which plots P_l under the same conditions. It is clear that P_l increases slightly (observe that P_l remains in the 10^{-3} range) by increasing n_{max}, which is due to the corresponding increase of P_f. By looking at figures 3.5 and 3.6 together, one can infer the tradeoff regarding n_{max} from attacker's viewpoint.

3.2.3 Scheme II: SYN-Number based Filtering (SNF)

TCP connection establishment procedure is characterized by its deterministic nature regarding the type and content of messages exchanged between the communicating parties. This forms the basis of our second victim-assisted scheme for defense against TCP-based RDoS at-

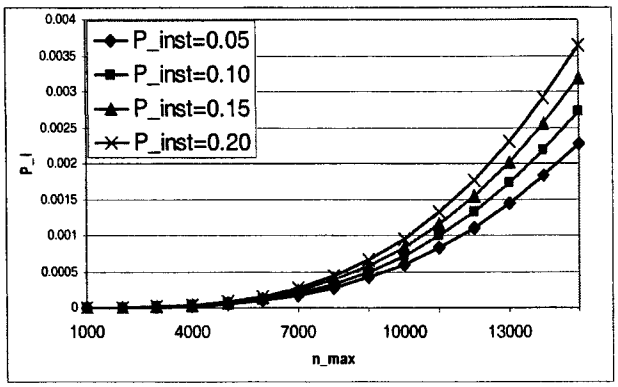


Figure 3.6 Probability of filtering a legitimate SYN-ACK packet (P_l). P_{new} and P_{asymm} were fixed to 0.5 and 0.2 respectively.

tacks. Theoretically, an incoming SYN-ACK packet (destined to the victim) can be validated by inspecting its sequence number to see if it matches the ISN of the corresponding SYN packet plus one. Although such validation could be done very effectively at the victim itself, it would not be useful to stop network resource exhaustion attacks.

The proposed scheme, called Sequence Number based Filtering (SNF), enables edge routers of the ISP network to validate incoming SYN-ACK packets destined to the victim without maintaining state information about *individual* SYN packets sent originally by the victim. The main idea of the SNF scheme is to restrict the choice of the Initial Sequence Numbers (ISNs) of SYN packets generated by the victim to certain pattern, such that corresponding SYN-ACK packets can be validated at the ISP perimeter. In the following subsections, we develop three variants of the SNF scheme to counter both *classical* and *advanced* TCP-based RDoS attacks. As discussed in Chapter 1, classical attacks are those in which the attacker does not react to defenses employed by the victim during attack period, while advanced attacks are those in which the attacker monitors victim's reaction and acts accordingly. We develop a simple analytical models to evaluate the performance of the proposed scheme focusing on the following performance metrics:

- False positive rate (FPR): The percentage of attack SYN-ACK packets that are falsely

allowed to pass the ISP perimeter toward the victim.

- False negative rate (FNR): The percentage of legitimate SYN-ACK packets that are falsely filtered.

Clearly, both metrics need to be minimized.

3.2.3.1 Countering Classical TCP-based RDoS Attacks

In the classical TCP-based RDoS attacks, SYN packets generated by attack nodes will continue to hold ISNs according to the rules specified originally by the attack tool itself. To counter such attacks, we propose the Basic-SNF scheme in which the victim chooses a *secret pattern* for its ISNs. The secret pattern, C_s, is set by fixing a randomly chosen k-bits out of the 32-bits used to represent the sequence number. To activate the schemes, the victim initiates a filtering request by multicasting a message to all edge routers of the ISP network. The message includes the specific secret pattern, C_s, that should be used to validate incoming SYN-ACK packets. Edge routers inspect incoming SYN-ACK packets destined to the victim according to the algorithm shown in Figure 3.7. In this algorithm, if the (ACK number - 1) of the SYN-ACK packet contains the secret pattern, C_s, then the packet is passed. Otherwise, it is filtered.

Algorithm name: Basic-SNF
Input: SYN-ACK packet, P, destined to the victim, V
Output: A decision whether to pass or drop P

1. X=P.ACK_number-1
2. *if* (X contains C_s) pass P
3. *else* drop P

Figure 3.7 Basic-SNF algorithm. This algorithm is performed at each edge router while attack is going on. It is applied only to SYN-ACK packets destined to the victim.

Assuming that the ISNs of attack packets are generated randomly, the probability that a given attack packet will contain a bit pattern that matches the secret pattern currently in

use, C_s, is equal to $\frac{1}{2^k}$, where k is the length of C_s. This implies that the Basic-SNF scheme is very effective against classical attacks. However, a major weakness of the this scheme is that the attacker does not have to discover the actual secret pattern used by the victim. In fact, it is sufficient to intercept a single legitimate SYN or SYN-ACK packet and using its corresponding ISN as an input to subsequent attack packets. In order to address this issue, we propose to change the secret pattern, C_s, in a way that it significantly reduces attacker's chances of launching replay attacks (i.e., using previously used secret pattern). To achieve this, the secret pattern can be changed either periodically where the C_s is changed regularly every T time units or reactively where C_s is changed whenever a replay attack is detected by the victim.

3.2.3.2 Countering Advanced TCP-based RDoS Attacks

In this model, we assume that an attacker experiences, some delay, D_a before being able to reconfigure its attack tool to generate SYN packets that carry valid SYN-numbers to the reflectors (i.e., packets that hold the secret pattern currently in use by the victim). This delay consists of the time required to intercept SYN packets generated by the victim itself, the communication time to the zombies under attacker's control, and the attack tool reconfiguration time. To deal with this type of attacks, we propose two modified versions of the Basic-SNF scheme. The first version involves changing the secret pattern periodically. Hence, it is called the Periodic-SNF. The second version involves changing the secret pattern reactively. Hence, it is called reactive-SNF.

Analysis of Periodic-SNF: In this scheme, we propose changing the secret pattern of the SYN number periodically (i.e., every T time units). This is done by dividing the time since an attack is detected into fixed intervals, each of length T. A secret pattern, C_i, is then assigned for each interval. It is assumed that interval length and secret pattern assignment are known to edge routers via authentic multicasting. Any SYN packet generated by the victim during the i-th interval must hold C_i as part of its ISN. For SYN-ACK packet validation, each edge router acts independently according to the filtering algorithm shown in Fig. 3.8.

The edge router specifies the interval in which the packet to be inspected has arrived (step 1). It is expected that SYN-ACK packets that correspond to SYN packets which are sent at the end of a given interval (e.g., the $(i-1)$-th interval) may arrive at the edge router during the subsequent interval (i.e., the i-th interval). Such packets will be dropped if the validation is done based on the current secret pattern in use (i.e., C_i) alone. To reduce the impact of this problem, a packet is considered to be valid if its secret pattern matches either of C_i or C_{i-1} given that its arrival time falls between $(i-1)T$ and $(i-1+\alpha)T$, where $0 \leq \alpha \leq 1$ (step 3). Otherwise the validation is performed based on C_i alone (step 4).

Algorithm name: Periodic-SNF
Input: SYN-ACK packet, P, destined to the victim, V
Output: A decision whether to pass or drop P

1. $i = \lceil \frac{P.at}{T} \rceil$
2. X=P.ACK-1
3. *if* ($(i-1)T \leq P.at \leq (i-1+\alpha)T$)
   (a) *if* ((X contains C_i) OR (X contains C_{i-1})) pass P
4. *else if* (X contains C_i) pass P
5. *else* drop P

Figure 3.8 Periodic-SNF algorithm. This algorithm is performed at each edge router while attack is going on. It is applied only to SYN-ACK packets destined to the victim. $P.at$ denotes the packet arrival time. P. C_i denotes the secret pattern assigned for interval i. α is the overlapping ratio.

Fig. 3.9 shows two scenarios for connection establishment by the victim assuming α to be zero. In scenario 1, the request (i.e., a SYN packet) is sent by the victim at time t_1 during the $(i-1)$-th interval, the reply (i.e., the corresponding SYN-ACK packet) takes x_1 time units to arrive at the ISP perimeter. Since $(t_1 + x_1 < T)$, the reply is passed. In scenario 2, the reply that corresponds to the request made at t_2 arrives the ISP perimeter at $t_2 + x_2 > T$ (i.e., in the i-th interval. The reply is filtered in this case because it holds a secret pattern (C_{i-1}) that is no longer in use.

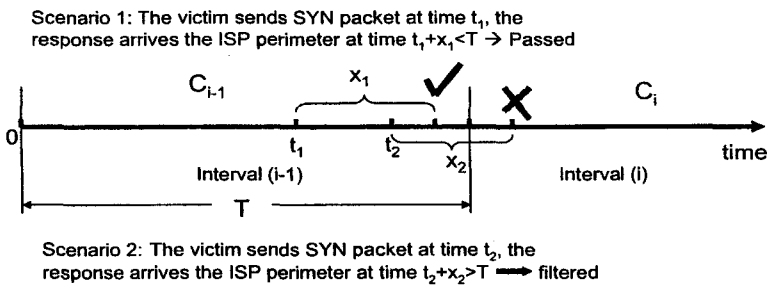


Figure 3.9 The impact of changing the secret pattern of the ISN for connections generated by the victim. The secret pattern is set to C_{i-1} during the $(i-1)$-th interval, and to C_i during the i-th interval.

Generally, because of the attacker's ability to generate valid attack packets after D_a seconds each time a new secret pattern is applied (recall that D_a represents the amount of delay experienced by the attacker before being able to reconfigure its tool), it can be seen that the attacker can pass its packets during the window of time specified by $(1+\alpha)T - D_a$. It is clear that the choice of T and α introduces a tradeoff that shapes the false positive and false negative rates. The following analysis focuses on evaluating both metrics.

FPR for Periodic-SNF ($FPR_{periodic}$): $FPR_{periodic}$ can be expressed as the expected value of the ratio of time in which the attack traffic is allowed to pass the ISP perimeter. Let $A_{periodic}$ represent this ratio. It is clear that:

$$A_{periodic} = \begin{cases} \frac{(1+\alpha)T - D_a}{(1+\alpha)T} & \text{if } D_a < (1+\alpha)T; \\ 0 & \text{otherwise.} \end{cases}$$

Assuming that D_a has an exponential distribution[7] with rate μ_a, the expected value of $A_{periodic}$ is given by Equation (3.5).

$$FPR_{periodic} = E[A_{periodic}] = 1 + \frac{1}{\mu_a(1+\alpha)T}(e^{-\mu_a(1+\alpha)T} - 1) \tag{3.5}$$

It is to be noted that the value of $FPR_{periodic}$ does not depend on the actual values of μ_a and T. Instead, it depends on the relationship between them which can be expressed as $\mu_a = \frac{1}{rT}$, where r is a positive real number. Fig. 3.10 plots the effect of α on the percentage of

[7]This assumption is made because actual measurements of D_a are not available.

false positives for different values of r. It is obvious that as α increases, $FPR_{periodic}$ increases as well in all cases. This can be explained by recalling that increasing α allows the attacker to pass more attack packets. Also, it is clear that $FPR_{periodic}$ increases by increasing μ_a (equivalent to decrease r), given that T is fixed. This is expected because larger values of μ_a, indicates faster reaction by the attacker. In practice, it is difficult to predict attacker's reaction rate. Therefore, fixing T to a high value may result in a large false positive rate. This is due to the fact that increasing T provides larger window of time for the attacker to pass his traffic toward the victim, especially when his reaction rate is very high.

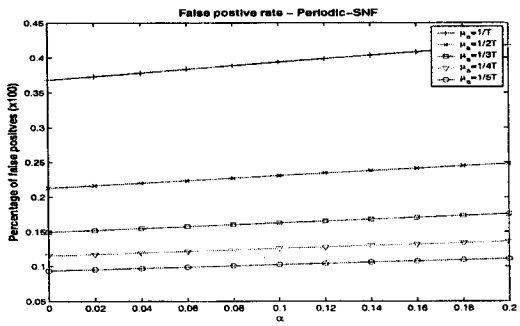


Figure 3.10 The effect of α on the percentage of false positives for different rates of attacker's reaction as captured by the relation between μ_a and T.

FNR for Periodic-SNF ($FNR_{periodic}$): In order to gain an insight on the performance of the periodic SNF scheme in terms of the false negative rate, $FNR_{periodic}$, we assume that the victim is generating TCP connections at a Poisson rate of λ per unit time, and that the distribution of connection response time, X, (defined as the time between sending the SYN packet by the victim and receiving the corresponding SYN-ACK at the perimeter) is given by $X = c + Y$, where c is a constant and Y is a variable component. This assumption is based on the definition of the round trip time (RTT), which consists of a *fixed* component given by the sum of link propagation latencies and transmission and processing delays on all nodes in the forward and reverse direction, and additional *variable* components due to queuing and processing delays at overloaded routers and end-hosts. Assuming that Y can be expressed as the summation of n identical exponentially distributed random variables each with rate β, Y is said to be an Erlang random variable with parameters β and n.

Based on these assumptions, we are interested in finding an expression for $FNR_{periodic}$, which is the same as the probability of filtering a legitimate packet. The following Equation represents the probability distribution function of X:

$$f_X(x) = \begin{cases} \frac{\beta^n (x-c)^{n-1} e^{-\beta(x-c)}}{(n-1)!} & \text{if } x > c; \\ 0 & \text{if } x \leq c. \end{cases}$$

We proceed in our analysis for $x > c$. The cumulative distribution function (CDF) of X is given by:

$$F_X(x) = 1 - \sum_{k=0}^{n-1} \frac{e^{-\beta(x-c)} \beta^k (x-c)^k}{k!} \tag{3.6}$$

It can be seen that Pr(a legitimate packet P is filtered | P is sent at time t) = Pr($X > (1+\alpha)T - t$ | P is sent at time t) = 1 - Pr($X \leq (1+\alpha)T - t$ | P is sent at time t). We call this probability the conditional probability of legitimate packet filtering $CPLP_{periodic}$. This probability is given by the following Equation:

$$CPLP_{periodic} = 1 - F_X((1+\alpha)T - t) \tag{3.7}$$

Substituting (3.6) with ($x = (1+\alpha)T - t$) in 3.7, we obtain:

$$CPLP_{periodic} = \sum_{k=0}^{n-1} \frac{e^{-\beta((1+\alpha)T-t-c)} \beta^k ((1+\alpha)T - t - c)^k}{k!} \tag{3.8}$$

Unconditioning on the time at which each request is generated, we obtain:

$$FNR_{periodic} = \frac{1}{T} \int_0^T CPLP_{periodic} dt \tag{3.9}$$

In practice, each end-to-end path is characterized by its own minimum possible RTT (i.e., the fixed component c of the RTT along the given path). Also, the variable component represented by the Erlang random variable, Y, has its own parameters β and n for each TCP segment due to the diversity of destinations and the variability of load on network routers. However, for the purpose of obtaining an upper bound on the probability of a legitimate SYN-ACK packet being filtered, due to secret pattern change, we can assume worst case values for c, β, and n.

We performed extensive simulation experiments to evaluate the value of $FNR_{periodic}$ for several values of α and T. It is to be noted that $FNR_{periodic}$ does not depend on λ since all arrivals are independent. However, the value of λ was fixed to 0.004 just for the sake of experimentation. To obtain a worst case value for the random variable X, we assigned a value of 1 second to the constant c, which was found to be the maximum reported minimum RTT [83]. Also, to eliminate the variability of parameters n and β, we assigned the values 30 and 2 for them, respectively. This corresponds to the maximum reported path length [84], and a relatively large queuing and processing delay at intermediate routers. α was varied along the x axis in the range of 0 to 0.2. Fig. 3.11 plots the effect of α on the percentage of false negatives for different values of T.

By recalling that αT represents the amount of extra time in which a secret pattern of a given interval remains valid throughout the subsequent interval, it is easy to explain the decrease of the false negative rate by increasing α. This is expected since secret pattern change will be less frequent, resulting in lower percentage of legitimate packets being dropped. The results we presented about $FPR_{periodic}$ and $FNR_{periodic}$ lead to an important conclusion: The Periodic-SNF scheme is expected to perform well in terms of false negative rate. However, it is expected to perform well in terms of false positive rate only if the value of T happens to be close to the rate at which the attacker reconfigures its tool with a valid secret pattern. Since this is difficult to predict in practice, we propose the following reactive solution.

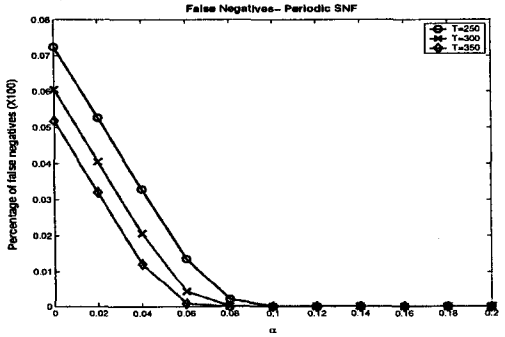


Figure 3.11 The effect of α on the percentage of false negatives for different values of T. μ_a, λ, β, and n were fixed to 0.004, 5, 2, and 30, respectively.

Analysis of Reactive-SNF: Periodic changing of secret pattern may result in unnecessary legitimate packet loss, especially if $D_a \geq (1+\alpha)T$. As an alternative solution, we propose changing the secret pattern reactively. In this solution, we assume that the victim incurs some delay, D_v, before detecting that the attacker is generating attack packets that hold the secret pattern currently in use. Therefore, the attacker can pass its packets during the window of time specified by D_v. It is to be noted that the secret pattern of the legitimate packets SYN number would change every $D = D_a + D_v$ time units. For simplicity, we do not adopt the notion of overlapping ratio (α) in this version of SNF. Fig. 3.12 illustrates the reactive solution in which the secret pattern of the ISN for connections generated by the victim is changed from C_s to C_{s*} after D time units. The figure also shows that a legitimate response packet may still be filtered due to the secret pattern change.

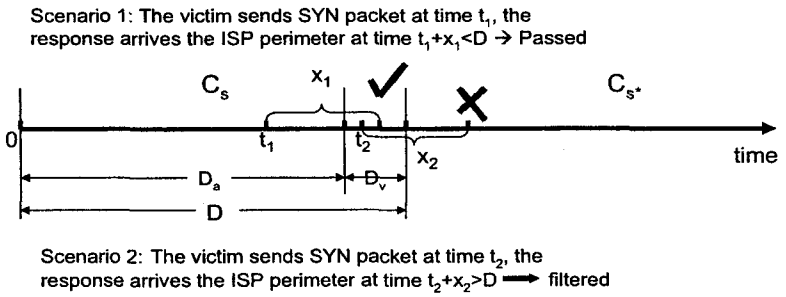


Figure 3.12 The reactive SNF scheme. The secret pattern of the ISN for connections generated by the victim is changed from C_s to C_{s*} after D units of time.

The distributions of D_a and D_v are assumed to be exponential with rates μ_a and μ_v, respectively. This assumption is made because actual measurements of D_a and D_v are not available. The distribution of D is given by:

$$f_D(t) = \begin{cases} \frac{\mu_a \mu_v}{\mu_a - \mu_v}(e^{-\mu_v t} - e^{-\mu_a t}) & \text{if } \mu_a \neq \mu_v; \\ \mu^2 t e^{-\mu t} & \text{if } \mu_a = \mu_v = \mu. \end{cases}$$

As in the previous subsection, we are interested in finding the false positive and false negative rates.

FPR for Reactive-SNF ($FPR_{reactive}$): Similar to the Periodic-SNF, the false positives rate can be expressed as the expected value of the ratio of time in which the attack traffic is allowed

to pass the ISP perimeter. Let $A_{reactive}$ represent this ratio. It is clear that:

$$A_{reactive} = \frac{D_v}{D_a + D_v} \tag{3.10}$$

It can be shown[8] that the expected value of $A_{reactive}$ is as follows:

$$FPR_{reactive} = E[A_{reactive}] = \frac{\mu_a \mu_v}{(\mu_v - \mu_a)^2}(\log \frac{\mu_v}{\mu_a} + \frac{\mu_a}{\mu_v} - 1) \tag{3.11}$$

The actual values of μ_a and μ_v depend on the reaction of the attacker and the victim, respectively. We can generally assume that detecting that the attacker is using valid ISNs is much faster than the process of reconfiguring attack tools to generate attack packets with valid ISNs. This means that μ_a should be less than μ_v. In general, we can say: $\mu_v = f\mu_a$. Where $f \geq 1$ is called the reaction factor. This means that victim's reaction is f times faster than attacker's reaction. Taking this assumption into account, Equation (3.11) can be rewritten as:

$$FPR_{reactive} = E[A_{reactive}] = \frac{f}{(f-1)^2}(\log f + \frac{1}{f} - 1) \tag{3.12}$$

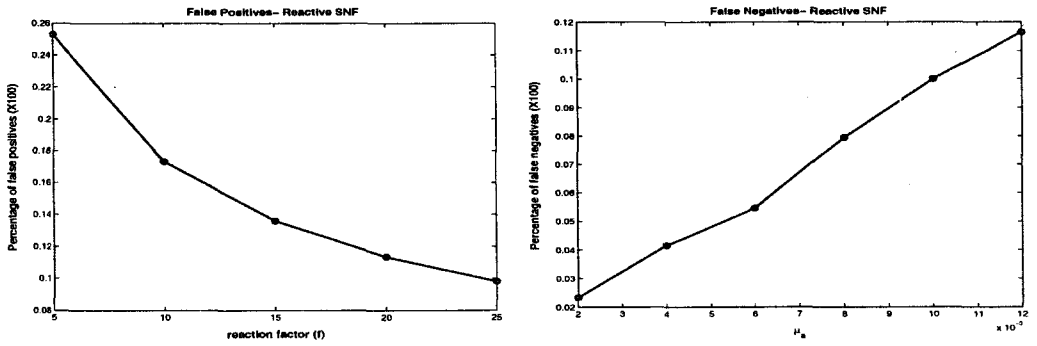
FNR for Reactive-SNF ($FNR_{reactive}$): The $FNR_{reactive}$ is the same as the probability of filtering a legitimate packet under Reactive-SNF. This can be obtained by unconditioning on T in Equation (3.9) as follows:

$$FNR_{reactive} = \int_0^\infty FNR_{periodic} f_D(T) dT \tag{3.13}$$

Fig. 3.13 (a) captures the effect of the reaction factor on the false positive rate. As expected, the victim's reaction determines the percentage of attack SYN-ACK packets that can pass the ISP perimeter toward the victim. It can be seen that faster reaction, represented by higher values of f, results in lower false positive rate. We performed extensive simulation experiments to evaluate the false negatives rate. Fig. 3.13 (b) shows the effect of μ_a on the false negatives rate. The reaction factor, f, is assumed to be 10 to indicate that the victim is 10 times faster than the attacker in detecting attackers discovery of the secret pattern in use. Small values of μ_a correspond to large values for delay experienced by the attacker (i.e., D_a) before being able to reconfigure its tool with a valid secret pattern. In such cases, the secret

[8]The derivation of $FPR_{reactive}$ is provided in appendix A.

pattern change by the victim will be less frequent. This explains the low false negatives rate when μ_a is relatively small (e.g., 0.005). However, if the attacker is very fast in secret pattern discovery, the victim will be forced to change its secret pattern more frequently, leading to higher false negatives rate, because many legitimate SYN-ACK packets will be dropped after each secret pattern change. Different than the periodic SNF which performs well only if the secret pattern change period is close to D_a , the reactive SNF achieves the best of both worlds; low percentage of false positive and false negative regardless the value of D_a.



(a) (b)

Figure 3.13 (a) Effect of the reaction factor, f, on the false positive rate. (b) Effect of μ_a on the false negative rate. λ and f (the reaction factor) were fixed to 2 and 10, respectively.

3.3 Protocol Determinism-based DoS Mitigation

In this section, we employ the proposed concept of protocol-determinism in a novel scheme for SYN flooding attack mitigation. The scheme, called *intentional dropping based filtering*, is based on the observation of *client's persistence* (i.e., client's reaction to packet loss by subsequent retransmissions) which is very widespread as it is built in TCP's connection setup. The main idea in this scheme is to *intentionally drop* the first SYN packet of each connection request. Subsequent SYN packet from a request is passed only if it adheres to the TCP's timeout mechanism. Our analysis shows that the proposed scheme reduces attacker's effective attack rate significantly with an acceptable increase in connection establishment latency.

The rest of this section is organized as follows: An overview of SYN flooding attacks is provided in subsection 3.3.1. Previous work is discussed in subsection 3.3.2. The proposed

scheme is described in subsection 3.3.3. Performance studies are discussed in subsection 3.3.4.

3.3.1 An Overview of SYN Flooding Attacks

TCP continues to be the dominant transport protocol used by several popular Internet applications (e.g., http, email, and ftp). Recent studies show that TCP carries 95% of today's Internet traffic and 80% of the total number of flows in the Internet [77]. Therefore, it is no wonder that TCP-based DoS attacks are the most common. TCP traffic itself can be generally classified as control traffic (e.g., SYN, SYN-ACK, FIN, etc.) and data traffic. Although an attacker can flood its target by any type of traffic, the magnitude of damage caused to the target and the difficulty of attack mitigation is maximized when TCP control packets are used as the weapon of an attack. Among TCP control packets, SYN packets are the most dangerous if used by an attacker in the form of SYN flooding, because every faked SYN packet, in addition to bandwidth consumption, can disproportionately consume a system's resources for a disproportional amount of time.

In SYN flooding attacks [19, 85], the victim is overwhelmed by very large number of spoofed SYN packets (i.e., packets that hold spoofed IP source addresses). This consumes victim's bandwidth and forces it to allocate resources for connections that will never complete. A TCP listen port has a finite number of slots in its listen queue and normally that number of slots is relatively small. When an attacker sends enough faked SYN packets, the listen queue can be fully occupied and subsequently deny any legitimate SYN packet from entering into the listen queue. Therefore, until the connection establishment process times out, a disproportional amount of system resources are occupied: a slot in the attacked port's listen queue, memory to maintain connection information, and CPU and network bandwidth to retransmit the SYN-ACK packet.

As in any other DoS mitigation schemes, the main challenge in SYN flooding mitigation is to accurately identify attack packets and filter them without causing collateral damage to legitimate traffic designated to the victim. The proposed scheme takes the TCP client's persistence behavior into account to distinguish legitimate connection requests from attack

connection requests far away from the victim, leading to protection from bandwidth exhaustion attacks as well as victim's exhaustion attacks. We perform simple analysis to evaluate the proposed scheme focusing on its effectiveness in reducing attacker's effective attack rate, and showing its impact on connection establishment latency.

3.3.2 Previous Work

Previous work on mitigating SYN flooding attacks aimed at avoiding allocating resources for TCP connections before proving their legitimacy (e.g., SYN cookies [86] and SYN cache [87]), or by freeing the allocated resources when certain conditions are satisfied (e.g., Synkill [85]). SYN cookies work to alleviate SYN floods by calculating cookies that are functions of the source address, source port, destination address, destination port, and a random secret seed. When a SYN packet is received, the server calculates a SYN cookie and sends it to the requesting client as part of the SYN-ACK packet without allocating resources for that request. When ACK packet is received, the connection is established if a valid cookie is included.

With the SYN cache, when the SYN backlog queue overflows, a minimal amount of state is stored for each SYN request in a data structure known as the SYN cache, and a SYN-ACK response is sent. If a valid ACK comes back, a complete connection is created. If there is no route or a TCP RST or ICMP Unreachable comes back, the entry is deleted. Otherwise, the entries will just time out. In both cases, since state is not kept, the SYN backlog queue is not exhausted, and normal TCP communications can continue. Synkill, on the other hand, frees allocated resources by following two methods: (1) it basically sends RST packets whenever it observes connection establishment attempts from suspicious IP addresses, and (2) it completes TCP connections by generating the third message of the three-way handshake, and sending it to the destination. The major problem with these approaches is that they cannot alleviate bandwidth exhaustion attacks resulting from SYN flooding.

Generally, filtering of spoofed IP packets leads indirectly to TCP flooding mitigation. For example, in ingress filtering [31], routers are configured to block packets that arrive at the edge router of the source network with illegitimate source addresses. This may violate some

existing setups and protocols such as Mobile IP and multi-homing. It is also difficult to convince ISP administrators to support ingress filtering because the benefit is not felt directly by the deploying ISP. Another example is the SAVE protocol [33], which is designed to provide routers with the information needed for source address validation. The main problem of this protocol is that legitimate packets may be filtered even in the absence of an attack. This is due to routing instability which leads to errors in the source address validation tables maintained at the SAVE enabled routers. In general, such schemes require large scale deployment to prevent IP source address spoofing efficiently.

3.3.3 Intentional Dropping-based Filtering

3.3.3.1 Intentional Dropping Concept

Consider the client server model shown in Fig. 3.14. It is assumed that packets from client C to server S pass through router R. When the client issues a "connect()" command, the stack sends a SYN packet, P. If R is overloaded, it may drop packet, P. We call this dropping *accidental* because it is not under the router's control. However, if R is not overloaded and yet decides to drop packet, P, then we call this dropping *intentional* because it is under the router's control. The effect of dropping packet, P, either accidentally or intentionally, is the same from the client's viewpoint. Since the server does not get the SYN packet, the client never gets the corresponding SYN-ACK packet. So the client waits for some time, T_0, thinking that the packet might be lost on its way, and sends a SYN packet again. This behavior is known as *client persistence*. We assume that end-points adhere to a TCP-Reno style congestion control mechanism [88]. Therefore, the client initiates subsequent retransmissions before aborting its connection. According to [89], the initial timeout value before the first connection attempt is $T_0 = 3$ seconds. After each retry, the amount of time to wait is doubled (i.e., the time to wait after the i-th retransmission is $T_i = 2^i T_0$ seconds, $i = 1, 2, ...$).

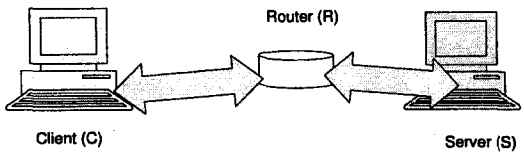


Figure 3.14 Client-Server model. Bidirectional arrows represent communication channels between C, R, and S.

3.3.3.2 Intentional Dropping-based Filtering

A recent study by Jamjoom et. al., [90] shows that client persistence is very widespread as it is built in TCP's congestion control. This behavior can be observed whenever network congestion occurs. For example, during flash crowd event (FCE), very large number of *legitimate* clients try to establish connections with a popular server. Theses clients keep retransmitting their SYN packets until they gain access or they give up. Also this behavior is observed during a SYN flooding attack, where very large number of *fake* SYN packets mixed with few legitimate SYN packets overload a given server. In this scenario, attackers continue to flood the server, and legitimate clients, due to their persistence behavior, keep trying to gain access to the targeted server before giving up. In both cases, we emphasize that SYN packets are the main contributing factor to network and end server congestion.

During a SYN flooding attack, we apply intentional dropping on incoming SYN packets destined to the victim to prevent network congestion, and yet cause client persistence, which is a basic characteristic of legitimate clients. Typically, it is impossible to distinguish legitimate SYN packets from attack SYN packets by inspecting each packet alone. However, by taking advantage of the client persistence behavior, we can make an educated guess of whether a given connection request is malicious or legitimate. *We emphasize here that the distinction is made at the connection level rather than at the packet level.* The main idea is to *intentionally drop* the first SYN packet of each connection request. Subsequent SYN packet from a request is passed only if it adheres to the TCP's time out mechanism. Since attackers do not adhere to TCP's timeout mechanism, and in most cases they assign random addresses to their SYN packets, all SYN flooding packets are supposed to be filtered. In contrast, due to their persistence behavior, legitimate clients can be identified and allowed to pass one SYN packet per connection request.

Supporting intentional dropping requires state information to be maintained about connection requests and their arrival times. It is clear that performing this scheme at the victim itself reduces resource exhaustion without eliminating bandwidth exhaustion. However, performing it at the edge routers[9] of an ISP network has the advantage of eliminating bandwidth exhaustion as well as victim's resource exhaustion[10]. This conclusion is based on the assumption of *routing stability*, referred to as the situation of having the same ingress router for all packets sent from a given source to a destination located in the ISP network, which implies that there is a mapping between attack nodes and the edge routers of the ISP's edge routers. By this mapping we mean that each subset of attack nodes have to forward their attack packets through the same edge router during the attack period.

3.3.3.3 An Illustrative Example

To illustrate the operation of the proposed scheme, consider the two timing diagrams shown in Fig. 3.15. These diagrams reflect the reaction of a legitimate client (X) and an attacker (Y) to intentional dropping. It is assumed that both of them forward their traffic to server S through edge router R. Our objective is to show how router R can distinguish a legitimate request from a fake request. The following two cases are considered:

- Legitimate request: At time t_1, X initiates a TCP connection to S. The corresponding SYN packet, P_X^1, arrives the router R at time $at_1 = t_1 + t_{XR}^1$, where t_{XR}^1 represents the time it took the packet to travel from X to R. At this instant, R drops P_X^1 intentionally, and records connection request information, such as: connection ID and arrival time. As a result, X repeats its request by sending another identical SYN packet, P_X^2, at time $t_2 = t_1 + T_0$. When the packet arrives R at time $at_2 = t_2 + t_{XR}^2$, it is considered to be valid if the following inequality holds: $T_0 - \Delta \leq (at_2 - at_1)$. Theoretically, $t_{XR}^2 = t_{XR}^1$. However, due to variability of network load, the two values are expected to be different. Δ, called the delay tolerance parameter, is introduced here to tolerate this difference.

[9]TCP header should be accessed by edge routers.

[10]In this dissertation, we assume that the scheme is to be implemented at the edge routers.

The inequality requires the interarrival time between the first two SYN packets of a given connection request to be at least $T_0 - \Delta$.

- Attack request: We assume that Y attacks S by sending randomly spoofed SYN packets at a constant rate of 1 packet/second. Applying intentional dropping to incoming SYN packets (P_A^1, P_B^1, P_C^1, etc. with spoofed source addresses A, B, C, etc.), at router R, leads to filtering of all of them because each SYN packet represents a new connection in R's viewpoint. Even if subsequent SYN packets hold the same spoofed source address, they are filtered because their interarrival time is $\leq T_0 - \Delta$.

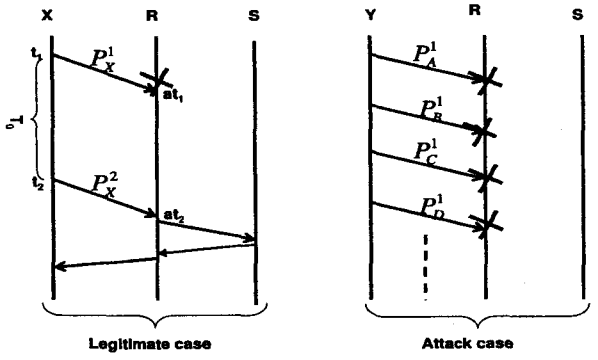


Figure 3.15 Timing diagrams that show the impact of intentional dropping on connection establishment. Left: legitimate request. Right: attack requests. P_S^i represents the i-th SYN packet with source address S.

3.3.3.4 SYN Packet Filtering Algorithm

By taking client's persistence into account, connection establishment request can be viewed as a sequence of SYN packets generated repeatedly at time instants specified by the TCP's congestion control mechanism until a response is received or a maximum number of retransmissions is reached. Therefore, SYN packets can be generally categorized into *classes* that reflect the client's persistence level. A class of a packet is defined as the actual or estimated number of retransmissions of that packet. For example, class 0 SYN packets represent *new* connection requests, class 1 SYN packets represent *repeated* connection requests for the first time (i.e., first retransmissions), class 2 SYN packets represent *repeated* connection requests

for the second time, and so on. Generating the i-th SYN packet for a particular connection, implicitly implies the loss of the previous $(i-1)$ SYN packets of that connection[11].

For security purposes, the proposed scheme can be generalized such that intentional dropping is applied to the first k consecutive SYN packets from each connection request, where k is called the *dropping parameter*. In this case, subsequent SYN packet from each request (i.e., the $(k+1)$-th) is passed only if it adheres to the TCP's time out mechanism. Obviously, the value of k has direct impact on the connection establishment latency as it determines how many SYN packets to drop within each connection. The effect of k is discussed in section 3.3.4.

The proposed scheme is initiated by the victim upon detection of SYN flooding attack. The victim informs its ISP's edge routers (e.g., through an authentic multicast message) to activate the filtering scheme. Fig. 3.16 shows the SYN packet Filtering Algorithm (SFA) to be performed by each edge router of the victim's ISP. It is important to mention that this algorithm is applied only to SYN packets destined to the victim. When a SYN packet, P, is received, its class, i, is determined (step 1). Clearly, if P is a new connection request, then its class is equal to zero. However, if P is a repeated connection request, then its class is equal to the class of the preceding SYN packet of the same connection plus one[12]. Based on the class of the packet, i, and the dropping parameter, k, a decision is taken whether to pass the packet or to drop it. If $(i < k)$ (step 2), P is dropped and request information, such as connection ID, arrival time, and class are recorded. If $(i = k)$ (step 3), the packet is passed given that its interarrival time is at least $2^{i-1}T_0 - \Delta$. It is to be noted that the value $2^{i-1}T_0$ represents the timeout value after retransmitting a packet of class $i-1$. The packet is dropped if it does not adhere to the timing constraints (step 4).

3.3.4 Performance Evaluation

The proposed scheme has a dual impact. On the positive side, it reduces the attack rate significantly. On the negative side, it increases the connection establishment latency for legitimate clients. We assume that clients would favor extra latency to gain access to a web

[11]This claim is based on the assumption that packet loss is observed on the forward direction only.

[12]We assume that connection request information is already recorded at the edge router.

Algorithm name: SYN filtering algorithm (SFA)
Input: SYN packet, P, destined to the victim, V
Output: A decision whether to pass or drop P

1. i= class of packet P
2. *if* $(i < k)$
   (a) drop P
   (b) record (P.ID, $P.at$, i)
3. *else if* $(i = k$ AND $P.iat \geq 2^{i-1}T_0 - \Delta)$
   (a) pass P
4. *else* drop P

Figure 3.16 SYN packet filtering algorithm. P represents SYN packet destined to the victim, V. $P.ID$ denotes the packet's ID, $P.at$ denotes packet's arrival time, $P.iat$ denotes packet's interarrival time. k is the dropping parameter, and Δ is the delay tolerance parameter.

server over a complete blocking of service. We quantify the performance of the proposed scheme in terms of the following performance metrics[13]:

3.3.4.1 Attacker's Effective Attack Rate (EAR)

EAR is defined as the number of attacker's SYN packets that pass filtering per time unit. This should be distinguished from attacker's actual attack rate (AAR) which is defined as the number of attacker's generated SYN packets per time unit. Theoretical maximums for AAR depend on the connection type (e.g., using analog modem, ISDN, T1, the maximum AAR is 87, 200, 2343 SYNs/sec, respectively [91]). Under intentional dropping scheme, what determines the EAR is the way the attack is performed. In our analysis, we consider the following types of SYN flooding attacks:

- Type 1: Attack SYN packets are generated at the rate specified by the attacker with randomly spoofed source addresses.

[13]It is to be noted that the aggregate traffic before and after applying the scheme will increase by the same amount approximately due to packet retransmissions in both cases.

- Type 2: Attacker mimics the behavior of a legitimate client. Therefore, attack SYN packets that hold the same spoofed source address are generated at time instants as if the attacker is experiencing real packet loss.

- Type 3: Attacker mimics the behavior of a legitimate client for a group of spoofed source addresses. This type is similar to type 2 except that the attacker changes the source addresses alternatively.

Currently, type 1 is the most common as attacker's are not aware of the proposed scheme. However, type 2 and type 3 are expected to appear if intentional dropping is deployed as a defense mechanism. It is easy to see that EAR is exactly zero for type 1. This is due to the fact that the first k SYN packets are dropped from each connection request. In fact, it would be sufficient for k to be equal to one for the EAR to be zero. In type 2, the attacker is able to pass one SYN packet every $\sum_{j=0}^{k-1} 2^j T_0$ seconds. Therefore,

$$EAR = \frac{1}{\sum_{j=0}^{k-1} 2^j T_0} \ldots\ldots\ldots packets/second \tag{3.14}$$

This explains why we introduced the dropping parameter, k. Fig. 3.17 (a), supports our intuition that intentional dropping scheme results in drastically low EAR for type 2 attacks[14]. It also shows the effect of the dropping parameter, k, on EAR; k denotes the number of SYN packets that are intentionally dropped, it is varied from 1 to 4 as most TCP implementations abort connection establishment after 4 to 6 failures. As expected, larger k results in lower EAR, because a single SYN packet of a given connection request is passed on the k-th attempt. In its current form, intentional dropping scheme does not mitigate type 3 attacks effectively. Extensions to the proposed scheme that would make it effective against type 3 attacks are going to be the focus of our future research.

3.3.4.2 Connection Establishment Latency Increase (CELI)

Originally, in addition to the round trip time, connection establishment latency is due to SYN packet loss in the forward direction (i.e., client to server), SYN-ACK packet loss in the

[14]It is to be noted that the EAR shown in the figure is for a single attacker.

reverse direction (i.e., server to client). The loss in both directions is accidental as it is caused by network congestion. Obviously, intentional dropping of SYN packets in the forward direction increases the connection establishment latency. It can be seen that accidental dropping affects the operation of the SFA algorithm in the sense that SYN packet's class (defined in subsection 3.3.3.4) can be inaccurate if SYN packets experience congestion loss. For example, if the second SYN packet of a given connection request is lost before reaching the intentional dropping filter, the sender will retransmit the packet after $2T_0$ seconds. The retransmitted packet is the third from sender's view point. However, when it reaches the intentional dropping filter it will be wrongly classified as the second packet. To avoid this confusion, we modify the SFA to become aware of congestion loss. The only modification required is to determine packet's class, i, based on the interarrival time, T, of successive SYN packets for the same connection attempt. Theoretically,

$$T = 2^{i-1}T_0 \tag{3.15}$$

Therefore,

$$i = 1 + \log_2(\frac{T}{T_0}) \tag{3.16}$$

However, since T is not going to be exactly $2^{i-1}T_0$, the fraction $\frac{T}{T_0}$ must be rounded to the closest exponent of 2 before substituting in Equation (3.16). It is to be noted that this analysis does not take into consideration consecutive packet loss from the same connection.

The maximum connection establishment latency increase, $CELI_{max}$, happens when there is no congestion loss for the first k SYN packets of a given connection request. Therefore,

$$CELI_{max} = \sum_{j=0}^{k-1} 2^{j-1}T_0 \ldots\ldots\ldots \; seconds \tag{3.17}$$

Fig. 3.17 (b) shows the value of $CELI_{max}$ that correspond to k intentional droppings of SYN packets that belong to certain connection request. It can be seen that as k increases, $CELI_{max}$ increases drastically. This increase, in general, is within acceptable range (3 to 27 seconds) that are typically tolerable by legitimate clients. It is also important to observe the tradeoff introduced by k between EAR and $CELI_{max}$.

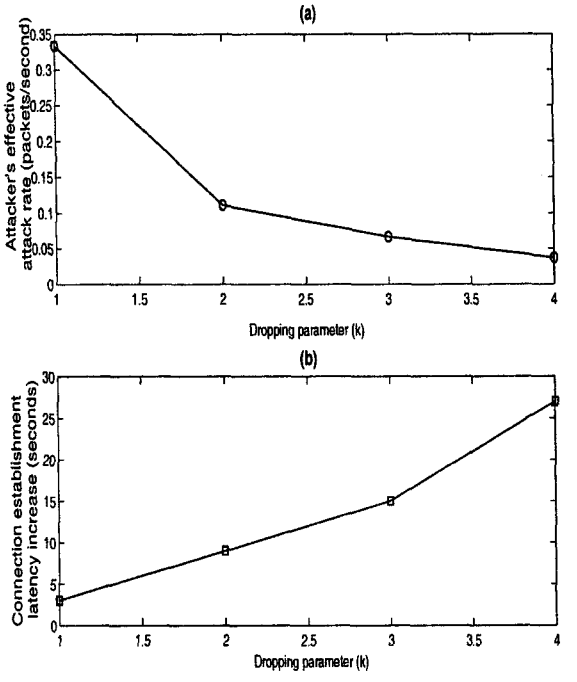


Figure 3.17 (a) Effect of the dropping parameter, k, on the attacker's effective attack rate. (b) Effect of the dropping parameter, k, on the maximum connection establishment latency increase.

3.4 Practical Considerations

There are several important issues that determine the practicality of the DoS mitigation schemes presented in this chapter. In this section, we discuss the issue of inspecting the TCP's packet header, which is required by the three proposed schemes. Also, we discuss issues related to the practicality of the PF scheme, and the impact of restricting the ISN to a specific pattern, which is a specific issue to the SNF scheme.

3.4.1 The Need to Perform TCP's Header Inspection

The proposed schemes require that all edge routers of the ISP network that contains the victim to inspect each packet closely enough to determine if it is a TCP SYN-ACK packet (in the case of TCP-based RDoS attacks), or if it is a TCP SYN packet (in the case of SYN flooding attacks), so the scheme can more closely analyze the packet to see if it is a legitimate or not. This requires, at least for all TCP packets destined to the victim, examining the TCP header fields. Although modern routers [92] have started implementing this capability, most of

the existing edge routers may not have this capability. So installing such mechanisms at those routers would slow them down. To reduce the overhead imposed on edge routers, a lightweight algorithm similar to the one proposed in [93] could be used to distinguish TCP control packets from TCP data packets. Based on that algorithm, a router can tell TCP control packets from data packets without accessing the TCP header by checking the "total length field" in the IP header. If the total length of an IP packet is 40, then it is most probably a TCP control packet (given that its protocol type is TCP and its fragmentation offset is zero). By following this approach, among the IP packets destined to the victim, only TCP control packets undergo TCP header inspection by edge routers. Once a TCP control packet is identified, the edge router has to inspect the corresponding flags in the TCP header to determine its type. This implementation based modification reduces the overhead of TCP header inspection. The actual TCP header inspection can be done by the router itself or by a special device attached to it.

3.4.2 Practicality of the PF Scheme

In this subsection, we consider the following issues, which basically determine the practicality of the PF scheme.

- *Generality of the PF scheme:* It is important to emphasize that the PF scheme is not specific to TCP-based reflector attacks since it has the ability to filter incoming reply packets of any protocol type (assuming that it would be activated according to the attack packets type). In practice, an attacker can just generate random packets from his zombies and flood the target's link. If the attacker particularly likes sending TCP SYN-ACK packets, for some reason, he can fabricate them and send them directly to the target without the help of reflectors. Although this is generally true, it is not useful for the attacker because: (1) the amplification effect cannot be achieved this way, and (2) the zombies under attacker's control can be located if a traceback scheme is employed.

- *Avoiding filtering of legitimate packets:* We have shown in Subsection 3.2.2.4 that legitimate reply packets may get filtered due to routing asymmetry, routing instability, or lack of knowledge about the edge routers at which these packets are expected to arrive.

Therefore, all reply packets originating from a certain source may get filtered. To avoid this kind of filtering, the PF scheme can be modified as follows. Instead of notifying a single edge router about an expected reply packet, the victim can notify several edge routers at the same time, which means that packet pairing can be performed at any of these routers. Given that an ISP network usually has few edge routers that are connected to other ISP networks, the number of notification messages that are to be sent by the victim for each request it initiates will be limited as well.

- *SFL table blackholing:* An attacker may attempt to force an edge router to filter legitimate reply packets originating from certain sources by filling the SFL table at that edge router with the addresses of these sources. Therefore creating a black hole in that SFL table. Although nothing can prevent an attacker from doing so, the PF scheme can still pass legitimate reply packets based on the fact that the legitimacy of incoming reply packets is first determined by inspecting the LPL table rather than inspecting the SFL table.

3.4.3 The Impact of Restricting the ISN to a Specific Pattern in the SNF scheme

It is known that when an end system sends its SYN packet to establish a TCP connection, it chooses an (ISN) for that connection. Typically, the ISN should change over time, so that each connection has a different ISN. New instances of a connection may be established if the connection is opened and closed in quick succession, or if the connection breaks with loss of memory and is then reestablished. The TCP should be able to identify duplicate segments from previous instances of the same connection. To achieve this, RFC 793 [94] specifies that the ISN should be viewed as a 32-bit counter that is incremented by one every 4 microseconds. Although this practice is common in most TCP implementations, it is not necessary for proper functionality of the protocol, and it can be violated in favor of mitigating the effect of an ongoing attack. The main impact of restricting the choice of the ISN is the increase the possibility of *duplicate segment problem* (i.e., having duplicate segments from previous instances of the same connection) if multiple instances of the same connection are created frequently. However, this

drawback can be tolerated during TCP-based RDoS attack, because it is more important to mitigate the effect of the attack. It is also, important to point out that restricting the ISN to certain pattern does not increase the vulnerability of connection hijacking through TCP sequence number guessing. This argument is based on the fact that for a connection to be hijacked, the sequence number of the server, rather the client, needs to be guessed.

3.5 Chapter Conclusions

DoS attacks represent an imminent threat to the Internet infrastructure. Effective mitigation of these attacks is an extremely important and strategic countermeasure. The focus of this chapter has been on supplementing ISP's edge routers with special mechanisms that aid them in performing online DoS mitigation, wherein each edge router becomes independently capable of distinguishing attack packets from legitimate packets. This feature addresses many of the performance and practical problems of existing DoS mitigation schemes. Two novel concepts were advocated in order to envision this functionality, namely, the concept of victim-assistance and the concept of protocol-determinism.

We have shown how to utilize the concept of victim-assistance to counter a class of attacks known as TCP-based RDoS. In particular, we proposed two victim-assisted schemes to mitigate these attacks. The first scheme, called pairing-based filtering (PF), is based on the idea of pairing request packets and their corresponding reply packets in a distributed manner. Packet pairing is performed at the edge routers of the ISP network that contains the victim, such that attack reply packets can be identified and filtered directly, leading to protection from bandwidth exhaustion. Through analytical studies, we showed that PF scheme offers protection for legitimate packets going to the targeted system during a RDoS attack, while filtering attack packets. Our analysis shows that the probability that a legitimate packet being dropped under the PF scheme is less than 0.001, when suitable parameters are chosen.

The second scheme, called SYN number Based Filtering (SNF), is based on the idea of restricting victim's choice of the initial sequence numbers of the generated connection establishment requests during an attack, such that legitimate incoming SYN-ACK packets can be

verified at the ISP perimeter by checking specific bit combination. Our analysis shows that the percentage of false positives is extremely low in the case of classical attacks. For advanced attacks, the analysis shows that periodic SNF performs very well in terms of false positives and false negatives rates only when the period of changing the secret pattern is close to attacker's response time. However, reactive SNF performs very well for both cases.

Obtaining victim's assistance represents the basis for defending against TCP-based RDoS attacks in both schemes. The importance of this approach is reflected in the fact that a new source of information (i.e., the victim) is now available to make distinction between attack and legitimate packets. This approach opens new directions of research in designing efficient defenses for DoS attacks, which includes (1) integrating the proposed schemes in such a way to provide a hybrid service of attack prevention and mitigation and (2) investigating the viability of victim's assistance in defending against other types of DoS attacks. In this context, several issues needs to be addressed. For example, what information should be provided by the victim? How this information is to be communicated to edge routers? When to activate/terminate the mitigation scheme and on which edge routers? How to extend the concept of victim-assistance to inter-domain setting? Does collaboration among edge routers help in anyway?

We have shown how to utilize the concept of protocol-determinism in mitigating SYN flooding attacks. In this context, a novel scheme, called intentional dropping based filtering, has been proposed. The proposed scheme enables edge routers of an ISP network to distinguish between legitimate and fake connection requests taking into account the persistent behavior of legitimate clients. The main idea of this scheme is to intentionally drop the first SYN packet of each connection request. Subsequent SYN packet from a request is passed only if it adheres to the TCP's timeout mechanism. We showed that the proposed scheme reduces the effective attack rate significantly. However, it imposes additional delay on connection establishment. Several issues need to be addressed in the future. For example, what improvements should be made in order to take the parallelism of HTTP 1.0 connections into account? How to mitigate type 3 attacks? Is it possible to perform adaptive activation and termination of the filtering scheme by each edge router individually?

It is clear that the co-existence of the SNF and intentional dropping-based filtering schemes can effectively mitigate TCP-based DoS attacks which are the most common and the most challenging. The issue of co-existence of these two schemes with the PF scheme or with any other mitigation scheme opens up future research directions with respect to scalability, interference, and effectiveness.

CHAPTER 4. NOVEL HYBRID SCHEMES EMPLOYING PACKET MARKING AND LOGGING FOR IP TRACEBACK

Tracing DoS attacks that employ source address spoofing is an important and challenging problem. As explained in Chapter 2, traditional traceback schemes provide spoofed packets traceback capability either by augmenting the packets with partial path information (i.e., packet marking), or by storing packet digests or signatures at intermediate routers (i.e., packet logging). Such approaches require either a large number of attack packets to be collected by the victim to infer the paths (packet marking), or a significant amount of resources to be reserved at intermediate routers (packet logging). We adopt a hybrid traceback approach in which packet marking and packet logging are integrated in a novel manner, so as to achieve the best of both worlds, that is, to achieve small number of attack packets to conduct the traceback process and small amount of resources to be allocated at intermediate routers for packet logging purposes.

Based on this notion, two novel traceback schemes are presented. The first scheme, called *Distributed Link-List Traceback* (DLLT), is based on the idea of preserving the marking information at intermediate routers in such a way that it can be collected using a link list based approach. The second scheme, called *Probabilistic Pipelined Packet Marking* (PPPM), employs the concept of a "pipeline" for propagating marking information from one marking router to another so that it eventually reaches the destination. We evaluate the effectiveness of the proposed schemes against various performance metrics through a combination of analytical and simulation studies. Our studies show that the proposed schemes offer a drastic reduction in the number of packets required to conduct the traceback process and a reasonable saving in the storage requirement.

The rest of this chapter is organized as follows. In the next section, we discuss the attack traceback problem in order to motivate our work. In Section 4.2, we present the first scheme: distributed link-list traceback. In Section 4.3, we present the second scheme: probabilistic pipelined packet marking. Practical considerations are discussed in Section 4.4. In Section 4.5, we evaluate the proposed schemes through combination of theoretical and experimental analysis. Finally, summary is given in Section 5.

4.1 Background and Motivation

4.1.1 Attack Traceback Problem

Let R_{i1}, R_{i2}, ..., R_{in} be the ordered list of routers between attacker (A_i) and victim (V). This ordered list of routers defines the attack path for A_i. We call each of these routers involved in forwarding malformed packets to the victim, as an *Attack Router*. For any attack router R_{ij} in the list, all routers between R_{ij} and the victim are called the *Predecessor List* of R_{ij}, while all routers between attacker and R_{ij} are called *Successor List* of R_{ij}. The main objective of attack traceback problem is to identify the attack router connected directly to A_i (i.e., router R_{i1} which has empty successor list). In our view, this is equivalent to identifying the end point of a link list starting at the victim, where each element in the list represents an intermediate router along the path from victim to attacker as can be seen in Fig. 4.1. Multiple attackers case corresponds to a tree of link lists rooted at the victim (V), where each leaf represents a link list end point.

Figure 4.1 An instance of the attack traceback problem. The list of routers R_{i1}, R_{i2}, ..., R_{in} represents the attack path followed by packets sent from A_i to V.

The main assumptions made in our work are outlined as follows:

- The attackers overwhelm the victim by large number of packets. This is a valid assump-

tion which reflects the nature of most common DoS attacks. However, different than the assumptions made in [51, 53, 54], we do not necessarily assume that individual attack sources has to send numerous packets.

- Routing paths are stable (i.e., all packets follow the same path for a given (source, destination) pair). This is not necessarily true for all packets due to load balancing or network configuration changes. However, it can be accepted since these route changes are infrequent and can be ignored for the sake of simplifying the problem.
- Packets travel from source to destination in a relatively short time (i.e., few milliseconds).
- Attackers are aware of the traceback scheme in use. Therefore, they try every possible way to mislead the victim.
- Routers are not widely compromised. Our schemes rely on well behaved routers.

4.1.2 Motivation and Contributions

The imminent threats imposed by DoS attacks call for efficient and fast traceback schemes that enjoy the following features:

1. Providing accurate information about routers near the attack source rather than those near the victim.
2. Recognition and exclusion of false information injected by the attacker.
3. Avoiding the use of large amount of attack packets to construct the attack path or attack tree.
4. Low processing and storage overhead at intermediate routers.
5. Efficient collection of marking information stored at intermediate routers (if any).

Previous schemes failed to satisfy the above features collectively. For example, in PPM [51], routers that are far away from the victim have very low chance to pass their marking information to the victim because intermediate routers overwrite this information, which leads

to the loss of valuable marking information written by routers near the attacker(s). This is contradictory to our goal of having more knowledge about these routers. PPM requires considerable amount of packets to be collected at the victim before conducting the traceback process. Waiting for a large number of attack packets to be collected at the victim will significantly increase the response time of countering an attack. This argument may not be valid when a small value (e.g., 0.05) for the marking probability is used in PPM. However, such marking probability introduces a serious vulnerability in PPM that was pointed out in [52], where the attacker has the ability to pass spoofed marking information to mislead the victim.

The major drawback of Hash-based traceback [62] is that it incurs a heavy burden on routers by requiring them to log information about every forwarded packet. Moreover, the method employed to collect packet information from network routers is inefficient and requires special resources; the scheme assumes that a central management unit is available in each domain to download and search all packet digests looking for specific packet footprints. This results in a tedious process and an unnecessary overhead. The deterministic storage algorithm increases the memory requirement of the scheme. Moreover, a major concern in Hash-based traceback is the small window of time through which packets can be successfully traced.

Generally, routers write their IP addresses in the forwarded packets under the sampling based schemes. In contrast, packet information (digests or signatures) is written in router's memory under the logging based schemes. Each approach has its own pros and cons. We believe that developing a hybrid of both approaches, as predicted in [95], can lead to an improved traceback capability. Different than earlier work, we focus on key aspects of an efficient traceback scheme. In particular, we show how to provide the victim with more information about the path followed by an attack packet using constant space in the packet header in order to reduce the number of packets required to localize attack sources, and we address the issue of eliminating the attacker's ability to mislead the victim in the traceback process. Also, we focus on reducing the router's storage requirement.

We propose two alternative implementations of a hybrid marking and logging based IP traceback. The first implementation is called *Distributed Link-List Traceback* (DLLT). DLLT has an efficient implementation that combines the good features of both PPM and Hash-based schemes based on the idea of preserving the marking information at intermediate routers in such a way that it can be collected using a link-list based approach. The second implementation is called *Probabilistic Pipelined Packet Marking* (PPPM). This scheme aims at propagating the IP addresses of the routers that were involved in marking certain packet by loading them into packets going to the same destination. Therefore, preserving these addresses while avoiding the need for long term storage at intermediate routers. DLLT and PPPM exhibit the features of PMM [51] in the sense that routers mark forwarded packets probabilistically. Also, they exhibit the features of Hash-based scheme [62] in the sense that processing and storage at intermediate routers are necessary. The main advantage of PPPM scheme over the DLLT scheme is that long term storage at intermediate routers is not necessary.

We discuss the details of both schemes showing the procedures performed by Internet routers, the fields need to be allocated in each packet for marking purposes, the methods used to collect marking information from intermediate routers, and the attack source identification algorithm used by the victim. Practical considerations are discussed as well. We perform preliminary theoretical study to evaluate the proposed schemes and compare them with the PPM scheme in terms of the number of packets required to conduct the traceback process. Also, we quantify the storage requirement of the proposed schemes. New performance metrics are defined and used to evaluate the hybrid schemes through extensive simulation experiments.

4.2 Scheme I: Distributed Link List Traceback (DLLT)

4.2.1 Distributed Link-List (DLL) Concept

The main idea of DLL is to keep track of a subset of the routers that are involved in forwarding certain packet by establishing a temporary link between them in a distributed manner. DLL is based on a "store, mark and forward" approach. A fixed-size marking field is allocated in each packet. Any router that decides to mark the packet, stores the current

content of the marking field (which was written by the previous marking router) in a special data structure called *Marking Table* maintained at the router. The router generates an ID for that packet to index its marking information in the marking table. The router marks the packet by overwriting the marking field by its own IP address, and then forwards the packet as usual. Any router that decides not to mark the packet just forwards it.

A link list is inherently established because the marking field serves as a pointer to the last router that did the marking for a given packet, and the marking table of that router contains a pointer (i.e., the IP address) to the previous marking router, and so on. Therefore, each packet received by the destination contains the start point of a link list that is part of the packet path. We call it distributed link-list because each router decides by its own to be on the list or not according to certain marking probability.

We illustrate the main idea of DLL by considering the scenario shown in Fig. 4.2. Let packet x be forwarded from Attacker to Victim through the list of routers (R_1, R_2, ..., R_8). Assume that routers R_2, R_5, and R_7 marked the specified packet according to certain marking probability, q. In the DLL approach, an entry will be added to the marking table of each of these routers as shown in the figure. The destination in this case receives a packet marked by R_7. The remaining marking information (i.e., router's IP addresses) of packet x can be obtained by following the link-list at R_7 and then propagating to other routers (i.e., R_5 and R_2). The main advantage of this scheme is to keep the size of the packet from growing while preserving marking information along the path to be collected if necessary. Obviously, the cost paid for this gain is the additional storage requirement at intermediate routers.

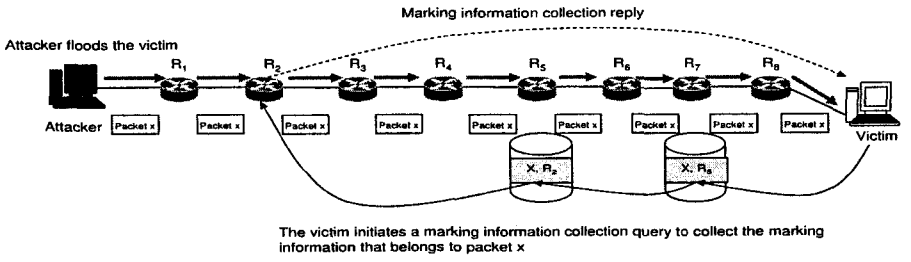


Figure 4.2 An example of distributed link-list marking. In this example, R_2, R_5, and R_7 decided to mark packet x.

4.2.2 Details of Distributed Link-List Attack Traceback

Distributed Link-List Traceback (DLLT) uses DLL concept to keep track of the routers that have been involved in forwarding malformed packets toward the victim. The basic components of this scheme are the marking and storage procedure, and the marking information collection protocol. What follows is a discussion of each of these components.

4.2.2.1 Marking and Storage Procedure

DLLT employs a probabilistic marking and storage scheme. When a router receives a packet, it will mark the packet with probability q. If the packet has previously been marked, then the router will store that information before remarking the packet. Therefore, packet marking and storage is an integrated procedure. Before going into details of this procedure, we show the main data structure used for storing packet information.

Logging packet information at intermediate routers is not a new idea. Storing the packet features was considered in [61]. Also storing packet digests was considered in [62]. The major drawback of these schemes, as pointed out in subsection 4.1.2, is that they put a heavy burden on routers by requiring them to log information about every forwarded packet. Moreover, there is a need to download and search all this information looking for specific packet footprints, which results in a tedious and an unnecessary process. Our storage scheme is probabilistic in nature, which means that only fraction of the traffic is to be logged at each router. Also, we store this information in such a way as to ensure that it can be collected in a predetermined manner. For storage implementation, We borrow the idea of using Bloom filters [69] from [62]. However, we modify it to satisfy our requirements:

- Storing the packet digests to be able to verify that a given packet has been forwarded by the router.
- Mapping the digests of a given packet to a certain memory location where the marking information of that packet can be stored.

The first requirement can be achieved exactly the same way as in [62], where a Bloom

filter computes j distinct packet digests for each marked packet using j independent uniform hash functions, and uses the j (n-bit) results to index the 2^n-sized bit Digests Array (DA) (please refer to Appendix B for details). The array is initialized to all zeros, and bits are set to one as packets are received. The second requirement can be achieved by storing the marking information of a given packet in the Marking Information Table (MIT) at the memory location indexed by the first hash function that maps to zero bit in the digests array. Fig. 4.3 depicts both the DA and MIT with j hash functions. It also shows the marking information of a given packet before and after being marked. In each packet, we reserve the following fields to hold the marking information:

- Single 32-bits marking field. This field holds the IP address of the marking router.
- $\lceil \log_2(j) \rceil$ bits hash function number. This is the number of the hash function used to index the location of the marking information of the given packet in the MIT.

In the MIT, we store the following information:

- The IP address of the previous router that marked a given packet. This serves as a pointer to that router while collecting marking information.
- The hash function number (hfn) found in the marked packet. This specifies which hash function was used to index the MIT of the router pointed by the IP address.

It can be realized that *probabilistic edge marking* (an edge is composed of two adjacent routers on the packet path) is simple to implement in our scheme. Whenever a router decides to mark a packet, the subsequent router can be forced to mark the same packet. This can be achieved by maintaining a 1-bit field called *marking flag* as part of the marking information to be held in the packet. This flag is used to enforce deterministic marking when it is *on*. When it is *off*, the marking becomes probabilistic. With this flag, the probabilistic edge marking in DLLT can be implemented as follows: When a router receives a packet, it checks the marking flag. If it is on, it has to do the marking and storage procedure and then reset the flag. Otherwise (i.e., when the flag is off), it takes the decision based on some probability, q. If

the decision outcome is to mark the packet it will do so, and then set the flag such that the next adjacent router will do the marking deterministically. The detailed marking and storage procedure is shown in Fig. 4.4.

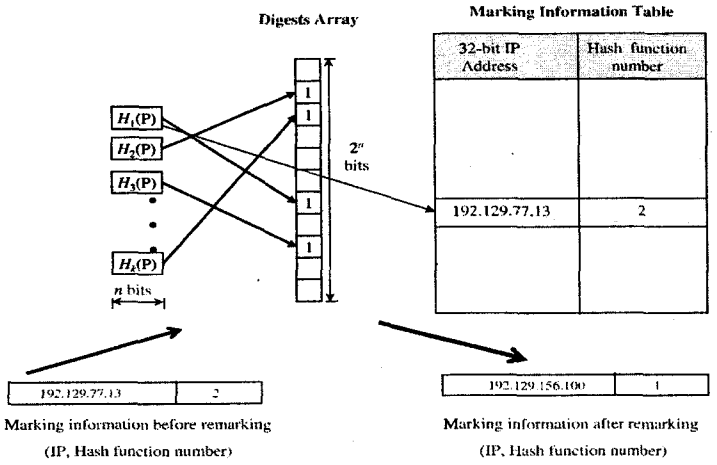


Figure 4.3 Digests array (DA) and marking information table (MIT) at router R. The marking information of a given packet before and after being marked at router R which has the IP address of 192.129.156.100.

We have observed that there will be a lot of unused memory locations in the MIT because the actual number of packets whose digests are stored in one DA is certainly less than 2^n, and it depends on the capacity factor of the DA, which defines how many bits are required in the DA in order to store the digests of certain number of packets. For efficient storage utilization, the MIT is shared among several DAs. This issue is revisited in subsection 4.5.2.

4.2.2.2 Marking Information Collection Protocol

Once an attack is detected, then the first thing to do is to collect marking information from intermediate routers. A straightforward approach is to use a centralized scheme as in [62]. Such approach requires a central management unit that is responsible for polling each router in its domain to download a copy of its currently stored information. Although easy to implement, this scheme lacks scalability and efficiency. Moreover, it is an expensive approach since special

Marking and Storage Algorithm at Router R

1. *for* each packet, P
   (a) *if* (P.marking flag == 1) Mark and Store (P)
   (b) else
      i. choose a random number x in [0, 1]
      ii. *if* ($x < q$) Mark and Store (P)
      iii. *else* forward P

Procedure Mark and Store (Packet P)

1. compute $H_1(P)$, $H_2(P)$, ..., $H_j(P)$
2. *for* ($i = 1$ to j)
   (a) *if* (Digests Array $[H_i(P)]$ == 0) {index = i; break;}
3. set Digests Array bits indexed by $H_1(P)$, $H_2(P)$, ..., $H_j(P)$
4. store the previous marking information of P in the MIT at location indexed by $H_{index}(P)$
5. remark P. (i.e., P.Marking field = R, flip P.Marking flag, P.hash function number = index)

Figure 4.4 DLLT marking and storage algorithm at router R.

resources (for example central management units) have to be deployed in each Internet domain. Instead of downloading the whole marking information from each router, we propose a *Marking Information Collection Protocol* (MIC) to collect relevant marking information from routers that are believed to be involved in forwarding attack packets.

The main objective of the MIC protocol is to collect the addresses of the routers that actually marked a particular attack packet. In this protocol, we define three types of packets: (i) MIC-request packets, (ii) MIC-reply packets and (iii) MIC-deadend packets. First, we describe the format of each of these packet types. Then, we discuss the operation of this protocol. All MIC packets have the following general format: <Packet Type, Source Address, Destination Address, Victim Address, Hash Function Input, Hash Function Number, Marking Routers List>. Where packet type field specifies whether the packet represents request, reply,

or deadend.

- *MIC-request packet:* It is used to obtain the ordered list of routers that marked a particular attack packet. Each MIC request packet propagates to the set of marking routers of a particular attack packet. Therefore, source address, destination address and hash function number fields keep changing from one visited marking router to another. Each of the visited marking routers appends its address to the marking routers list.

- *MIC-reply packet:* It is generated by a node that identifies itself as the end list point of the given attack packet (i.e., when the MIC-request cannot be propagated further). This packet is sent back to the victim. It is possible for the attacker to spoof the marking field of its outgoing packet such that the first marking router will store this information. This has the impact of propagating the MIC-requests to a false router that does not have the digests of the given packet. Such router is called a dead-end router. This router is responsible of sending the MIC-deadend packet described below to the requesting party.

- *MIC-deadend packet:* It is generated by a router called the deadend router in response to an invalid MIC request (i.e., when no marking information is available for the given attack packet). It is sent back to the requesting party to indicate that party must identify itself as the end router of the link list obtained by the given attack packet. Such router is called the *end-list* router. The contents of the received MIC request packets are copied into the outgoing MIC-deadend packet except for the packet type, source and destination address fields need to be changed accordingly.

The MIC protocol is initiated by the victim once an attack is detected. Each MIC-request packet generated by the victim corresponds to one particular received attack packet. The number of generated requests is an input parameter specified by the administrator. Each MIC-request packet is sent to the router whose address is found in the marking field of the corresponding attack packet. This request propagates to other routers that marked that particular packet. This is accomplished by processing the MIC-request as follows: the router who receives a MIC-request, performs a membership test (by computing j hash functions with the

given packet as an input) to check whether the packet belongs to the router's DA or not. If the membership test is positive, then the router extracts the marking information of that attack packet from its MIT at the location specified by $H_{hfn}(P)$, where 'hfn' is the hash function number found in the MIC-request packet. Also, it appends its own IP address to the Marking Routers List and updates the 'hfn' field. Then it propagates the request packet to the next marking router.

It is important to distinguish the deadend router from the end-list router in DLLT. The deadend router of a given packet is the router that does not maintain the digests of that packet. While the end-list router is by default the router which could store spoofed source address, because the attacker is expected to spoof the marking field. (recall that the end-list router is the first router on the actual attack path that decides to mark the packet). When the end-list router receives a MIC-request, it has to propagate it (after appending its IP address) to the router specified by the stored IP address. Since the address is spoofed, there are two cases possible:

1. *Valid IP address:* It will lead to a router that is not on the actual packet path. This router which we call the deadend router realizes that it is not part of the packet path (based on not having the packet's digests). Therefore, it sends a reply message to the end-list router indicating no farther propagation for the request is possible.

2. *Non-valid IP address:* If the MIC-request is forwarded to a router that is not implementing the scheme, or to a non-existing address, then the end-list router will either receive an error message or it will time out before receiving any reply.

In both scenarios, it can be assumed that no farther propagation of the request message is possible. Fig. 4.5 depicts the operation of this protocol. Shaded nodes represent routers involved in forwarding attack packets to the victim V coming from nodes A_1 and A_2. Dashed arrows show how the MIC-request for a particular attack packet propagates from the victim to X and Y. As no further propagation is possible, Y, which is called end-list router in this case, sends an MIC-reply packet back to V.

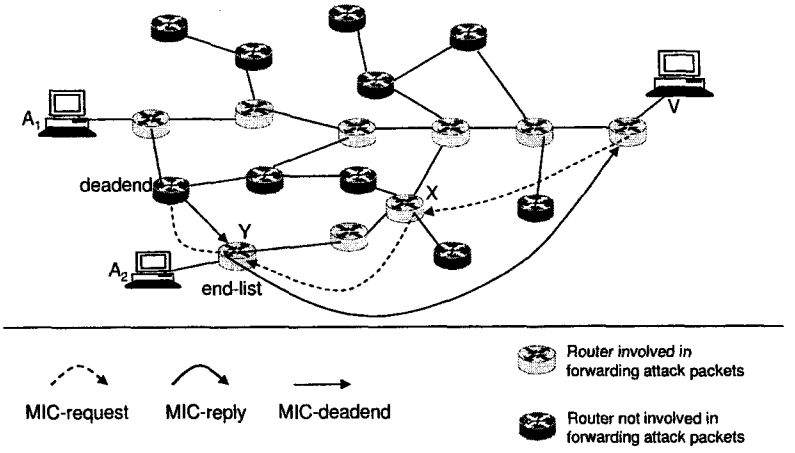


Figure 4.5 Basic messages used in MIC protocol. Notice that the deadend router does not have the packet's digests. Therefore, it sends the MIC-deadend message to router Y (the end-list corresponding to the given MIC-request).

4.3 Scheme II: Probabilistic Pipelined Packet Marking (PPPM)

A disadvantage of DLLT is that long term storage at intermediate routers is necessary. In this section, we propose an alternative scheme, called probabilistic pipelined packet marking (PPPM), that takes such issue into account.

4.3.1 Pipelined Packet Marking Concept

In computer architecture, pipelining is used to allow more than one instruction to be in some stage of execution at the same time. Usually, each instruction passes through sequence of stages during its life cycle. The outcome of one stage is forwarded to the next stage before being replaced by another instruction from the previous stage. The whole process is repeated at each stage until the whole instruction is executed. We propose a similar approach for packet marking. In computer architecture analogy, a router that marks a packet represents a pipeline stage, the marking process represents the instruction execution, and the propagation of marking information from one marking router to another represents the flow of instructions in a pipelined system. *The main idea is to transfer the marking information that belongs to certain packet by propagating it from one marking router to another using subsequent packets*

traveling to the same destination.

We propose a packet marking scheme, called probabilistic pipelined packet marking (PPPM), based on the pipelined marking concept. The objective of PPPM is to let the destination know about all routers that were involved in marking certain packet, P, using *constant* space in the IP packet header without incurring long term storage overhead at intermediate routers. In the following discussion we assume that P has enough space to hold the marking information of one router[1]. In its basic form, the marking information field allocated in each packet consists of two parts: the IP address of the marking router, MR, and an ID used to link marking done for a given packet by different routers[2].

Suppose that m routers (R_1 through R_m) decide to mark packet P (based on certain marking probability, q). The total marking information to be written by the m routers can be viewed as a long message that needs to be transferred to P's destination. The main idea is to distribute this message among m packets. For the message to be reassembled at P's destination, a unique ID must link the m packets. Distributing the message among several packets is accomplished in our scheme through pipelining. The first router[3], R_1, that decides to mark P, copies its IP and a randomly chosen ID, x, into the corresponding marking fields of the packet. The second marking router of P (i..e., router R_2) buffers P's marking information ($MR = R_1, ID = x$) in a destination based table, before remarking it by its own. It is important to mention that P's ID remains the same. The same process is repeated at each marking router. The content of P's marking information field when arriving its destination would be (R_m, x). The remaining marking information for packet P, which is still buffered at routers R_2 through R_m are transferred to P's destination by loading them into packets going to the same destination.

Fig. 4.6 shows an example of pipelined packet marking. In this example, packets are forwarded from S to D through the path consisting of routers R_1, R_2, R_3, and R_4. To simplify the discussion, we assume that each forwarded packet is marked by at most two routers chosen

[1]We discuss where to place this information in 4.4.3.1.

[2]Actual contents of the marking field are discussed in subsection 4.3.2.

[3]A router can identify itself as the first marking router of a given packet if the marking field found to be empty.

arbitrarily. The names of the routers that decide to mark certain packet are shown in the table associated with the source S. The figure shows the content of the marking field allocated in each packet as it traverse its path from S to D. It also shows the content of the buffer at each router. In this example, routers R_1 and R_3 decide to mark the first packet. Since R_1 is the first marking router of packet 1, it puts its own mark into the packet. This consists of its IP address, R_1, and a randomly chosen ID, x. Notice that nothing is buffered at R_1. The same packet is remarked by R_3. Since the packet is already marked, R_3 has to buffer the marking information for the packet's destination before writing its own IP address in the packet. Notice that R_3 keeps the same ID, x, in the marked packet. The buffered information is transferred to D when R_3 decides to mark another packet destined to D. This happens to be the forth packet. Since both packets (i.e., the first and the fourth) hold the same ID, x by the time they arrive at D, it is easy for D to conclude that packet x has been marked by R_1 and R_3. Similar discussion applies for other packets in this example.

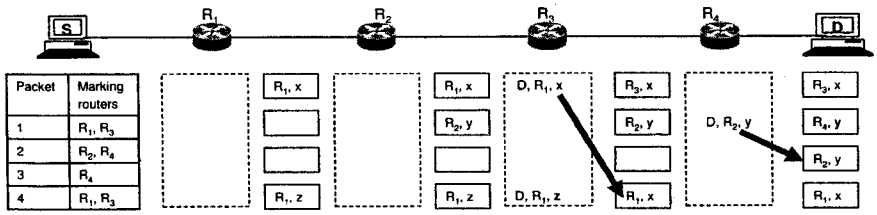


Figure 4.6 An example of pipelined-based packet marking. The table shows the names of the routers that did the marking for each packet. Dashed rectangular boxes represent buffers associated with each router. Marking information augmented with each packet is shown, in a solid rectangular box, for each packet at each router along its path.

4.3.2 Details of Probabilistic Pipelined Packet Marking (PPPM)

In this subsection, we provide details about PPPM. We start by describing the fields that need to be allocated in each packet for marking purposes, then the information that needs to be buffered at each marking router and lastly we describe the pipelined marking and buffering

algorithm. Similar to DLLT, packet marking and logging in PPPM is an integrated procedure. The following fields are allocated in each packet for marking purposes:

- *32-bit IP address:* This field represents the IP address of the marking router.
- *8-bit $T\bar{T}L$:* This field is used to obtain the actual distance between the marking router and the victim. $T\bar{T}L$ is normally set to the value of TTL found in the packet before remarking it except when the packet is used to transfer a buffered marking information that belongs to another packet, say P_x. In which case, the $T\bar{T}L$ is set to $(TTL + T)$, where T represents the distance between the previous marking router of P_x and the current router.
- *c-bit ID:* This is an ID that uniquely identifies the marked packet. It is chosen randomly by the first marking router (i.e., the first router to mark a certain packet).

Each PPPM enabled router maintains a destination based buffer for the purpose of marking information logging. For each destination, the most recent marking information is buffered. This includes the same marking fields found in the marked packet except that instead of storing the $T\bar{T}L$ value, the difference T=$T\bar{T}L$-TTL is stored. This difference represents the distance between the previous marking router and the current marking router. The advantage of this field is to determine the distance of a certain marking router from the destination. For a high speed implementation, there should be a mechanism by which a router can directly identify whether a given destination has marking information in the buffer, or not. Similar to DLLT, we suggest using a Bloom filter at each router to achieve this objective. When a router decides to buffer a packet's marking information, it applies j hash functions to the packet's destination and set the corresponding bits of the Bloom filter to one.

Fig. 4.7 shows the marking and buffering algorithm that is to be performed at a PPPM enabled router. A router, R, decides to mark a packet, P, based on certain marking probability, q. If there is no marking information to be transferred to P's destination and if P is not already marked, then R writes its own mark into P. This includes its own IP address, a randomly assigned ID, and a $T\bar{T}L$ value equals to that of P's current TTL. Otherwise (i.e., if P is already

marked), R buffers P's marking information before remarking it. On the other hand, if there is marking information to be transferred to P's destination, then that information is loaded into P before P's current marking information is buffered at R. It is to be noted that, in the latter case, the router keeps swapping the marks but cannot perform its own remarking on the packets for the same destination. However, we do not consider this as a major problem, because in PPPM scheme, it is sufficient to obtain only one mark per router to conduct the traceback process. This is based on the fact that each router is associated with a distance parameter that specifies its location on the packet path. To ensure that a router writes its own mark into at least one packet destined to a given destination during attack, the router clears the Bloom filter bits from time to time and writes its own mark probabilistically into packets destined to destinations that are not already inserted in the Bloom filter. Recall that, a Bloom filter is used in PPPM to indicate whether certain destination has a buffered marking information or not. When a router buffers marking information for a given destination, it inserts that destination in the Bloom filter by setting the bits indexed by the hash functions outputs. It is to be noted that clearing the Bloom filter is also necessary to get rid of any marking information that was buffered before the attack. Otherwise, such information will be written into packets destined to the victim.

Fig. 4.8 shows the benefit of using the $T\bar{T}L$ field in determining the distance of certain marking router from the destination. Let packet P_1 be sent from S_1 along the shown path to D. Packet P_2 is sent from different source, S_2, to the same destination D. We assume that P_1 was sent before P_2, and we assume that P_1 was marked by routers R and W respectively, and that its ID was set to x by router R; while P_2 was marked by routers F and W and its ID was set to y by router F. The figure shows the values of (packet ID, marking router, TTL, $T\bar{T}L$) for both packets after they leave each router along their paths. Also, it shows the content of the buffer at router W. In this example, P_2 is used to transfer the marking information that belongs to packet P_1 which was buffered at router W. After receiving the two packets, the destination can identify that packet x was marked by routers R and W which are 3 and 1 hops away, respectively. Determining the distance of each marking router from the destination can

Marking and Buffering Algorithm at Router R

1. *for* each packet, P
   (a) choose a random number x in [0, 1]
   (b) *if* ($x > q$) forward P
   (c) *else*
      i. *if* (P.destination is NOT in R.MB)
         A. *if* (P.ID == 0) mark P
         B. *else* buffer P.MI, remark P
      ii. *else*
         A. save P.MI in a temporary variable
         B. remark P
         C. buffer $P.MI$

Figure 4.7 Marking and buffering algorithm at router R. MB denotes Marking Buffer. MI denotes Marking Information.

be easily incorporated in the marking procedure shown in Fig. 4.7.

Similar to DLLT, probabilistic edge marking can be realized in the proposed marking and buffering procedure by adding a 1-bit marking flag. Once a DoS attack is detected, the victim starts the source identification process using k attack packets as an input. By realizing that packets holding the same ID carry marking information that belongs to the same packet, the victim can extract the marking information of each of the distinct IDs found in the k packets by running the simple algorithm shown in Fig 4.9. The outcome of this algorithm is an ID table with two columns: *Packet ID* and *Marking Routers List* and their *distances* from the victim.

4.4 Practical Considerations

4.4.1 Choice of the Marking Probability

The value of the marking probability, q, plays a direct role on the effectiveness of the proposed schemes. On one extreme, if we set q to 1 (i.e., deterministic marking) then only 1 attack packet would be sufficient to locate the attacker in the case of DLLT, while d attack

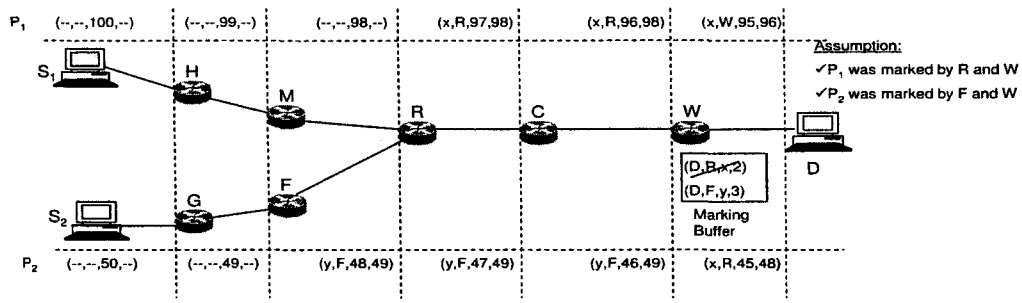


Figure 4.8 An example of distance calculation. The values of (packet ID, marking router, TTL, $T\bar{T}L$) are shown at each route for both packets. By receiving P_1 and P_2, D concludes that packet x was marked by W which is $98 - 96$ hops away, and by R which is $48 - 45$ hops away.

ID Table Construction Algorithm

1. *for* each packet, P
   (a) *if* (P.ID NOT in the ID Table (IDT)) add entry for P.ID in IDT
   (b) append P.MR and ($P.T\bar{T}L$ - P.TTL) to the ID.marking routers list

Figure 4.9 ID table construction algorithm at the victim. P.ID denotes P's ID, P.MR denotes P's marking router, and ID.marking routers list denotes the list of routers that correspond to the IDT's entry that contains ID.

packets are required in the case of PPPM (where d is the number of of PPPM enabled routers in the pipeline between attacker and victim). This choice is excluded in practice to avoid excessive overhead on network routers. On the other extreme, if we set q very close to 0, then larger amount of packets would be required to reconstruct the attack path. In general, the following issues must be considered when choosing q.

1. *Ensuring that at least one router along the attack path will mark a given packet.* This has the benefit of overwriting any false marking information injected by the attacker if he acts as a marking router.

2. *Minimizing the number of packets required for attack traceback.* This bound is expressed

as a function of q (see Equation (4.2)), where it can be seen that the bound is minimized by increasing q.

3. *Minimizing the overhead on network routers.* The amount of overhead imposed by DLLT/PPPM on network routers can be lowered by decreasing q.

While issues 1 and 2 are consistent, they conflict with issue 3. Therefore, there is a tradeoff between the efficiency of the proposed schemes and the amount of overhead imposed on network routers. In [52], it was shown that the ability to hide attacker's location (by injecting false marking information into its attack packets) is curtailed by increasing the marking probability, however in PPM [51] the degree to which the victim can delimit the attacker's injection of false marking information is bounded by the sampling constraints. In DLLT and PPPM, there is no maximum bound on the value of q, because information about all marking routers of a given packet will eventually propagate to the victim. In our simulation studies (section 4.1), we use relatively large values for q to illustrate the efficiency of the proposed schemes. In practice, lower values for q would be used to minimize overhead on network routers. The choice of q is a design issue that involves a tradeoff between efficiency and router's processing overhead.

4.4.2 Security of the Proposed Schemes

4.4.2.1 DLLT security

The correctness of the DLLT scheme is based on the validity of the information collected by the MIC protocol. It is possible that attackers may abuse this protocol in different ways.

- *The protocol can be abused by generating fake MIC-requests and/or fake MIC-replies.* This is inevitable as it is inherent problem with any protocol and it is not specific to the proposed MIC protocol. Fake MIC-requests may increase processing overhead at network routers. However, they do not affect the outcome of the traceback process. Fake MIC-replies, on the other hand, can affect the traceback outcome, because such replies inject false information about the attack path/tree. To overcome this problem, the MIC protocol can be modified in such a way to provide light wight authentication of MIC-

replies. A straightforward modification is to let the victim attach a secret code to each generated MIC-request. This code propagates with each MIC-request as it becomes part of the request. The same secret code is copied into the the corresponding MIC-reply. Fake MIC-replies can be easily filtered as they are not supposed to contain a valid code. For increased security, the secret code should be distinct for each MIC-request.

- *The deadend router which may be chosen in advance by the attacker can be compromised.* For example: (i) it can create *looping* by propagating an MIC-request packet to one of the routers found in the Marking Routers List of that packet (ii) it may ignore any MIC-request packet (iii) it may propagate received MIC-request packets to wrong routers. Issue (i) can be avoided by preventing a router from propagating a MIC-request packet if its address is already in the Marking Routers List. Issues (ii) and (iii) can be detected by modifying the MIC protocol such that each router acknowledges the requesting router. If no acknowledgment is received within a specific amount of time. The requesting router detects that some something is wrong, identifies itself to be the end-list router, and generates the MIC-reply packet.

4.4.2.2 PPPM Security

The PPPM scheme is vulnerable to the following types of attacks.

- *The attacker is expected to behave like a marking router.* Therefore, marking its outgoing packets with false information before being marked by any router along their path. For example, it can spoof the IP-marking field. Also, it can write any value in the $T\bar{T}L$ field. If the attacker uses a $T\bar{T}L$ value that is less than the original packet's TTL, then it is very easy to distinguish and drop attack packets before reaching the victim, because $T\bar{T}L$ must be larger than the TTL value of a given packet. If the $T\bar{T}L$ written by the attacker is larger than the packet's original TTL, then intermediate routers cannot distinguish attack packets. This false information is more likely to be overwritten by subsequent routers if a reasonable marking probability is used. Unfortunately, there is nothing that can prevent the false information from propagating to the victim. For a given packet ID,

the victim will include the spoofed marking router in the marking routers list for that ID. However, because of the restriction that we have imposed on $T\bar{T}L$ (i.e., $T\bar{T}L$ must be larger than TTL), *the distance obtained at the victim for the spoofed marking router will always be larger than that obtained for any other marking router for the same packet. Therefore, the victim can easily distinguish and exclude this information.* We emphasize here that it is impossible to determine whether the first mark is bogus or genuine. What we meant here is that the first mark can be identified (Recall that the first mark has the largest distance) and it can be ignored as a precaution regardless of being bogus or genuine. Given that the marking probability has to be large enough to ensure that at least one router along the packet's path do the marking (as discussed in subsection 4.4.1), this precaution is to be taken into account only if the victim obtains information about two or more marking routers of a given packet (because if it obtains one mark only, then that mark has to be genuine). In this case, if the first mark happens to be from the attacker, then ignoring it does not affect the outcome of the traceback process. However, if it happens to be from a genuine router, then ignoring it has the impact of not localizing the attacker exactly.

- *As another vulnerability of the PPPM scheme, the attacker may use the same packet ID for all its outgoing packets.* Worse than that, multiple attackers can coordinate with each other to use the same packet ID either continuously or from time to time. In PPPM, the first marking router cannot recognize itself as the first router to do the marking for a given packet. Therefore, the ID injected by the attacker is used. We suggest the following slight modification to the marking and buffering algorithm shown in Fig. 4.7, to prohibit attackers from using the same ID for all packets: if a router receives a packet, P, that has identical (P.dest, P.ID) to those buffered for previously marked packet, it will drop the incoming packet. The suggested modification is based on the fact that it is very unlikely for two distinct packets traveling to the same destination within small window of time to have identical IDs.

4.4.3 Implementation and Deployment Issues

4.4.3.1 Marking Information - How many bits and Where to Place?

A major challenge to packet marking schemes in general is the limited space available in the IP packet header for marking purposes. Based on the fact that packet fragmentation is very infrequent in today's Internet [50], the focus in previous packet marking schemes (e.g., [51, 53, 54, 55]) has been always on encoding the marking information in the 16-bits ID fragmentation field and few other unused bits in the IP packet header. The proposed traceback schemes face the same challenge. However, due to design constraints, encoding schemes used in previous work cannot be extended to DLLT and PPPM. Fortunately, the marking information in DLLT scheme which has a size of 34 bits can be accommodated without encoding by utilizing the 25 bits recommended by Dean et. al., [54] which include 16 bits ID field, 1 bit fragmentation flag, and 8 bits type of service. In addition to these bits, we suggest utilizing the 13 bits fragmentation offset, because these bits will anyway go waste if the ID field is overwritten. This brings the total space available for packet marking to 38 which satisfies DLLT's requirements. However, this does not satisfy the requirement of the PPPM scheme (recall that the marking information size in PPPM is 57 bits). To overcome this problem, we propose an implementation based modification where the PPPM scheme is restricted to use up to 38 bits for marking purposes based on the idea of using Internet topology information for efficient encoding of the marking information.

Our analysis of the raw measurements of the topology information of nine major ISP networks, which was obtained from the Rocketfuel project [81], reveals that the majority of nodes in these networks have relatively low degrees. The value of this parameter varies based on the location of the node. Core nodes usually have high degrees, while edge nodes have low degrees. It has been shown earlier [96] that Internet maps follow power law, meaning that Internet connectivity is concentrated at few nodes, which can be interpreted by saying that few nodes have high degrees, and all other nodes have low degrees. Fig. 4.10 shows the node degree distribution for nine major ISP networks. As expected, node degree distribution of a given ISP network depends directly on the connectivity of that network. This explains the

variation in distribution among the considered ISPs. It can be seen that highly connected ISP networks, such as Level3 and VSNL, have higher degree distributions as compared to other ISPs. For example, it can be seen that more than 20% of Level3 nodes have degrees that range from 20 to 90, while the same percentage of nodes in VSNL have degrees between 11 and 28.

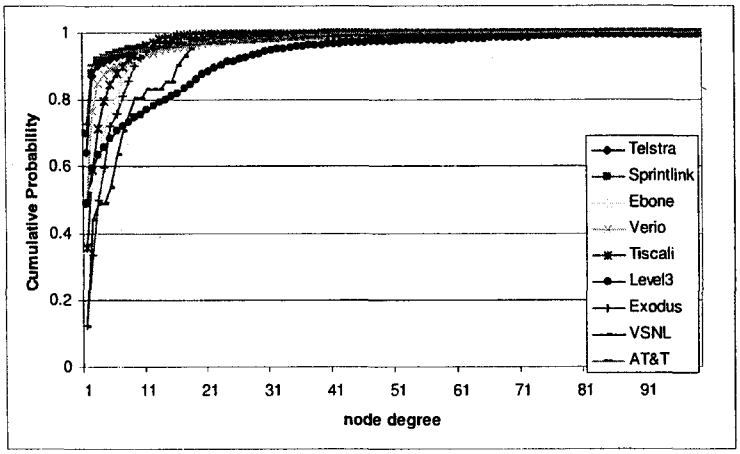


Figure 4.10 Node degree distribution based on Rocketfuel data [81].

Based on these statistical degree distributions, we propose the following modification to the original PPPM scheme. Instead of marking a packet by the IP address of the marking router, the packet is marked by a codeword of size x that represents the edge through which the packet has been received. An edge is a link between two adjacent nodes. It is clear that a value of 5 for x would be sufficient to represent an edge at any router along the path because most nodes have degrees less than 2^5. However, to accommodate some exceptional nodes (those with high degrees) we suggest using 8 bits (out of the available 38 bits) to represent x. This choice is based on the node degree distribution shown in Fig. 4.10. The remaining bits (i.e., 30) are used to accommodate the packet ID and auxiliary TTL fields as in the original scheme. The impact of this modification is reflected in the fact that the victim receives edge codes rather than routers' IP addresses. Decoding of the collected marking information can be done by propagating a traceback query starting at router to which the victim is connected.

From there, the query is forwarded to the router at the end of the link represented by the first codeword in the query. This process is repeated until all routers along a given attack path are identified.

Inspired by the idea of using network topology for encoding marking information, we tried to answer the following question: How many bits are required to encode an entire Internet path[4]? To answer this question, we investigated real Internet maps to obtain specific statistical topology information. This study is based on five data sets of raw Internet measurements obtained from the Internet Mapping project [84] at different dates. Internet mapping project is one of the earliest research efforts to construct a full map of the Internet. This project tries to identify all reachable Internet hosts from a central measurement point and record the list of router IP addresses found along the path to each host. Starting with a raw data representing reachable Internet hosts and their corresponding paths, we constructed an Internet tree rooted at the measurement point. The tree is then searched to extract the distribution of sum of ceiling of $\log_2$(node degree) along each Internet path.

Ceiling of $\log_2$(node degree) represents the number of bits required to uniquely encode one of its incoming/outgoing links. Therefore, the sum of ceiling of $\log_2$(node degree) along certain path represents the number of bits required for entire path encoding. This parameter captures both node degree and path length in a manner that cannot be obtained from their individual distributions. This forms the basis for our claims about space requirement for marking purposes. Fig. 4.11 shows the distribution of this parameter based on different data sets from the Internet mapping project [84]. It is to be noted that the distributions at different dates are not identical. This is due to two reasons. First, applying the ceiling function affects the distributions. Second, even if multiple paths have the same length, their corresponding node degrees will be different in most cases. The five graphs merge when the value on the x-axis equals to 46. At this point the cumulative distribution function is almost 1. Therefore, we can safely assume that almost all Internet paths can be encoded using at most 46 bits. This does not deny the fact that some exceptional, but very few paths, require larger number

[4]This is a general question and it is not related to the proposed DLLT and PPPM schemes.

of bits.

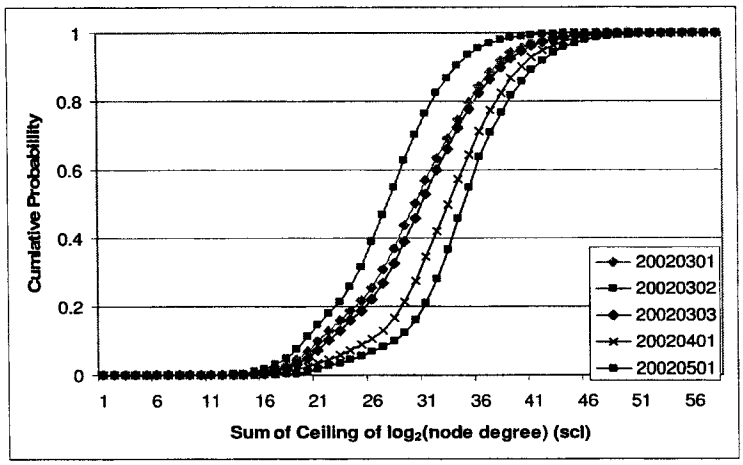


Figure 4.11 Distribution of sum of ceiling of $\log_2$(node degree) along Internet paths for five data sets obtained from the Internet Mapping Project [84].

4.4.3.2 Activation of the Proposed IP Traceback Schemes

It is obvious that the proposed schemes demand packet-level computation at the routers. Therefore, activating packet marking and logging all the time might be unnecessary and it has scalability implications. It is possible to adopt existing approaches to activate the traceback schemes when necessary. We briefly describe two approaches to achieve this objective.

1. *Router-driven activation:* Packet marking and logging could be activated at each router independently by following an approach similar to that used in the Path Information Cashing and Aggregation (PICA) scheme [60]. Recall that PICA suggested to leverage routers forwarding table to keep track of per-subnet traffic aggregate rate, which makes it possible to base path message generation (i.e., traceback activation) decisions on traffic rate. DLLT/PPPM enabled routers can take advantage of this approach to trigger packet marking and logging for packets destined to certain destinations.

2. *Destination-driven activation:* The only thing that is common among attack packets

is their destination address and, in most cases, their type (i.e., TCP, UDP, ICMP). Therefore, it is possible to activate the proposed schemes on packets destined to certain destination when that destination indicates its interest of receiving marking information about particular type of packets. To achieve this, we adopt an approach similar to that used in "intention driven ICMP traceback (iTrace)" scheme [59] to perform packet marking and logging on selective flows. It is to be noted that such approach has the same practical limitations of the iTrace proposal.

As discussed, such activation approaches can be integrated into our traceback schemes as the former do not interfere with the latter. Such modification would make the proposed hybrid schemes more scalable. Scalability analysis based on this modification is beyond the scope of this dissertation.

4.4.3.3 Deployment of the Proposed IP Traceback Schemes

The proposed schemes do not require cooperation among multi-domain network operators to perform the traceback across domains that implement the schemes. In other words, the whole process can be automated. In the PPPM scheme, the marking information buffered for a certain destination at different routers is transferred within packets destined to the same destination. In this regard, PPPM scheme is similar to the PPM scheme, because marking information in both schemes is transferred through the packets. On the other hand, DLLT scheme requires an explicit protocol to collect the marking information. This makes it similar to the Hash-based traceback scheme which deploys a central management unit in each domain for digests collection and partial path reconstruction. A full path is then constructed by combining the partial paths generated at each domain.

In a multi-domain scenario, the link list approach enables the victim to retrieve its marking information from DLLT enabled routers along the packet path regardless of being in the same or different domain. Another issue in this context is the partial deployment of the proposed schemes. In such a scenario, attack paths that span the domains which implement the traceback schemes will be constructed. This leads to partial path construction. Therefore, the proposed

schemes do not require every router on the Internet to implement the marking and buffering procedure. In fact, it is sufficient to implement the proposed schemes at ISP's edge routers which are assumed to have enough computational resources to perform packet marking and logging. Although, this would have the limited benefit of identifying the autonomous system that contains the attack sources rather than identifying the sources themselves, it remains as the first practical choice.

4.4.3.4 Applicability to Multiple Attackers Traceback

Both schemes are applicable to multiple attackers traceback. In the case of multiple attackers, attack paths converge to form a tree rooted at the victim. Therefore, path segments close to the victim are expected to be shared among several paths. The marking information associated with a given attack packet, P, consists of the end-list router and its predecessor list. If P's end-list router appears in the predecessor list of any other router, then all P's marking information must be added to the predecessor list of that router. Otherwise, P's end-list router must be on a distinct path and its predecessor list may or may not be shared among other paths (each router in the list must be checked to determine whether it appears in other routers predecessor lists or not). By taking this observation into account, it is easy to group routers that are shared among different paths and to identify routers that represent the leaves of the attack tree. The identification and reconstruction algorithm (Fig. 4.13) is based on this observation. Enforcing edge based marking and augmenting packets with distance parameters will accelerate the attack tree construction process.

4.4.3.5 Timely Collection of the Marking Information in DLLT

Upon detecting an attack, the traceback information preserved for attack packets should be collected immediately from the routers. Otherwise, there is a risk for this information to be cleared off. This is due to the fact that a Bloom filter of a given size can maintain the digests of certain number of packets without exceeding the specified false positives rate. When that number is reached, the router has to clear the Bloom filter and use it to store the digests of the

next round of packets. The probabilistic nature of the proposed schemes and the possibility of activating them on selective flows (As discussed earlier in this subsection) increases the window of time through which the traceback information can be collected successfully.

4.5 Performance Evaluation

We evaluate the performance of the proposed schemes through a combination of theoretical and simulation studies. Theoretically, we bound the number of packets required by DLLT/PPPM for full path construction and compare it to that required by the PPM scheme [51] (subsection 4.5.1), and we quantify the storage requirement of DLLT and PPPM (subsection 4.5.2). In addition, we evaluate the performance of the proposed schemes through extensive simulation experiments against various performance metrics (subsection 4.5.3).

4.5.1 Number of Attack Packets Required to Construct Full Attack Path

It is obvious that the proposed schemes are similar in functionality, in the sense that marking information that belongs to a certain packet is preserved to be later collected by the victim. While the number of packets required by the victim to construct a full attack path is different in both schemes (few additional packets are required by the PPPM scheme to flush the pipeline), the number of packets that must be sent by an attacker such that every router along the path to the victim is involved in marking at least one of these packets is the same in both schemes (assuming the same q in both cases). Our objective is to find a bound on the minimum number of packets that has to be sent by the attacker such that every router along the path to the victim is involved in marking at least one of these packets with high confidence probability u. Let k represent this lower bound. Let the marking probability at router R be q. Let P_f be the probability that R fails to mark any packet out of k packets. Clearly, $P_f = (1-q)^k$. Therefore, the probability that R succeeds in marking (we call it the success probability) at least one packet is given by:

$$P_s = 1 - P_f = 1 - (1-q)^k \tag{4.1}$$

To obtain the desired bound, we can safely assume that the success probability for all routers along a path of length d is the same and equal to that of the farthest router (i.e., as given in Equation (4.1)). If we define X to be a random variable that represents the number of routers out of d that succeed in the marking process, then X follows the binomial distribution with success probability P_s given by Equation (4.1). We need to find k such that: $P(X = d) \geq u$ (i.e., the probability that each router succeeds in marking at least one packet is larger than u). But, $P(X = d) = (1 - (1 - q)^k)^d \geq u$, solving for k, we obtain:

$$k \geq \frac{\log_{10}(1 - u^{1/d})}{\log_{10}(1 - q)} \tag{4.2}$$

By following a similar approach, it can be shown that the minimum number of packets required by the PPM scheme [51] (here we consider edge sampling technique without edge encoding) is given by:

$$k_{ppm} \geq \frac{\log_{10}(1 - u^{1/d})}{\log_{10}(1 - q(1 - q)^{d-1})} \tag{4.3}$$

It may be argued that the 16-bit mark restriction of existing probabilistic packet marking schemes (e.g., PPM [51]) increases their packet requirement. However, it is important to point out that this restriction is not the main reason for their requirement of large number of packets for traceback. Such schemes require large number of packets originally because downstream routers overwrite upstream router's marking. It is important to mention that the bound in Equation (4.3) does not assume edge encoding (i.e., no restriction to 16-bit mark). In contrast, it assumes that an edge can be represented by three fields (of a total size of 72 bits): start, end, and distance. On the other hand, the significant reduction of the number of packets achieved in our schemes is due to the marking information preservation, which is part of the hybrid design. Therefore, it is not because of the relaxed assumption of a long mark.

Fig. 4.12 (a) provides a comparison between k and k_{ppm} as the marking probability, q, varies from 0.05 to 0.3. d and u were set to 15 and 95%, respectively. In PPM, whenever a router decides to mark a packet, it overwrites the packet's marking information. Therefore, increasing the marking probability, q, implies that the chances for a packet to be marked by

a router far away from the victim and not remarked by any subsequent router will decrease. For a given q, this explains the sharp increase in the number of packets required by PPM to obtain a mark from each router along the path. In contrast, DLLT and PPPM are designed to preserve the marking information written by any router along the packet's path (recall that whenever a router decides to mark a packet, it preserves the packet's marking information). Increasing q implies that more routers are expected to mark a given packet. Since all markings are preserved and then collected by the victim, the amount of path information obtained per packet increases by increasing the marking probability. It is important to mention that very small values of q should be avoided in PPM to prevent the problem of marking field spoofing [52][5].

We also study the effect of path length on the number of packets required for attack path reconstruction. Fig. 4.12 (b) provides a comparison between k and k_{ppm} as the path length, d, varies from 5 to 25. q and u were set to 0.2 and 95%, respectively. Since routers in a path appear serially, the probability that a packet will be marked by a router and then left unmarked by all downstream routers is a strictly decreasing function of the distance to the victim in the case of PPM. Therefore, increasing the path length reduces the chance to obtain marks from far away routers. Hence, more packets are required to ensure that all routers along the path are represented by at least one mark. Coming to DLLT/PPPM, the number of packets required for traceback does not remain constant as path length increases. In fact, there is a slight increase which is difficult to notice in the plot because it is too small compared to the scale of the Y axis. This increase, though slight, is intuitive because more packets would be required to cover a longer path. However, it is not significant because as the path length increases more routers are expected to mark a given packet, which means that the amount of path information obtained per packet increases as well.

[5]Notice that there is no restriction on q under DLLT/PPPM.

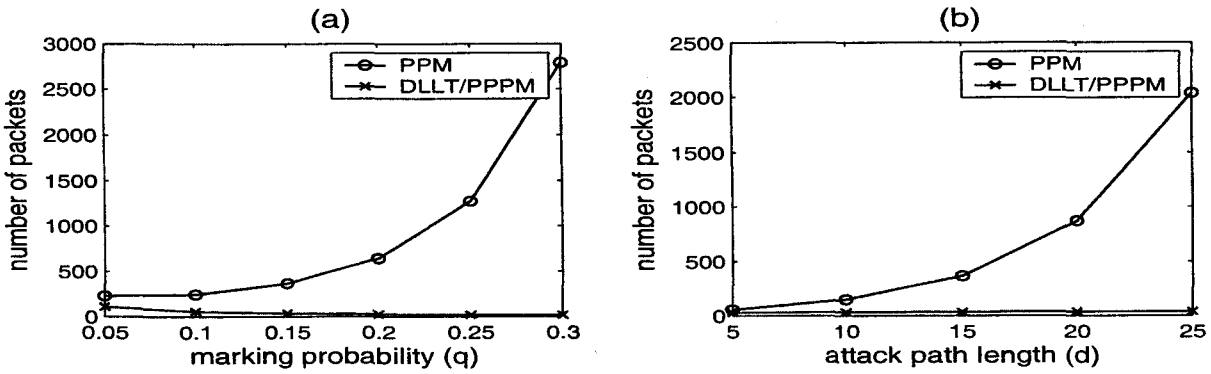


Figure 4.12 A comparison between the number of packets required by DLLT/PPPM and that required by PPM: (a) Effect of marking probability, q. Attack path length, d, was fixed to 15 (b) Effect of attack path length. Marking probability, q, was fixed to 0.2.

4.5.2 Storage Analysis

The amount of storage that needs to be allocated for a traceback scheme is an important issue. In this subsection, we quantify the amount of memory required in both DLLT and PPPM schemes. First, we review some characteristics of the Bloom filters that would be necessary in our analysis (detailed description of Bloom filters is available in Appendix B). A Bloom filter is characterized by its size, s bits, the number of hash functions used, j, and its capacity factor, f. A Bloom filter of size s and capacity factor f can be used to store the digests of at most s/f packets. The effective false positive rate of a Bloom filter is directly dependent on the previous parameters. What follows is a quantification of memory requirement at each router in the two schemes.

4.5.2.1 Memory Requirement of DLLT (M_{dllt})

Let b denote the number of bits required to store the marking information of one packet in the MIT (i.e., this includes 32-bit IP address plus $\lceil \log_2(j) \rceil$ bits for the "hash function number" field). To store the marking information of x packets, we need an MIT of size xb bits to be *shared* among f digests arrays each of size x bits. Therefore, the total memory requirement at each DLLT enabled router, (M_{dllt}), to store the marking information of x packets is given

by: $M_{dllt} = x(b+f)$. Assuming an aggregate incoming link capacity of n packets/sec and a marking probability of value q at each router, x has a binomial distribution with parameters (n, q). Therefore, the amount of memory required to store a second's worth of digests has the following probability mass function.

$$P(M_{dllt} = i) = \begin{cases} \binom{n}{(i/(b+f))} q^{i/(b+f)}(1-q)^{n-(i/(b+f))} & \text{if } i = k(b+f), k = 0, 1, 2, ...; \\ 0 & \text{otherwise.} \end{cases}$$

With an average value given by $\mathrm{E}[M_{dllt}] = qn(b+f)$.

4.5.2.2 Memory Requirement of PPPM (M_{pppm})

Typically, at each PPPM enabled router, a marking buffer that has 2^{32} entries would be required (i.e., one entry for each possible destination). In practice, the number of destinations seen by a router during a small window of time is limited. By taking advantage of this observation, we limit the size of the marking buffer to 2^a entries, where a is the size of the IP destination address suffix used to index the buffer. As mentioned in subsection 4.3.2, a Bloom filter is also used to indicate whether a given destination has a marking information or not. Therefore, at each PPPM enabled router, a fixed size buffer with 2^a entries would be required to preserve the marking information of the forwarded packets. Each entry has a size of 57 bits, which include 32-bit IP address, 17-bit ID, and 8-bit distance field. In addition, a Bloom filter of size fx would be required to store the digests of at most x different destination addresses. Assuming the percentage of different destination addresses seen by a router in a sample of n packets is p, then the number of addresses, x, to be inserted in the Bloom filter within one second has a binomial distribution with parameters (n, pq), where n is the aggregate link capacity per second, and q is the marking probability. Therefore the memory requirement of PPPM, M_{pppm}, can be expressed as the sum of a fixed memory of size 57×2^a bits, and a Bloom filter of size S, where S has the following probability mass function.

$$P(S = i) = \begin{cases} \binom{n}{(i-b)/f} (qp)^{(i-b)/f}(1-qp)^{n-(i-b)/f} & \text{if } i = b + kf, k = 0, 1, 2, ...; \\ 0 & \text{otherwise.} \end{cases}$$

With an average value given by $\mathrm{E}[S]$=$qpnf$.

4.5.3 Simulation Studies

The simulation studies presented in this section focus on the process of identifying attack sources without specifying the scheme used for packet marking. The assumption made here is that marking information that belongs to a certain packet can be collected by the victim either under DLLT scheme, or under PPPM scheme. The objective is to show how the victim can identify attack sources, reconstruct attack paths, and determine the actual number of attackers. The extensive simulation experiments carried out in this section are: (i) to evaluate the proposed schemes in terms of different performance metrics, and (ii) to support our claim about their superior performance compared to PPM. We start by discussing the source identification algorithm performed by the victim. Then, we define the performance metrics, followed by a description of the simulation method and the results.

4.5.3.1 Attack Source Identification

There is an ongoing debate regarding the outcome of a traceback scheme. Some researchers support the notion of locating attackers regardless of providing any information about the paths followed by the attack packets. However, other researchers support the notion of constructing the complete path followed by the attack packets leading to the attacker's location. We believe that such decision must be made based on the actual number of attackers (classifying DoS attacks based on the number of attackers was considered in [78]). In the case of having few attackers (e.g., less than 50), it will be necessary to locate most of them in order to stop an ongoing attack, or to hold attackers accountable. However, in the case of having large number of attackers, it would be difficult (technically and administratively) to place filters at their sites. Even, knowing all their identities would not be necessary to find out the original attacker (recall that the systems identified by a traceback process are those who have been compromised earlier by the master machine which is under the control of the original attacker). Therefore, it would be more useful to have knowledge about the paths followed by attack packets such that filters can be placed at their intersection points.

We propose an algorithm, called Identification and Reconstruction (IR) algorithm, that is

able to identify attack sources and/or reconstruct attack paths. For any router R, recall from section 4.1 that the set of routers on the attack path between the victim and R is called the predecessor set of R, whereas the set of routers on the attack path between R and the source is called the successor set of R. Clearly, if we are interested in locating attackers, then we have to find routers that have an empty successor set. The IR algorithm, shown in Fig.4.13, takes the marking information that belongs to k attack packets as an input. The algorithm performs two passes on the given input. In the first pass, IR algorithm creates a table with two columns: end-list routers and their partial predecessor lists. For each subset of routers that marked a certain packet (i.e., routers found in a MIC-reply packet if DLLT is used, or routers found in the packets that have the same ID if PPPM is used), it adds a new entry for the end-list router if it does not already exist and adds its predecessors found in that packet. In the second pass, IR eliminates the end-list routers that appears in the predecessor lists of other routers. The remaining end-list routers represent the set of identified attack sources. If each router is associated with distance parameter (as in the PPPM scheme), then the routes within each predecessor list can be ordered according to their positions in the actual path, which helps in the process of full path construction.

4.5.3.2 Performance Metrics

We define and use the following new metrics to evaluate the proposed schemes:

- *Path coverage ratio (PC)*: This metric is defined as the ratio of the length of the reconstructed attack path to the length of the actual attack path. A value of 1 indicates that the path is fully reconstructed.

- *Attack source localization distance (ALD)*: This metric defines the distance between the identified attack source and the actual attack source. If more than one attack path is discovered by the algorithm, then an average ALD is computed for all paths. A value of zero for ALD means that the exact attack source is identified.

- *Number of attackers ratio (AR)*: This metric is defined as the number of leaves in the

IR (Marking Info. of k Attack Packets)

Pass I

1. *for* each subset of routers that marked certain packet (P)
   (a) *if* (P.end-list router is not in the table)
      i. add new entry for P.end-list router
      ii. add P.remaining routers to the predecessor list of P.end-list router
   (b) *else* add P.remaining routers to the predecessor list of P.end-list router

Pass II

1. *for* each end-list router, ER_x, in the table
   (a) *if* (ER_x appears in the predecessor list of any other end-list router, ER_y)
      i. append predecessor list of ER_x to the predecessor list of ER_y
      ii. remove ER_x and its predecessor list from the table

Figure 4.13 Identification and reconstruction (IR) algorithm. Remaining end-list routers represent the identified attack sources, and their predecessor lists represent attack paths.

reconstructed attack tree to the number of leaves in the actual attack tree (recall that attack paths converge at the victim to form an attack tree). Typically, this ratio should be 1. However, in the proposed identification and reconstruction algorithm, it is possible to identify more than one node on the same actual attack path to be attack sources (i.e., tree leaves). This represents a deviation from the desirable outcome of identifying only one attack source (the farthest from the victim among the identified sources on the same path) for each attack path. AR captures this deviation.

- *Detection percentage (DP)*: This metric is defined as the percentage of exactly identified attack sources. For example, if j attack sources are exactly located out of m attack sources, then we express the detection percentage as $\frac{j}{m} \times 100$ %.

Table 4.1 Simulation parameters

parameter	range (default value)
number of attackers (m)	1, 100
number of attack packets used (k)	5 ... 100
marking probability (q)	0.05 ... 0.15
attack path length (d)	(20)

4.5.3.3 Simulation Method

In each simulation experiment we generated a random attack tree with m attackers and one victim. The attack path length, d, was the same for all attackers. Packets were marked according to the specified marking probability, q. Attackers were instructed to inject their packets simultaneously with a rate of 1000 packets/attacker. Table 4.1 shows the range of each parameter (values in parenthesis represent the default values of the corresponding parameters when applicable). We performed simulation experiments under the assumption that: (i) Edge based marking is not enforced by the underlying marking scheme. We use the letter (n) to refer to this assumption. (ii) Edge based marking is enforced by the underlying marking scheme. We use the letter (e) to refer to this assumption. In the graphs, the legend (n, q) represents assumption (n) under the given marking probability, q. While legend (e, q) represents assumption (e) under the given marking probability, q. Each of the following results represents the average of 200 independent simulation runs based on the simulation parameters values shown in Table 4.1 unless otherwise specified. In a practical scenario, the only parameter under the control of the victim is the number of packets, k, for which the marking information has to be collected. We take this fact into account by making this parameter the main variable in our simulation[6] .

[6]In the case of having m attackers, mk packets were actually used by the victim. However, only variable k is shown in the graphs.

4.5.3.4 Simulation Results

Figures 4.14 (a) through 4.14 (d) show the path coverage ratio, the attack localization distance, the detection percentage, and the number of attackers ratio, respectively, as a function of the number of packets used, k, for different values of q, under both assumptions (n) and (e). These figures correspond to single attacker case (i.e., $m = 1$). Figures 4.15 (a) through 4.15 (d) show the same performance metrics, but correspond to multiple attackers case (m was set to 100). The following discussion applies for both cases (i.e., single and multiple attackers). The main difference to be observed between them is that the multiple attackers case corresponds to smoother graphs and better performance in general. This is due to the fact that we obtain more marking information about the same path (or parts of the path) when it is shared among several attackers.

Path coverage ratio: This parameter depends on the amount of information obtained by the victim about the path followed by the attack packets. Therefore, it is natural to see this parameter increasing by increasing the value of the marking probability, q, or by increasing the number of packets, k, used by the victim, or by increasing both of them as can be seen in Fig. 4.14 (a). It can also be seen that enforcing edge based packet marking leads to better path coverage in general, because obtaining edge information facilitates full path reconstruction compared to the case in which separate router information is obtained. It is also important to point out that the simulation results agree with the theoretical analysis regarding the number of packets required for full path reconstruction. For example, when $q = 0.1$, 60 packets are required, which is very close to the minimum bound specified by Equation (4.2) which suggests using 55 packets under similar conditions.

Attack localization distance: Fig. 4.14 (b) compares the average attack localization distance obtained for different values of q under assumptions (n) and (e). By increasing the number of packets used, more path information is obtained, and the victim starts to realize the actual shape of the attack path. This explains the continuous decrease of the average attack localization distance as k increases. Increasing the marking probability has similar effect as can be seen in the figure. It is also clear that enforcing edge based packet marking reveals

more details about attack path, contributing to locating attacker's location precisely.

Attack source detection percentage: As pointed out earlier, in some cases it is important to detect attack source without necessarily reconstructing the complete attack path. To be able to identify an attack source (i.e., the closest router to the attacker), the victim must obtain the IP address of that router, and must recognize it as an end-list router along some path. This does not depend on whether the marking scheme enforces edge-based marking or not, which is reflected in the results shown in Fig. 4.14 (c), where the attack source detection percentage is the same under both assumptions (n) and (e). As expected, the attack source identification percentage increases by increasing k, q, or both of them.

Number of attackers ratio: The number of nodes identified by the victim as an attack nodes exhibits an interesting behavior as k increase. This is depicted in Fig. 4.14 (d). Initially, if only one packet is used (i.e., $k = 1$), then one node will be identified as an attack source (this node may not necessarily be the actual attack node). By increasing k, more nodes will be identified as attack sources. This increase will continue until enough path information is obtained, where the victim becomes able to place nodes into their right positions, eliminating false attack source identification. In practice, the victim can determine the actual number of attackers by observing that the number of identified attack sources remains constant no matter how much k is increased. The ratio of the number of identified attack sources is a measure of how much information is required by the victim to determine the exact number of attackers. It can be seen that increasing q, enforcing edge based marking, or both, results in lower ratio.

Comparison of PPPM/DLLT with PPM scheme: We performed extensive simulation experiments to compare the performance of the proposed schemes to that of the PPM [51] scheme. Packet marking in the forward direction was performed based on the hybrid schemes (i.e., DLLT/PPPM) in our case, and based on the PPM scheme in another case. The number of packets used for traceback, k, was varied from 5 to 100 in steps of 5. To capture the effect of the marking probability, q, the results were taken for $q = 0.10$ and $q = 0.15$. For the purpose of discussion, we consider single attacker case (i.e., $m = 1$) with attack path length of 20 hops. Figures 4.16 (a) through 4.16 (d) show the performance comparison between the proposed

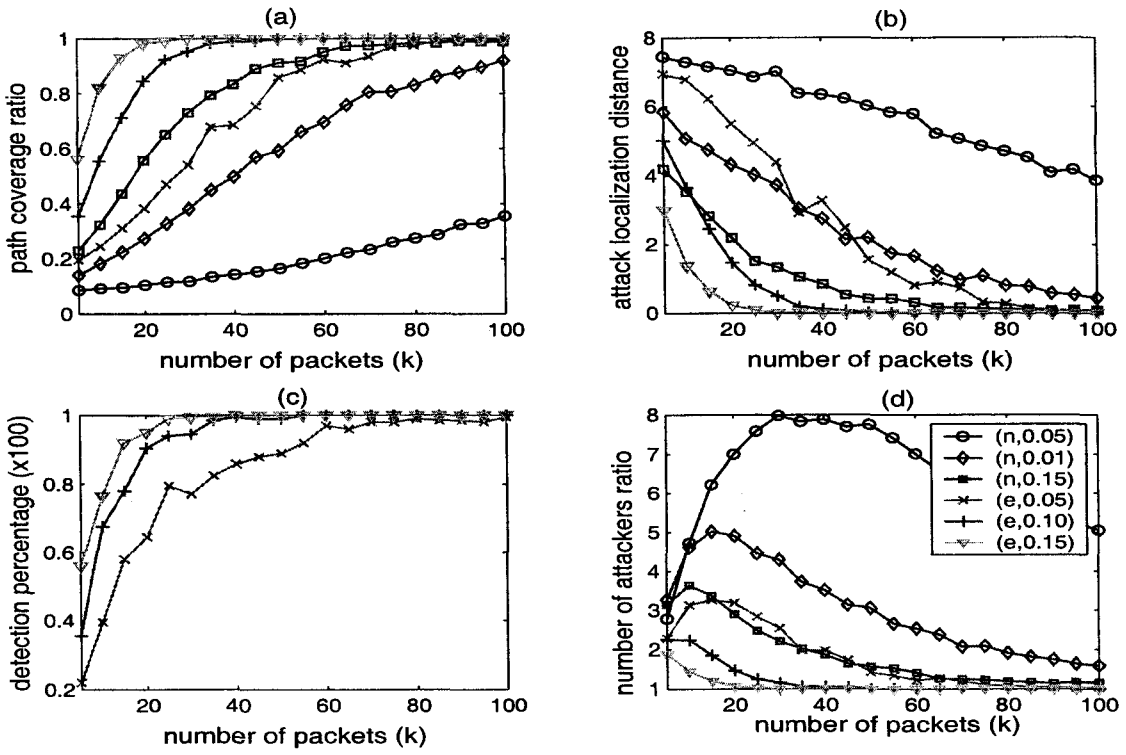


Figure 4.14 Single attacker case. (a) Path coverage ratio. (b) Average attack localization distance. (c) Attack source identification percentage. (d) Number of attackers ratio.

schemes and the PPM scheme. It is obvious that the hybrid schemes outperform the PPM scheme significantly. By increasing k, DLLT/PPPM converges very fast to the actual path. For example, path coverage ratio, average attack localization distance, detection percentage, and number of attackers ratio are exactly, 1, 0, 100%, and 1, respectively, when $k = 50$. At the same time, the path obtained by PPM remains far away from the real one for the range of k used in this part[7]. This is reflected in the low path coverage ratio (Fig. 4.16 (a)), high attack localization distance (Fig. 4.16 (b)), zero attack detection percentage (Fig. 4.16 (c)), and relatively large number of attackers ratio (Fig. 4.16 (d)).

[7]PPM requires a lot of packets to converge to the actual path.

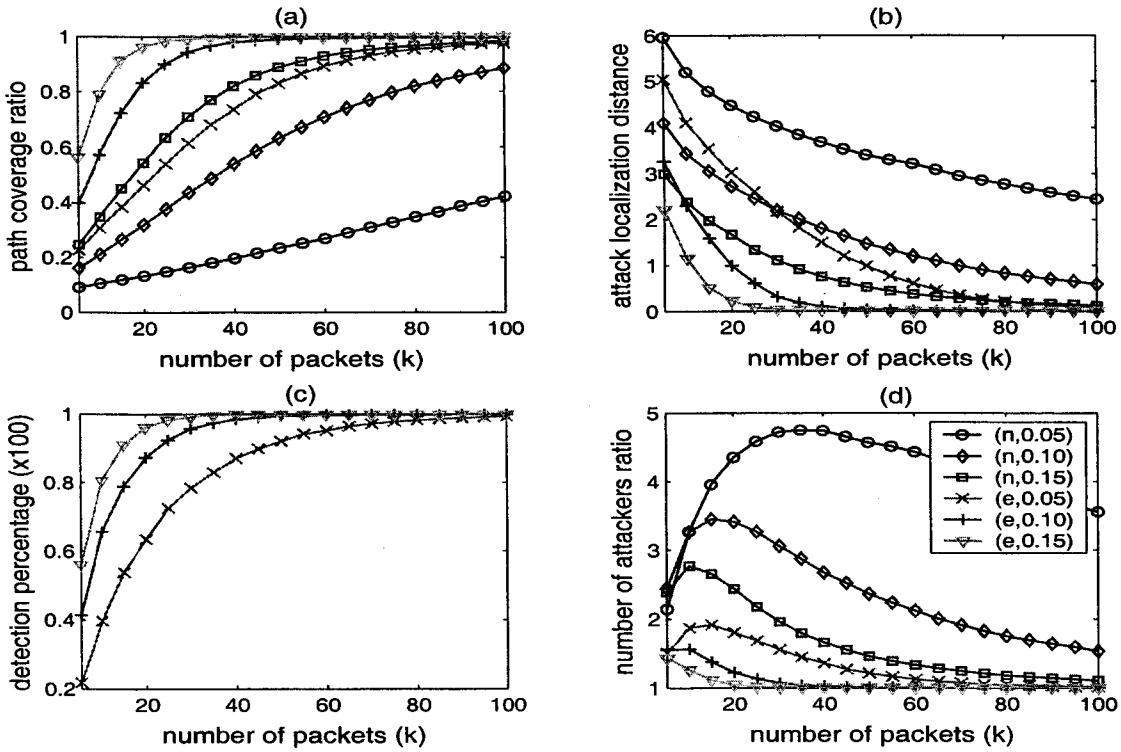


Figure 4.15 Multiple attacker case. (a) Path coverage ratio. (b) Average attack localization distance. (c) Attack source identification percentage. (d) Number of attackers ratio.

4.6 Chapter Conclusions

An efficient traceback scheme is necessary to identify the sources of DoS attacks which impose an imminent threat to the availability of Internet services. The work presented in this chapter adopts a hybrid traceback approach in which packet marking and packet logging are integrated to achieve the best of both worlds (i.e., small number of attack packets to conduct the traceback process, and small amount of resources to be allocated at intermediate routers for packet logging purposes). Based on this notion, two traceback schemes were proposed. The first scheme, called *Distributed Link-List Traceback* (DLLT), is based on the idea of preserving the marking information at intermediate routers in such a way that it can be collected in an efficient manner. The second scheme, called *Probabilistic Pipelined Packet Marking* (PPPM), employs the concept of "pipeline" for propagating marking information from one marking

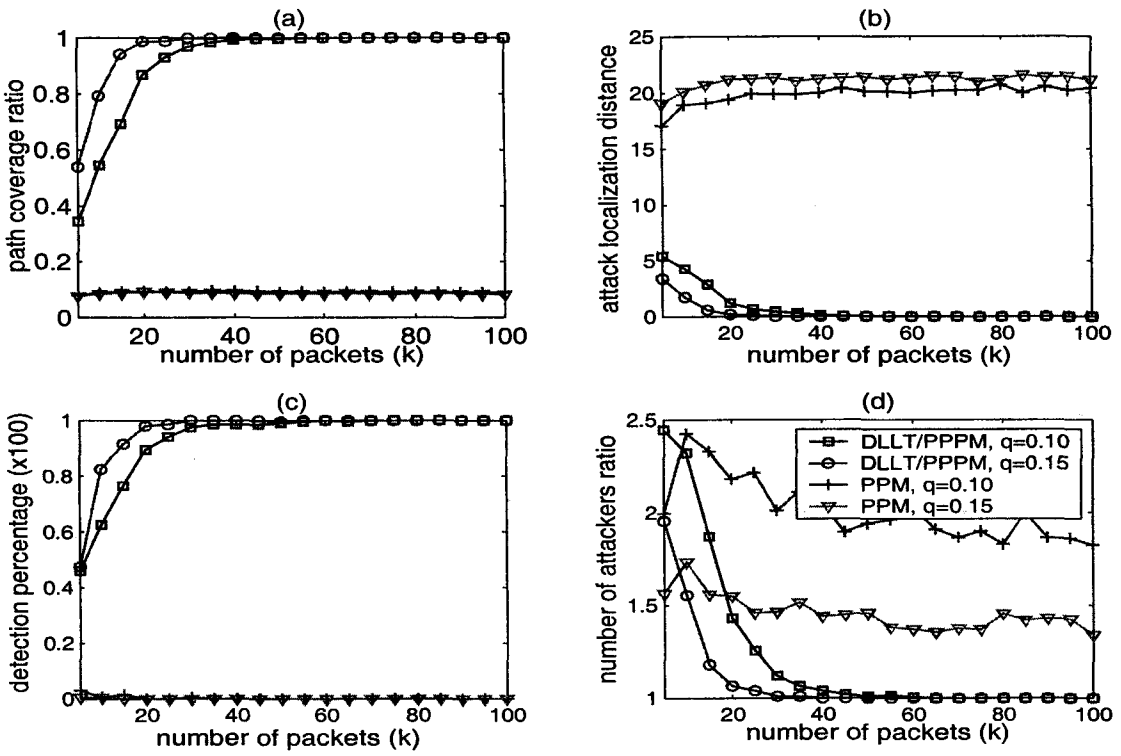


Figure 4.16 Comparison between DLLT/PPPM and PPM [51] in terms of: (a) Path coverage ratio. (b) Average attack localization distance. (c) Attack source identification percentage. (d) Number of attackers ratio.

router to another so that it eventually reaches the destination.

The proposed schemes enjoy many of the features mentioned in section 4.1. Their probabilistic nature of marking and storage offers the advantage of minimizing router's processing and storage overhead. Also, both schemes eliminate attacker's ability to mislead the victim. This is achieved in DLLT by storing the packet digests at intermediate routers, which provides an authentic way to verify that a given router has actually forwarded certain packet. In PPPM, spoofed marking information written by the attacker can be discarded by observing that the distance associated with it is always the largest among distances obtained for marking information that correspond to the same packet. Marking information is collected from intermediate routers efficiently. For example, DLLT collects relevant marking information from specific routers in a predetermined manner using the link list approach. PPPM collects the

marking information by loading them into packets going to the same destination.

The analysis carried out in section 4.5 provides a theoretical background about the performance of DLLT and PPPM in terms of the number of packets required for full attack path reconstruction and in terms of their storage requirements. It has been shown also that the number of packets required in DLLT/PPPM is substantially less than that required by PPM [51].

We draw the following conclusions from the simulation results presented in subsection 4.5.3:

- DLLT/PPPM offer high detection percentage under different design parameters.
- Enforcing edge based packet marking results in better traceback capability.
- Attack path(s) reconstruction is one way in determining the number of attackers participating in the attack.
- The source of attack can be localized exactly if reasonable number of packets are used for traceback.
- There is a trade-off between different parameters. We showed how different parameters can affect the performance metrics in various ways.

We do not claim that DLLT and PPPM are perfect IP traceback schemes. In general, for such schemes to be practically efficient they must be secure, backward compatible and requires low storage. A study in terms of saving information for "selective packet flows" instead of "per packet information" may improve the scalability of the proposed hybrid IP traceback schemes. Also, it is possible to use some machine learning algorithms to initiate the proposed schemes on the suspected paths so as to capture enough information for IP traceback.

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

Denial of service attacks represent an imminent threat to the availability of Internet services. These attacks can quickly incapacitate a targeted business, leading to lost revenue and productivity. The alarming increase in the number of these attacks against e-commerce companies and other organizations coupled with the emergence of newly sophisticated attacks have increased the fear from potential powerful coordinated attacks. All these factors have led to a significant amount of research in recent years aiming at countering these attacks on all fronts. Given that attack packets differ from the legitimate ones in intent but not in content, the focus has been always on developing mechanisms to distinguish attack packets from legitimate packets, and on the issue of locating actual sources of attack packets. This has resulted into three major approaches to counter DoS attacks, namely, prevention, mitigation, and traceback. Although DoS prevention is the most desirable solution, it is very difficult to achieve due to: (1) the fact that DoS attacks are emerging in various forms and becoming more complicated, and (2) the proactive nature of prevention schemes makes their deployment less incentive, especially because DoS attacks are the exception rather than the norm. This explains why most of the research efforts in this area have been mainly on DoS mitigation and IP traceback.

The work presented in this dissertation advances the state of the art in DoS mitigation and IP traceback. In particular, we have shown that it is possible to perform online perimeter-based DoS mitigation. Also, we have shown that it is possible to combine packet marking and packet logging in a novel manner such as to achieve efficient IP traceback functionality. More specifically, the following is a summary of the work presented in this dissertation.

- In Chapter 1, we discussed the problem of DoS attacks focusing on their original causes, and on their various mechanisms. We also identified several attributes to characterize

DoS attacks. These attributes include header spoofing, attack indirection, attack amplification factor, attack rate dynamics, number of attackers vs. number of victims, and attacker's awareness to the defense mechanism. In addition, we projected known and potential DoS attacks onto a means-based taxonomy as well as onto an impact-based taxonomy, and we discussed the main research challenges associated with the DoS attacks problem.

- In Chapter 2, we provided an extensive discussion of the state of the art research in countering DoS attacks, focusing mainly on the major defense mechanisms including prevention, mitigation, and IP traceback. In addition, we discussed the main approaches in each category including source-based prevention, network-based prevention, victim-based DoS prevention, rate limiting-based mitigation, statistical-based mitigation, path-based mitigation, packet marking-based traceback, packet logging-based traceback, and traffic engineering-based traceback.

- In Chapter 3, two novel concepts were proposed for DoS mitigation, namely, victim-assistance and protocol-determinism. These concepts were mainly proposed to supplement edge routers of an ISP network by mechanisms that enable them to distinguish attack traffic from legitimate traffic. In this context, the novelty of the proposed concepts is due to their benefit in performing online DoS mitigation at an ISP perimeter. A feature that is not available in current perimeter-based DoS mitigation schemes.

- In Chapter 4, two novel IP traceback schemes were presented. The first scheme, called *Distributed Link-List Traceback* (DLLT), is based on the idea of preserving the marking information at intermediate routers in such a way that it can be collected using a link list based approach. The second scheme, called *Probabilistic Pipelined Packet Marking* (PPPM), employs the concept of a "pipeline" for propagating marking information from one marking router to another so that it eventually reaches the destination. We evaluated the effectiveness of the proposed traceback schemes against various performance metrics through a combination of analytical and simulation studies. Our studies show that the

proposed schemes offer a drastic reduction of the number of packets required to conduct the traceback process in comparison to the well known PPM scheme.

Despite the significant amount of research that has been made in the area of DoS attacks in the past few years, there is no real evidence that these efforts have fully succeeded in countering these attacks. On the contrary, the recent reports about DoS attacks confirm that the Internet infrastructure is still exposed to the danger of these attacks, and the problem will not disappear any time soon. We believe that the work presented in this dissertation represents an important step towards reducing the impact of these attacks. However, several issues still need to be investigated. In particular, we identify the following issues which open up new research directions in the field of DoS attacks.

- *Employing the concept of protocol-determinism in mitigating protocol exploitation-based DoS attacks.* As discussed in Chapter 1, protocol exploitation-based DoS attacks (also known as control flow-based DoS attacks) take advantage of the deterministic nature of Internet protocols. We believe that an approach similar to the one we proposed for SYN flooding mitigation (which is a special case of control flow-based DoS attacks) can be followed for mitigation of other types of control flow-based DoS attacks.

- *Employing the concept of victim-assistance in mitigating direct DoS attacks.* We have shown that the concept of victim-assistance aids significantly in the mitigation of RDoS attacks. It would be interesting to investigate the viability of employing this concept in defending against direct DoS attacks as well. In order to achieve this goal, several issues need to be addressed. For example, what information should be provided by the victim? How this information is to be communicated to edge routers? When to activate/terminate the mitigation scheme and on which edge routers? How to extend the concept of victim-assistance to inter-domain setting? Does collaboration among edge routers help in anyway?

- *Improving the scalability of the proposed traceback schemes.* In general, for DLLT/PPPM schemes to be practically efficient they must be secure, backward compatible and requires

low storage. A study in terms of saving information for "selective packet flows" instead of "per packet information" may improve the scalability of the proposed hybrid IP traceback schemes. Also, it is possible to investigate using some machine learning algorithms to initiate the proposed schemes on the suspected paths so as to capture enough information for IP traceback.

- *Designing hardware components to support the proposed schemes.* A major challenge in dealing with DoS attacks, in any countermeasure, is the need to perform per packet processing at very high Internet speeds. The proposed DoS mitigation and traceback schemes (PF, SNF, intentional dropping, DLLT, and PPPM) are not an exception. In fact, these schemes introduce additional processing delays due to TCP header inspection (mitigation), or due to packet marking and logging logging (traceback). A possible research is to (1) study the impact of this overhead on the performance of the proposed schemes, and (2) design hardware components that support the operation of these schemes at very high Internet speeds.

- *DoS detection and mitigation using edge routers cooperation.* Cooperation among an ISP's edge routers can aid significantly in detecting attacks of type multi-attackers multi-victims, which is one of the most complex form of attacks. We suggest investigating the following two approaches to determine which one is more suitable for detecting this type of DoS attacks.

  - *Approach 1: IDS-based DoS attack detection.* Usually, an intrusion detection system (IDS) raises an alarm indicating the existence of a DoS attack only if the rate of the attack traffic reaching the monitored system is above certain threshold. This means that the system will not be considered to be under attack even if it receives attack traffic unless the rate of this traffic is high enough. In our view, the fact that there is some sort of unwanted traffic reaching the system, even at a low rate, should be taken seriously. By recalling that such traffic can be part of an engineered attack that aims at disturbing the ISP's network operation rather than bringing that particular

system down. If each system ignores such attack, then attack traffic will overload the ISP's network without being detected. We believe that sharing the information about incoming attack traffic among these systems can provide complete picture about the actual threat level. By exchanging this information, each edge router will have complete attack rate information. Hence, all edge routers will come to the same conclusion about the existence of an attack.

  - *Approach 2: Health-based DoS attack detection.* In this approach, there is no need for individual systems to report attack rates. The existence of DoS attacks is usually reflected on the ISP's network health (i.e., delay, loss rate, etc.) immediately, especially if the attack is very intense. By sharing information about delay and packet loss parameters, edge routers obtain global approximate picture about the state of the network. A decision about whether the network is under attack or not can be made accordingly.

- *Internet worms: detection and containment.* Internet worms represent another persistent threat to the Internet infrastructure. Despite their devastating effects, the research done so far in the area of detection and containment of Internet worms is very limited and has been focused mainly on issues such as the analysis of Internet worm impact, and modeling of Internet worm propagation. However, only few and not very effective mechanisms have been proposed for detection and preventing propagation of Internet worms.

- *Developing a comprehensive defense strategy.* We believe that integrating the major defense paradigms (i.e., prevention, mitigation, and traceback) may lead to an effective DoS defense strategy. In fact, very few research efforts had been made in this direction. For example, the authors in [43] proposed a DoS mitigation scheme that takes advantage of existing IP traceback schemes. As another example, the authors in [56] showed that their distributed packet filtering architecture (which is basically intended for DoS prevention) can localize the attacker's source AS's within 5 sites. These research efforts represent an important step toward realizing the vision of integrated prevention, mitigation, and traceback.

APPENDIX A. DERIVATION OF THE EXPECTED VALUE OF $A_{reactive}$

First, we outline the basic steps of finding the distribution of

$$A_{reactive} = \frac{D_v}{D_a + D_v} \tag{A.1}$$

1. We find an expression for $f_{A,V}(a, v)$, the the joint PDF of D_a and D_v.

2. We obtain an expression for $f_{U,Z}(u, z)$, the joint PDF of the two random variables U and Z. Where,

$$U = D_v \tag{A.2}$$

and

$$Z = A_{reactive} = \frac{D_v}{D_a + D_v} \tag{A.3}$$

3. The PDF of Z is obtained by integrating $f_{U,Z}(u, z)$ with respect to u from 0 to ∞.

What follows is a detailed derivation of $f_Z(z)$ based on the above outline.

D_a and D_v are assumed to have exponential distributions with rates μ_a and μ_v, respectively. Therefore,

$$f_A(a) = \mu_a e^{-\mu_a a} \tag{A.4}$$

and

$$f_V(v) = \mu_v e^{-\mu_v v} \tag{A.5}$$

Since D_a and D_v are independent, their joint PDF is given by:

$$f_{A,V}(a, v) = \mu_a \mu_v e^{-\mu_a a} e^{-\mu_v v} \tag{A.6}$$

It is to be noted that U and Z satisfy the following conditions:

1. The equations $u = v$ and $z = \frac{v}{v+a}$ have as their solution $v = u$, $a = \frac{v - zv}{z}$.

2. u and z have continuous partial derivatives at all points (v, a) and are such that the following determinant

$$J(D_v, D_a) = \begin{vmatrix} \frac{\partial U}{\partial D_v} & \frac{\partial U}{\partial D_a} \\ \frac{\partial Z}{\partial D_v} & \frac{\partial Z}{\partial D_v} \end{vmatrix} = \frac{-v}{(v+a)^2} \neq 0 \tag{A.7}$$

at all points (a,v), where J(D_v,D_a) is the Jacobian matrix.

It follows that the desired density is:

$$f_{U,Z}(u, z) = f_{V,A}(v, a) \times \frac{-1}{J(D_v, D_a)} \tag{A.8}$$

$$f_{U,Z}(u, z) = f_{V,A}(v, a) \frac{(v+a)^2}{v} \tag{A.9}$$

substituting $v = u$ and $a = \frac{v-zv}{z}$, we obtain

$$f_{U,Z}(u, z) = \frac{u\mu_a\mu_v}{z^2} e^{-(\mu_v - \mu_a + \frac{\mu_a}{z})u} \tag{A.10}$$

Therefore,

$$f_Z(z) = \int_0^\infty \frac{u\mu_a\mu_v}{z^2} e^{-(\mu_v - \mu_a + \frac{\mu_a}{z})u} du = \frac{\mu_a\mu_v}{(z\mu_v - z\mu_a + \mu_a)^2} \tag{A.11}$$

$$E[A_{reactive}] = \frac{\mu_a\mu_v}{(\mu_v - \mu_a)^2} (\log \frac{\mu_v}{\mu_a} + \frac{\mu_a}{\mu_v} - 1) \tag{A.12}$$

APPENDIX B. BLOOM FILTERS

A Bloom filter [69] is a data structure for representing a set of n elements (also called keys) to support membership queries. The idea (illustrated in Fig. B.1) is to allocate a vector R of m bits, initially all set to 0, and then choose k independent hash functions, each with range $\{1,...,m\}$. For each element, A, the bits at positions $H_1(A)$, $H_2(A)$, ..., $H_k(A)$ in R are set to 1. (A particular bit might be set to 1 multiple times.) Given a query for B, we check the bits at positions $H_1(B)$, $H_2(B)$, ..., $H_k(B)$. If any of them is 0, then certainly B is not inserted in the filter. Otherwise we conjecture that B is inserted in the filter although there is a certain probability that we are wrong. This is called a "false positive". The parameters k and m should be chosen such that the probability of a false positive is acceptable.

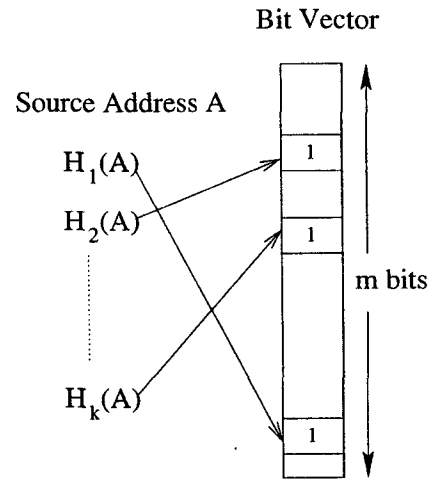


Figure B.1 A Bloom filter with k hash functions

The false positive rate of a Bloom filters is given by the following equation:

$$p_f = (1 - (1 - \frac{1}{m})^{kn})^k \approx (1 - e^{\frac{kn}{m}})^k \tag{B.1}$$

BIBLIOGRAPHY

[1] CERT Coordination Center. CERT Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL.
http://www.cert.org/advisories/CA-2001-19.html.
(date retrived: 15 March 2005).

[2] CERT Coordination Center. CERT Advisory CA-2001-26 "Nimda Worm".
http://www.cert.org/advisories/CA-2001-26.html.
(date retrived: 15 March 2005).

[3] CERT Coordination Center. Code Red II.
http://www.cert.org/incident notes/IN-2001-09.html.
(date retrived: 15 March 2005).

[4] CERT Coordination Center. Denial of Service Tools.
http://www.cert.org/advisories/CA-1999-17.html.
(date retrived: 15 March 2005).

[5] CERT Coordination Center. IP Denial-of-Service Attacks.
http://www.cert.org/advisories/CA-1997-28.html.
(date retrived: 15 March 2005).

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

[7] BizReport, "DDoS Attacks Still Pose Threat to Internet," Nov. 2003. http://www.bizreport.com/news/5413/
(date retrived: 10 February 2005).

[8] R. Lemos, "Attacks disrupt some credit card transactions," Sep. 2004. http://news.zdnet.com/2100-1009_22-5378217.html.
(date retrived: 10 Febreuary 2005).

[9] J. Leyden, "Online payment firm in DDoS drama," Nov. 2004.
http://www.theregister.co.uk/2004/11/03/protx_ddos_attack/
(date retrived: 10 February 2005).

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

[11] J. Mirkovic and P. Reiher, "A Taxonomy of DDoS Attacks and Defense Mechanisms," in *Proc. ACM SIGCOMM Computer Communications Review*, vol. 34, no. 2, April 2004. pp. 39-54.

[12] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," in *Proc. ACM SIGCOMM Computer Communication Review*, vol. 31, no. 3, July 2001.

[13] H. Wang, A. Bose, M. El-Gendy, and K. G. Shin, "IP Easy-pass: Edge Resource Access Control," in *Proc. of IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[14] The Swiss Education and Research Network. Default TTL values in TCP/IP, 2002. Available at: http://secfr.nerim.net/docs/fingerprint/en/ttldefault.html. (date retrived: 10 February 2005).

[15] C. Jin, H. Wang, and Kang G. Shin," Hop-Count Filtering: An Effective Defense Against Spoofed DDoS Traffic," in *Proc. ACM Conference on Computer and Communications Security (CCS)'2003*, Washington, DC, October 2003.

[16] CERT Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks. http://www.cert.org/advisories/CA-1998-01.html. (date retrived: 15 March 2005).

[17] CERT CC. DoS using nameservers. Attacks. http://www.cert.org/incedint_notes/IN-2000-04.html. (date retrived: 15 March 2005).

[18] Q. Li, E. C. Chang, and M. C. Chan, "On the Effectiveness of DDOS Attacks on Statistical Filtering," in *Proc. IEEE INFOCOM 2005*, Miami, FL, March 2005.

[19] CERT Advisory CA-1996-21, "TCP SYN Flooding and IP Spoofing Attacks," http://www.cert.org/advisories/CA-1996-21.html. (date retrived: 15 March 2005).

[20] S. Kandula, D. Katabi, M. Jacob, and A. Burger, "Botz-4-Sale: Surviving DDos Attacks that Mimic Flash Crowds," in *Proc. USENIX NSDI 2005*. To appear, Boston, MA, May 2005.

[21] S. Blake and et al., "An architecture for differentiated services," in *RFC 2475*, Dec. 1998.

[22] A. Habib, S. Fahmy, S. R. Avasarala, V. Prabhakar, and B. Bhargava, "On Detecting Service Violations and Bandwidth Theft in QoS Network Domains," in *Proc. Computer Communications*, vol. 26, no. 8, pp. 861-871, May 2003.

[23] A. Habib, M. Hefeeda, B. Bhargava, "Detecting Service Violations and DoS Attacks," in *Proc. Network and Distributed System Security Symposium (NDSS 2003)*, San Diego, CA, February 2003.

[24] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP)– Version 1 Functional Specification", RFC 2205, September 1997.

[25] A. Kuzmanovic and E. Knightly, "Low-rate TCP-targeted denial of service attacks," in *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, August 2003.

[26] X. Luo, R. K. C. Chang, "On a New Class of Pulsing Denial-of-Service Attacks and the Defense," in *Proc. Network and Distributed System Security Symposium (NDSS 2005)*, San Diego, California, February. 2005.

[27] S. Ebrahimi, A. Helmy, S. Gupta, "A Systematic Simulation-based Study of Adverse Impact of Short-lived TCP Flows on Long-lived TCP Flows", in *IEEE INFOCOM 2005*, Miami, FL, March 2005.

[28] J. Bellardo and S. Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions", in *Proc. USENIX Security Symposium*, Washington D.C., August 2003.

[29] P. Kyasanur and N. Vaidya, "Detection and Handling of MAC Layer Misbehavior in Wireless Networks", in *Proc. Dependable Systems and Networks (DSN 2003)*, San Francisco, CA, June 2003.

[30] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of Service Attacks at the MAC Layer in Wireless Ad Hoc Networks," in *Proc. of MILCOM 2002*, Anaheim, CA, October 2002.

[31] P. Ferguson and D. Senie, "Network ingress filtering: Defeating denial-of-service attacks which employ IP source address spoofing," RFC 2827, May 2000.

[32] K. Park and H. Lee, "On the Effectiveness of Route-Based Packet filtering for Distributed DoS Attack Prevention in Power-Law Internets," in *Proc. ACM SIGCOMM 2001*, San Diego, CA, August 2001.

[33] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang., "SAVE: Source address validity enforcement protocol," in *Proc. of IEEE INFOCOMM 2002*, New York, June 2002.

[34] A. Bremler-Barr and H. Levy, "Spoofing Prevention Method", in *Proc. IEEE INFOCOM 2005*, Miami, Fl, March 2005.

[35] J. Ioannidis and S. M. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks", in *Proc. Network and Distributed System Security Symposium*, San Diego, CA, February 2002.

[36] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High- Bandwidth Aggregates in the Network (Extended Version)", http://www.icir.org/pushback/, July 2001. (date retrived: 10 March 2005).

[37] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling High-Bandwidth Flows at the Congested Router", in *Proc. IEEE ICNP 2001*, Riverside, CA, November 2001.

[38] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High Bandwidth Aggregates in the Network", in *Proc. ACM SIGCOMM Computer Communications Review*, vol. 32, no. 3, pp. 62-73, July 2002.

[39] T. Peng, C. Leckie and K. Ramamohanarao, "Protection from Distributed Denial of Service Attack Using History-based IP Filtering," in *Proc. of IEEE ICC 2003*, Anchorage, AK, May 2003.

[40] Y. Kim, W. Lau, M. Chuah, J. Chao, "PacketScore: A statistical-based overload control against DDoS attacks", in *Proc. IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[41] M. Chuah, W. Lau, Y. Kim, J. Chao, "Transient Performance of PacketScore for Blocking DDOS attacks", in *Proc. IEEE ICC 2004*, Paris, France, June 2004.

[42] A. Yarr, A. Perrig and D. Song, "Pi: A path Identification Mechanism to Defend against DDoS Attacks," in *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 2003.

[43] M. Sung and J. Xu, "IP Traceback-based Intelligent Packet Filtering: A Novel Technique for Defending Against Internet DDoS Attacks," in *Proc. IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 9, pp. 861-872, Septemper 2003.

[44] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An Architecture for Mitigating DDoS Attacks," in *IEEE Journal on Selected Areas in Communications (JSAC), special issue on Recent Advances in Service Overlay Networks*, vol. 22, no. 1, pp. 176-188, January 2004.

[45] D. Xuan, S. Chellappan, X. Wang, and S. Wang, "Analyzing the Secure Overlay Services Architecture under Intelligent DDoS Attacks," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, Tokyo, Japan, March 2004.

[46] J. Mirkovic, G. Prier and P. Reiher, "Attacking DDoS at the Source," in *Proc. IEEE ICNP 2002*, Paris, France, November 2002.

[47] J. Mirkovic, "D-WARD: Source-End Defense Against Distributed Denial-of-Service Attacks," Ph.D. Thesis, UCLA, August 2003.

[48] G. Yang, M. Gerla, and M. Y. Sanadidi, "Defense against Low-rate TCP-targeted Denial-of-Service Attacks," in *Proc. ISCC 2004*, Alexandria, Egypt, June 2004.

[49] D. K. Yau, John C. S. Lui, and F. Liang, "Defending Against Distributed Denial of Service Attacks with Max-min Fair Server-centric Router Throttles," in *Proc. IEEE IWQoS 2002*, Miami Beach, FL, May 2002.

[50] I. Stoica and H. Zhang, "Providing guaranteed services without per flow manaegment," in *Proc. ACM SIGCOMM 1999*, Cambridge, MA, August 1999.

[51] S. Savage, D. Wetherall, A. Karlin and T. Anderson, "Practical network support for IP traceback," in *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, August 2000.

[52] K. Park and H. Lee, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, April 2001.

[53] D. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," in *Proc. IEEE INFOCOMM 2001*, Anchorage, AK, April 2001.

[54] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback," in *Proc. Network and Distributed System Security Symposium (NDSS 2001)*, San Diego, CA, February 2001.

[55] M. T. Goodrich, "Efficient Packet Marking for Large-Scale IP Traceback," in *Proc. ACM CCS 2002*, Washington, DC, November 2002.

[56] A. Belenky and N. Ansari IP, "Traceback With Deterministic Packet Marking,"in *Proc. IEEE COMMUNICATIONS LETTERS*, vol. 7, no. 4, pp. 162-164, April 2003.

[57] A. Belenky and N. Ansari, "Accommodating fragmentation in deterministic packet marking for IP traceback", in *Proc. IEEE GLOBECOM 2003*, San Francisco, CA, December 2003.

[58] S. M. Bellovin, "ICMP traceback messages," Mar. 2000, Internet Draft: draft-bellovin-itrace-00.txt (expires Septemper 2000).

[59] A. Mankin, D. Massey, C.L Wu, S.F Wu, L. Zhang, "Intention-Driven ICMP Traceback," in *Proc. IEEE ICCCN 2001*, Scottsdale, AZ, October 2001.

[60] F. Hsu and T. Chiueh, "A Path Information Cashing and Aggregation Approach to Traffic Source Identification," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, Providence, RI, May 2003.

[61] T. Baba and S. Matsuda, "Tracing network attacks to their sources," in *Proc. IEEE Internet Computing*, vol. 6, no. 2, pp. 20-26, 2002.

[62] A. C. Snoeren, C. Partiridge, L. A. Sanchez, C. E. Jones, F. Tchhakountio, S. T. Kent, and W. T. Strayer, "Hash-Based IP TraceBack," in *Proc. ACM SIGCOMM 2001*, San Diego, CA, August 2001.

[63] L. A. Sanchez, W. C. Milliken, A. C. Snoeren, F. Tchakountio, C. E. Jones, S. T. Kent, C. Partridge, and W. Timothy Strayer, "Hardware Support for a Hash-Based IP Traceback," in *Proc. The 2nd DARPA Information Survivability Conference and Exposition (DISCEX-II)*, Anaheim, CA, June 2001.

[64] R. Stone, "Centertrack: An IP overlay network for tracking DoS floods," in *Proc. The* 9_{th} *USENIX Security Symposium*, Denver, CO, August 2000.

[65] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," in *Proc. 2000 USENIX LISA Conference*, New Orleans, LA, December 2000.

[66] H. Aljifri, M. Smets, A. P. Pons,"IP Traceback using header compression," in *Proc. Computers and Security*, vol. 22, no. 2, pp. 136-151, 2003.

[67] H. Aljifri, "IP Traceback: A New Denial-of-Service Deterrent?" in *Proc. IEEE Security and Privacy*, vol. 1, no. 3, pp. 24-31, May-June 2003.

[68] J. Li, M. Sung, J. Xu, L. Li, and Q. Zhao, "Large-Scale IP Traceback in High-Speed Internet: Practical Techniques and Theoretical Foundation," in *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004.

[69] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," in *Proc. Communications of ACM*, vol. 13, no. 7, pp. 422-426, July 1970.

[70] J. Postel, "Character Generotor Protocol," RFC 864, May 1983.

[71] S. Chen and Q. Song, "Perimeter-Based Defense against High Bandwidth DDoS Attacks," to appear in *Proc. IEEE Transactions on Parallel and Distributed Systems (TPDS 2005).*

[72] Cisco systems, "Defeating DDoS Attacks,"
http://www.cisco.com/application/pdf/en/us/guest/products/
ps5888/c1244/cdccont_0900aecd8011e927.pdf.
(date retrived: 2 April 2005).

[73] S. Gibson,"Distributed Reflection Denial of Service," February 22_{nd}, 2002.
http://grc.com/dos/drdos.htm.
(date retrived: 16 May 2004).

[74] J. Postel, and J. Reynolds,"File Transfer Protocol (FTP)," RFC 959, October 1985.

[75] A. S. Tanenbaum, "Computer Networks, Fourth Edition," Prentice Hall, 2003.

[76] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, "SOCKS Protocol Version 5," RFC 1928, March 1996.

[77] M. Mellia, I. Stoica, and H. Zhang, "TCP Model for Short Lived Flows," in *Proc. IEEE Communication Letters*, Vol. 6, no. 2, pp. 85-87, February 2002.

[78] A. Hussain, J. Heidemann, and C. Papadopoulos,"A Framework for Classifying Denial of Service Attacks," in *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, August 2003.

[79] H. Wang, D. Zhang, K. G. Shin, "Detecting SYN Flooding Attacks,"in *Proc. IEEE INFOCOM 2002*, New York, June, 2002.

[80] Y. Kim, J.-Y. Jo, H. J. Chao, and F. L. Merat, "High-speed router filter for blocking TCP flooding under distributed denial of service attack," in *Proc. IEEE IPCCC 2003*, Phoenix, AZ, April 2003.

[81] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP Topologies with Rocketfuel," in *Proc. ACM SIGCOMM 2002*, Pittsburgh, PA, August 2002.

[82] S. Ross, "A First Course in Probability," Fourth Edition, Prentice Hall, 1994.

[83] J. Aikat, J. Kaur, F. D. Smith, and K. Jeffay, "Variability in TCP Round-trip Times," in *Proc. ACM SIGCOMM Internet Measurement Conference (IMC 2003)*, Miami, FL, October 2003.

[84] B. Cheswick, H. Burch, and S. Branigan, "Mapping and Visualizing the Internet," in *Proc. USENIX Annual Technical Conference 2000*, San Diego, CA, June 2000.

[85] C.L. Schuba, I. V. Krusl, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni, "Analysis of denial of service attack on TCP," in *Proc. IEEE Symposium on Research in Security and Privacy*, Oakland, CA, May 1997.

[86] A. Zuquete, "Improving the functionality of SYN cookies," in *Proc. The 6th IFIP Communications and Multimedia Security Conference*, Portorosz, Slovenia, September 2002

[87] J. Lemon,"Resisting SYN flooding DoS attacks with a SYN cache," in *Proc. USENIX BSDCon 2002*, San Francisco, CA, February 2002.

[88] V. Jacobson, "Congestion Avoidance and Control," in *Proc. ACM SIGCOMM 1988*, Stanford, CA, August 1988.

[89] R. Braden, "Requirements for Internet Hosts- Communication Layers," RFC 1122, October 1989.

[90] H. Jamjoom and K. G. Shin, "Persistent Dropping: An Efficient Control of Traffic Aggregates," in *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, August 2003.

[91] R. Oliver,"Countering SYN Flood Denial-of-Service (DoS) Attacks," invited talk at *The* 10_{th} *USENIX Security Symposium,* Washington, D.C. August, 2001.

[92] Cisco systems, "Configuring TCP Intercept(Prevent Denial-of-Service Attacks)," http://www.cisco.com/univercd/cc/td/doc/product/software/ ios113ed/113ed_cr/secur_c/scprt3/scdenial.htm. (date retrived: 2 April 2005).

[93] H. Wang and K. G. Shin, "Transport-Aware IP Routers: A Built-In Protection Mechanism to Counter DDoS Attacks," in *Proc. IEEE Transactions on Paralle and Distributed Systems,* vol. 14, no. 9, pp. 873-884, Septemper 2003.

[94] J. Postel, "Internet Protocol - DARPA Internet Program Protocol Specification," RFC 791, September 1981.

[95] T. E. Daniels, "Reference Models for the Concealment and Observation of Origin Identity in Store-and-Forward Networks," Ph.D. Dissertation. Purdue University, West Lafayette, Indiana. 2002.

[96] M. Faloutsos and P. Faloutsos and C. Faloutsos, "On Power-Law Relationships of the Internet topology", in *Proc. ACM SIGCOMM 1999*, Cambridge, MA, August 1999.