HONEYPOT FRAMEWORK FOR NETWORKS UNDER DISTRIBUTED DENIAL OF SERVICE ATTACKS

A THESIS

Submitted in partial fulfilment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRONICS AND COMPUTER ENGINEERING

by

ANJALI SARDANA

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)

JULY, 2009

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"HONEYPOT FRAMEWORK FOR NETWORKS UNDER DISTRIBUTED DENIAL OF SERVICE ATTACKS"** in partial fulfilment of the requirements for the award of the Degree of Doctor of Philosophy and submitted in the Department of Electronics and Computer Engineering of the Indian Institute of Technology Roorkee, Roorkee is an authentic record of my own work carried out during a period from July 2006 to July 2009 under the supervision of **Dr. R. C. Joshi**, Professor, Department of Electronics and Computer Engineering of Indian Institute of Technology Roorkee, Roorkee.

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

Anjali Sardana

(ANJALI SARDANA)

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

(R. C. Joshi)
Supervisor

Date: 20.7.09

The Ph.D. Viva-Voce examination of **Ms. Anjali Sardana**, Research Scholar, has been held on 17.12.09....

Signature of Supervisor

Signature of External Examiner

Abstract

Internet was designed for openness, functionality and with the aim of sharing resources. Security was not the prime concern during the initial phase of the design. Phenomenal growth of the Internet during the last two decades has converted the Internet into a public connected network used for daily important communications such as stock trades, financial management, banking, medicine, education etc. However, lack of built in security has caused the Internet to be vulnerable to intrusions and has facilitated break-ins of a variety of types, leading to heavy financial losses, delays, customer dissatisfaction etc. Over the past few years, denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks have become a costly threat. These attacks are critical, as they are aimed at denying or degrading service for a legitimate user. Defending the DDoS attacks involves three phases: before the attack, during the attack and after the attack. Defense corresponding to the first phase is prevention; second phase is detection and characterization. Lastly, defense after the attack makes use of mitigation techniques. In the last decade there have been several attempts to defend against DDoS attacks using one or more of the above approaches. Though an array of schemes have been proposed, most of them are point technologies or perimeter solutions which are unable to withstand the advancing attack techniques. There is a dire need of complete framework to defend against various phases of DDoS attacks. This is presented in detail through citing key works in the first part of the thesis.

In the second part of the thesis, we propose a honeypot based framework that provides integrated defense during various phases of a DDoS attack. Our framework is proactive and works on three lines of defense, namely, detection, characterization and response, and mitigation. The work presented in this thesis proposes new and efficient techniques for each of the lines of defense. We propose the use of honeypot that appears to be part of a network but which is actually isolated, (un)protected, and monitored, and which appears to contain information or a resource that would be of value to attackers. Our framework does not replace the existing technologies like firewalls and IDS but is used in conjuction with them to defend the attacks.

We model Internet as transit-stub network. Our aim to defend the DDoS attack is to prevent the attack flow reach the target to ensure its availability. The traffic is analyzed on the edge router of transit domain before entering the network. The detection thresholds are

optimized and the detector is calibrated according to network load and client requirements. If the attack is detected, the flows corresponding to attackers are identified. At macroscopic-level, the flows that maximally contribute to the DDoS attacks are identified and dropped before they enter the network. Rest of the flows undergoes microscopic detection and characterization. Legitimate flows are routed to active servers. The anomalous and suspicious flows are tagged as attacks, directed to honeypots and further monitored before any action is taken on them. Number and location of honeypots and servers is varied dynamically depending on the network load and client requirements. Hence our scheme encompasses various aforementioned phases of defense and is tuned to operate in one of the three modes of defense, namely naïve, normal or best, depending on the client load, attack load and client requirements.

In the third part of the thesis, we introduce novel dual - level attack detection (D-LAD) scheme which is the first line of defense in the proposed framework. The first level attempts to detect congestion inducing macroscopic attacks which cause apparent slowdown in network functionality. Using the Macroscopic – Level Attack Detectors (Ma-LAD), macroscopic or large volumes of attacks are detected early at border routers in transit network before they converge at the victim. On the other hand, sophisticated attacks that cause networks to degrade gracefully, and stealth attacks that may not necessarily impact the network and remain undetected in transit domain but have dramatic impact on server are detected at second level by Microscopic – Level Attack Detectors (Mi-LAD) at border routers in stub domain near the victim. The techniques for attack detection are based on entropy which measures traffic feature distribution and utilize cumulative sum and change point detection computed in moving time window. Honeypots help achieve high detection rate and filtering accuracy.

Our proposed scheme is a hybrid that combines anomaly detection and honeypots in a way that exploits the best features of these mechanisms while shielding their limitations. The compromise between detection accuracy and time of confirming is a critical aspect and the proposed technique provides the quite demanded optimal solution to this problem. Our scheme is simple to understand and implement. Results demonstrate that in addition to being competitive than other techniques, our scheme is very effective and works well in the presence of different DDoS attacks. It is capable of handling infiltrating, sophisticated, meek as well as highly distributed DDoS attacks. Besides being computationally fast and accurate, it adapts to varying network conditions with minimum collateral damage and false alarms.

The fourth part of the thesis introduces the problem of characterization of traffic. Attack characterization forms second line of defense in the proposed framework. Our techniques for attack characterization are based on examining traffic feature distributions which identify the attacks in systematic manner. Our detection and characterization overlap, since the method used to detect the existence of an attack provides necessary information to characterize the traffic. Characterization is extended to take an immediate response decision. The response system either drops the attack traffic in a timely fashion or renders them harmless by redirecting them into a trap for further evaluation and analysis.

Similar to detection, characterization is performed at two levels. In case of macroscopic level characterization, most of the macroscopic attack traffic is identified early at the border router of transit network. Response mechanism at this level selectively drops the congestion inducing attack traffic. The microscopic level attack characterization is triggered at border router of stub network by Mi-LAD. The response mechanism then redirects the suspicious traffic of anomalous flows to honeypot trap for further evaluation. Legitimate traffic is directed to servers.

Our response mechanism works by implementing three rules, namely allow rule, redirect rule and drop rule to implement the above functionality. Hence, our response mechanism selectively drops the attack packets and minimizes collateral damage in addressing the DDoS problem.

In the fifth part of the thesis, we propose a proactive 'autonomous dynamic' honeypot redirection approach for attack mitigation which forms the third line of defense of the proposed framework. In our approach to attack mitigation, the total budget (total number of machines) gets partitioned into two groups, active servers and honeypots. The traffic is handled through honeypots or active servers contingent on the input derived from characterization at microscopic level. The good traffic is routed to one of the active servers, while the attack is diffused across honeypots. By 'dynamic' we mean that the number of active servers and honeypots is adaptive and changing and by 'autonomous' we mean change in number and locations of active servers and honeypots is triggered independently with changing network conditions. Legitimate clients, depending upon their trust levels (defined by organization according to its security policy configurations), can track the actual servers for certain time period. Anomalous flows reaching honeypots are logged by honeypot. Our mitigation techniques use light weight backward hash chains for tracking the location, number and duration of active servers and honeypots.

Our mitigation technique has an edge over existing techniques as it provides service continuity to legitimate clients with guaranteed Quality of Service (QoS) in addition to stable network functionality under dynamically changing network conditions even for attacked network.

Our work also includes analytical modeling and extensive simulations in ns-2 carried out over AT&T topology generated by GT-ITM topology generator with realistic network parameters. Various attack scenarios have been synthetically generated specially for testing the suggested techniques. Implementation of basic proof-of-concept, cost benefit analysis and exhaustive analysis of network performance parameters like goodput, mean time between failure and average response time have been performed to evaluate the various techniques and demonstrate feasability of the proposed framework. The simulation results are very promising and show that the proposed framework is robust, resilient and can withstand high levels of DDoS attacks.

Finally the contributions made in the thesis are summarized and scope for the future work is outlined.

Acknowledgements

I wish to express my heartfelt gratitude for the encouragement, broad knowledge and the expert guidance that Prof. R. C. Joshi has given me throughout my work on this thesis at the Indian Institute of Technology Roorkee, Roorkee, India. I consider myself very fortunate for having been associated with him. It is his vision and insight that inspired me to undertake research in this interesting field of information security. He has been very generous in providing me the necessary resources to carry out my research. Prof. Joshi's insistence on preparing and giving the best possible presentation, writing the best possible paper, and being the best possible student, definitely changed my way of thinking and enriched my growth as a researcher. I have benefited immensely from his inspiration and advice. I fall short of words to express his excellent and able guidance.

The co-operation and help extended by the Head and faculty members, Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee is gratefully acknowledged. I specially thank my research committee members for their patience during discussions and evaluations.

I would like to thank several anonymous reviewers whose constructive comments and suggestions on the work from time to time since its initial stages helped me in improving the quality of the work and manuscript substantially.

The department of Electronics and Computer Engineering provided an excellent environment for my research in information security. I spent many enjoyable hours working with department members and fellow students. Without this friendly environment and freedom, many of my ideas would not have come to fruition. A special mention of Ms. Chanchal Gupta, Mr. Emmanuel S. Pilli and Ms. Lata Chauhan who took the arduous step of proof reading the drafts of the thesis. I record here a note of thanks to Mrs. Poonam Choudhary and Mr. T. P. Sharma for sharing their ideas and providing light moments, especially while writing this thesis. I acknowledge my senior Dr. Krishan Saluja's help for initial rigorous discussions in the field. Sincere efforts of Mr. Raj for providing assistance with the use of laboratory facilities are acknowledged.

I would also like to thank all the friends, specially, Mrs. Shilpa Pal, Mrs. Prerana Gupta, Mrs. Deepti Thakur, Mrs. Dhanpati Rana, Ms. Lata Chauhan, Mrs. Nidhi Singh, Ms. Priya Kapur, Mrs. Kushuma Negi, Mrs. Sreela Dey, Mrs. Avlokita Agrawal and Mrs.

Pallavi Dwivedi. They provided the joyful, affectionate and healthy environment during my stay at Rooorkee.

On a personal note, I owe everything to the Almighty and my parents. I feel deep sense of gratitude for my mummy and papa for their unending love, inseparable support and prayers during this research experience. Without the sacrifices from my parents, I could never have the opportunity to undertake this study. They always encouraged me in all my endeavors and felt proud of every achievement of mine. I would like to extend my sincere thanks to my grandparents for their affection and wishes. I would like to thank my brother for his constant care and concern that made this journey so congenial and easy.

I feel grateful for the support I have received from the Indian Institute of Technology, Roorkee through Ministry of Human Resource Development (MHRD) scholarship. I am thankful and acknowledge the help extended by the Council of Scientific and Industrial Research (CSIR), Indian National Science Academy (INSA), Ministry of Communications and Information technology (MIT) and Microsoft for sponsoring my conference participations.

God understands our prayers even when we cannot find words to say them. I am indebted to the Almighty for making my life meaningful even outside the confines of this Ph.D candidature. I thank God for everything I am blessed with.

Anjali Sardana

Anjali Sardana

Contents

List of Abbreviations

AL	Attack Load
ART	Average Response Time
B	Best
BW	Bandwidth
CBR	Constant Bit Rate
CERT	Computer Emergency Response Team
CL	Client Load
CNN	Cable News Network
CSI	Computer Security Institute
CUSUM	Cumulative Sum
DDoS	Distributed Denial-of-Service
Destination IP	Destination IP address
DHE	Dynamic Honeypot Engine
D-LAD	Dual-Level Attack Detector
DoS	Denial-of-Service
FBI	Federal Bureau of Investigation
FL	Flow List
FN	False Negatives
FP	False Positives
FTP	File Transfer Protocol
GTITM	Georgia Tech Internetwork Topology Models
H	High
HC	Honeypot Controller
IAT	Inter Arrival Time
IDS	Intrusion Detection System
IP	Internet Protocol
ISP	Internet Service Provider
L	Low
M	Moderate
Ma-LAD	Macroscopic-Level Attack Detector
Mbps	Megabits per second

Mi-LAD	Microscopic-Level Attack Detector
miv	Migration interval
ms	millisecond
MTBF	Mean Time Between Failure
Na	Naive
Nam	Network Animator
N_H	Number of Honeypots
No	Normal
N_s	Number of Servers
Ns	Network simulator
QoS	Quality of Service
ROC	Receiver Operating Characteristic
s	Second
Source IP	Source IP address
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
UDP	User Datagram Protocol

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The Internet (originally known as ARPANET) was created in 1969 by the Department of Defense in the USA. The aim was to develop a means of survivable communication in case other modes of communication got disrupted during a nuclear attack [1]. The Internet has grown phenomenally during the last two decades and success of the Internet has changed the way traditional essential services such as banking, transportation, medicine, education, defense etc. are operated. They have been replaced by cheaper and more efficient network based services and applications. For example, governments use the Internet to provide information and services to the citizens, companies share and exchange information with their divisions, suppliers, partners and customers through the Internet, research and educational institutes depend on the Internet for collaboration and for disseminating their research discoveries rapidly.

In the present era, the Internet is used in daily routine for important communications such as stock trades, financial management, and customer services, delays in which can lead to big losses in terms of revenue and customer dissatisfaction. Enterprises have become highly dependant on networks for their functionality. As the Internet becomes an integral part in many people's lives [2], the need to keep networks and servers protected, available and secure has become increasingly important. Any threat to the network's security can lead to heavy financial and economic losses and cause other fatalities. So, protection of network against security threats is a crucial prerequisite.

When the TCP/IP protocols were designed in the 1980s, openness and growth of the network were design priorities. As the network was private and isolated, the security was not the main concern. This resulted in many vulnerabilities in the basic design of the Internet. As the Internet grew, these vulnerabilites continued to increase. As a result, today's Internet, which is a public and connected network with its backbone as TCP/IP, lacks even the most basic mechanisms of security. According to latest CERT statistics [3],

the number of reported vulnerabilities in the Internet has increased from 171 in 1995 to 8,064 in 2006, as shown in Figure 1.1. The number of vulnerabilities in the first two quarters of 2008 alone had reached to a figure of 6,058.

The inherent weaknesses and lack of built in security in the architecture of the Internet resulted in successful origin and execution of attacks to TCP/IP suite, such as Denial-of-Service (DoS), Distributed Denial-of-Service (DDoS), IP spoofing, sequence number hijacking, DNS attack etc. According to CERT [3] the number of reported Internet security incidents jumped from 6 in 1988 to 82,094 in 2002, and were 137,529 in 2003 as shown in Figure 1.2.

Internet failures can be accidental or intentional. The design of the Internet has been made to an extent of controlling accidental failures. But intentional attacks have no answer in the original Internet design. A DoS attack [4] is such an intentional attempt by attackers to completely disrupt or degrade the services to authorized users. As a proof of the disturbing trends in Figure 1.2, a FBI/CSI survey conducted in 2004 [5] concluded that DoS attacks were one of the major cause of the financial losses resulting from cyber crime. Therefore DoS attacks are real and very costly threat to the steady functioning of any network.

DoS attack is characterized by "an explicit attempt by an attacker to prevent legitimate users of a service from using that service" [6]. DoS attacks can be launched against a host (e.g. FTP server) or a network, (e.g., the network connection to a server). Along the path between a client and a server, network packets consume various resources like access link bandwidth, router buffers etc. DoS attacks inject maliciously-designed packets into the network. These numerous, useless packets can deplete the resources of server or network. They can consume memory, CPU and network resources and damage or completely disrupt the operation of the resource under attack.

DDoS attack is an extension of the DoS attack. A DDoS attack [7] is a coordinated DoS attack on the availability of services of a given target system or network that is launched indirectly through many compromised computing systems by sending stream of useless traffic meant to explode host or network resources.

The timeline of most significant DoS attack incidents in real life is shown in Figure 1.3. The timeline shows that DoS attacks have been known since the early 1980's. Although

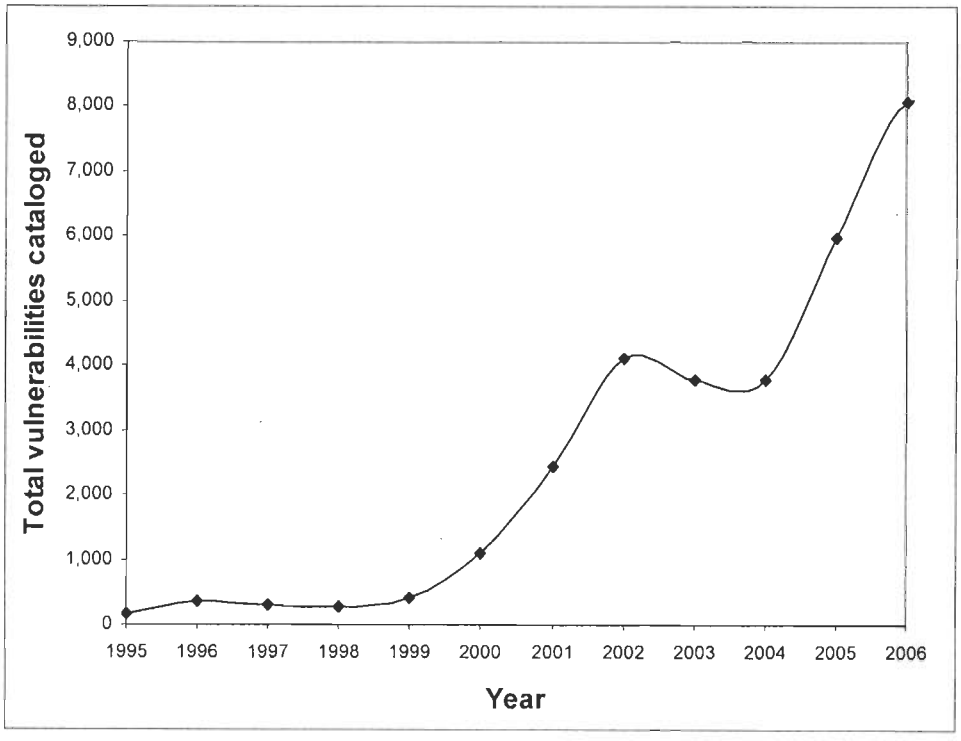


Figure 1.1. The number of total vulnerabilities catalogued from 1995 to 2006

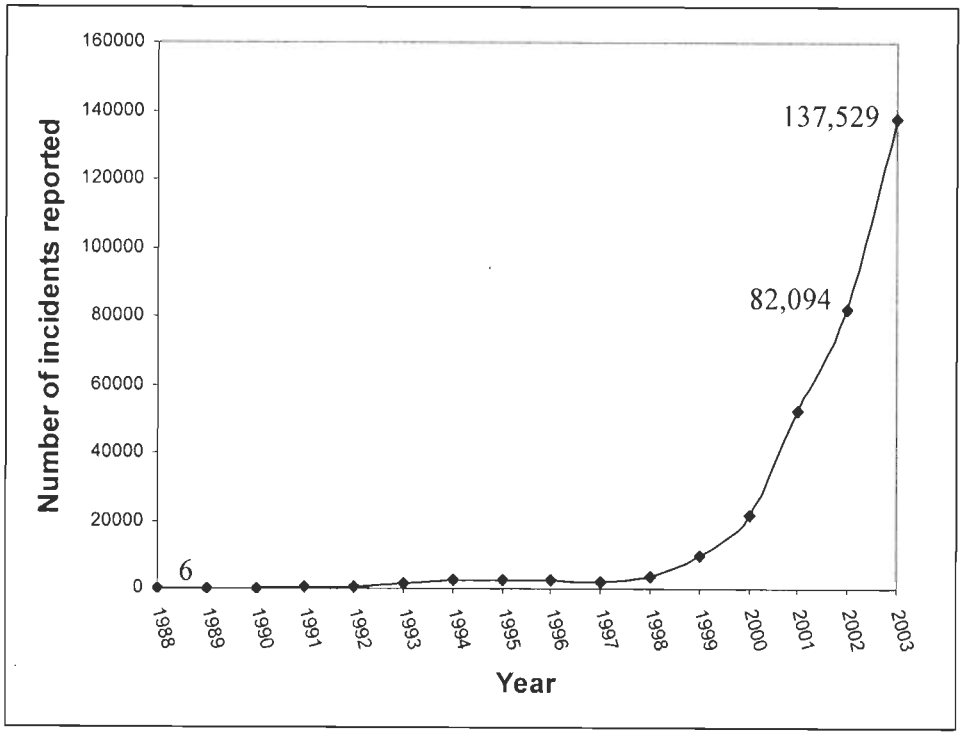


Figure 1.2. The number of Internet security incidents reported from 1988 to 2003

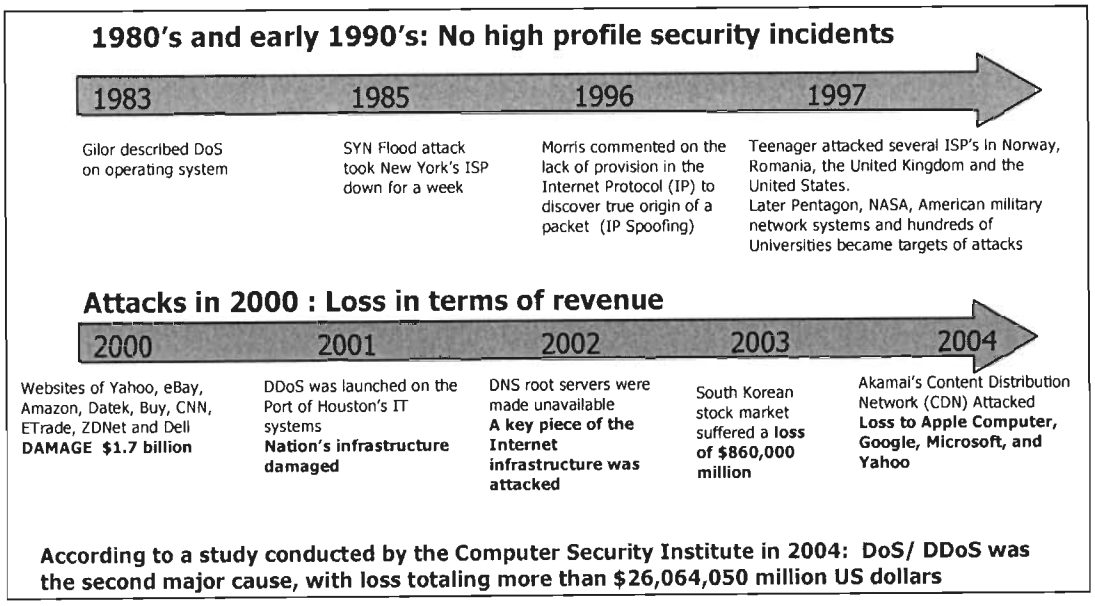


Figure 1.3. Timeline of DoS attack incidents

DoS attacks existed during the 1980s and early 1990s, at that time they were not viewed publicly as being high-profile security incidents. They became costly and crucial only when the Internet became a mainstream medium due to the phenomenal growth in late 1990's.

Gilor in 1983 provided the first description of DoS in operating systems [8]. Morris in 1985 commented on the lack of provision in the Internet Protocol (IP) to discover true origin of a packet [9]. The attacker routinely exploited this weakness by faking their source address, which a decade later came to be known as "IP Spoofing".

On 6th September 1996, SYN Flood [10] attack took Panix, New York City Internet service provider offline for a whole week. "Realsecure" [11] by IBM was one of the first products to halt the attacks which failed in December 1996 when attack on Webcom's server knocked thousands of commercial websites offline [12]. In January 1997, a teenager attacked an Internet Relay Chat (IRC) network [13] and several ISPs in Norway, Romania, the United Kingdom and the United States [14] with combination of ping attacks [15] and SYN floods [10]. In January 1998, DALnet and other IRC networks became targets of smurfing [16, 17]. During the same period, an 18-year-old targeted the Pentagon, NASA, American military network systems and hundreds of universities using "Teardrop" and "Bonk" [18].

Frequency of the attacks increased from the year 2000 due to sudden popularity of the Internet. The attacks started causing losses in terms of revenue. In February 2000, websites of Yahoo, eBay, Amazon, Datek, Buy, CNN, ETrade, ZDNet and Dell were attacked by a 15-year-old causing a total damage of $1.7 billions [19]. In July 2000, BT.com, BTInternet.com and Gameplay.com were attacked by a customer as a revenge for bad service. Register.com in January 2001 [20] and GRC.com in 2002 became victims of Distributed Reflector DoS attacks [21, 22]. "Code Red" worm in 2001 infected 2,50,000 computers in nine hours and programmed them to simultaneously attack the website of White House causing a damage of about $2.6 billion [23]. For the first time, part of a country's national infrastructure was disabled when DDoS was launched on the Port of Houston's IT systems [24], which contained data on navigation, tides, water depths and weather. In June 2002, the website of the government of Pakistan was victim of politically motivated attack launched by "YAHA" [25]. In October 2002, a key piece of the Internet infrastructure was attacked when DNS root servers were made unavailable by DDoS for about an hour [26]. In December 2002, the "King of Spam", Alan Ralsky became victim of DoS attack when several Internet users subscribed him to thousands of catalogs and mailing lists [27]. A few months later, Byers *et al.* [28] published how to automate such attacks using scripts. In January 2003, the worm "SQL Slammer" [29] hit South Korea having world's highest penetration of broadband Internet services at that time, causing loses of about US$860,000 to South Korean stock market [30]. A few months later, same worm caused a 5-hour outage to the safety monitoring system of the nuclear power plant in Ohio [31]. In March 2003, the Arabic and English-language websites of the satellite television network Al-Jazeera suffered two days of DoS attacks [32]. Another major DoS attack on June 15, 2004 [33], against name servers in Akamai's Content Distribution Network (CDN) blocked nearly all access to many sites for more than two hours. The affected sites included Apple Computer, Google, Microsoft, and Yahoo. These companies have outsourced their DNS service to Akamai to enhance service performance. In August 2004, a corporate executive in Massachusetts was charged with hiring hackers to launch DoS attacks using SYN and HTTP floods causing a total of $2 billion losses [34].

Since 2004, DoS incidents have deliberately not been widely publicized, as they have shifted to the sensitive field of economic crime, and harm the victim's reputation. DoS-extortion has become common, in which the victims are threatened to be put offline until they pay, with typical targets being credit card processing companies. Since the Internet downtime is very damaging in terms of finances, and the reputation of online companies is

very important, most victims choose to pay. In January 2006, the www.milliondollarhomepage.com, a British teenager's novel advertising idea to earn $1 million in four months, became famous very quickly and drew instant attention of cyber-extortionists, who bombarded the website with intense DoS. They initially asked $5,000 to avert them and finally $50,000 to stop them [35]. The text-to-speech translation application running in the Sun Microsystem's Grid Computing system was disabled on the very day it was launched with a DoS attack in March, 2006 [36]. The very latest DDoS incident that occurred on 29 May, 2009 turned down Internet in various parts of China [37], putting few million Chinese Internet users into trouble. Thus, the timeline shows that over the time, DoS attacks have become more frequent, sophisticated and effective in obstructing the availability of services and causing significant damages.

The DDoS attacks have become attractive for attackers because of two reasons. The first reason is that there are effective automatic tools [38] available for attacking a victim, hence launching a DDoS attack does not require any expertise. As can be seen from examples, any unsophisticated user can easily locate and download DDoS tools and perform successful, large-scale flood attacks. The second reason is that it is usually impossible to locate an attacker without extensive human interaction or without new features in most routers of the Internet [39]. DDoS attacks make use of vulnerabilities in end-hosts, routers and other systems connected to a computer network. As the vulnerabilities continue to increase (refer to Figure 1.1), installing malicious software on these hosts becomes easy which sends DDoS attack traffic.

Design of the Internet has raised several issues that are concerned with the opportunities for DDoS attacks. Firstly, Internet security is highly interdependent. Even if the victim is secured, its susceptibility to DDoS attacks depends on the state of security in the rest of the global Internet. DDoS attacks are commonly launched from systems that are subverted through security-related compromises. Secondly, resources of the Internet are limited and can be consumed by too many users. Thirdly, intelligence and resources are not located at the same place. Intelligence is needed for service guarantees. This requires minimizing processing time in intermediate network so that packets can be forwarded quickly. Therefore intelligence is stored in end hosts. At the same time, a desire for large throughput led to the design of high bandwidth links in the intermediate network. Thus, attackers misuse the abundant unintelligent resources of the intermediate network for delivery of numerous messages the victim. Fourthly, accountability is not enforced. IP spoofing gives attackers a powerful mechanism to launch attacks and escape easily.

Lastly, control is distributed and each network is run according to local policies defined by its owners. There is no way to enforce global deployment of a particular security mechanism or security policy, and due to privacy concerns, it is often impossible to investigate cross-network traffic behavior.

Given the ease and sophistication of attacks and the present state of the Internet, dealing with DDoS attacks has become one of the thorniest problems. Initial ideas to defend against the attacks were aimed to halt the escalation of the attacks. Hence, traditionally, the defense against DDoS attacks was based on restricting the access to authorized users using Access Control Lists (ACLs) [40] at the routers. However, the administrative time spent in determining the rules and updating ACLs and lack of processing power during the attack rendered such techniques unsuccessful. Since the attacks aim at overwhelming the target resource altogether, victim alone cannot employ defense on its own. Moreover, due to increasing vulnerable hosts, limited security in older networking protocols, increased sophistication of attack tools and attacks shifting towards economic crime, the researchers were motivated to tackle DDoS attacks on the basis of various phases of attack process.

Defending against the DDoS attacks involves three phases [41-43]: (i) before the attack, (ii) during the attack and (iii) after the attack. The general goal of DDoS defense in first phase is prevention [41-64], in the second phase is detection [65-80] and characterization [45, 46, 51-55, 65-70, 72, 75, 77-110], and finally, the last phase of defense after the attacks involves response [51, 52, 54, 55, 75-80, 111-119] and mitigation [39, 51, 52, 55, 75-80, 87, 88, 98-100, 106, 108, 111, 113-117, 120-145][7, 146-171].

Prevention aims stop the attacks before they are actually launched. It requires methods that can block the unwanted traffic. The techniques include installing filters at Internet Service Providers (ISPs) [51, 75], fixing security holes at end hosts and updating latest security patches [55], installing point or perimeter solutions like firewalls [41, 42] and Intrusion Detection System (IDS) [43, 45-47], disabling unused and non-essential network services, overprovisioning [57] to improve capacity and resiliency etc. This approach aims to improve global security level and is the best solution to DDoS attacks in theory. However, the disadvantage is that it needs global cooperation which is extremely difficult to achieve in reality. Most of these techniques are very expensive and greatly limit the functionality of a network node, often in ways that are incompatible with the Internet's mission. Because of inevitable software vulnerabilities, with the appearance of a number

of new attacking techniques and with the new attacks originating, it became practically impossible to halt the attacks.

Detection techniques developed so far are either signature based [45, 46] that look for specific attack signatures or anomaly based [65-68, 71] that build normal network behavior and watch for any divergence from the normal profile. The normal traffic behavior models are mainly based on flow rates and volume of traffic. They detect attack near the victim, when attack has already consumed significant resources on its path. Moreover, due to the diversity of user behavior, it is difficult to obtain a general and robust model for describing the normal traffic behavior. As a result, all the techniques suffer from false positives (FP) i.e. legitimate traffic classified as attack traffic and false negatives (FN) i.e. attack traffic classified as legitimate traffic. The challenge is to detect every attack at the earliest without misclassifying any legitimate traffic.

If an attack is detected, the network must be able to identify the attack traffic so that appropriate response decision can be taken. Similar to detection, characterization techniques can be broadly divided into signature based techniques [45, 46] and anomaly based techniques [65-70, 72, 78, 81-85, 88, 103, 108]. Characterization requires high level of accuracy. Most of the work in anomaly based techniques involves the use of volume based metrics which do not provide sufficient information to distinguish attacks. Network traffic is noisy, which makes it difficult to extract meaningful information about attacks from any kind of traffic characteristics.

Responsive techniques [51, 52, 54, 55, 75, 78, 80, 111, 113-117] are based on filtering or rate limiting the traffic. In most of the techniques, filtering and rate limiting is performed based on attack detection and attack traffic identification using the above characterization techniques. The success of response depends on accuracy of characterization. It is very difficult to reliably recognize and filter only attack traffic without causing any collateral damage.

Mitigation techniques aim to minimize the effect of ongoing DDoS attacks. These techniques include link based defense techniques like filtering [51, 52, 75, 87, 88, 116, 126, 128], IP Traceback [76, 133-141, 143, 146, 172], pushback [80, 153], link testing [111, 113, 144], special routing [116, 117, 140, 156, 173], Centertrack [111], aggregate based schemes [78, 79, 154, 155], and node based defense techniques [55, 57, 98-100, 108, 120-123, 127] and queue management techniques [7, 77, 161-165]. Aim of all these techniques has been limited to tolerate the attacks, avoid network and service failure and

provide network survivability. Most of these techniques require cooperation among ISPs and global deployment which is difficult to attain. Even in presence of mitigation techniques, legitimate users suffer large delays in service response times due to unstable network functionality during attack.

Most of the work done in this area is an effective stopgap measure, but does not eliminate nor deter the attackers. Despite methods that are in existence today, the threat of an attack still lingers and future attacks will likely be more powerful and the aftermath could be considerably worse. The main difficulty for defending these attacks lies in keeping up with the pace of the attackers and new attacking techniques. It requires enough knowledge about new and unknown attacks to remain a step ahead of attackers. Gathering this kind of information about attacks is not easy but important. If the attack strategy is known, countermeasures can be taken and vulnerabilities can be fixed. To gather as much information as possible is one main goal of a honeypot [174-176]. Generally, such information gathering should be done silently, without alarming an attacker. All the gathered information leads to an advantage on the defending side and can therefore be used to prevent attacks. Moreover, honeypot provides a controlled environment where the attack traffic can be redirected to isolate the actual resources from effects of attack.

Honeypot [174-176] is an important security technology used to understand and defend DoS attacks. It gets its name from the age old saying, "You can catch more flies with honey rather than vinegar". It is primarily an instrument for information gathering and learning. It lures the attackers away from the real system, and closely monitors the attacks. Honeypots can prevent attacks by deception, isolate attack traffic as well as gather information about attackers, which is synonymous to steps for defense against DDoS. Coupled with an Intrusion Detection System (IDS), honeypots are effective in detecting victim hosts [176].

Recent events indicate that DoS and DDoS attacks will be persistent, posing a severe threat to the stability of the Internet and could undermine the usability of the Internet. Researchers are still struggling to devise an effective solution to the DDoS problem. Although many commercial and research defenses have appeared, none of them provide comprehensive solution. Rather, they detect a small range of attacks that either use malformed packets or create severe disturbances in the network; response and mitigation techniques handle the attacks by non-selectively dropping a portion of the traffic destined for the victim. Clearly this strategy relieves the victim from the high-volume attack, but also inflicts damage to legitimate traffic that is erroneously dropped. Ultimately network

should respond when under attack: identifying attack traffic, implementing appropriate filters, isolating attack traffic, mitigating attacks and performing the follow-up investigation.

As the existing methods alone are insufficient to handle DDoS attacks, there exists a need to propose a comprehensive solution with simple and effective techniques for DDoS attack detection, traffic characterization, response and mitigation which work in conjunction with each other and proactively defend the DDoS attacks.

1.2 Statement of the Problem

The main objective of the present research work is as follows:

"To formulate a honeypot framework that encompasses the activities of attack detection, characterization, response and mitigation for defense against DDoS attacks."

The availability of such a framework shall enable reasonable network performance for DDoS attacked networks which has not been feasible till now. It requires research based solutions for issues involved in building practical and effective mechanisms to detect, characterize, respond to and mitigate DDoS attacks. These defense mechanisms should coordinate with each other to detect and characterize attacks quickly and accurately. They should ensure reasonable performance for networks or systems under attacks taking into account dynamically changing nature of the Internet.

We have several issues to contend while developing the framework to defend against DDoS attacks. These can be subdivided into smaller objectives as follows:

- To investigate the key existing techniques for detection of DDoS attacks.
- To explore and propose new and improved techniques for detection of DDoS attacks.
- To examine and suggest novel and better techniques for characterization of DDoS attacks.
- To propose and implement new and efficient techniques for responding to the attacks in an efficient way.
- To investigate and propose new techniques for mitigating DDoS attacks and for maintaining reasonable network performance.

- To validate the effectiveness of proposed techniques and hence, establish the validity of the framework by analytical modeling and by exhaustive simulations on realistic topology.

1.3 Organization of the Thesis

The rest of this thesis is organized as follows. Chapter 2 reviews the key techniques for DDoS attack prevention, detection, characterization, response and mitigation and brings out the research gaps.

Chapter 3 discusses the overall design of the honeypot framework proposed by us. It describes the various lines of defense used by the framework and details out the building blocks of the framework namely, detection, characterization, response and mitigation. It then discusses the features of the proposed framework.

Novel approaches for detection of DDoS attacks have been proposed in Chapter 4 which forms the first line of defense. The chapter details out the use of entropy as metric for measurement of traffic feature distribution and choice of traffic features. It presents the dual-level attack detection schemes, namely macroscopic-level and microscopic-level. Transit-stub model of Internet has been described followed by the sampling and detection mechanism at the two levels. It describes decision of optimum thresholds, various modes of defense, honeypot based suppression of false negatives at macroscopic-level and CUSUM algorithm for attack detection at microscopic-level. It also includes the simulation study on AT&T networks for the performance evaluation.

In Chapter 5, we propose new techniques for characterization and response which form the second line of defense. Characterization of DDoS attacks at dual-level is presented and discussed. Details of macroscopic-level and microscopic-level characterization techniques and technique for identification of victim are elaborated. The various response rules are discussed. Finally, the overall detection and characterization algorithms running at macroscopic-level and microscopic-level in transit stub network are presented. Simulation study on AT&T network has been presented to show the effectiveness of the techniques.

Chapter 6 proposes and discusses the technique for the DDoS attack mitigation which forms the third line of defense in the framework. Building blocks of the technique including service replication, dynamic roaming honeypots and service migration are presented. The chapter discusses the system model and design details of honeypot controller and dynamic honeypot engine. Various parametric dependencies are presented

followed by performance evaluation of mitigation techniques and overall framework under various modes of defense.

Finally, the summary of contributions of the present work and the scope for future work are discussed in Chapter 7.

Chapter 2

Literature Review

2.1 Introduction

During the past years, there have been several attempts to classify DoS attacks. Various proposals for defense against such attacks can be broadly classified into following approaches: (i) prevent the attacks, (ii) detect the attacks, (iii) identify the attack traffic, (iv) respond to the attacks and (v) mitigate the effects of the attacks. This chapter presents a survey of DoS attacks, including DDoS attacks and reviews the key proposals based on the above mentioned approaches for defending against the attacks. We discuss the strengths and weaknesses of each proposal and present countermeasures that an attacker may employ to defeat the protection provided by each proposal. We highlight research gaps in each of the approaches and present a background of honeypot technology.

2.2 Survey of DDoS Attacks

The DoS attack can be defined as any activity that aims to disable the services provided by the victim by sending an excessive volume of useless traffic. This is in contrast to the flash crowd which occurs when a large number of legitimate users access a server at the same time. It is necessary to be able to discriminate DoS attacks from flash crowds. The comparison between DoS attacks and flash crowds is shown in Table 2.1.

Table 2.1. Comparison between DoS attack and flash crowd [177]

Parameter	DoS Attack	Flash Crowd
Network impact	Congested	Congested
Server impact	Overloaded	Overloaded
Traffic	Illegitimate	Genuine
Response to traffic control	Unresponsive	Responsive
Traffic type	Any	Mostly web
Number of flows	Any	Large number of flows
Duration	Long	Small
Predictability	Unpredictable	Mostly predictable

A false positive (FP) is a normal operation that is misdiagnosed as an attack. A false negative (FN) is an attack that has not been identified by the defense scheme.

2.2.1 DDoS Attack: Modus Operandi

DDoS attacks consist of an overwhelming quantity of packets being sent from multiple attack sites to a victim site. These packets arrive in such a high quantity that some key resource at the victim like bandwidth, central processing unit time to compute responses etc. is quickly exhausted. The victim either crashes or spends so much time handling the attack traffic that it cannot attend to its actual work. Thus, legitimate clients are deprived of the victim's service as long as the attack lasts. A DDoS attack is composed of four elements namely the real attacker, the handlers or masters, the attack daemon agents or zombie hosts and the victim. The interaction between these elements [178] is shown in Figure 2.1.

- To launch the attack, an attacker first exploits the vulnerabilities of the installed network software on computers that are connected to the Internet.
- Using the vulnerabilities, the computers are converted into masters or handlers. These are compromised hosts with a special program running on them, capable of controlling multiple agents.

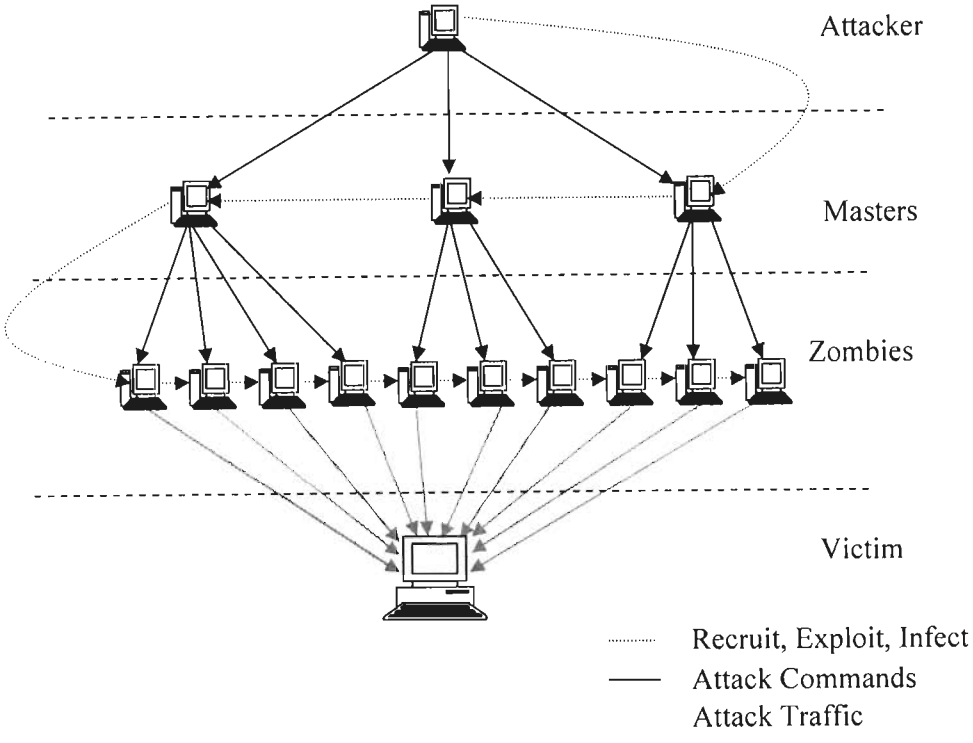


Figure 2.1. DDoS attack: Modus Operandi

- The masters compromise and infect machines by installing DDoS daemons and turn them into zombies. The zombies or attack daemon hosts are responsible for generating a stream of packets towards the intended victim.
- The attacker can issue commands, at any time, to all masters to trigger the zombies to simultaneously flood a very large number of attack packets to the victim. The attack traffic consumes resources of victim which may be network or target machine.

2.2.2 Classification of DoS Attacks

The scale of the damage caused by the attacks varies according to the type and volume of attack traffic. DoS attack defense systems perform differently against different types of DoS attacks. Hence, it is essential to categorize the main types of DoS attacks so that appropriate countermeasures for each type of DoS attack can be carefully developed [179].

2.2.2.1 Victim Type

DoS attacks consume both host and network resources. They are categorized according to victim type as shown in Figure 2.2, namely service, host, network and infrastructure attacks.

Each host normally provides multiple services, where each service is a separate application. A *service attack* aims to disable one particular service by exploiting an inherent vulnerability of that service. During a service attack, the destination port number of the attack packets is the same. It is difficult to detect this type of DoS attack. First, as

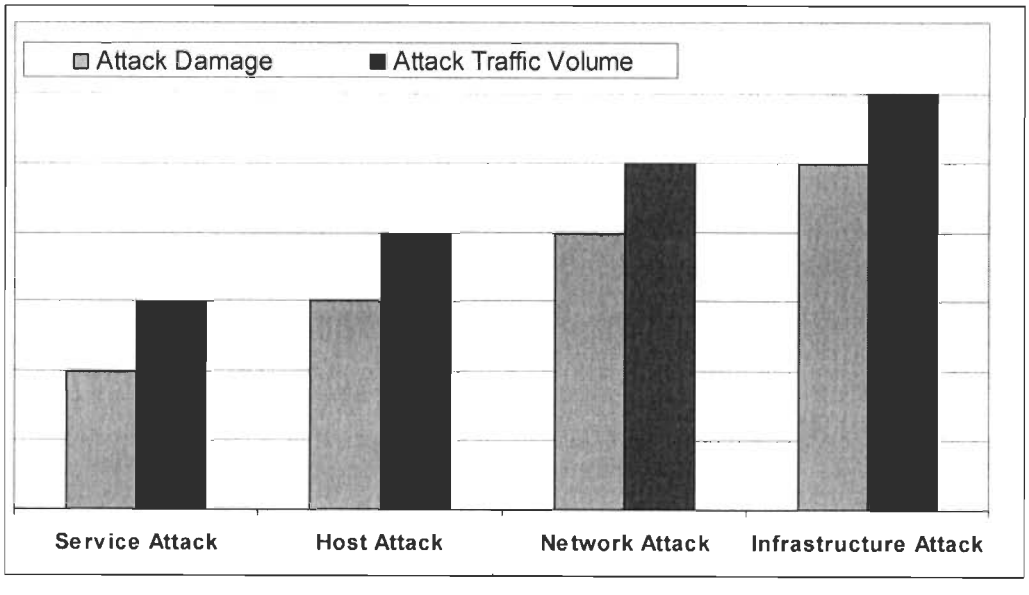


Figure 2.2. Categorization of DoS attacks according to victim type

the service attack aims to target one particular service that has a relatively small proportion of resources, its traffic volume is low. Second, as applications that are not under attack operate normally, the host is likely to be unaware of the attack. In order to detect the attack, one needs to configure a detection scheme for each service of the host. The *host attack* aims to disable the communication with the host. During a host attack, the destination IP addresses of the attack packets is the same. The host can detect this type of attack easily as the attack impact is significant. *Network attacks* aim to consume the bandwidth of the target's network. During a network attack, the destination IP addresses of the attack packets share the same network address. As the network attack consumes the network bandwidth, the attack traffic is generally high. To curtail this attack, the attack traffic should be filtered close to the source. The *infrastructure attack* aims to disable the services of critical components of the Internet. The result of an infrastructure attack is potentially catastrophic as the whole Internet may be affected. Significant power is required to launch a successful infrastructure attack. Global cooperation is essential for an effective defense against infrastructure attack.

2.2.2.2 The Parameters of Attack Power

The attack power consists of two parameters: traffic volume and the resources consumed per packet. DoS attacks can be categorized according to the values of these two parameters as shown in Figure 2.3.

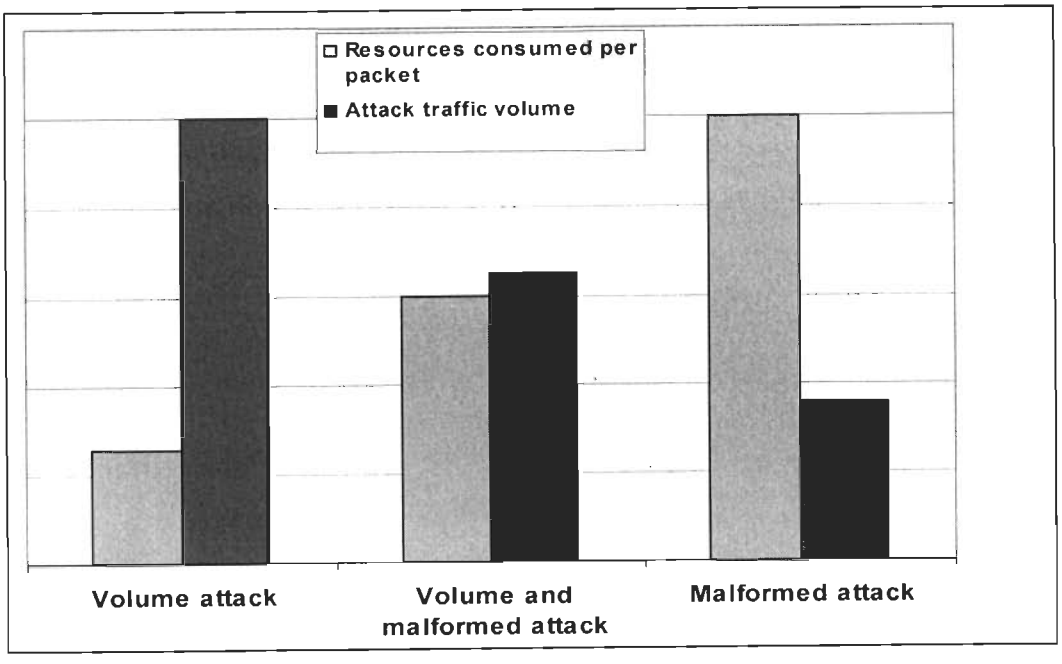


Figure 2.3. Categorization of DoS attacks according to parameters of attack power

The *volume attack* is based on a massive number of packets. Each attack packet consumes the same level of resources as a normal packet. The *malformed attack* sends a set of malformed packets, which consume more resources than normal packets. These malformed packets generally contain an invalid source address, fragmentation field, or protocol number. The traffic volume of a malformed attack need not necessarily be high to be effective. Generally, most operating systems are now resilient to malformed packets. As a result, attackers have to increase the volume of malformed packets to maintain the attack power. This type of attack is called the *volume and malformed attack*, which is based on both the traffic volume and resource consumed per malformed packet.

2.2.2.3 Average Flow Rate and Number of Flows

Given a constant attack traffic volume, there are two variable parameters: average flow rate and number of flows. According to different values of these two parameters, DoS attacks can be centralized or distributed. The *centralized or concentrated attack* contains a single flow with extremely high flow rate. The *distributed attack* contains a small number of flows with average flow rate that is higher than normal but comparatively lower than centralized attack. The highly distributed attack contains an extremely large number of flows with high, normal or low average flow rate.

2.2.2.4 Attack Traffic Rate Dynamics

During a DoS attack, an attack can choose the attack traffic rate dynamics. The result can be either a constant rate attack or a variable rate attack. During a *constant rate attack*, the attack sources send attack traffic to the target at a constant traffic rate, which causes continuous service disruption to the target. During a *variable rate attack*, the attack sources send attack traffic to the target with changing traffic rate.

2.2.2.5 Impact of Attack

The impact of an attack depends on attack power of the attacker in terms of the traffic volume and resources consumed per packet, and the resources available at the target. The result can be either a *malign attack*, which causes complete disruption, or a *benign attack*, which causes only partial disruption and graceful degradation. Since these attacks do not lead to total service disruption, they could remain undetected for a long time.

Macroscopic Attacks: They are congestion inducing attacks like the network and infrastructure attacks, volume attacks, concentrated high rate attacks, extremely distributed attacks and malign attacks that cause immediate damage and complete disruption.

Microscopic Attacks: They are benign attacks and distributed attacks that target host or service. They are crafted to match legitimate traffic statistics and constitute stealth, sophisticated or infiltrating attacks. They cause graceful degradation of the network and services.

2.2.3 Popular DoS Attacks

In this section, we discuss attacks that have been commonly observed in the majority of DoS incidents. The first category is protocol attacks that take advantage of the Internet protocols and include TCP SYN flooding attack, UDP flooding attack, ICMP flooding attack and DNS attack. The second category is DDoS attacks that amplify attack power using a large number of distributed attack sources and include typical DDoS attacks and distributed reflector denial-of-service (DRDoS) attacks.

2.2.3.1 TCP SYN Flooding Attack

At the beginning of each TCP connection, the client negotiates with the server to set up a connection, which is called 3-way handshake and is illustrated in Figure 2.4. The SYN flood attack [10] exploits a vulnerability of the TCP/IP protocol. During SYN flood attacks, the attacker sends SYN packets with source IP addresses that do not exist or are not in use.

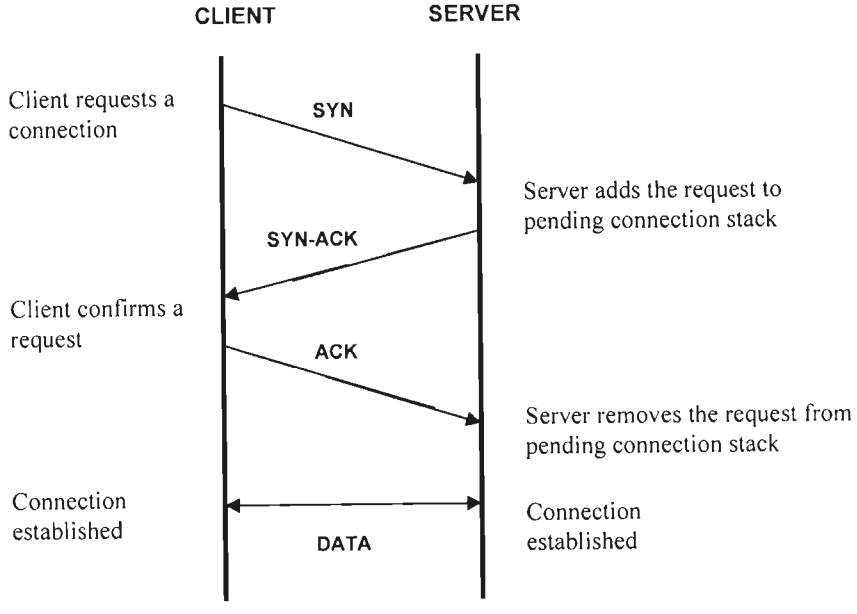


Figure 2.4. TCP 3-way handshake

During the 3-way handshake, when the server puts the request information into the memory stack, it will wait for the confirmation from the client that sends the request. Before the request is confirmed, it will remain in the memory stack. Since the source IP addresses used in SYN flood attacks are non-existent, the server cannot receive confirmation packets for requests created by the SYN flood attack. Thus, more and more requests will accumulate and fill up the memory stack. Therefore, no new request, including legitimate requests, can be processed and the services of the system are disabled. Generally, the space for the memory stack allocated by the operating system is small, and even a small scale SYN flood attack can be dangerous. In order to keep buffer space occupied for desired time, the attacker needs to generate steady stream of SYN packets towards the victim to again reserve those resources that have been freed by timeouts [180].

2.2.3.2 UDP Flooding Attack

The User Datagram Protocol (UDP) is a connectionless protocol that does not have flow control mechanisms, i.e., there is no built-in mechanism for the sender and receiver to be synchronized to adapt to changing network conditions. The UDP flood is a type of bandwidth attack that uses UDP packets. Since UDP does not have flow control mechanisms, when traffic congestion happens, both legitimate and attack flows will not reduce their sending rates. Figure 2.5 gives an example of how a single spoofed UDP packet can initiate a never-ending attack stream.

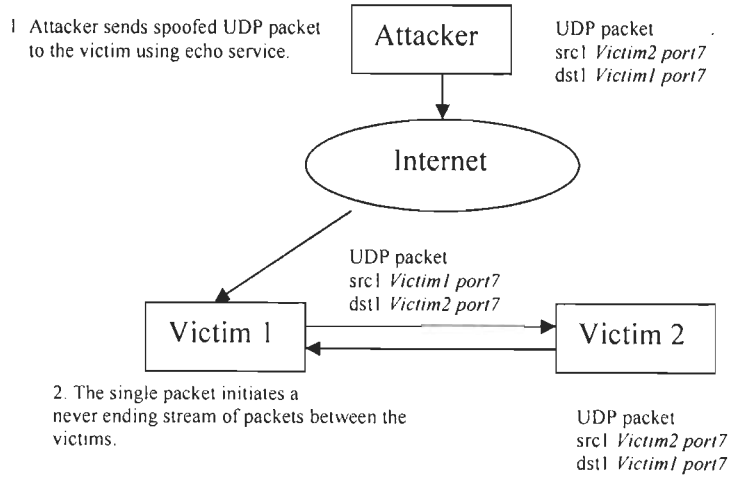


Figure 2.5. UDP flooding

The attacker sends a UDP packet to victim 1, claiming to be from victim 2, requesting the echo service. Since victim 1 does not know this is a spoofed packet, it echoes a UDP packet to victim 2 at port 7 (echo service). Then victim 2 does exactly the same as victim 1 and the loop of sending echo requests will never end unless it is stopped by the external source [181]. In addition, if two or more hosts are so connected, the intervening network may also become congested and deny service to all hosts whose traffic traverses that network. Solutions to the UDP flood are discussed in [181], which include disabling any unused UDP services, e.g., the echo service.

2.2.3.3 ICMP Flooding Attack

The Internet Control Message Protocol (ICMP) is based on the IP protocol and is used to diagnose network status. An ICMP flood uses ICMP packets. On IP networks, a packet can be directed to an individual machine or broadcast to an entire network. Packets may be broadcasted to all machines on or outside the local network. IP broadcast addresses are usually network addresses with the host portion of the address having all one bits. For example, the IP broadcast address for the network 10.*.*.* is 10.255.255.255. Network addresses with all zeros in the host portion, such as 10.0.0.0, can also produce a broadcast response. The smurf attack is a type of ICMP flood, where attackers use ICMP echo request packets directed to IP broadcast addresses from remote locations to generate DoS attacks. There are three parties in these attacks: the attacker, the intermediary, and the victim [17]. Figure 2.6 gives an example of the smurf attack.

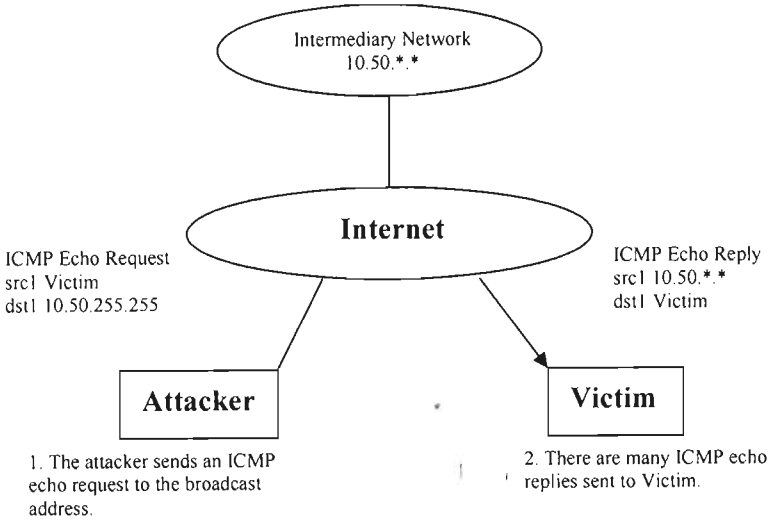


Figure 2.6. Smurf attack

First, the attacker sends one ICMP echo request packet to the network broadcast address and the request is forwarded to all the hosts within the intermediary network. Second, all of the hosts within the intermediary network send the ICMP echo replies to flood the victim. Solutions to the smurf attack are discussed in [17], which include disabling the IP-directed broadcast service at the intermediary network.

2.2.3.4 Domain Name Server (DNS) Attack

The attack sends a stream of DNS requests to multiple name servers, spoofing victim's address in their source address fields. Because name server responses can be significantly larger than DNS requests, there is potential for bandwidth amplification. Attackers usually request the same valid DNS record from multiple name servers [182].

2.2.3.5 Typical DDoS Attack

The steps to launch a DDoS attack has been discussed in Section 2.2.1. The attack traffic could use genuine or spoofed source IP addresses [183]. However, there are two major motivations for the attacker to use randomly spoofed IP addresses: (i) to hide the identity of the zombies and reduce the risk of being traced back via the zombies, (ii) to make it hard or impossible to filter this type of traffic without disturbing the legitimate traffic.

2.2.3.6 Distributed Reflector Denial-of-Service (DRDoS) Attack

A DRDoS attack uses a third-party (routers or web servers) to bounce the attack traffic to the victim. The DRDoS attack contains three stages. The first stage is very similar to the typical DDoS attack. However, after the attacker has gained control of a certain number of zombies, instead of instructing the zombies to send attack traffic to the victims directly, the zombies are ordered to send the spoofed traffic with the victim's IP address as the source IP address to the third parties. Compared with the typical DDoS attack, the DRDoS attack is more dangerous, for the following reasons. First, the DRDoS attack traffic is further diluted by the third parties, which makes the attack traffic even more distributed. Second, DRDoS attack [21] has the ability to amplify the attack traffic, which makes the attack even more potent.

2.2.4 Popular DoS Attack Tools

The various attack tools for DDoS can be divided into agent based and IRC based tools [38]. Agent-based DDoS attack tools are based on agent handler DDoS attack model (discussed in Section 2.2.1) and include Trinoo, Bandwidth depletion Trinoo, Tribe Flood

Network (TFN), TFN2K, Encrypted message based TFN2K, Stacheldraht, Mstream, Shaft etc. IRC-based DDoS attack tools have important features of agent-based attack tools but are more sophisticated and include Trinity v3, Random Spoofing based Trinity, Knight, Kaiten etc.

2.3 Existing DDoS Attack Defense Approaches

According to [178, 179], various proposals for defense against DDoS attacks can be broadly classified into following approaches: prevention, detection, characterization, response and mitigation. The use of these approaches depends on the phase of the attack. Prevention aims to fix security holes and vulnerabilities before the attack is launched. Detection aims to detect the presence of attacks and characterization aims to discriminate attack traffic from legitimate traffic during the attack. They are important procedures to direct any further action. Finally response and mitigation aims to eliminate effects of an attack. This section presents the key proposals developed so far under each of these approaches along with their limitations.

2.3.1 Prevention

Prevention is a mechanism which stops the attacks before they are actually launched.

Access control list (ACL) [40] in the routers or switches can be used to block traffic flow with specific characteristics (e.g. addresses, protocols, etc.) but only if the characteristics can be known in advance. The routers/switches lack the processing power and profiling intelligence to make such recognition on their own. On the other hand, the administrative time spent on determining ACL rules is cost overhead. The management of a large number of temporary ACLs that may have performance impacts, is non-trivial, very labor intensive and error prone. ACLs alone can not serve as a DDoS mitigation solution.

Firewalls have been used to protect the networks from DoS attacks. A firewall [41, 42] enhances network security by filtering suspicious traffic at the border of the network. Based on the characteristics of the network traffic, to include requested services, source and destination addresses, and individual users, a firewall will make a decision on whether to allow the traffic to pass through the network. Firewalls can also be utilized on individual host based systems. Firewall can be configured as default permit (blacklist based filtering) or default deny (whitelist based filtering). A good firewall configuration implements whitelist based filtering and denies everything other than explicitly allowed necessary traffic. However there are several shortcomings associated with a firewall.

Firewalls cannot protect against attacks that bypass it. The firewall at the network interface does not protect against internal threats. In certain cases high volumes of network traffic may overwhelm the network monitoring capability of the firewall resulting in the possible passing of malicious traffic between networks.

An Intrusion Detection System (IDS) [43, 45, 46] is used to detect and alert on possible malicious events within a network. IDS sensors may be placed at various points throughout the network. An IDS is normally signature based, i.e., it will look for predefined signatures of bad events. These signatures normally reside in a database associated with the IDS. The IDS detect attack signatures and update the firewall's filtering rules. IDS like Snort [45] are open source which can be customized on Linux to suit an organization's requirements [48]. The use of IDS as a network security device also leads to shortcomings. They generate far too much data and large percentage of false positives and are unable to detect new attacks. IDS also generate false negative alerts when IDS systems fail to detect a valid attack.

Ubiquitous Ingress/Egress Packet Filtering (UIPF) [51] filters incoming traffic according to a specified rule. There are two types of ingress filtering. One is ISP-to-customer ingress filtering, which filters the traffic from the external networks to the customer. Another is customer-to-ISP ingress filtering (known as egress filtering), which filters the traffic from the customer to the external networks. Figure 2.7 illustrates the operation of ingress and egress filtering.

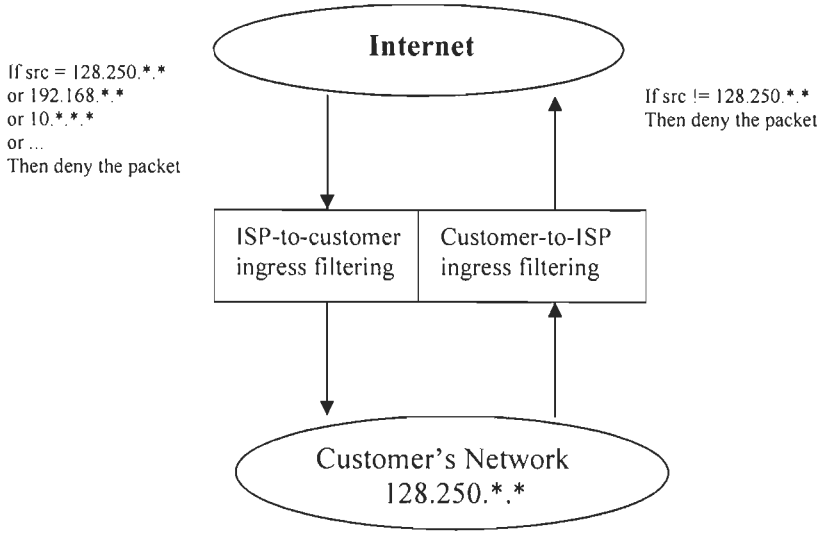


Figure 2.7. Defense against DDoS using ingress/egress filtering

For the ISP-to-customer ingress filtering, any internal IP address, private network IP address (e.g., 192.168.*.*) and specified IP addresses (e.g. the IP addresses of known malicious users) will be filtered. The filtering is normally integrated with firewall technology. However, the efficacy of the ISP-to-customer ingress filtering is limited. First, since the filtering is done close to the victim, it cannot prevent network bandwidth consumption. Second, it only denies a small proportion of IP addresses, so attacks can still be launched using the rest of the IP address space. To bypass the ingress filter, the attacker can carefully spoof the IP addresses.

For the customer-to-ISP ingress (or egress) filtering, all the IP addresses which do not belong to the ISP's network will be filtered. The egress filtering is effective in defending against IP source address spoofing since it can filter spoofed traffic close to the source. However, a series of costs are raised by implementing the customer-to-ISP ingress filtering. First, as it needs to do per packet checking, it will result in potential router overhead. Second, to beat the customer-to-ISP ingress filtering, the attacker can spoof the IP source address from the customer's network. Thirdly, it requires universal deployment.

The Router-based Packet Filtering (RPF) proposed by Park and Lee [52] extends ingress filtering to the core of the Internet. Generally, IP packets travel between the source and the destination using the same path. Hence, one border router would only expect traffic from a stable group of autonomous systems (ASes) on each link. RPF is implemented in border routers and filters any unexpected traffic on each link. The filtering rule for RPF is based on the topology of the autonomous systems and the policies for each AS. Simulation results show that a significant fraction of spoofed IP addresses can be filtered if RPF is implemented in at least 18% of ASes in the Internet [52]. Given that RPF should be implemented in 18% of the ASes, to make the scheme effective is a difficult task to accomplish. Moreover, RPF needs the BGP [53] messages to carry the source addresses, which significantly increases the BGP message size and processing time for the BGP message. Next, the dropped packets by RPF can be legitimate if there has been a recent route change. Similar to ingress filtering, RPF can only restrict the space for IP spoofing instead of completely stopping IP spoofing. Furthermore, the RPF cannot prevent non-spoofed DDoS attacks. In addition, since RPF depends on the BGP message to configure the RPF filter, the attacker can hijack a BGP session and disseminate bogus BGP messages to mislead border routers to update filtering rules in favor of the attacker.

The router-based packet filter [52] is vulnerable to asymmetrical and dynamic Internet routing as it does not provide a scheme to update the routing information. To overcome

this disadvantage, Li et. al proposed the Source Address Validity Enforcement (SAVE) Protocol [54]. It enables routers to update the information of expected source IP addresses on each link and block any IP packet with an unexpected source IP address. Similar to the existing routing protocols, SAVE constantly propagates messages containing valid source address information from the source location to all destinations. Hence, each router along the way is able to build an incoming table that associates each link of the router with a set of valid source address blocks. Therefore, SAVE is a protocol that enables the router to filter packets with spoofed source addresses using incoming tables. It shares the same idea with ingress filtering and RPF that the source address space on each link of the router is stable and foreseen. Any packet that violates the expected source address space will be regarded as forged and will be filtered. SAVE outperforms ingress filtering and RPF in that it overcomes the asymmetries of Internet routing by updating the incoming tables on each router periodically. However, SAVE needs to change the routing protocol, which will take a long time to accomplish. If SAVE is not universally deployed, attackers can always spoof the IP addresses within networks that do not implement SAVE. Moreover, even if SAVE were universally deployed, attackers could still launch DDoS attacks using non-spoofed source addresses.

Changing IP address of victim computer proposed by Geng *et al.* [55] is a simple solution to a DDoS attack. It invalidates the victim computer's IP address by changing it with a new one to switch the congestion point. This is called moving target defense. Once IP address change is completed, all Internet routers will have been informed, and edge routers will drop the attacking packets. However, this action still leaves the computer vulnerable because the attacker can launch the attack at the new IP address. Attackers can render this technique ineffective by adding a domain name service tracing function to the DDoS attack tools.

Geng *et al.* [55] also suggested some common preventive measures which should be taken up by individual servers and ISPs to defend DDoS attacks and are listed below:

1. In general, if network services are not needed or used, the services should be disabled to prevent attacks.
2. The host computers should update themselves with latest security patches for the bugs present and should use latest techniques available to minimize the effect of DDoS attack.

3. By disabling IP broadcasts, host computers can no longer be used as amplifiers in ICMP flood and smurf attacks. However, a defense against this attack will be succeeded only if all the neighboring networks disable IP broadcasts.

Kim *et al.* [56] proposed to apply security engineering and calculated system security level by identifying threat level to build a security countermeasure.

Overprovisioning [57] and resource multiplication mechanisms provide an abundance of resources to counter DDoS threats. The straightforward example is a system that deploys a pool of servers with a load balancer and installs high bandwidth links between itself and upstream routers [57]. For example, Microsoft has used it to withstand large DDoS attacks. Another approach is the use of Akamai [58] services for distributed web site hosting. User requests for a web page hosted in such a manner are redirected to an Akamai name server, which then distributes the load among multiple, geographically distributed web servers hosting replicas of the requested page. Costs are incurred in terms of extra resources in absence of attacks or low attack loads. It does not provide perfect protection, but for those who can afford the costs of resource multiplication, it has often proved sufficient. Such approaches essentially raise the bar on how many machines must participate in an attack to be effective. On demand resource allocation [59] can be used to complement overprovisioning, however the request completion time increases [59]. This deviates from the main goal of timely information [60].

Scheme proposed in [61] is effective in limiting the access of intended illegitimate users to information systems by detecting the privacy violations. However it is based on building privacy policy to detect probable privacy policy violations which requires post event information. In addition, several fault tolerant routing schemes have been proposed [62] . Also, changes in the router architecture have been proposed [63]. However, they are not effective. Moreover, they require major changes in the protocols and hardware, which is very hard to achieve looking at the current scenario of the Internet.

2.3.2 Detection

Apart from attack prevention, the first step to defend against DoS attacks is attack detection. Detection is the process of identifying that a network or server is under attack. Detection can be passive if logs are analyzed after attacker fulfils his/her desire and attack is over, it can be on time if we detect when attack is ongoing or it can be proactive if either attack is detected before it reaches target or before an appreciable degradation of service.

There are three reasons for attack detection. First, if a target can detect an attack before the actual damage occurs, the target can implement attack reaction and protect legitimate users. Second, if attacks can be detected early i.e. close to attack sources, attack traffic can be filtered before it wastes any network bandwidth. However, there is generally insufficient attack traffic in the early stage of an attack and at links close to attack sources. Consequently, it is easy to mistake legitimate traffic as attack traffic. Therefore, it is challenging to accurately detect attacks quickly and close to attack sources. Third, it is important to differentiate DoS attacks from flash crowds so that targets can react to them separately. Generally, there are two measures for DoS attack detection. The first is detection time and the second is false positive rate. The DoS attack traffic may look very similar to legitimate traffic. This means that any detection scheme has a high risk of mistaking legitimate traffic as attack traffic. A good detection technique should have a short detection time and low false positive rate.

There are three methods to detect DDoS attacks:

1. Rule or Signature Based which are normally used in prevention [40-43, 45, 46] (discussed in Section 2.3.1).
2. Anomaly or misuse based which models the behavior of normal traffic, and then reports any anomalies [65-75].
3. Congestion based which are now-a-days used in response, mitigation and tolerance [77-80, 154, 162].

Gil *et al.* [65] propose a scheme called MULTOPS to detect DoS attacks by monitoring the packet rate in both the up and down links. MULTOPS assumes that packet rates between two hosts are proportional during normal operation. A significant, disproportional difference between the packet rate going to and from a host or subnet is strong indication of a DoS attack. MULTOPS assumes that the incoming packet rate is proportional to outgoing packet rate, which is not always the case. For example, real audio/video streams are highly disproportional, where the packet rate from the server is much higher than from the client. This gives rise to false positives. The simplest way to cripple MULTOPS is to use randomly spoofed IP addresses, which makes the calculation based on genuine IP addresses inaccurate and consumes resources by storing spoofed IP address information. Another countermeasure is to connect to the target from a large number of attack sources in a legitimate manner. Therefore, the packet rate ratio between in flows and out flows during the attack will appear to be normal and undetected by MULTOPS.

Wang *et al.* [66] proposed SYN detection to detect SYN floods and Blazek *et al.* [67] proposed batch detection to detect DoS attacks. Both methods detect DoS attacks by monitoring statistical changes. The first step for these methods is to choose a parameter for incoming traffic and model it to be a random sequence during normal operation. In [66], the ratio of SYN packets to FIN and RST packets is used while in [67] a variety of parameters such as TCP and UDP traffic volume, are used. The attack detection is based on assumption that there will be a statistical change when an attack happens. The detection scheme in [66] is based on the fact that a SYN packet will end with a FIN or RST packet during normal TCP connection. When the SYN flood starts, there will be more SYN packets than FIN and RST packets. The attacker can avoid detection by sending the FIN or RST packet in conjunction with the SYN packets. To beat the detection scheme in [67], the attacker can carefully mix different types of traffic to ensure the proportion of each traffic is the same as it is in normal traffic.

Cheng *et al.* [68] propose to use spectral analysis to identify DoS attack flows. In this approach, the number of packet arrivals in a fixed interval is used as the signal. A normal TCP flow will exhibit strong periodicity around its round-trip time in both flow directions, whereas an attack flow usually does not. First of all, spectral analysis is only valid for TCP flows. As UDP and ICMP are connectionless protocols, the periodic traffic behavior is unexpected. Attackers can use UDP or ICMP traffic to confuse the detection scheme. Moreover, the attacker can mimic the periodicity of normal TCP flows by sending packets periodically.

Kulkarni *et al.* proposed a Kolmogorov complexity based detection algorithm [69] to identify attack traffic. The assumption of the Kolmogorov test is based on the fact that multiple attack sources use the same DoS attack tool. Therefore, the resulting traffic is highly correlated. Unfortunately, there is no theoretical analysis to support this assumption. Attacker sources can be devised to break the correlation by sending attack traffic at different times, with different traffic types, packet sizes, and sending rates. For example, attackers can use the IP address of a compromised computer as the random seed to generate a set of parameters for configuring attack traffic. By doing this, attack traffic will appear random, which can bypass detection.

Cabrera *et al.* [70] proposed a scheme to proactively detect DDoS attacks using time series analysis. Time series analysis is based on the strong correlation between traffic behavior at the target and traffic behavior at the attack source. There are three steps to this scheme. The first step is to extract the key variables from the target. For example, the

number of ICMP echo packets is the key variable for Ping Flood attacks. The second step is to use statistical tools (e.g., Auto Regressive Model) to find the variables from the potential attackers that are highly related to the key variable. For example, the number of ICMP echo reply packets at the potential attackers is highly correlated with the key variable for Ping Flood attacks. The third step is to build a normal profile using the found variables from the potential attackers. Any anomalies from potential attackers compared with the normal profile are regarded as strong indications of an attack. The vulnerability of this scheme is that the efficacy of training is based on the features of known attacks. The attacker can disturb or disable the detection scheme by inventing new attacks.

Bencsath *et al.* [71] proposed to monitor the incoming traffic and detected attacks on the basis of traffic level measurements for e.g., if buffer length exceeded a threshold or aggregate traffic exceeded a threshold. Though their scheme minimized false positives, it could handle only special cases of DDoS attacks.

Feinstein *et al.* [72] proposed statistical approach to DDoS detection [73] in which they calculated randomness in a particular feature of packet for normal flows. Whenever randomness crossed threshold in actual scenario, they termed it as anomaly or attack. As it was not applied for training using proper internet topologies, so it did not give any better results and produced high false positives and negatives.

A structural analysis of network traffic flows is proposed by Lakhina *et al.* [74]. As network traffic arises from the superposition of Origin-Destination (OD) flows, they argue that the understanding of OD flows becomes essential for addressing a wide variety of problems, including traffic engineering and anomaly detection. They propose the use of sampled flow measurements in an IP network for detecting and understanding network-wide traffic anomalies. They analyze randomness in feature distribution for highly sensitive detection.

History based IP filtering proposed by Peng *et al.* [75] relies on the basic idea that history repeats itself. In DDoS context, the legitimate users access same website regularly. As per this approach, all the IP addresses of the previous successful connections are recorded in order to compile an IP address database (IAD). The stale IP addresses are left out as per sliding window based on timestamps and prefixed window time. Then when network or website experiences a high level of congestion, edge routers admit the incoming packets according to pre-built IAD. Hash–based/ Bloom filter [76] techniques are used to quickly search IP in IAD. This scheme is robust, and does not need the

cooperation of the whole Internet community. However, if the attacker gets estimate of the window time, then IAD can become database of attackers. A new legitimate client in the event of attack is not entertained. To improve percentage of legitimate traffic getting through, the size if IAD need to be increased, the lookup time will also increase in the process. Even in normal operation, all packets need to be checked for their source legitimacy, so consume resources to store connection history. Defense is vulnerable to high rate attack traffic and flash crowds.

Mahajan *et al.* proposed Aggregate-based Congestion Control (ACC) [77-79] which could give some relief to congested links due to DDoS attacks and flash crowds. The ACC was broken into two phases, namely detection and control. In the detection phase, ACC aims at learning a congestion signature and identifying a small number of aggregates responsible for congestion. The ACC agent tries to find the congestion signature using the packet drop history (or the random samples) of the last K seconds during times of sustained high congestion. The authors propose the use of a destination-based identification algorithm, which first draws out a list of high-bandwidth addresses (32-bit) based on the drop history (or random sample) and then clusters these addresses into 24-bit prefixes. For each of the clusters, it tries to find a longer prefix that still contains all the dropped packets, since a longer prefix characterizes a congestion signature better and does not punish a large category of traffic. Although this algorithm is simple to implement, congestion signature is based on volume of traffic and it results in unfairness for the legitimate traffic to the congested destination. A more accurate and flexible identification algorithm is needed to maintain the fairness between friendly and misbehaving aggregates. Once the router knows the congestion signature, it then filters the bad traffic according to this signature. The implementation under FreeBSD was done by Ioannidis *et al.* [80].

2.3.3 Characterization

As with detection, characterization is performed by measuring features of the incoming traffic and comparing them either to a normal profile in anomaly-based methods or to a attack profile in signature-based methods. These features may be actual statistical features measured in real-time or simple observations acquired by actively testing the users of the network and asking them to prove their legitimacy. Anomaly-based methods are less accurate, but apply to a broader range of attacks, while signature-based methods are more dependable, but apply only for the attacks that they have been designed to detect and counter. Although both are used, academic research tends to prefer anomaly-based classification methods, since it is far easier to keep a signature of the legitimate users'

normal traffic for one's network than the signatures of a range of known attack types, which can never be complete.

Mechanisms that deploy pattern detection like IDS [45, 46] store the signatures of known attacks in a database. Each communication is monitored and compared with database entries to discover occurrences of DDoS attacks. Occasionally the database is updated with new attack signatures. Although known attacks are easily and reliably detected, and no false positives are encountered, the obvious drawback of this mechanism is that it can only identify known attacks, and it is usually helpless against new attacks or even slight variations of old attacks that cannot be matched to the stored signature. Snort [45] provides one example of a DDoS defense system that uses pattern attack detection.

Kim *et al.* [81] devised a number of heuristics to detect specific attack patterns from flow header data. They use flow measurements taken on a single router or link. However, no evaluation on real system is given.

Kong *et al.* [82] propose a practical solution to defend against DDoS by combining complementary countermeasures along with identifying DoS attacks and use random flow network model. Manikopoulos *et al.* [83] propose a statistical anomaly approach based on neural network classification to identify DDoS traffic. Jin *et al.* [84] propose covariance analysis method to effectively differentiate between normal and attack traffic. However, no theoretical justifications are provided.

Hussain *et al.* [85] made use of distributed nature of the attacks to identify them. All attack flows do not traverse the Internet through the same paths; thus they reach the victim destination at different times, resulting in a gradual increase of the incoming traffic. This ramp-up behavior was initially proposed as a means to tell whether an attack is distributed or single-source. A longer ramp-up time will also be associated with a greater number of spoofed source IP addresses. Consequently, it also means that the IP addresses which arrive after the DDoS attack and until it reaches its peak are more likely to be illegitimate.

In Self-Aware Quality of Service (QoS) driven network environments [86], clients may specify that they belong to a specific type or request a certain level of QoS. The degree with which a QoS agreement is honored by the client is a strong indication of his/her validity. Both attackers and misbehaving clients will fail such a test.

Hop Count Filtering (HCF) [87] exploits the fact that although the attacker can forge any field in the IP header, he/she cannot falsify the number of hops a packet needs to reach its destination starting from its source address. Their very simple algorithm infers the

number of hops traversed (from the packet's TTL field) and compares it to the value that can be inferred for the source, which is stored in a relevant table. If these two values are significantly different, it is a clear indication of IP spoofing, and is a good reason to treat this source's packets as illegitimate.

These and other similar passive tests have the advantage of being relatively lightweight, but are by themselves not sufficient to achieve accurate classification. More accuracy inevitably requires more specialization.

Mirkovic *et al.* [88, 89] proposed a source end defense which uses a set of anomaly-based classification criteria for flows and connections. For instance, a TCP or an ICMP flow may be classified as an attack flow depending upon their packet ratio. For the UDP protocol, a "normal flow model" is proposed to be a set of thresholds based on the upper bound of allowed connections per destination, the lower bound of allowed packets per connection, and the maximum allowed sending rate per connection. The classification of connections is also done based on limits of the connections' allowed packet ratios and sending rates. D-WARD addresses the fundamental DoS attack defense rationale: removing attack traffic at its source. However, it faces the following two challenges. First, for a large scale of DDoS attack, attack traffic generated by one source network can be very small and unnoticed compared with legitimate traffic flows. Hence, detecting attack traffic accurately can be difficult or impossible. A well-organized, geographically distributed DoS attack is likely to defeat this scheme as attackers can control the attack traffic originated from each source network to be within normal range. Second, while D-WARD plays a similar role as ingress filtering, it is more expensive to implement. Consequently, the deployment motivation is a big concern.

Scheme in [90] uses the Bayesian concept of the conditional legitimate probability (CLP) as the basis for a packet filtering scheme. Traffic characteristics during an attack are compared with previously measured legitimate traffic characteristics, and the CLP provides an indication of the legitimacy of suspected packets. An extension to reduce complexity and enhance performance is presented in [91]. Evidence can be combined from various sources [92]. However, as with all profile-based, and particularly Bayesian profile-based DoS approaches, the greatest challenge is not the fine-tuning of the defense mechanism but acquiring dependable traffic profiles.

Aggregate based congestion control techniques [77-80] detect attacks and identify aggregates of flows containing bad packets using congestion signatures as described in

Section 2.3.2. Since normal traffic may also be a part of the identified aggregate containing bad traffic, it will also be dropped along with the identified aggregate. Hence ACC techniques are inaccurate and lead to collateral damage.

A human can easily tell the sequence of letters that appear in a CAPTCHA's (Completely Automated Public Turing Test to Tell Computers and Humans Apart) image, while computers usually cannot, so that CAPTCHAs have been suggested to counter DDoS attacks against web servers [93, 94]. They have been used to block automated requests to websites, such as the automated email account registration which has plagued Hotmail and Yahoo in the past. However, issuing a graphical Turing test and expecting an answer requires that a connection be established between the attacker and the web-server, thus rendering the authentication mechanism itself a potential DoS target. Kandula *et al.* [95] address this issue and suggest minor modifications to the TCP protocol to overcome it. They also argue that it is not the actual answer to the test but the behavior of the client that matters. A human would solve the puzzle either immediately or after reloading the page a few times. A computer would probably continue requesting the web page. Despite their value, a DoS detection mechanism cannot depend solely on CAPTCHAs. Dedicated applications built by Computer Vision researchers achieve upto 92% success in solving commonly used types of CAPTCHAs [93]. Advances in Artificial Intelligence along with simple craftiness limit the value of CAPTCHAs, while visual CAPTCHAs are also a major impediment to computer users whose vision is impaired.

Several methods have been proposed for actively challenging the clients' legitimacy. The work presented in [96] detects aggressive behavior and changes the frequency of tests to identify attack flows. The frequency of test is randomly determined for high rate flows. A system called Netbouncer [97], is representative of this category. Netbouncer keeps a list of authorized users (beyond suspicion), while the rest undergo a series of tests, divided into packet-based, flow-based, application-oriented and session-oriented. Various QoS techniques are utilized to assure fair sharing of the resources by the traffic of the legitimate clients, while this legitimacy expires after a certain interval and needs to be challenged again.

A very similar concept is explored in the puzzle based defense solutions [98-100]. The clients are asked to solve a little cryptographic puzzle before their connection request is authorized. The puzzle may take a little time to solve, while the defending server can rapidly verify the result. This does slow down the attacker, but does not guarantee that it

will suffice, since overwhelming the puzzle-generation process will still be possible if the attacker's rate is sufficiently high.

Some of these active test solutions may achieve impressive levels of accuracy, but all suffer from the common weakness of being exploitable as DoS vessels themselves.

Signal processing techniques have been used to analyze malicious network traffic and to detect ongoing attacks. Cheng *et al.* [68] use spectral analysis to identify high volume DoS attack flows due to change in periodicities in the aggregate traffic. Barford *et al.* [101] focused on detecting and classifying anomalies from flow measurements taken at a single router or link. They employed a wavelet-based signal analysis of flow traffic to characterize single-link byte anomalies. Wavelets and other signal processing techniques have been extensively used to analyze both wired and wireless network traffic [102]. Scheme in [103] uses statistical signal processing based on abrupt change detection to identify attack, however it requires dealing with missing samples for applications running over UDP.

Mutaf [104] proposed a real-time anomaly detection scheme to identify TCP SYN flood attacks by analyzing daily maximum arrival rate. Similar work was done by Haggerty *et al.* [105]. The above two schemes were designed for SYN flood attacks, which occurred before connection establishment and failed to identify malicious TCP flows after successful TCP connections. Xu *et al.* [106] proposed to isolate malicious traffic via HTTP redirect messages. Since most of the attack flows employ spoofed source IP addresses, their sources cannot receive the redirect messages, and thus the subsequent packets from them will be blocked. This scheme is simple and may be readily implemented. However, their mechanism worked only for web servers.

Kim *et al.* [107] suggest a technique for traffic anomaly identification based on analyzing correlation of destination IP addresses in outgoing traffic at an egress router. This address correlation data are transformed using discrete wavelet transform for effective detection of anomalies through statistical analysis close to the source. They present a multidimensional indicator using the correlation of port numbers and the number of flows as a means of detecting anomalies. Their scheme shows that the proposed signals are more effective in detecting attacks than the analysis of traffic volume alone.

Xenoservice is based on the approach of resource multiplication and acquire resources dynamically once the attack has been detected [108]. It enables accounted execution of

untrusted code. Quality of Service is monitored and if it starts to deteriorate, the website is replicated to other servers.

Genetic programming [109] methods have been used for attack classification. However, they require training data sets and such exhaustive training datasets are hard to achieve. Scheme in [110] uses multimodal nature of traffic to classify it on the basis of packet train length and packet train size.

2.3.4 Response

In terms of response, the existing literature uses either packet dropping (filtering) or redirection of the packets that the classification methods identified as illegitimate. Redirecting the offending packets to a controlled part of the network not only decreases the congestion in the victim network, but also provides with the opportunity to analyze the attack. However, very limited work exists [184-186] and such a safe and controlled environment is difficult to maintain. The family of response methods based on filtering is much more common. Packet filtering can be done according to classification rules [52, 54, 55, 75] or can be based on link testing [111-113] schemes and pushback schemes [77-80] that traceback to the source and drop the attack traffic.

Many routers include a feature called input debugging [111, 112] that allows an operator to filter particular packets on some egress port and determine which ingress port they arrived on. This capability is used to implement a trace as follows: First, the victim must recognize that it is being attacked and develop an attack signature. The network operator installs a corresponding input debugging filter on the victim's upstream egress port. This filter reveals the associated input port, and hence which upstream router originated the traffic. The process is then repeated recursively on the upstream router, until the originating site is reached. Once this reroute is complete, network operator can then use input debugging at the tracking router to investigate where the attack enters the ISP network. The most obvious problem with the input debugging approach is that it requires considerable management overhead, time, attention and commitment of both the victim and the remote personnel and appropriate technical skills.

Burch *et al.* [113] developed link testing traceback and filtering technique called controlled flooding that does not require support from network operators. It tests links by flooding them with large bursts of traffic and observing how this perturbs traffic from the attacker. Using a pre generated map of Internet topology and by observing changes in the rate of packets received from the attacker, the victim can therefore infer which link they

arrived from. As with other link testing schemes, the basic procedure is then applied recursively on the next upstream router until the source is reached. The disadvantage is that controlled flooding should have ability to generate huge traffic and is itself a DoS attack. Also, controlled flooding requires the victim to have considerable knowledge of network topology. It can be difficult to recognize the set of paths being exploited when multiple upstream links are contributing to the attack. Like all link-testing schemes, controlled flooding is only effective in tracing an on-going attack and cannot be used "post-mortem". Moreover, high speed routers lack tracking ability, such as the ability to tell which link one packet came from.

CenterTrack [111] uses overlay network architecture to overcome the limitations in [113]. It employs overlay network of special tracking routers which links all edge routers to a central tracking router. During an attack, the traffic to the victim is routed through the overlay network by dynamic routing. The attack packets can be easily tracked using tracking routers, hop-by-hop, through the overlay network, from the routers close to the target to the attack entry point of the ISP and then filtered. This reduces number of hops required to trace back to the approximate source of attack to 2-3 hops. The scheme is based on assumption that attack and attack signature is identified by third party detection system or already implemented in IDS. Extra resources in terms of overlay networks are required. Moreover changes are required to global routing table. In addition, DoS attacks that originate from within the overlay network cannot be tracked. Finally, there are high overheads of storage and processing at tracking routers as entire traffic database has to be maintained.

Several techniques have been proposed to generate response to aggregates causing congestion using pushback and filtering.

Yong Xiong [114] took the defense of DDoS attack as a congestion control problem. They propose to use backward pressure propagation, feedback control scheme to defend DDoS attack. They used rate-based and queue-length based algorithms to create the feedback signal accordingly. Once the input traffic rate or output queue length has exceeded the desired threshold, a feedback signal is sent to adjust the admitted portion of traffic in different input and output ports to put the rate and queue length below the threshold. The method is effective to make sure the network traffic works in a tolerable level during DDoS attack. However, they don't set up a scheme to tell good traffic from bad traffic, which could make the good traffic under unnecessarily control during a DDoS attack.

Several schemes have been proposed by Mahajan *et al.* [77-79] to control high bandwidth aggregates in the network. As described in Section 2.3.2, ACC is broken into two phases namely detection and control. ACC can be local ACC and pushback based ACC. In the detection phase, ACC agent learns congestion signature and identifies aggregates containing bad traffic based on volume of traffic to target from different links. In control phase, the rate limit determines whether a packet is to be discarded or forwarded. In local ACC, both phases were applied on same congested router whereas in other referred to as pushback, local ACC is extended to upstream routers which filters the bad traffic according to this signature. The advantage is that when pushback is applied for upstream, there is more bandwidth for legitimate traffic in downstream routers. Architecture of pushback and its implementation under FreeBSD is discussed in [80]. This scheme is effective against most DDoS attacks except uniformly distributed attack sources. It needs a narrow and accurate congestion signature to make sure only attack traffic is filtered while legitimate traffic is not affected. Since the pushback scheme aggregates attack traffic according to destination IP addresses, it is vulnerable to attack traffic with spoofed source addresses. Moreover, this scheme infers attack sources by checking the traffic volume to the victim on each upstream link. If the attack sources are highly distributed, the traffic volume to the victim on each upstream link will appear to be similar, this gives rise to false negatives. As the limiting is applied based on victim address, so a lot of legitimate traffic is also throttled causing high collateral damage.

Yau *et al.* [115] used router throttles to combat DDoS attacks against Internet servers. A proactive approach is followed in the sense that before aggressive packets can converge to overwhelm a server, routers along forwarding paths, regulate the contributing packet rates to more moderate levels, thus averting an impeding attack. The throttle limits the rate at which packets can either be dropped or rerouted to alternate server. The throttle rate is determined by two strategies: just half or fairly equal throttling (fair throttling) at all routers. Here no pushback and response messages are required as in Pushback technique [80]. However, attackers can exploit communication part as no secure ways are used to send throttle messages in same and different domain. In case of meek slow rate attack, collateral damage is more as normal packet survival ratio (NPSR) is very low. Control parameters should be set more dynamically and intelligently.

Another of the significant approaches is Secure Overlay Services (SOS) [116] which supports emergency services. The architecture of SOS is constructed using a combination of secure overlay tunneling, routing via consistent hashing, and filtering. It reduces the

probability of successful attacks by: (i) performing intensive filtering near protected network edges, pushing the attack point perimeter into the core of the network, where high-speed routers can handle the volume of attack traffic, and (ii) introducing randomness and anonymity into the architecture, making it difficult for an attacker to target nodes along the path to a specific SOS-protected destination. The goal of SOS is to route only the "confirmed" users' traffic to the server and drop everything else. The clients are authenticated at the overlay entrance and they use the overlay network to reach the server. Only a small set of source addresses are approved to reach the server, while all other traffic is heavily filtered out. The main advantage of SOS is that it can be applied over the existing IP infrastructure and can guarantee to some extent that in times of crisis a "confirmed" user will have access to the victim server. However, SOS is more difficult to deploy in a fully public network, since the clients must be aware of the overlay network and use it to access the victim. Also, it does not offer protection for the incoming links of the filtering router in front of the client, which can be quite easily overwhelmed by sheer volumes of DoS traffic.

The SOS approach is generalized by Mayday [117], in which overlay networks and lightweight packet filtering are combined. The overlay nodes perform client authentication and protocol verification, and then relay the requests to a server, which is protected from the outside by simple packet filtering rules.

2.3.5 Mitigation

The main aim of mitigation is to provide optimum level of service as per QoS requirements to legitimate clients while the network is under attack. In network based attacks, congestion at access links causes the denial of service. In server or node based attacks, sheer volume and rate at which requests for service keep coming to servers, which the server is not able to handle, exhausts the finite server resources and causes the DoS. Depending upon the type of attacks, defense can either be link based or node based respectively. Figure 2.8 shows the classification of defense solutions proposed for mitigation. Node based defense solutions are discussed first.

Dynamic Resource Pricing [121] imposes dynamically changing prices on resources based on the system load. This cost has to be dispensed from the requesting client before the resource is allocated. Client puzzles [98, 99] (discussed in Section 2.3.3) is a special case of such a pricing mechanism, where the client has to solve a cryptographic problem with varying complexity before the server allocates resources to the request and starts

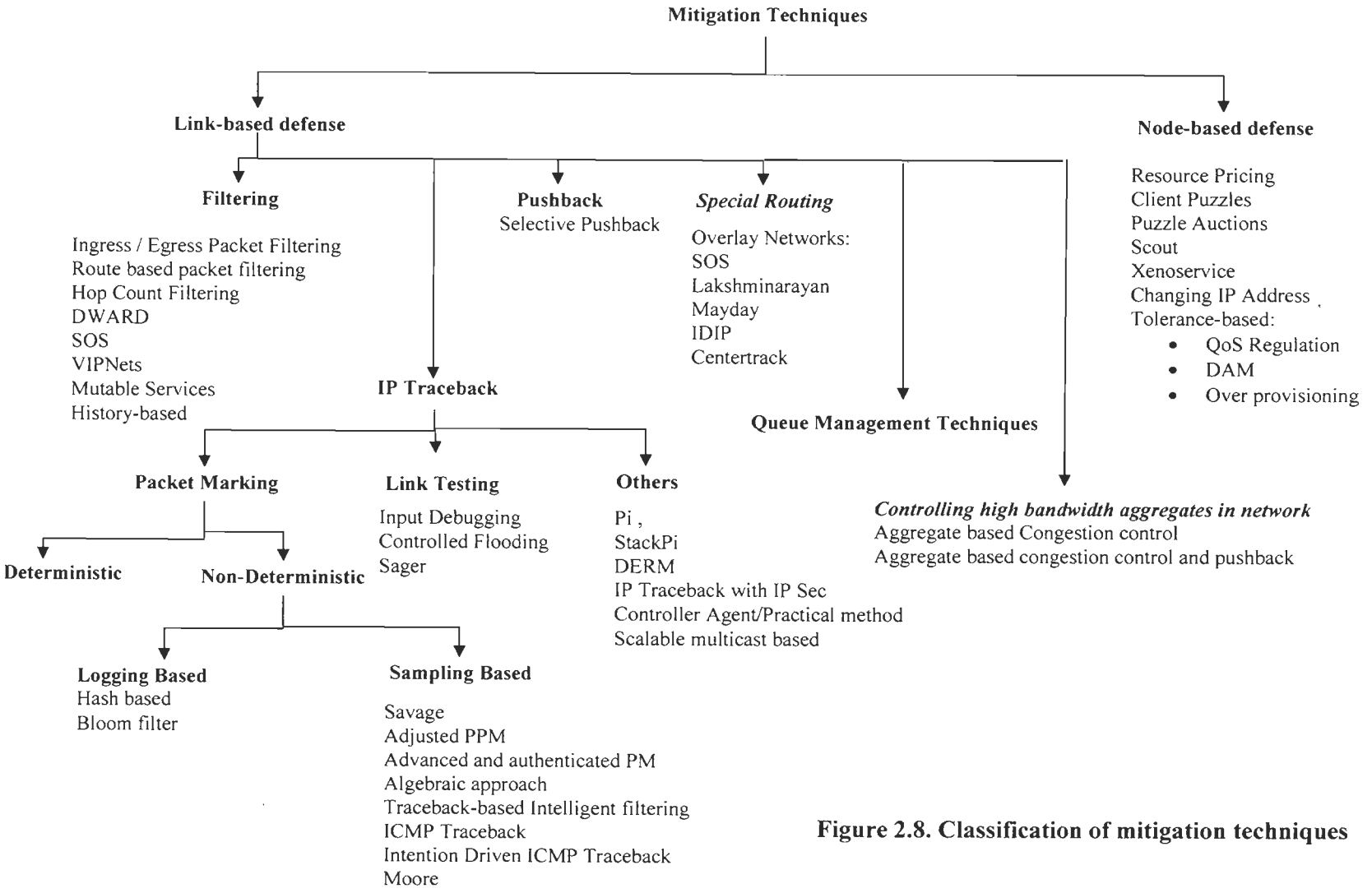


Figure 2.8. Classification of mitigation techniques

servicing it. Puzzle auctions [100] is based on similar concepts. They are implemented in Linux kernel as a framework to tune puzzle difficulty so as to minimize legitimate client cost in the presence of an adversary of unknown computing power. The main disadvantage is the requirement of special client software.

Resource accounting schemes have been proposed to defend against DoS attacks. The ability to account for resources allocated to each client according to negotiated contracts, detect contract violations and recover misused resources is a design target of the Scout operating system [120]. Using the techniques employed in QoS regulation, Garg *et al.* [122] proposed to regulate resource consumption that belong to category of resource accounting. They suggest resource regulation can be done at flow level, where each flow gets a fair share of resource. However it is still possible to mount a DoS attack by having a large number of hosts connecting to the server each claiming their slice of resource, thus causing resource starvation. Other techniques include resource reservation for different class of applications to provide guaranteed QoS [171, 187].

Changing victim IP address [55] and Xenoservice [108] described earlier provide methods for node based DDoS defense.

The idea of introducing a layer of indirection to defend against DoS attacks was presented in DoS Attack Mitigation (DAM) [123] framework similar to SOS [116]. DAM uses the network overlay to hide the locations of gateways to protect a Content Distribution Network (CDN). Firewalls at the entry point of each replica in the CDN allow only traffic sourced at the secret gateways. The requirements of both continuous changes of gateway locations, to avoid detection, and flexibility in the assignment of overlay nodes to services would cause an overhead of maintaining an up-to-date list of gateway addresses at the replica firewalls.

Overprovisioning [57] (discussed in Section 2.3.1) serves as tolerance based scheme for node based defense. However, it has costs associated with it due to the absence of attacks in majority of time or at low attack loads, especially when client load is low.

Client based schemes like proactive server roaming highlighted by Khattab *et al.* [124] was further extended by Sangpachatanaruk [125] . Here one server from cluster of servers is made active at a particular time. However, it requires secure communication methods.

Kargl *et al.* [127] suggested load balancer based technique in which cluster of web servers are protected by firewall and load balancer. Firewall implements traditional prevention measures and filters suggested by load balancer time to time. Load balancer act

as translator and allocator for all requests to appropriate web server as per load. Moreover traffic monitors at web servers and load balancer in consultation with manager deduce classification for packets to be treated by load balancer using class based queuing (CBQ). The same CBQ is also used at web servers for sending response to various classes. Long delays are caused to legitimate packets because of CBQ and slow rate attacks go unnoticed.

Link based defenses for mitigation include filtering, IP traceback, aggregate based congestion control and pushback, special routing and queue management techniques that are discussed next.

Many filtering schemes have been discussed with their obvious disadvantages. Ingress packet filtering [51] (discussed in Section 2.3.1) requires filter to be placed on the boundary of every single sub network. Route-based distributed packet filtering (DPF) [52] (discussed in Section 2.3.1) does not block all spoofed packets. The main difficulty is how to collect and maintain routing information within the participating routers. In the secure overlay services (SOS) architecture [116] (discussed in Section 2.3.4), only packets coming from a small number of nodes, called servlets, are assumed to be legitimate. However, this scheme is difficult to deploy. Hop count filtering [87] (discussed in Section 2.3.3) is based on classification technique which is not very accurate. As the ultimate goal for DoS attack defense is to filter attack traffic at the source, Mirkovic *et al.* [88] proposed a scheme called D-WARD [89] (discussed in Section 2.3.3) to defend against DoS attacks at the source network, where the attack sources are located. However, their scheme is difficult to deploy and easy to defeat. In [128], it is proposed that the ISPs carry the packets of the victim's "VIP" clients in a privileged class of service, protected from congestion, whether malicious or not, while all non-VIP traffic is considered as low-priority and can be dropped in the case of an attack. The approach is simplistic and can prove very useful for transaction-based websites, such as e-commerce. The architecture relies on the provision of QoS mechanisms, such as diffserv [129] in intermediate routers. Mutable services [130] is a framework to allow for relocating service front-ends and informing legitimate clients of the new location. History based filtering [75] (discussed in Section 2.3.2) is another example of a link based defense based on filtering.

IP traceback techniques [170] can be subdivided into three categories, packet marking, link testing and others. Most of the IP traceback techniques [131] are based on packet marking. Packet marking can be deterministic or non-deterministic. Deterministic packet marking (DPM) [132] requires each router to contribute into the packet path information

field, while in non- deterministic or probabilistic packet marking (PPM), a router makes this contribution with probability p. The non-deterministic packet marking schemes can further be subdivided into logging based or sampling based techniques.

The main idea of PPM is to let routers mark packets with path information probabilistically and let the victim reconstruct the attack path using these marked packets. In sampling based techniques, Savage *et al.* [39] proposed to traceback the IP source by marking path information into incoming packets probabilistically while they travel between source and destination. They use fragmented marking scheme (FMS) and suggest that routers probabilistically mark the 16 bit IP identification field, and that the receiver reconstructs the IP addresses of routers on the attack path using these markings. The authors describe some basic marking algorithms, such as appending each node's address to the end of the packet, node sampling, where each router chooses to write its address in the node field with some low probability p, or edge sampling of participating routers. Although, no specific field has been reserved for tracking purpose in the current Internet protocol IPv4, it is included in IPv6 and is called the identification field. The FMS do not work well if only a small number of routers implement them.

Park and Lee [133] study tradeoffs for various parameters in PPM. They proposed to put the distributed filters on the routers and filter the packets according to the network topology. This scheme can stop the spoofed traffic at an early stage. However, in order to be effective, there is a need to know the topology of the Internet and the routing policy between Autonomous Systems, which is hard to achieve in the expanding Internet.

Song *et al.* [134] proposed advanced and authenticated packet marking scheme (AMS) to improve the efficiency and security of PPM. It was based on a new hashing scheme to encode the path information and an authentication scheme to ensure the integrity of the marking information. It used the identification field of IP header for storing the hash as well as the distance from the router which marked the packet. AMS suffered from high false positive rate and lack of the enough range for the hash values which increased the probability of collision.

With the aim to minimize the time required to reconstruct the path, [135] proposed adjusted probabilistic packet marking. The time required to reconstruct the path depends on the time it takes to receive packets that have been marked by each router on the attack path. This in turns depends on choice of marking probability. Adjusted PPM adjusts the marking probability used by each router to reduce the number of packets needed to

reconstruct the attack paths as compared to PPM and can locate attack sources regardless of spoofed source addresses.

Dean *et al.* [136] follow a slightly different approach to probabilistic packet marking. They reframe the IP traceback problem as a polynomial reconstruction problem and use algebraic techniques and present a series of schemes. Coding scheme using an algebraic approach to embed path information reduces the number of packets needed to reconstruct the attack path. The scheme does not require an upstream router map to construct an attack path but is less efficient in presence of multiple attackers.

In [137] a protocol-independent DDoS defense scheme has been proposed that is able to dramatically improve the throughput of legitimate traffic during a DDoS attack. It works by performing "smart filtering"; dropping DDoS traffic with high probability while allowing most of the legitimate traffic to go through. This clearly requires the victim to be able to statistically distinguish legitimate traffic from DDoS traffic. The proposed scheme extends IP Traceback techniques to gather "intelligence" or information such as whether or not a network edge is on the path from an attacker or "infected". By preferentially filtering out packets that are inscribed with the mark (identity) of an "infected" edge, the proposed scheme filters out most of the traffic from attackers, since each and every edge on an attacker's path to the victim is infected. Packets from a legitimate client, on the other hand, with high probability will not be filtered out, since typically most of the edges on the client's path to the victim are not infected.

Bellovin [138] proposed an approach called the ICMP traceback. In this scheme [138, 146], routers generate an ICMP traceback message (called an iTrace packet [139]) to the destination containing the address of the router and information about adjacent routers with a low probability (eg. 1/20,000). For a significant traffic flow, the destination can gradually reconstruct the route that was taken by the packets in the flow. However, the iTrace packets are generated with a very low probability by routers to reduce the additional traffic, which undermines the effectiveness of the scheme. Moreover, large number of ICMP packets is required to construct the attack graph. No means of authentication of source of ICMP messages is described. To prevent attackers from spoofing the ICMP packets, an authentication field is used in the iTrace packet. This scheme is later improved by Wu *et al.* [140].

The second category of non – deterministic IP Traceback is hash based IP Traceback. In [141, 142], hash based IP traceback schemes are proposed to trace even single-packet

attacks, such as those which exploit vulnerabilities in the packet processing of TCP/IP stack implementations. This system, which the authors name SPIE ("Source Path Isolation Engine"), supports tracing by storing a few bits of unique information, essentially packet digests, for a period of time as the packets traverse the Internet. In this proposal, routers keep a record of every packet passing through the router. In order to minimize the storage needs and memory requirements for the digest tables, they use Bloom filters [76], which are space-efficient data structures with independent uniform hash functions. By sending traceback query for a packet to upstream routers, packet origin can be located. The software implementations of SPIE perform adequately for slow to medium speed routers. To achieve better results for faster routers, in [143] a hardware implementation as a processing unit inserted into the router, or as stand-alone connected to the router through an external interface, is described. This scheme is arguably the most effective scheme to traceback DDoS attacks. However, the success of traceback depends on the number of tracking routers installed, and the area covered by these routers. There is a huge storage overhead for a router to implement this scheme. The implementation cost and computation overhead of this approach are also high.

In Deterministic Packet Marking (DPM) [132] , only the edge routers participate in the marking procedure. DPM tries to construct the ingress address of the router closest to the source by fragmenting the IP address and sending it in two packets. The ID field is used to carry one half of an IP address and the RF bit is used to denote whether it is the first half or the second half of the IP address. In the initial proposal, the source address in the IP packet was used to construct the IP address of the router. This has a very serious consequence when the attacker uses many randomly generated source addresses. Thus the scheme was modified by also sending a hash of the IP address along with the fragmented IP address to aid the victim in the reassembly procedure. Since a hash is also sent along, the numbers of fragments now increase. Constructing an IP address from fragments would require trying out all the possible permutations. Not only would this require much processing overhead, but would also result in a high number of false positives while assembling the fragments, especially in the case of a DDoS attack where multiple fragments from multiple attackers are collected.

Besides the above packet marking schemes, another way for IP traceback is based on link testing. It includes input debugging [111, 112] and controlled flooding [113] schemes as described in Section 2.3.4. Sager [144] suggested logging packets at key routers and then using data mining techniques to determine the paths that the packets traversed. This

scheme has an advantage that it can trace an attack long after the attack has completed. However, it also has obvious drawbacks, including potential enormous resource requirements and a large scale inter provider database integration problem.

Other techniques for IP traceback include Pi [145] , StackPi [147], DERM [148], IP traceback with IPSec [149], Controller Agent method [131, 150, 151] and scalable multicast based method [152]. Pi, proposed by Yaar *et al.* [145] is a victim-based defense, building on previous packet-marking techniques, that inserts path identifiers into unused (or underused) portions of the IP packet header. The main idea is that these path identifiers or fingerprints are inserted by the routers along the network path. The target or victim would then reject packets with path identifiers matching those packets that have been clearly identified as part of an attack.

Pi [145] claims to work after the first attack packet has been identified (if it can be identified by the target), and work without inter-ISP cooperation, and with minimal deployment. StackPi [147] is based on Pi and introduces new marking schemes and filtering mechanisms that substantially improve Pi's incremental deployment performance. These schemes, along with other packet marking schemes, require significant changes to routers throughout the Internet for its implementation and successful deployment. Deterministic Edge Route Marking (DERM) [148] does not provide the marks of the actual zombies. The victim actually gets the hashmarks of the edge routers, to which the reflectors involved in the attack are connected. In DERM [148], it is assumed that all ingress routers and only ingress routers participate in packet marking. The scheme is to insert a hash of the IP address of the edge router in the IP header. The victim is able to map the hash to a list of ingress routers. On being informed of an attack packet by the Intrusion Detection System, the victim is able to identify the attacker. DECIDUOUS [149] is built on top of the IETF's IPSEC infrastructure, and it does not introduce any new network protocol for source identification in a single administrative domain. It defines a collaborative protocol for inter-domain attack source identification.

Tupakula and Varadharajan [150] propose an agent-controller model to counteract DoS attacks within one ISP domain which they later extended to multiple domains [151]. It produces an intermediate network reaction. In this model, agents represent the edge routers and controllers represent trusted entities owned by the ISP. Once a target detects an attack, it sends a request to the controller, asking all agents to mark all packets to the target. After checking the marking field, the target can send out which agent (edge router) is the entry point for the attack traffic. The target then sends a refined request to the

controller, asking some particular agents to filter attack traffic according to the attack signature provided by the target. It uses third party detection for detecting and characterizing attack traffic. The aim of the controller-agent model is to filter attack traffic at the edge routers of one ISP domain. The attack signature should be as narrow as possible to lessen collateral damage. It is doubtful whether the attack reaction is quick enough to curtail the attack

Lee *et al.* [152] propose a technique to trace by employing multiple trace nodes in simulated network. Once detection system or victim detect an attack and characterize the attack signature, the same is sent to these trace nodes by multicasting in secure manner. These trace nodes try to identify attack signatures in their on going flows. If they find packets matching attack signatures, they send a traceroute request to victim. After getting the response, whole response is sent back to victim for identifying attack graph. Sending traceroute requests in a congested network near victim may or may not get response. Moreover secure communication between trace nodes and victim or detection system in terms of confidentiality, authentication and integrity at lower overheads is a big question

Besides Filtering and IP Traceback, pushback schemes [80, 153] provides link based defense against DDoS. As discussed in Section 2.3.4, a pushback message can be sent to a router upstream. The message is sent to the routers that are forwarding the traffic that is supposedly from the attacker. The purpose of this message is to invoke the ACC mechanism at the router upstream. Mahajan *et al.* [78] propose to recursively pushback control signal to the network which contains description of the traffic aggregate which causes congestion in the network. Ideally, these pushback messages should propagate as far upstream as possible to the point where the traffic enters the network so that rate-limiting can be performed without affecting the rest of the traffic in the network. However, they propose to let the downstream router send pushback request to all upstream neighbors. It requires changes to many routers throughout the Internet and additional processing increases the reaction time. Moreover, the scheme itself is not always perfect in correctly identifying attackers.

Unlike previous approaches, selective pushback [153] obtains source information by probabilistic packet marking and directs the pushback packet directly to routers near the source of attack. This reduces computing overheads for routers and reduces time for relevant routers to receive control signal. By filtering packets using source information, malicious traffic are filtered while protecting legitimate traffic.

Controlling high bandwidth aggregates in network [78] using aggregate based congestion control schemes [77, 79] and complementing it with pushback [80, 154, 155] has been used to mitigate the effects of DDoS attacks.

Special routing techniques to mitigate the DDoS attacks include overlay networks [116, 156]. Secure Overlay Services (SOS) [116] (discussed in Section 2.3.4) requires large numbers of overlay nodes are required to make the system resilient to DoS attacks. Also, SOS offers no protection from insider attacks. Centertrack [111] and Mayday [117] are similar overlay networks used for defense against DDoS. [156] is an application-level protocol that coordinates detection and responses of multiple intrusion detection systems. IDIP nodes are organized into neighborhoods and communities. Coordinated detection, attack tracing and response within a community are managed by a component called a Discovery Coordinator. IDIP assumes contiguous deployment of neighborhoods that share information, and thus its ability to suppress attacks is limited in a partial deployment scenario.

Router control plays a predominant role in defending DDoS attack. There are two types of algorithms for router control, scheduling and queue management.

Scheduling algorithms aim to maximize resource utilization [157]. Scheduling algorithms [158-160] can ensure the fairness between the traffic flows but they are too expensive in terms of delays, state monitoring and checking for broad deployment and do not scale well to large number of users. Moreover, large number of slow rate DDoS traffic flows can still prove lethal at victims.

Simulations were done for various queue management techniques like Random Early Detect (RED) and its variants [77, 161-165]. However, bandwidth available for the legitimate user is very small even for the best rated RED which is itself a certain kind of denial of service. Furthermore, the topology, traffic generation models and applications used in simulation are very simple compared to realistic network topology.

Global cooperation is a must in routers of multiple domains so as to implement preference based approaches like Integrated Services [166] and Differentiated Services [167] and labeling based scheme proposed in [168].

Wang *et al.* [169] propose that all types of packets do not deserve same treatment as done in best-effort model and even in packets of same class under DiffServ [129]. So packets can be differentiated into various bandwidth aggregates. They propose layer 4 service differentiation and resource isolation.

2.4 Research Gaps

Preventing DDoS attack to curb their devastating effect is the first choice of both commercial and research organizations. The three precautions are (i) install ACLs, firewalls and IDS to prevent from being compromised, (ii) install ingress filters and stop IP address spoofing, and (iii) repair security holes for well known software and protocol bugs. Table 2.2 shows the comparison of various prevention schemes.

If all the schemes mentioned above can be implemented effectively, the Internet would be much relieved from DDoS attacks. But the approaches to stopping intrusions [40-43, 45, 46], filtering malicious IP addresses [51, 52, 54, 75] and repairing security holes by patches [55] have lot of hurdles in terms of global deployment, installation of patches as soon as they are developed and released, overheads to check extra packet headers, updating IDS database with new attack signatures and high rate of false positives and negatives.

The next approach to deal with DDoS attacks is detection and characterization. Table 2.3 gives a comparison of various schemes for detection and characterization. Signature based schemes [40-43, 45, 46] are usually applied as prevention techniques and are limited to detecting known attacks. Due to the broad signatures used in congestion based schemes [77-80, 154, 162], punishing the aggregates causing congestion causes severe collateral damage as good traffic in the aggregates is also dropped. The most common used DDoS detection and characterization schemes are anomaly based [65-75]. In almost all of these schemes, the common challenge is to provide all types of normal traffic behavior. As a result, legitimate traffic can be classified as an attack traffic leading to high false positive rate. To minimize false positive rate, a larger number of parameters are used to provide more accurate normal profiles. However, with the increase in number of parameters, the computational overhead to detect attack increases. This becomes a bottleneck, especially for volume oriented DDoS attacks that are aggravated by computational overhead of detection scheme. Unlike other attacks which are constrained to sending traffic that exploits a special vulnerability, DDoS attackers can mimic legitimate traffic to avoid anomaly- based detection. As depicted in Table 2.3, since most of the schemes are volume based [65-69], they can detect the attack only near the victim when the attack has already consumed significant network resources on its path. Also, volume based techniques are unable to distinguish between DDoS attacks and flash crowds. Other schemes that have been proposed are applicable to either specific attack type or specific network scenario and have limited scope.

Table 2.2. Comparison of prevention schemes

Scheme	Rule or Anomaly Based	Basis of defense	Location	False Positives	False Negatives	Collateral Damage	Effective against IP spoofing	Global Cooperation	Ease of Deployment /Configuration	Overheads / Disadvantages
Access Control Lists[40]	Rule based	Header	Victim	Yes	Yes	_	_	_	Easy / Easy	Administrative and Management Overhead
Firewalls[42]	Rule based	Header	Victim	Yes	Yes	_	_	_	Easy / Easy	Configuration issues, can be bypassed
Intrusion Detection Systems[43]	Rule (Signature) based	Header	Victim	Yes	Yes	_	_	_	Easy / Difficult	Detect only known attacks; updating signatures..
Ingress Filtering[51]	Rule based	Header	Victim	No	Yes	No	No	Required	Easy to deploy	Computational overhead
Egress Filtering[51]	Rule based	Header	Source	No	Yes	No	No	Required	Easy to deploy	Computational overhead
Route-based Packet Filtering[52]	Rule based	Header	Intermediate border routers of AS	Yes	Yes	Yes	No	Required	Difficult to deploy	Processing due to large packet size, ineffective in dynamic network environment
Source Address Validity Enforcement[54]	Rule based	Header	Intermediate Routers	Yes	Yes	Yes	No	Required	Difficult to deploy	Time for routing table updates
History Based IP Filtering[75]	Anomaly based	Header	Source	Yes	Yes	Yes	No	Not Required	Difficult to deploy	Ineffective if window size known

Table 2.3. Comparison of various detection and characterization schemes

Scheme	Rule or Anomaly	Basis of defense	Location	False Positives	False Negatives	Effective against spoofing	Ease of Deployment	Overheads / Disadvantages
MULTOPS[65]	Anomaly based	Volume, IP protocol type, packet rate	Victim	Yes	Yes	No	Easy	Crippled by IP spoofing
SYN detection[66]	Statistical anomaly based	Volume, packet type, source IP address (header)	Victim	Yes	Yes	No	Easy	Only for TCP SYN attacks
Batch detection[67]	Anomaly based	Volume and header	Victim	Yes	Yes	No	Easy	Limited Scope
Spectral Analysis[68]	Anomaly based	Volume, packet rate	Victim	Yes	Yes	No	Easy	Only for TCP SYN attacks
Kolmogorov Test[69]	Anomaly based	Volume , payload, packet size packet rate	Victim	Yes	Yes	No	Easy	Detection can be bypassed
Time series analysis[70]	Correlation among packets	Volume , payload, packet size packet rate, header	Victim	Yes	Yes	No	Easy	Based on training, cannot detect new attacks
Bencsath et al. [71]	Anomaly based	Traffic level measurement e.g. buffer length	Intermediate network or victim	No	Yes	No	Medium	Can handle special cases of DDoS attacks
Mining anomalies[74]	Anomaly based	Traffic feature distribution	Intermediate network or victim	Limited	Limited	Yes	Medium	Early detection is not possible
ACC and Pushback[154]	Congestion signature based	Volume, header (destIP prefix, transmission rate) and packet drop history	Victim to intermediate network	Yes	Yes	No	Difficult	Aggregates may contain good traffic, high collateral damage
DWARD [88, 89]	Anomaly based	Connection thresholds on IP protocol type, packet rate, source IP and destination IP	Source end	Yes	Yes	No	Difficult	Enough attack data not present at source for accurate detection

Response against DDoS attacks is based either on filtering or redirecting the bad traffic to controlled part of the network. Filtering is based either on classification rules [52, 54, 55, 75] which are difficult to generate in dynamic Internet environment, link testing schemes [111-113] like input debugging and controlled flooding which may themselves become cause of DoS due to extra flooding in already congested path, and aggregate based congestion control and pushback schemes [77-80]. Most of the filtering schemes cause high collateral damage as good traffic is also punished or filtered along with the attack traffic.

Last but mostly used approach is mitigation. It assumes that because of limitations of prevention, detection and characterization, and response, it is almost impossible to defend against DDoS without false positives, false negatives and collateral damage. Node based mitigation techniques like dynamic resource pricing [98-100, 121], resource accounting [120] and QoS regulation [122, 171, 187] based solutions available so far result in high delays because of scheduling and queuing approaches to handle traffic. Moreover, slow rate attacks where a large number of attackers consume lot of bandwidth have no proper answer available so far. Some of these solutions require special client software to be installed. Client based program are also required to be loaded for proactive server roaming that have hampered their popularity. Due to limited attack scenario in terms of topology, number of attackers etc. its performance still needs to be evaluated. Overprovisioning [57] alone is not sufficient solution in highly attacked environment and has high costs associated for seldom and low attacked networks.

In link testing schemes, tracing is one of the best strategies to curb the menace of DDoS attacks. In all the Traceback solutions overheads like permission and extra bandwidth for link testing schemes like input debugging and controlled flooding [113], extra resources for overlay networks [111], ICMP messages [138, 140, 146] and IP packet marking overheads [39, 133-136, 141] are involved. Moreover security of this communication [188] so that these control messages should not be forged is a big hurdle to tackle. Overall research direction in this field has been limited mostly to finding zombies and path characterization up to zombies. At the moment Traceback in combination with tolerance and mitigation is popular methodology to defend DDoS attacks [151, 152]. Some algorithms are available for detecting high bandwidth aggregates based on destination address [77-79] complemented with pushback [80, 154, 155]. However, they use broad congestion signatures. They require finding source characteristics to narrow down congestion signatures to reduce collateral damage.

Degrading attacks which do not cause congestion at links cannot be grouped in any congestion aggregates without high false positives and negatives. Isotropic slow rate attacks which even cause congestion at links are not identified. Though fair queuing [158, 160], stochastic fair queuing [159], QoS based techniques [129, 167] and identifying thinner bandwidth aggregates [169] and then regulation are good solution, they require excessive state monitoring, calculating proper rate limits and testing for defaulters cause appreciable overheads. Applying dynamic rate limits as per legitimate traffic models are main challenges.

In summary, each DDoS approach has its strength and weakness. None of countermeasures so far provide exhaustive solution that can stop DDoS attacks immediately and efficiently. Combining the strength of different approaches and let them compensate each other's weakness is a viable solution to DDoS problem. This requires a framework that should be modeled to illustrate the shared features of the approaches.

2.5 Honeypots

Honeypot [176] is an important security technology that is used to understand and defend against DoS attacks. A honeypot is a resource whose value lies in unauthorized use of that resource [174]. The strategy behind honeypots is to isolate the intruders from production systems and to obtain information about the intruders by logging their actions [175]. Honeypot is used as an instrument for information gathering. Honeypots do not replace firewalls and IDS but are used in conjunction with them.

A honeypot is set up on a network for the sole purpose of being attacked. It is designed with deliberate vulnerabilities, which is exposed to a public network. The goal is first to lure intruders away from the real system, and secondly to closely monitor the intruder to study the exploits used to launch attacks. Honeypots are not supposed to receive any legitimate traffic and thus, any traffic destined to a honeypot is most probably an ongoing attack and can be analyzed to reveal vulnerabilities targeted by attackers. Coupled with an Intrusion Detection System (IDS), honeypots are effective in detecting victim hosts.

2.5.1 Classification of Honeypots

As shown in Figure 2.9, honeypots are broadly classified via two methods [176]

(i) Based on usage and, (ii) based on level of interaction.

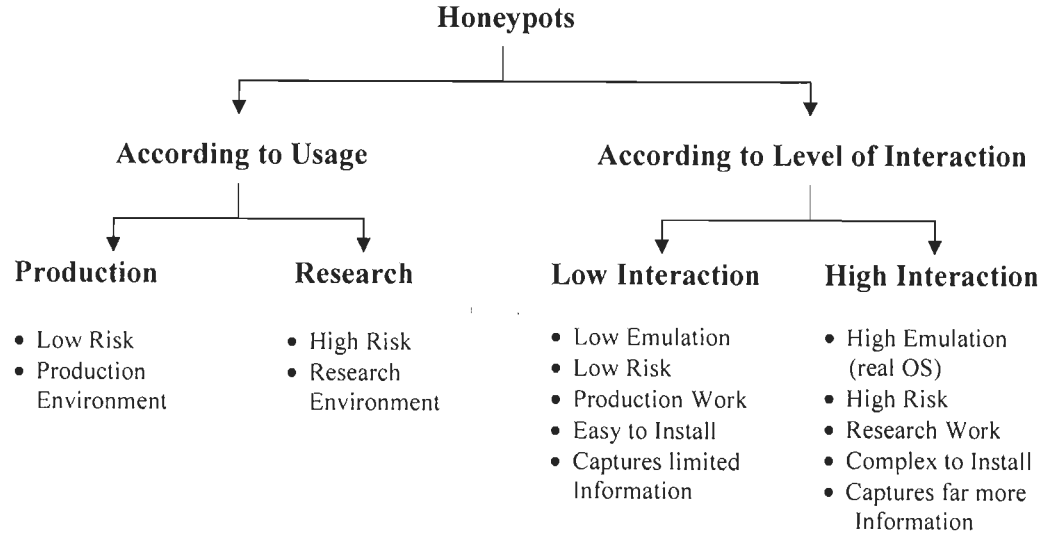


Figure 2.9. Classification of honeypots

Based on usage, honeypots can be production honeypots or research honeypots. Production honeypots are used to protect organizations. They add value to security measures of an organization. Research honeypots are used to research the threats organizations face, and how to better protect against those threats. They are more focused on researching the actions of intruder by using a number of different configurations to lure them in. The Honeynet Project [176], for example is a volunteer, non-profit security research organization that uses honeypots to collect information on cyber threats.

Based on level of interaction, honeypot systems can be implemented as low interaction and high-interaction systems [176]. Low Interaction Honeypots are characterized by its minimal interaction with the attacker. A low interaction honeypot involves a simple emulation of an operating system and network services like ftp or http. They are much simpler to deploy and maintain, but log only a limited amount of information regarding the attacker's activities. High Interaction honeypot gives a real operating system to attack upon. It is implemented as a real operating system and with services running on a real machine. This exposes the system to risk and complexity. At the same time the possibility to accumulate information about the attack as well as the attractiveness of the honeypot increases a lot, so they are specially used for research purposes.

2.5.2 Placement of Honeypots

Honeypots can be placed externally as well as internally [175] in an organization. Conceptually they can be placed at three main locations as shown in Figure 2.10.

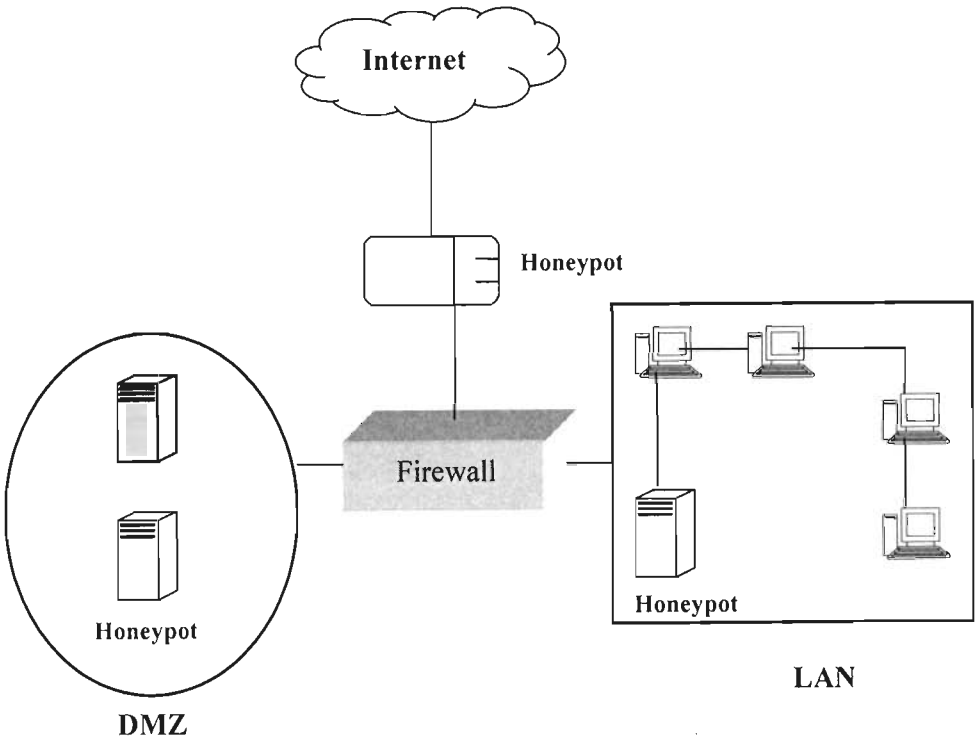


Figure 2.10. Placement of honeypots

Placing honeypots outside the firewall simulates a system without any firewall or Intrusion Detection System (IDS). The risk to the internal network reduces but it limits their ability to emulate the production systems and generate logs, which are relevant to the internal network. Being inside the internal network, they can emulate the production systems as well as can monitor the attacks made from inside the network. They give proper logs of all the activities and can be easily integrated with other security technologies to get the best output. Being inside the network they introduce some risks to the organization. By placing them inside the DMZ (De militarized zone), they can easily emulate the servers that are freely accessible to the public domains. This tracks the intention of attackers. It also increases the security of the production environment.

2.5.3 Current Honeypot Technology

Honeypots are investigated on parameters like security value, interaction and virtualization. Backofficer Friendly (BOF) is a lightweight honeypot and free to distribute; Specter is a commercial production and low interaction honeypot whose value lies in detection; Honeyd is an open source powerful production honeypot which can be used for

attack detection and reaction; Honeynet is the highest level of research honeypot which can be modified to production honeypot for attack detection and reaction. Honeypots at application level have been proposed in literature to detect targeted attacks [186] and for enhancing intrusion detection systems [189].

Honeypot technology has both advantages and disadvantages that are listed below.

Advantages

1. Honeypots are simple to deploy.
2. Unlike intrusion detection systems, which only identify when and how an attacker got into the network, a honeypot is a distraction as well.
3. Honeypots generate small data sets of high value and reduce noise. This means it is much easier to analyze the data a honeypot collects and derive value from it. Honeypots provide a lot of useful information on attacker's movement within the system.
4. A honeypot are used as an early warning system, alerting administrators of any suspect before the real system is attacked.
5. Honeypots require minimal resources.
6. Honeypots can collect in-depth information that no other technologies can match.

Disadvantages

1. Honeypots are worthless if no one attacks them.
2. Honeypots can be used as a launching platform to attack other machines.
3. Honeypots encourage an aggressive atmosphere and add risk to a network.
4. If the honeypot is attacked, the administrator should prevent major changes to the honeypot, because any noticeable changes will make the attacker more suspicious.

Role of honeypots in various phases of attacks

Honeypots can *prevent* automated attacks (targets of opportunity) that are based on tools that randomly scan looking for vulnerable systems. Honeypots monitor unused IP address space and when probed by such scanning activity, honeypots interact with and slow the attackers. Honeypots can also be used to prevent non automated attacks (targets of choice) by deception or deterrence.

Honeypots can protect an organization through *detection* of attacks. Detection has traditionally proven to be extremely difficult activity. IDS have proven ineffective for several reasons: They generate far too much data and large percentage of false positives

and are unable to detect on new attacks. Honeypots address these detection problems by reducing false positives by capturing small datasets of high value and capturing unknown attacks such as new exploits etc.

Honeypots can be used to generate *response* against attacks. They can be taken offline for analysis without impacting day-to-day operations. Any data retrieved from honeypot is most likely related to the attacker. Thus they give in depth information that is needed to rapidly and effectively respond to an incident.

Key Issues and Challenges

Honeypots have proven to be extremely successful and minimize false positives. Potential applications of honeypots include commercial enterprises, government agencies, surveillance and communication in battlefield and in production and research. Honeypots are a useful tool, however there are some limitations associated with their use. One problem is the issue of accountability. Intentionally placing an insecure machine on a network and allowing it to get compromised can be of high risk to an organization because the attacker can use that system to launch attacks on other organizations' or individuals' systems. Therefore, limitations have to be placed on honeypots' out-bound network connections in order to prevent abuse at the hands of attackers. Honeypots are hard to maintain and they need operators with in depth knowledge about computer and network security.

2.6 Conclusions

In this chapter we have given the critical review of various key techniques existing so far for defense against DDoS attacks. Already work done in DDoS defense has been concentrated either individually on prevention, detection, characterization, response and mitigation or in groups like detection and characterization with filtering, mitigation by tracing with filtering or rate limiting. The gaps in the existing work have been identified and highlighted. Efforts will be made to address some of these gaps as part of our work.

Chapter 3

Design of the Honeypot Framework

3.1 Introduction

Various DDoS attacks against high profile websites such as Yahoo, CNN, Amazon, E Trade, GRC.com etc. demonstrated devastating nature of DDoS attacks and the defenseless nature of the Internet under these attacks. The limitations existing with current approaches and motivation behind our proposed honeypot framework [190] are enlisted below:

i. Though an array of schemes have been proposed in last decade for defending against DDoS attacks using one or more approaches like prevention, detection, characterization, response and mitigation, there is still a dearth for an *integrated, dynamic and autonomous* framework that encompasses multiple stages in the process of defense. By *"integrated"*, we mean a solution that caters to all the phases of defense. *"Dynamic"* and *"autonomous"* refers to a solution that is adaptive and responds independently to dynamically changing network conditions and user requirements.

ii. Most of the point technologies like firewalls [41] and perimeter solutions like IDS [43] are unable to withstand the advancing attack techniques. It is required to move away from point and perimeter solutions to an integrated operational model which has all the capabilities to anticipate attacks, detect the point in time when attacks are in progress, identify attack flows, respond to DDoS attacks, mitigate the attacks and collect attack data for further investigation.

iii. Prevention techniques like firewalls [41] and IDS [43] use fortress mentality. However, defense should balance detection, characterization, response and mitigation. Prevention is a very static way for protection whereas detection, response and mitigation are dynamic and adaptive to situation and hence provide better overall protection. If the time the protection mechanisms can withstand an attack exceeds the

time that it takes to effectively detect, respond to and mitigate the attacks, then the network is secure.

iv. Existing detection tools detect only known attacks and result in high rate of false positives and negatives. Unknown attacks cannot be identified and restricted. Mitigation starts after the attack has occurred which leads to consumption of resources at victim end, leaving services unavailable for the legitimate users. There should be a proper response mechanism in place so that in case an attacker gets through, network should immediately and effectively respond, in order to maintain stable functionality. Moreover only the biggest attacks are fully investigated, the smaller attacks that gracefully degrade network performance remain unattended.

3.2 An Overview of the Proposed Framework

The framework proposed in this thesis contributes in the multiple phases of DDoS defense as follows:

i. Real time *detection* of variable rate DDoS attack, minimizing false positives (FP) and false negatives (FN).
ii. Accurate *characterization* of the traffic as attack or legitimate.
iii. *Accurate filtering* and *response* to attacks and suspects, minimizing collateral damage.
iv. Efficient *mitigation* of the effect of attack using autonomous dynamic honeypot redirection and thus maintaining stable network functionality in attacked network.
v. Optimum resource utilization by d*ynamic resource allocation* depending on the network conditions to provide guaranteed Quality of Service (QoS).

Figure 3.1 shows the architecture of the proposed framework.

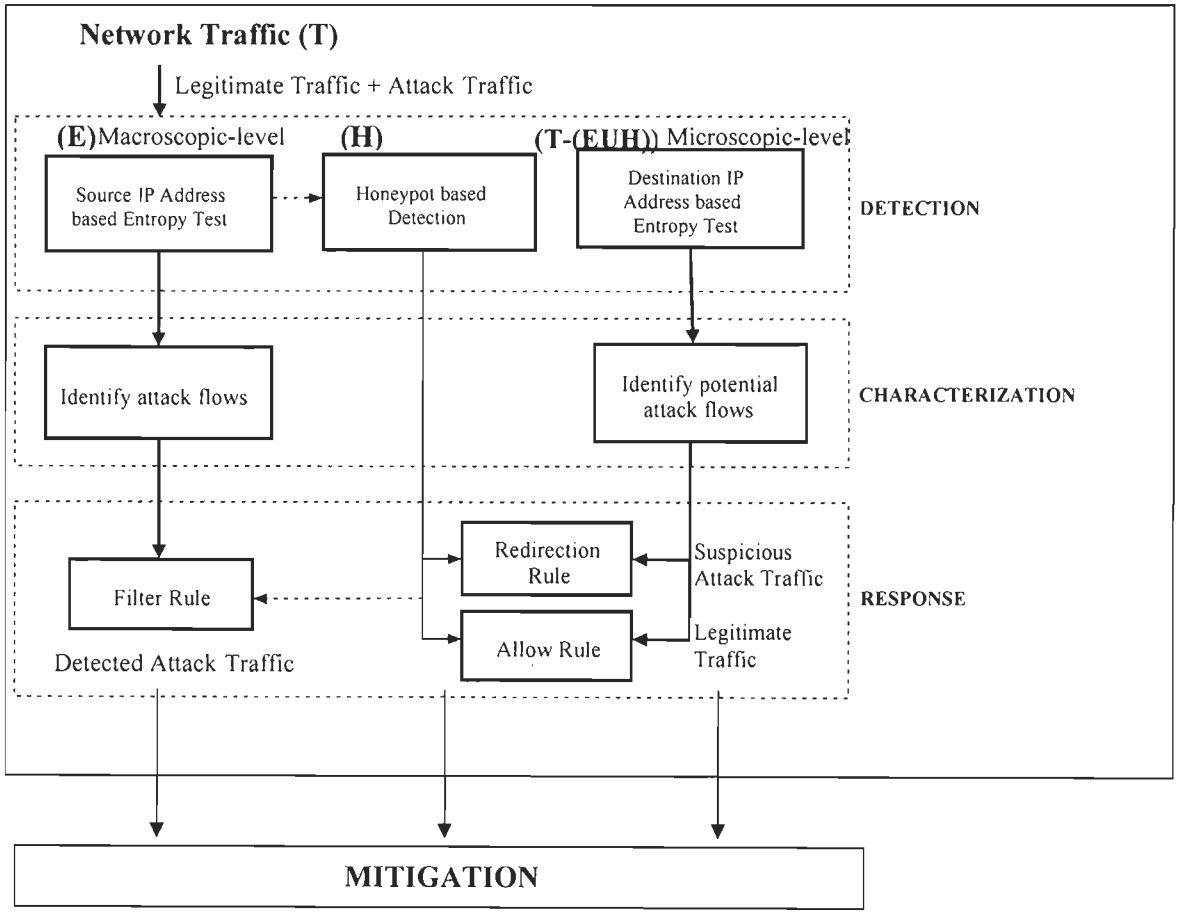


Figure 3.1. Architecture of the proposed framework

Our framework [190] works on three lines of defense, namely, detection; characterization and response; and mitigation. The work presented in this thesis is an effort to propose new and efficient techniques for each of these lines of defense. We propose use of honeypots [174-176] that appears to be part of a network but which is actually isolated, (un)protected, and monitored, and which seems to contain information or a resource that imitates actual server and would be of value to attackers. Honeypots reside in the same network as active servers which are to be protected. Our aim to defend the DDoS is to prevent the attack flow reach the target and ensure its availability. Our framework does not replace the existing technologies like firewalls [41] and IDS [43] but is used in conjuction with them to defend the attacks.

As shown in Figure 3.1, detection is first module that interacts with attack as well as legitimate traffic. The detectors after finding signs of attack try to characterize the attack flows. Once characterization is done, an appropriate response is triggered according to the

response rules. The strength of attack detection and characterization from time to time depends upon the arrival rate of traffic and amount of attack traffic. This decides appropriate values of threshold limits to be applied for filtering and server resource management.

Honeypots as preventive measure against DDoS

Though our proposed framework does not specifically aim at preventive measures, prevention of attacks is inherent to honeypots [175]. Honeypots prevent attacks by monitoring unused IP address space. For e.g., flows destined to non-existent IP address are identified as attacks and directed to honeypots. In our proposed framework, attacks are prevented in the presence of honeypots using deception and deterrence. We propose to set up honeypots in a manner similar to active servers and load it with numerous real files and directories. By making the honeypot information appear to be legitimate with legitimate files, it leads the attacker to believe that they have gained access to important information. The attackers are deceived and waste their time and resources attacking honeypots as opposed to attacking active servers. Therefore active servers remain isolated from attacks. Also, in our proposed framework, if the attackers know about the presence of honeypots, they do not know which systems are honeypots and which are active servers. They may be so concerned about being caught by honeypots that they decide not to attack. Thus, honeypots deter attackers.

3.2.1 Lines of Defense

3.2.1.1 Detection

We present a novel hybrid detection mechanism that combines the best features of honeypots and statistical anomaly detection. During the attack detection, the incoming network traffic undergoes various tests for anomaly detection as well as honeypot based detection (shown in Figure 3.1) at different points on their path from source to destination.

IDS like SNORT [45] and [173] have lately been popular mechanisms to detect and respond to attacks. Since such systems are rule based, they are limited to detecting already known attacks. In contrast to IDS, honeypots and anomaly detection system (ADS) offer the possibility of detecting previously unknown attacks, also referred to as zero-day attacks. Honeypots are machines that are not supposed to receive any legitimate traffic [174] and since, any traffic destined to a honeypot is most probably an attack, they act as a proactive detection mechanism.

Honeypots and ADS offer different tradeoffs between accuracy, scope of attacks that can be detected, and collateral damage, as shown in Figure 3.2. Honeypots have high accuracy but limited scope. Honeypots can be heavily customized to accurately detect attacks, but detection depends on an attacker attempting to exploit vulnerability against them. This makes them good for detecting only certain specific kind of attacks and therefore they have a narrow scope. Furthermore, honeypots can typically only be used for server-type applications.

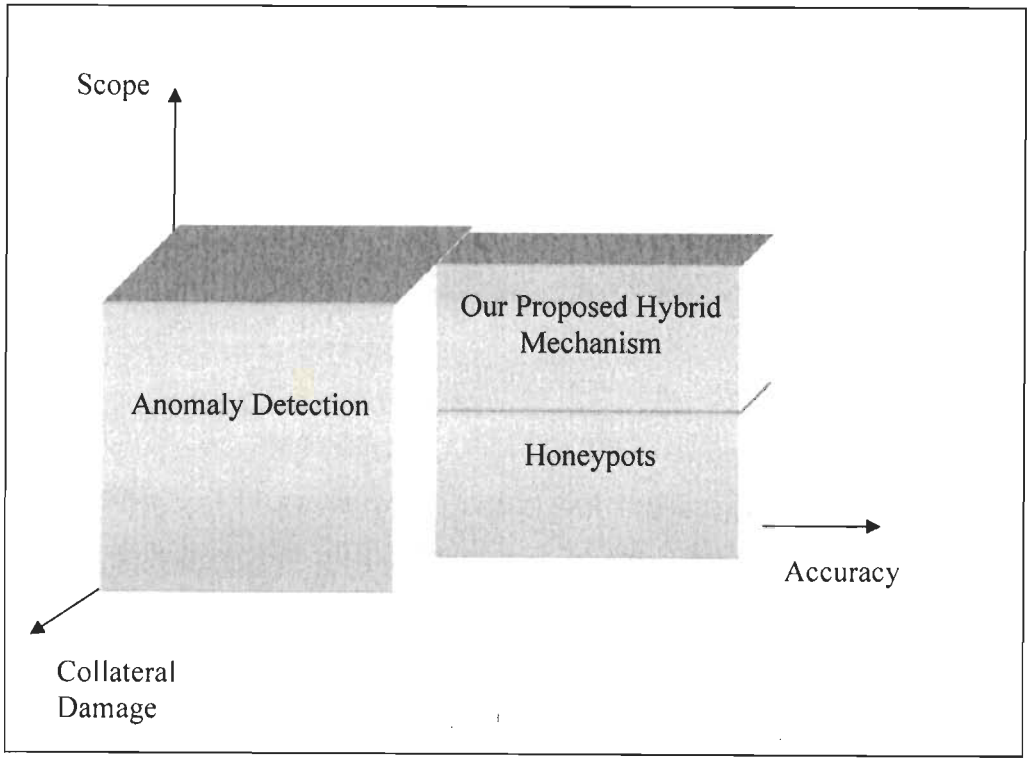


Figure 3.2. Trade-off in scope, accuracy and collateral damage

ADS have high scope but low accuracy. ADS can theoretically detect both automated as well as non-automated attacks, but are much less accurate. They offer a tradeoff between FP and FN rates. For example, it is often possible to tune the ADS to detect more potential attacks, at an increased risk of misclassifying legitimate traffic (low FN, high FP); alternatively, it is possible to make an ADS more insensitive to attacks, at the risk of missing some real attacks (high FN, low FP). In case of FP, ADS causes "collateral damage" by impairing, deactivating or disrupting legitimate users to access the services. Also, because an ADS-based intrusion prevent system (IPS) can adversely affect legitimate traffic and cause collateral damage (e.g. drop a legitimate request), systems are often tuned for low FP rates, potentially misclassifying attacks as legitimate.

Our proposed detection mechanism is based on detecting attacks at dual level. At a higher level or macroscopic-level, we use anomaly detectors to monitor all incoming traffic to a protected network and service. If an attack is detected, we characterize and filter out those attack flows that, if permitted into the network, may cause severe network performance degradation. However, the attackers may simulate the normal network behaviors, e.g. pumping the attack packets as Poisson distribution, to disable anomaly detection algorithms at high level. Low level or microscopic-level anomaly detectors along with instrumented honeypots are used to detect these potential attacks. The traffic that fails to pass low level anomaly detection tests is processed by honeypots to determine the accuracy of the anomaly prediction before dropping it.

Honeypot is capable of suppressing FP and FN. A honeypot has no real purpose, other than to capture attack [175]. So honeypot reduces false alarms by not capturing any normal traffic. Moreover, normal traffic or flash crowd initially misclassified by anomaly detector will be subsequently validated by the honeypot and handled correctly by the active server transparently to the end user, though with some latency. FN are another challenge. The honeypot reduces FN by capturing absolutely everything that is destined to it. This means all the activity that is captured is most likely suspect. Moreover an attack that remains undetected by anomaly detection tests at high level but exploits the vulnerabilities is captured by honeypots. Therefore, unknown activity, even if ADS misses it, is captured by the honeypot. Also, traffic identified as suspicious by low level anomaly detectors is redirected to honeypots before being processed by active servers. Active server remains isolated from potential attacks and hence reduce FN.

We combine filtering, anomaly detection and honeypots in a way that exploits the best features of these mechanisms while shielding their limitations. Our proposed mechanism increases scope and accuracy of detection and minimizes collateral damage.

3.2.1.2 Characterization

As soon as the attacks are detected, characterization is triggered. Similar to detection, characterization is performed at two levels, namely macroscopic-level and microscopic-level, to identify the attacks and suspicious flows respectively (shown in Figure 3.1). During characterization, out of all the incoming flows, some flows are identified with confidence as attack flows, and some are identified as suspicious flows that might belong to potential attacks. Flows characterized neither as attacks nor as suspicious are normal flows.

3.2.1.3 Response

As a response to the characterization, we propose three rules, namely filter rule, redirect rule and allow rule. "Filter" rule blocks the flows which are characterized with high confidence as attacks, "redirect" rule redirects the suspected attack flows to honeypots and "allow" rule allows the normal legitimate flows to access the active servers.

3.2.1.4 Mitigation

In our proposed framework, we deploy honeypots along with active servers. The honeypots and active servers are dynamic and roaming. The number and IP addresses of honeypots and active servers change after variable time intervals, therefore the attack connections are weeded off and this helps to mitigate the effect of the attack flows.

To optimize the detection strength and resource utilization, our proposed framework can be tuned to operate in one of the three modes of defense, namely naïve, normal and best mode, depending on client load (*CL*), attack load (*AL*) and client requirements.

3.2.2 Features of the Proposed Framework

Effectiveness: The presence of honeypots provides prevention from attacks as they deceive and deter the attackers. In case the attacks get through, our framework ensures effective response and mitigation ensuring that the DDoS effect goes away. The response is quick enough and ensures that the normal traffic is isolated from attack and victim does not suffer seriously from the attack.

Completeness: The presence of both anomaly detectors as well as honeypots in our framework makes it capable of handling many possible attacks. It has a high degree of

perfection as it can handle a large number of attacks like macroscopic attacks (concentrated high rate attacks and highly distributed low rate attacks) that cause immediate congestion in the network, microscopic attacks (distributed low rate attacks) that cause graceful degradation of the network, and stealth and sophisticated attacks that otherwise go undetected.

Attackers develop new types of attacks that bypass existing defenses. However, our proposed framework targets the fundamental basis of DDoS attacks; it validates normal traffic with high accuracy and concentrates on generating enough active servers that can service the normal packets. Moreover, it filters out congestion inducing flows before they enter the network, and is based on delivering only as many packets as the target network can handle. This reduces the FP and collateral damage. Finally, a fairly complete defense has been built by combining specific defenses.

Provide service to all legitimate traffic: The goal of our DDoS defense framework is not only to stop DDoS attack packets, but to ensure that the legitimate users can continue to perform their normal activities despite the presence of a DDoS attack.

Some legitimate traffic may be flowing from sites that are sending attack traffic. Other legitimate traffic is destined for nodes on the same network as the victim node. There may be legitimate traffic that is neither coming from an attacker site nor being delivered to the victim network, but shares some portion of its path through the Internet with some of the attack traffic. Some legitimate traffic may share other characteristics with the attack traffic, such as application protocol or destination port, potentially making it difficult to distinguish between them. These legitimate traffic categories should not be disturbed by the DDoS defense mechanism. Since DDoS attackers conceal their attack traffic in the legitimate traffic stream, it is common for legitimate traffic to closely resemble the attack packets. Since our characterization mechanism can identify attacks in all above cases, the collateral damage is minimized.

Our proposed framework filters congestion inducing attack packets before they enter the victim network. Hence it does not impede normal traffic and prevents collateral damage. For the suspicious traffic, instead of dropping them, it redirects them to honeypots. Legitimate traffic is isolated due to the presence of honeypots and directed to active servers. Suspicious traffic is further processed by honeypots before any action is taken on it. Lastly, our proposed framework dynamically generates enough active servers to service all legitimate clients, with the guaranteed QoS.

Low false-positive rates: In our proposed framework, the operation of detection, response and mitigation mechanisms depends on the client and attack load. False positives cause collateral damage. The various parameters and thresholds in the framework are dynamically optimized to minimize FP rates. The defense becomes more rigorous only when a DDoS attack is actually under way. The defense system is calibrated to strike a balance between detection rate and false alarm rate.

Low operational costs: Any defense mechanism uses some fraction of a system's resources and processing power, as well as induces some operational costs that relate to overheads imposed by the defense system like delay on packets etc. Unless such costs are extremely low or are rarely paid, they must be balanced against the benefits of achieving the degree of protection against DDoS attacks. A sufficiently frequent occurrence of attacks and hence the reaction demands more investment of resources. But if these costs are frequently paid when no attack is under way, then the costs of running the defense system may outweigh the benefits achieved except in those rare cases when an attack actually occurs.

The costs associated with our defense system are commensurate with the benefits provided by it. Our DDoS defense framework is fine tuned to optimize resource utilization according to network conditions and thus reduces the costs. It allows systems to continue servicing legitimate clients even when DDoS attack is underway, though at a slightly higher latency.

In the following chapters, we shall discuss in great depth the various facets of our proposed framework for defense against DDoS attacks. All the above strategies are integrated into a single autonomous framework and its performance is compared with existing standalone approaches. We use analytical and mathematical modeling along with extensive simulations for validation of the proof of concept and for measurement, analysis and comparison of network performance parameters.

The proposed framework provides a comprehensive solution for defense against DDoS attacks. It is centered on maintaining efficient network response while optimizing the network functionality, even in the presence of attacks through attack detection, attack characterization, attack mitigation and responding to attacks dynamically in real time. The functionality of each phase of defense is contingent on input derived from the previous phase. Each subsequent phase shields the limitations of the previous phase besides having its own advantages. Our framework is compliant with multiple phases of defense and

combines resource management techniques with network defense techniques to create a new hybrid defense.

3.3 Conclusions

The issues and challenges of current defense techniques under varied DDoS attacks require a well defined integrated solution to eliminate the problem of DDoS attacks. No work exists in literature where integration of all the approaches is available. Our proposed framework operates in various phases of DDoS defense process and combines attack detection and characterization with attack isolation and mitigation to recover networks from DDoS attacks. The proposed hybrid detection has high scope and accuracy with minimum collateral damage. Our proposed framework provides an effective and nearly complete solution to DDoS attacks with low FP rates and guaranteed QoS. The cost associated with our proposed framework is at par with the benefits it provides for defense against DDoS attacks.

Chapter 4

Dual-Level Attack Detection

4.1 Introduction

An ideal DDoS defense system should render any DDoS attack impossible. Apart from the fact that no prevention system is perfect, a proactive prevention approach requires too many resources to operate and is costly in the absence of attack. Therefore, attack detection is the first necessary element of a complete DDoS defense system. The DDoS attack detection problem consists of designating those points in time at which network is experiencing an attack. Only by timely detection of DDoS attacks, system can make proper response to escape big loss.

Many techniques have been suggested so far for DDoS attack detection [39, 45, 46, 65-69, 71-74, 77, 78, 80, 134, 150, 191]. Most of these techniques have certain limitations. The detection techniques based on monitoring the volume of traffic received by the victim [65, 67, 71] can detect a DDoS attack easily at the victim network. It is so because they can observe all the attack packets near the victim. However, attack packets clog a large part of the network before they are detected at the victim. On the other hand, volume based techniques that detect attack early in the network [39, 73, 77, 80, 134, 150] have to wait for the flooding to become widespread to be accurate, and consequently, they are ineffective to fence off the DDoS timely. Signature based schemes [45, 46] can detect only known attacks whereas anomaly based schemes [65-68, 71] give high FP rate. Many of the present DDoS attack detection techniques are complex, difficult to deploy or lead to computational and memory overheads [39, 45, 46, 65-69, 71-74, 77, 78, 80, 134, 150, 191]. A detailed discussion on major work on DDoS attack detection is given in Section 2.3.2.

A key challenge during DDoS attack detection is to detect attack traffic close to its source. This is particularly difficult when the attack is highly distributed, because the attack traffic from each source may be small compared to the normal background traffic. The second challenge is to detect the attack as soon as possible without raising a false

alarm, so that the victim has more time to take action against the attacker. Due to the inherently busty nature of Internet traffic, a sudden increase in legitimate traffic or a flash crowd may be mistaken as an attack. If we delay our response in order to ensure that the traffic increase is not just a transient burst, then we risk allowing the victim to be overwhelmed by a real attack. Therefore, we need to distinguish between attacks and flash crowds.

The detection schemes proposed in this chapter address the following challenges in attack detection:

i. Identify DDoS attack packets at early stage so as to eliminate congestion inducing attack packets before they reach the target.

ii. Keep the collateral damage minimum during early detection.

iii. Counter the stealth and sophisticated attacks which resemble normal network accessing patterns.

iv. Discriminate DDoS attacks from flash crowds.

v. Suppress false negatives (FN) during the detection process.

vi. Minimize computational and memory overheads.

vii. Adapt to different network environments and detect a range of different attacks without requiring detailed model for normal and attack traffic.

In this chapter we introduce a novel dual-level attack detection scheme which is the first line of defense in the proposed framework for defending against the DDoS attacks. We model Internet as a transit-stub network. In the proposed Dual-Level Attack Detector (D-LAD), the first level attempts to detect congestion inducing macroscopic attacks which cause apparent slowdown in network functionality. Using the Macroscopic-Level Attack Detectors (Ma-LAD), macroscopic or large volumes of attacks are detected early at border routers in transit network before they converge at the victim. On the other hand, sophisticated attacks that cause network performance to degrade gracefully, and stealth attacks that are crafted to match legitimate traffic characteristics remain undetected in transit domain. They are detected at second level by Microscopic-Level Attack Detectors (Mi-LAD) at border routers in stub domain near the victim. We also discuss and employ the concepts of change point detection on entropy with time to improve the detection rate. Honeypots help achieve high detection and filtering accuracy. The compromise of

detection accuracy and time of confirming the attack is a critical aspect and the proposed technique provides the quite demanded optimal solution to this problem.

Our proposed detection scheme is a hybrid that combines anomaly detection and honeypots [174-176] in a way that exploits the best features of these mechanisms while shielding their limitations. Unlike earlier proposals for attack detection [39, 45, 46, 65-69, 71-74, 77, 78, 80, 134, 150, 191] that are either based on unreliable assumptions or are too complicated to implement, our scheme is simple to understand and implement.

4.2 Traffic Feature Selection and Measurement

If DDoS attacks are carried forward in the network, the attack packets will flood the victim and submerge legal user's packets, making legal users unable to access server's resources. As discussed earlier, an effective detection method against DDoS should identify these attacks as early as possible, preferably much before the attack packets reach the victim. Current schemes for early attack detection are based on detecting aggregates causing sustained congestion on communication links [77, 80, 150], imbalance between incoming or outgoing traffic volume on routers [73] and probabilistic packet marking techniques [39, 134]. These early detection methods, unfortunately, have to wait for the flooding to become widespread, consequently, they are ineffective to fence off the DDoS timely. Moreover, techniques like [77, 80, 150] lead to severe collateral damage as legitimate traffic in the aggregates is also dropped. DDoS detection techniques based on volume of traffic (like number of packets, number of bytes, packet size and port distribution, attack speed, distribution of packet arrival time, concurrent flow number, in/out data rate etc.) [65, 67, 71, 78] can detect the attack only when they have reached the victim and are inefficient for early attack detection. The underlying reason is that DDoS attacks are launched from distributed sources. Hence the attack traffic is spread across multiple links. Maximum traffic is available near the victim point for analysis. As the distance from the victim increases, attack traffic is more diffused and harder to detect because the volume of attack flows are indistinguishable from legitimate flows.

Lakhina *et al.* [9] observed that most of traffic anomalies despite their diversity share a common characteristic, they induce a change in distributional aspects of packet header fields (i.e. source IP address, source port, destination IP address, and destination port etc called traffic features). For example, a scan for vulnerable port will have a dispersed distribution for destination addresses, and a skewed distribution for destination ports that is concentrated on the vulnerable port being scanned. Most of the DDoS events can be

treated as anomalies that disturb the distribution of certain traffic features. Our anomaly detection techniques use traffic feature distribution to detect DDoS attack. We choose entropy as a metric for measuring the distribution of traffic features as it exploits inherent feature of DDoS attacks, which makes it hard for attacker to counter this detection by changing their attack signature.

4.2.1 Entropy as a Metric for Measurement

According to the thermodynamics theory, the entropy accounts for the effects of disorder in the system [192]. When an abnormal factor arises to agitate the current system the entropy must show an abrupt change. In information theory, the Shannon [191] entropy or information entropy is a measure of the uncertainty associated with a random variable. Let an information source have n independent symbols each with probability of choice p_i. Then the entropy H is defined as [191]:

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{4.1}$$

Entropy can be calculated on a sample of consecutive packets. The entropy gives the distribution of randomness of some features which are fields in the network packets' headers. These features can be values like source IP address, TTL etc. that indicate the packet's properties. Entropy captures in a single value the distributional changes in traffic features, and observing the time series of entropy on the features exposes unusual traffic behavior. Our proposed algorithms focus on source IP address (or source IP) and destination IP address (or destination IP).

The entropy value gives a description about random distribution of the corresponding feature like source IP address. The bigger the entropy, more random the source IP is. The smaller the entropy, the narrow the distribution range of packets' source IP is, and some addresses have quite high appearance probability. Under normal network conditions, the entropy of network packets always fluctuates to some extent. But during attack, the entropy values have perceptible changes. The change in the feature distribution is detected through monitoring time series variation in the entropy, and reasons are provided for keeping or discarding those packets.

Next, we discuss the detection methods proposed in literature that are based on analyzing the distribution of packet's source IP. In [72] the authors proposed an improvement in the way of computation. In the implementation of their algorithm, the authors used a fixed-size sliding window to simplify the computation complexity. The

window size is W, the probability p_i here equals to the appearance probability of each distinct source IP address in W. Therefore, it does not need to consider all the packets for calculation, but just compute entropy on W packets. However, the authors used entropy very shallowly. They did not address stealth and sophisticated DDoS events that cause graceful degradation of service and used static thresholds for detection under dynamic network conditions.

We start with an empirical histogram $X = \{n_i, i = 1,.....,N\}$ that i occurs n_i times in the sample [193]. Let $S = \sum_{i=1}^{N} n_i$ be the total number of observations in the histogram. Then the sample entropy $H(X)$ is:

$$H(X) = -\sum_{i=1}^{N} (p_i) \times \log_2 (p_i) \tag{4.2}$$

where, . $p_i = n_i / S$

The value of sample entropy lies in the range $0 - \log_2 N$. The sample entropy takes on the value 0 when the distribution is maximally concentrated. It takes on the value $\log_2 N$ when the distribution is maximally dispersed, *i.e.* $n_1=n_2=$ ... n_n. Sample entropy is a convenient summary statistic for a distribution's tendency to be concentrated or dispersed. For e.g. to calculate source IP address based system entropy using Equation 4.2, p_i is the emergence probability of each distinct source IP address, n is the total number of packets being analyzed, and H is the entropy of the system. Similarly, to calculate destination IP address based system entropy using Equation 4.2, p_i is the emergence probability of each distinct destination IP address (irrespective of the source), n is the total number of packets being analyzed, and H is the entropy of the system. For e.g. consider a hypothetical network, in which each source sends an average of 5 packets under normal conditions. With 20 such different sources, according to Equation 4.2, the system entropy under normal conditions equals 2.9956. Consider a concentrated attack where there is a single source sending 50 packets along with normal traffic from 10 sources. The system entropy according to Equation 4.2 decreases to 1.8443. Next consider a distributed attack where there are 50 sources sending 1 packet each along with normal traffic from 10 sources. The system entropy rises to 3.7978.

In this chapter we put forward two new DDoS detection techniques based on the traditional entropy. We use the above entropy computing model as the basis for our macroscopic and microscopic attack detection techniques.

4.2.2 Choice of Traffic Features

Source IP based entropy detection algorithms are efficient in case of highly distributed DoS attacks or concentrated high rate attacks. However, a proficient and sophisticated attacker usually tries to defeat the source IP based entropy detection algorithm [72] by secretly producing flooding attack and simulating the detector's expected normal data flow. After knowing some packet attributes' entropy values, the attacker can use the automated attack tools to produce some flooding with adjustable entropy values. By guess, test or summary, these attackers can know the normal entropy range in the detector and adjust their own flooding to match it, although such stealthy attacks are not easy to realize.

We improve the previous entropy detection algorithms and propose enhanced algorithms for dual-level attack detection: Macroscopic-Level Attack Detectors (Ma-LAD) are based on entropy calculated over source IP and Microscopic-Level Attack Detectors (Mi-LAD) are based on entropy calculated over destination IP. We assume that any link or network has a characteristic probability distribution of IP addresses for initiators of IP traffic and another probability distribution for IP addresses that are the recipients of network traffic. A DDoS attack modifies these distributions of source and destination IP addresses in terms of new IP addresses entering the system or certain IP addresses becoming more dominant.

Ma-LAD uses source IP based entropy to detect the presence of DDoS attack. The source IP based system entropy fluctuates to some extent under normal network conditions. But when the entropy values have perceptible changes, attacks are detected. An increase in source IP based entropy indicates distributed attacks (according to Equation 4.2, entropy becomes maximum when distribution is maximally dispersed) whereas a decrease in source IP based entropy indicates a concentrated attack launched from single or a few sources (according to Equation 4.2, entropy becomes zero when distribution is maximally concentrated).

Mi-LAD uses destination IP based entropy to detect the presence of attack on server or victim. A single destination IP address (or alternatively, a very, very few number of unique destination IP addresses) receives many more packets during DDoS attack than other normal conditions. Consequently, a decrease in destination IP based entropy detects

the presence of DDoS attacks. Specifically, we use Shannon's mathematical definition of entropy [191] as a measure of uncertainty in a distribution to quantify source IP and destination IP address structure.

4.3 Dual-Level Attack Detection Scheme

In this section, the first motivation is to detect DDoS attack at an early stage and eliminate congestion inducing attack packets before they reach the target. The second motivation is to counter the attacks which resemble normal network accessing patterns and lastly to discriminate DDoS attacks from surge of legitimate traffic or flooding. We also aim to keep the FP and FN minimum during the detection process. In our proposed D-LAD scheme [194], Ma-LAD detects the voluminous attacks (that cause immediate network congestion and complete service disruption) early in the network. Mi-LAD detects the attacks that resemble normal network accessing patterns (cause graceful service degradation) which slip past Ma-LAD. Presence of honeypots detects the attacks that exploit vulnerabilities and suppresses FP and FN.

4.3.1 System Model

We use transit-stub network model [195, 196] of the Internet as shown in Figure 4.1. Transit-stub model is based on the hierarchical approach of the Internet [195, 196]. In such a model, every domain can be classified as either a transit network or a stub network. A stub network connects end hosts to the Internet. A transit network interconnects stub networks. Backbone ISPs and regional ISPs are examples of transit networks. The traffic generating nodes (end hosts) are only connected to stub networks. The packets have to travel, very often, through several networks before getting to the destination as shown in the Figure 4.1. As for the scenario of a DDoS attack, each of the attackers, legitimate users and the victim server are connected to a stub network. The traffic usually passes through two stub networks, one on the sender side and the other on the victim side, and one or more transit networks. Our aim is to protect the victim server and the corresponding network from DDoS attacks.

We model the Internet to measure the entropy in transit-stub network. During an attack, the Internet is divided into the two networks; one for inside to be protected and the other is for outside where attackers may reside. The entropy is measured by recording the dynamics of packets on the borders of the two networks. Packets flowing between these

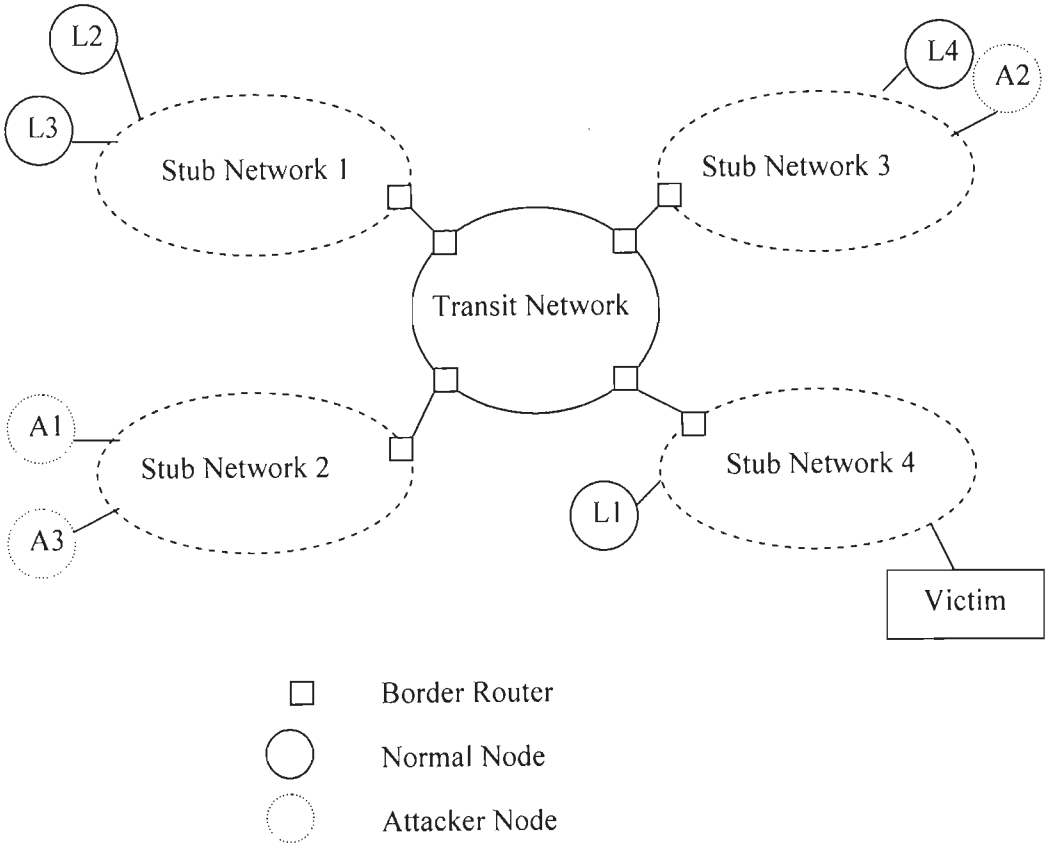


Figure 4.1. Transit-Stub Network

two networks may sustain the current entropy value in the range of the normal entropy if those packets are in harmony with the system or current entropy value may change abruptly and go out of normal range if packets agitate the system. In the proposed system we keep track of the value of entropy in time and identify the sudden changes in the value. These changes are regarded as the installation of attacks in the network.

Detection algorithms are running on the edge routers of transit network and stub network. Figure 4.2 gives the detection workflow. As discussed earlier, macroscopic or largest volume of attacks should be detected early and dropped before they enter the victim network. These attacks can create congestion in the network and stress resource utilization in a router and network, which make them crucial to be dropped before they enter the network from an operational standpoint. Macroscopic-Level Attack Detectors (Ma-LAD) on edge routers of transit network consistently detect these attacks. The edge

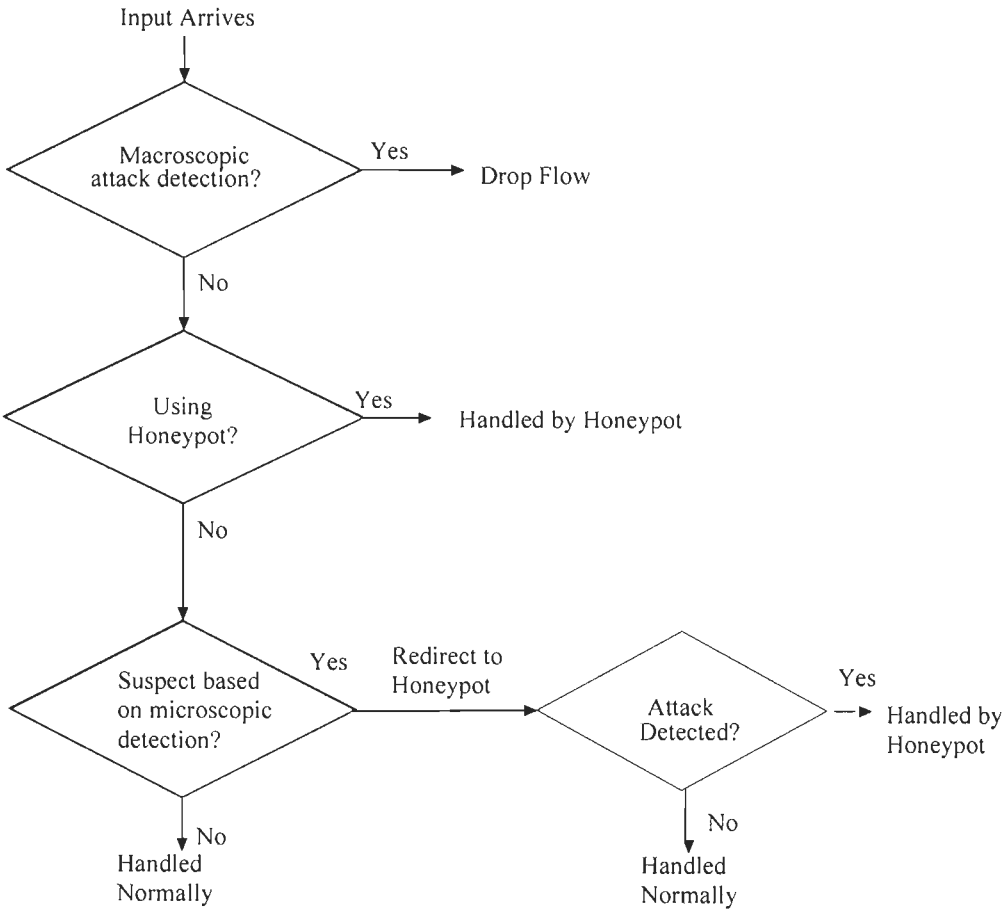


Figure 4.2. Detection Workflow

routers of transit network monitor the network traffic by calculating source IP based system entropy for the arriving flows. Ma-LAD treats all the packets coming from distinct source IP as a "flow". In the non-attack case, system entropy stays within a stable range. When there is an attack source IP based system entropy changes dramatically at router, because there is either one flow dominating the router (this indicates concentrated attack and entropy decreases) or multiple flows with a very few packet arrivals in each flow (this indicates distributed attacks and entropy increases). The detection of attack at edge router of transit network triggers the characterization process (discussed in next chapter) and flows identified as attacks at macroscopic-level are dropped as shown in Figure 4.2. The Figure also shows that the flows that are not detected as attacks by Ma-LAD but destined to honeypots are handled by honeypots. The traffic neither detected as attack by Ma-LAD nor destined to honeypots undergoes microscopic-level attack detection tests.

Microscopic-Level Attack Detectors (Mi-LAD) located on edge routers of stub domain are used for detecting attacks that may not necessarily impact the network, but can have dramatic impact on the victim or server. They enable highly sensitive detection. Edge routers of the stub network monitor the network traffic by calculating destination IP based system entropy. In case of destination IP based system entropy; all the packets going to a distinct destination IP constitute a "flow". When there is an attack, the destination IP based system entropy decreases dramatically; because there is one destination IP based flow dominating the edge router of the stub domain. In this case, the edge router treats the dominating flow as victim of DDoS attack. Once the victim of DDoS attack is identified, characterization process (discussed in next chapter) is triggered. The suspicious flows are directed to honeypots. The rest of the flows are normal and are directed to servers as shown in Figure 4.2.

4.3.2 Design of Macroscopic-Level Attack Detector (Ma-LAD)

Ma-LAD detects macroscopic attacks which are highly distributed DDoS flooding attacks (for e.g. distributed reflector denial of service attacks) or highly concentrated high rate attacks which induce immediate congestion in the network (for details, refer to Section 2.2.2). As discussed earlier, Ma-LAD is located on the edge routers of the transit domains and hence enable early DDoS attack detection without the need for traffic observation in the victim network. They make use of computing source IP based system entropy. If the system entropy crosses threshold limits, attack is detected. Further, if the flows are destined to honeypots, attack is confirmed. The corresponding attack flows are identified and dropped. Thresholds are optimized according to client requirements and network conditions.

4.3.2.1 Sampling and Detection Mechanism

Detecting DDoS attacks involves first knowing normal behavior of the system and then finding deviations from that behavior. The normal profile or behavior is obtained by using entropy $H(X)$ (refer to Equation 4.2) as a parameter to measure traffic feature distributions. Ma-LAD designates a different flow id to each unique source IP encountered in incoming packets. The edge router of transit domain collects traffic information in a time window and calculates system entropy $H(X)$.

Consider a random process $\{X(t), t = j\Delta, j \in N\}$, where Δ, a constant time interval is called time window, N is the set of positive integers, and for each $t, X(t)$ is a random

Table 4.1.Calculation of system entropy $H(X)$ in each time window Δ

Flow	1	2	3	...	...	N	H(X)
Δ	X(Δ,1)	X(Δ,2)	X(Δ,3)	...	...	X(Δ,N)	**H(Δ)**
2Δ	X(2Δ,1)	X(2Δ,2)	X(2Δ,3)	...	...	X(2Δ,N)	**H(2Δ)**
3Δ	X(3Δ,1)	X(3Δ,2)	X(3Δ,3)	...	...	X(3Δ,N)	**H(3Δ)**
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
$n\Delta$	X($n\Delta$,1)	X($n\Delta$,2)	X($n\Delta$,3)	...	...	X($n\Delta$,N)	**H($n\Delta$)**

variable. Here $X(t)$ represents the number of packet arrivals for a flow in $\{t-\Delta, t\}$. $X(t)$ as a whole represent our empirical histogram for computing entropy. Table 4.1 shows the calculation of system entropy $H(X)$ in each time window Δ.

It is found in our simulation without attack that entropy value $H(X)$ varies within very narrow limits after slow start phase is over. This variation becomes narrower if we increase Δ i.e. monitoring period. We take average of $H(X)$ and designate that as normal entropy $H_n(X)$. The basic idea is to remove small scale perturbations by averaging over slightly longer-intervals of time. However it is also desirable that the window duration should not exceed a limit as the Internet traffic shows large variations across different times of the day. By this way, normal profile of traffic in terms of Entropy $H_n(X)$ is obtained by our approach. To detect the attack, the entropy $H_c(X)$ is calculated in shorter time window Δ continuously, whenever there is appreciable deviation from $H_n(X)$, attack is said to be detected [193].

We assume that the system is under attack at time t_a, which means that all attacking sources start emitting packets from this time. The network is in normal state for time $t < t_a$ and turns into attacked state at time t_a. Let t_d denote our estimate on t_a. At time t_d following event triggers:

$$(H_c(X) > (H_n(X) + a \times d)) \cup (H_c(X) < (H_n(X) - a \times d)) \quad (4.3)$$

attack = true;

Here $a \in I$ where I is set of integers and a is the tolerance factor or threshold. Tolerance factor or threshold a is a design parameter and d is maximum absolute deviation in entropy $H(X)$ from average value $H_n(X)$ while profiling for network without attack. The flowcharts for macroscopic-level attack detection scheme are given in Figure 4.3, Figure 4.4 and Figure 4.5. Figure 4.3 shows the procedure for sampling in small time windows. Figure 4.4 illustrates the calculation of current entropy $H_c(X)$. Figure 4.5 shows the attack detection procedure.

4.3.2.2 Decision of Optimum Threshold and System Calibration

Many detection tools use a fixed threshold to alarm on anomalous traffic. This method makes sense for some applications. For example, in control systems, there may be a certain tolerance level for products to be considered "acceptable". If this tolerance level is exceeded, then the product is considered "bad". But for network traffic monitoring, the background is continually changing. If the background traffic changes, these thresholds may become meaningless and need to be changed. If the threshold a is set too high, normal entropy range $(H_n(X) \pm a \times d)$ that classifies traffic as legitimate would be broad, the FP rate will be low, but detection rate will be low too. Similarly, if the threshold a set too low, it may detect most attacks, but suffer from a high FP rate. Therefore, in network traffic monitoring, it is critical to update these estimates adaptively. We use adaptive threshold [197] which means that the threshold is updated regularly depending upon network conditions and user requirements. Since this adaptive approach continually updates the threshold or a tolerance factor a, the model adjusts to reflect changes in background traffic. FP gives the effectiveness of the system whereas FN gives a measure of the system reliability. As discussed, variations in threshold or tolerance factor a quantifies FP and FN [197]. Minimization of FP and FN assist in making decision on the optimum value of the threshold a. By adjusting a baseline, estimates adjust quickly to calibrate the system for normal entropy range.

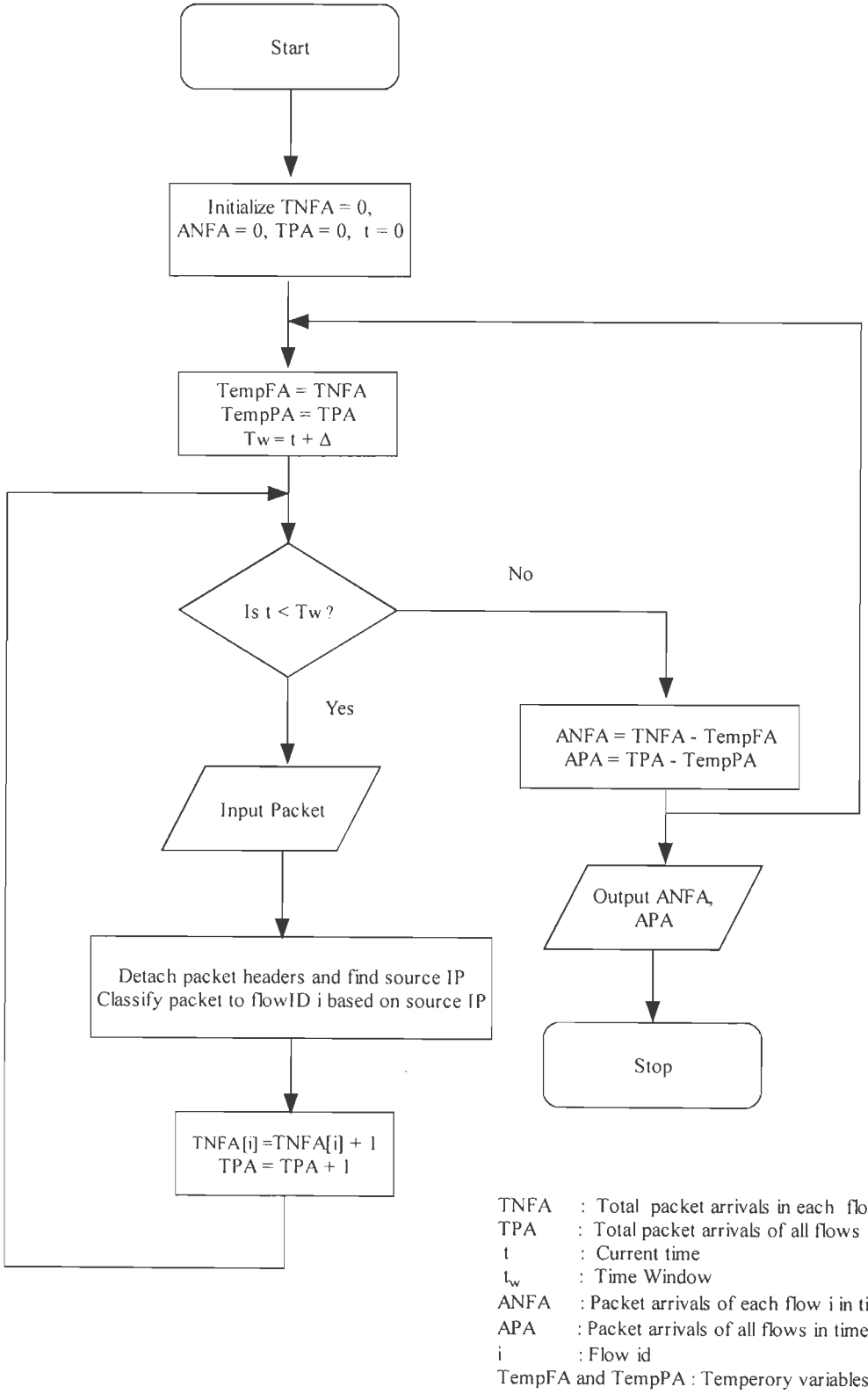
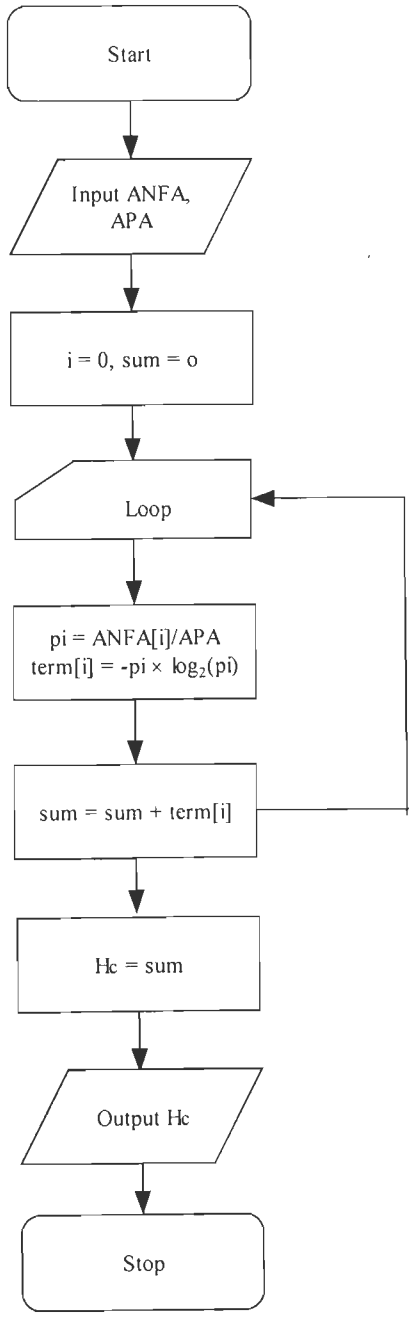
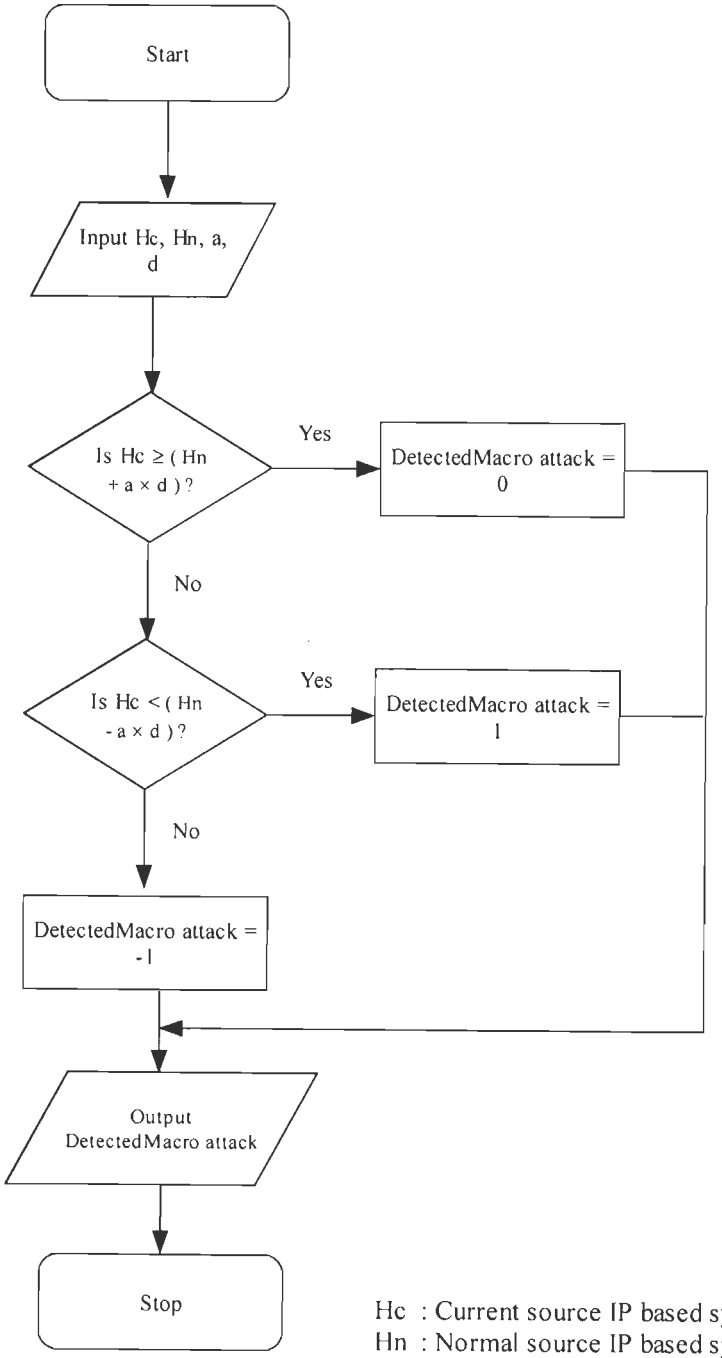


Figure 4.3. Macroscopic-level attack detection: Sampling in time window t_w



ANFA : Packet arrivals of each flow i in a time window
APA : Total packet arrivals of all flows in time window
Hc : Current entropy

Figure 4.4. Macroscopic-level attack detection: Calculation of H_c on the sample in t_w



Hc : Current source IP based system entropy
Hn : Normal source IP based system entropy
a : Threshold
d : Maximum absolute deviation
DetectedMacro attack =0 : Highly distributed low rate attack
DetectedMacro attack =1 : Concentrated high rate attack
DetectedMacro attack =-1: No attack

Figure 4.5. Detection of attack by Macroscopic-Level Attack Detector (Ma-LAD)

4.3.2.3 Defense Modes

Depending on the threshold or tolerance factor a (refer to Equation 4.3), the detection technique of the framework operates in one of the following modes of operation: Best defense, normal defense and naïve defense.

Best Defense: Figure 4.6 shows the best defense mode of operation. X-axis represents the value of source IP based system entropy and Y-axis gives the number of measurements i.e. number of time windows in which the entropy value is obtained. The Figure shows entropic profiles of legitimate and attack traffic. In case of best defense, the threshold or tolerance factor a is set to low value and hence the normal entropy range during attack detection is very small. The choice is made to reduce the FN to minimum, and is zero in ideal case.

Normal Defense: Figure 4.7 shows the normal defense mode of operation. The threshold or tolerance factor a is set neither too high nor too low. Hence the normal entropic range that classifies traffic as legitimate is moderate, and the FP and FN are balanced.

Naïve Defense: Figure 4.8 shows the naïve defense mode of operation. The threshold or tolerance factor a is set to a high value. Hence the normal entropic range that classifies traffic as legitimate is broad, and the FP rate is low, but detection rate is low, too. Naïve defense has the lowest detection sensitivity level and hence it has lowest FP rate at the cost of increased FN rate.

The mode of operation is decided according to network conditions so as to minimize FP and FN. Figure 4.9 gives the comparison between the three modes of operation with respect to FP and FN.

4.3.2.4 Honeypot-Based FN Suppression

We propose the deployment of honeypot [198] along with the server to suppress the FN as discussed in Section 3.2.1.1. The presence of honeypots reduces FN under naïve defense mode of operation. If the client load is high, to avoid adverse effects to legitimate traffic (keep FP low); detectors are often tuned in naïve defense mode. Due to the presence of honeypots, attack traffic that lie in permissible entropy range and remain undetected but attempt to exploit the vulnerabilities are destined and directed to honeypots, thus reducing the FN.

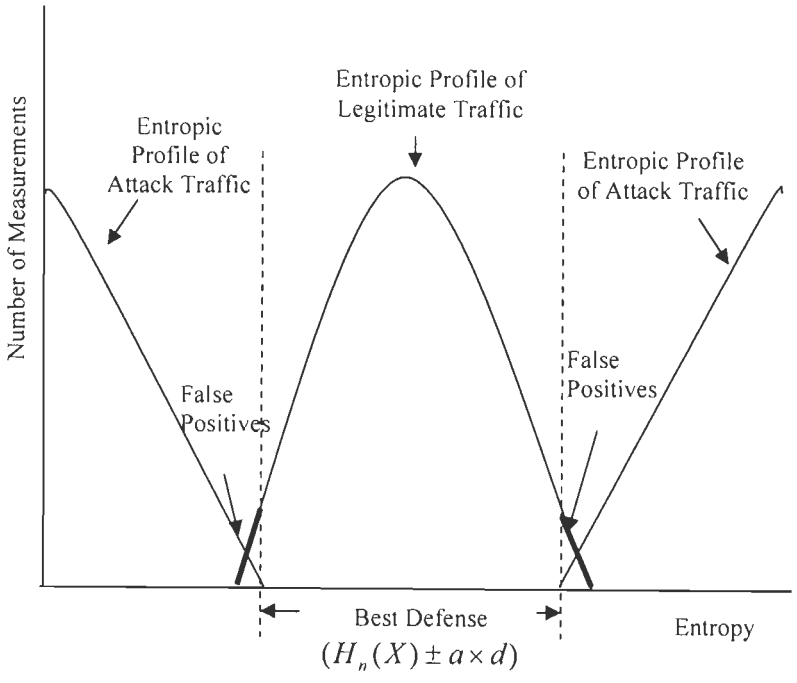


Figure 4.6. Best defense

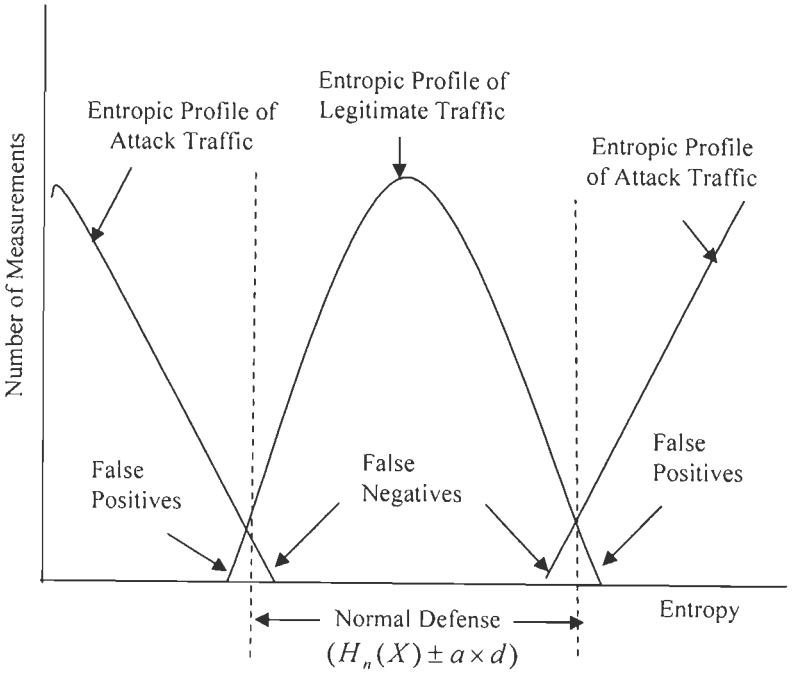


Figure 4.7. Normal defense

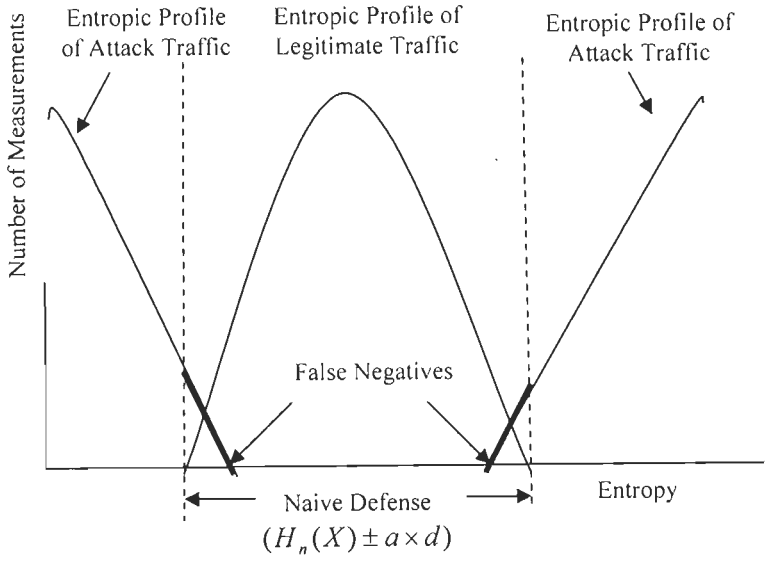


Figure 4.8. Naive defense

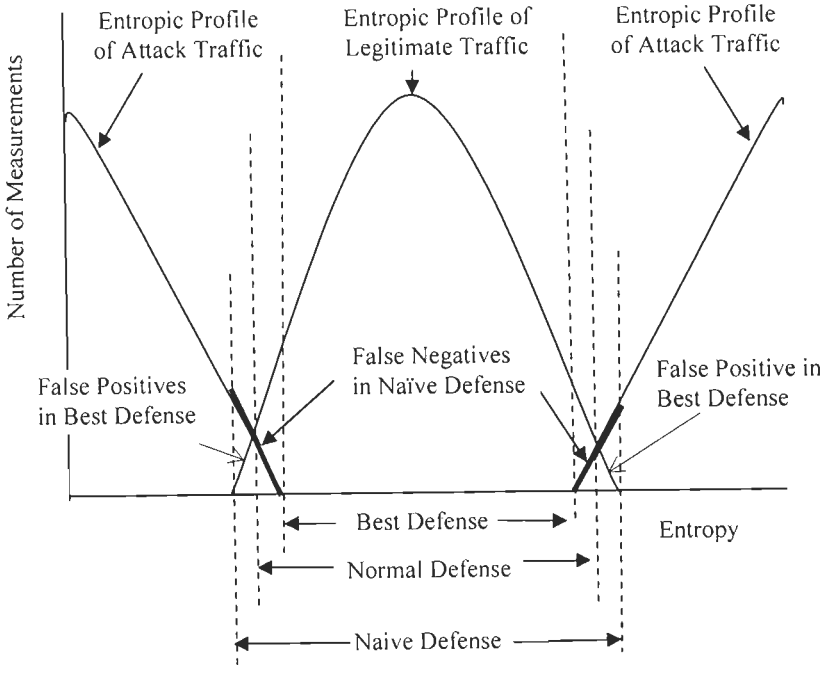


Figure 4.9. Comparison between naïve, normal and best defense

4.3.3 Design of Microscopic-Level Attack Detector (Mi-LAD)

Slow rate, isotropic attacks do not cause immediate congestion and do not abruptly stress the resources. They may go undetected by Ma-LAD. Moreover, distributional changes captured by entropy observed on source IP alone cannot detect stealthy and sophisticated attacks that are crafted to match statistics of normal traffic. For example, the attackers may simulate the normal network behaviors, e.g. pumping the attack packets as Poisson distribution, to disable Ma-LAD. Also, how to discriminate DDoS attacks from flash crowd is a major challenge. Current volume based detection schemes like [65, 67, 71] for attack detection at the victim cannot detect slow rate, isotropic attacks because these attacks do not cause detectable disruptions in traffic volume. Moreover, solutions [65, 67, 71] suffer from collateral damage when attack is carried at slow rate or when volume per attack flow is not so high as compared to legitimate flows. Hence we propose Mi-LAD to detect such attacks at edge routers of stub network.

A DDoS attack, regardless of its volume and source, will cause the distribution of destination IP address to be concentrated on the victim address. In DDoS attack scenario, a single destination IP address (or alternatively, a very, very few number of unique destination IP addresses) receives much more traffic than other normal conditions. Destination IP address information is not available at transit domains or core routers if NAT or proxies are used. So it can only be analyzed on border router of stub domain of victim. Observing the time series of entropy on destination IP exposes unusual traffic behavior which source IP alone could not detect. A decline in destination IP based system entropy in time series indicates an attack.

However, this happens as well when there is a flash crowd to server. Based on destination IP based system entropy calculation alone, we cannot identify flash crowd from DDoS attacks. In order to detect a DDoS attack, we need to test for changes in our detection feature over time. However, our detection feature i.e. destination IP is a random variable due to the stochastic nature of Internet traffic. Consequently, we require a mechanism that can accurately discriminate between the onset of a DDoS attack and a temporary random fluctuation in traffic. We therefore apply cumulative sum (CUSUM) [199] to solve this problem. CUSUM is calculated over destination IP based system entropy to detect the attacks. It makes use of the concept of time along with threshold to judge the network condition. If the abnormal condition persists for a certain period or crosses threshold, attack is detected. Destination under attack is identified in case an attack is present.

4.3.3.1 Sampling and Detection Mechanism

Mi-LAD designate a different flow id to each unique destination IP address encountered in incoming packet. As described earlier, for microscopic-level detector, we define flow as the packets that share same destination address at the edge router of stub network.

Consider a random process $\{X(t), t = n\Delta, n \in N\}$, where Δ a constant time interval is called time window, N is the set of positive integers, and for each t, $X(t)$ is a random variable. Here $X(t)$ represents the number of packet arrivals for flow in $\{t - \Delta, t\}$. $X(t)$ as a whole represent our empirical histogram for computing entropy. It is found in our simulation without attack that destination IP based system entropy value $Y(X)$ (calculated according to Equation 4.2) varies within very narrow limits after slow start phase is over. We take average of $Y(X)$ and designate that as expected value of entropy $E[Y] = \alpha$. By this way, normal profile of traffic in terms of destination IP based system entropy $Y(X)$ is obtained by our approach.

Our attack detection algorithm is based on the sequential change point detection [199]. The objective of change point detection is to determine if the observed time series is statistically homogeneous, and if not, to find the point in time when the change happens. It has been studied extensively by statisticians [199]. There have been various tests for different problems. They can be largely divided into two categories, namely, posterior and sequential. Posterior tests are done off-line where the whole data segment is collected first and then a decision of homogeneity or a change point is made based on the analysis of all the collected data. On the other hand, sequential tests are done on-line with the data presented sequentially and the decisions are made on the run. We adopt a sequential test for a quicker response when an attack occurs. It also saves memory and computation.

4.3.3.2 Cumulative Sum (CUSUM) Algorithm

Researchers use change point detection theory [199] to detect abnormal Internet traffic caused by DDoS attacks [66, 67, 200, 201]. Cumulative Sum (CUSUM) is an algorithm from statistical process control that detects the mean variation of statistical process [199]. CUSUM is based on the fact that if there is some change, the probability distribution of random sequence will also change.

We improve the previous entropy detection algorithm [193] by incorporating the idea of sequential variation using cumulative sum and variation detection. In the non parameter CUSUM algorithm, the idea of sequential variation is proposed [200, 201]. But its

approach is to analyze the ratio between new arrival IP number in a time and the total IP number, and thus construct a random sequence. To implement that algorithm, we need to create database containing large amount of legal IP addresses, and each time we should compare and calculate number of all new IP in each time unit. The calculation is complicated and has low efficiency. In our improvement, we use the destination IP address based entropy statistics. We try to cumulate the entropy according to some rules, thus it has more accurate DDoS attack detection.

In our proposed destination IP based entropy detection method, suppose Y_n is the destination IP based system entropy value calculated on edge router of stub at n^{th} sampling interval, and the random sequence $\{Y_n\}$ is the network service model. In the normal condition, this sequence is independent and distributed. Assume the variation parameter is the average value of sequence $\{Y_n\}$. Before change, this value $E(Y_n) = \alpha$.. Before attack, when the network is normal, the distribution of destination IP addresses is stable, and has certain randomness. But when DDoS attack happens on one of the destinations, this average value will decrease suddenly. $E(Y_n)$ will become far smaller than α.

CUSUM algorithm also has an assumption that in the normal case, the average value of random sequence should be negative and it becomes positive after change. Therefore, without losing any statistics properties, we transfer the sequence $\{Y_n\}$ to another random sequence $\{Z_n\}$ with negative average value.

$$Z_n = -(Y_n - \beta) \tag{4.4}$$

In a given network environment, parameter β is a constant used for producing a negative random sequence $\{Z_n\}$. In our detection algorithm, we define $\beta = \alpha$. When the attack happens, Z_n will suddenly become very large and positive. The detection threshold is the limit for the positive, which is the cumulative value of Z_n.

We use the following recursive formula for cumulative sum:

$$S_0 = 0$$

$$S_n = \max(S_{n-1} + Z_n, 0) \tag{4.5}$$

S_n represents the cumulative positive value of Z_n. The bigger the S_n, the stronger the attack is.

We calculate the time of increase of cumulated value using S_n using the following formula:

$$C_0 = 0$$

$$\begin{aligned} C_n &= (C_{n-1} + 1), \quad (S_n > S_{n-1}) \\ &= 0, \qquad\quad otherwise \end{aligned} \tag{4.6}$$

C_n represents a counter and signifies the duration of increase in S_n. It uses the concept of time to judge the network condition. The bigger the C_n, higher the probability that there is an attack.

The judgment function is:

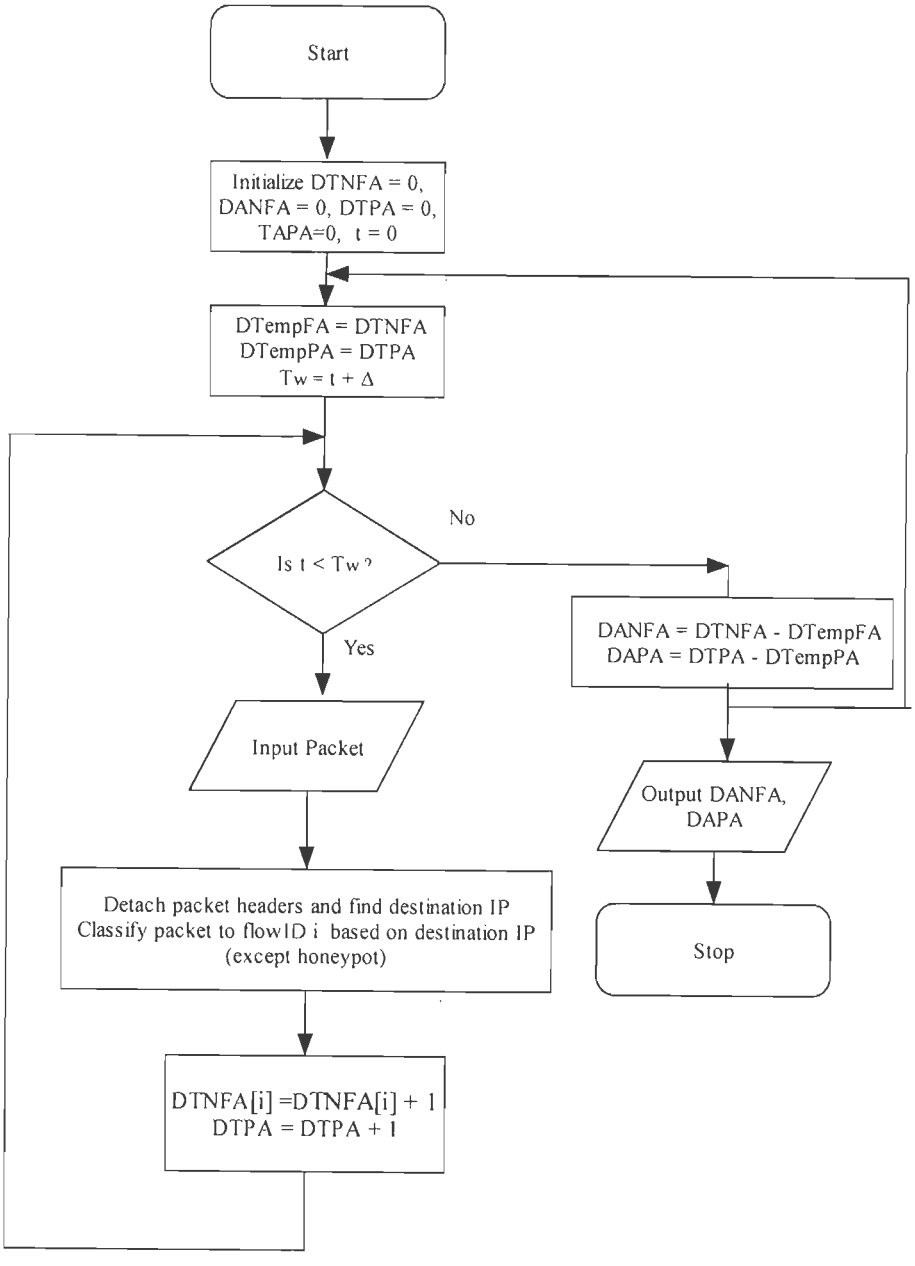
$$\begin{aligned} d(S_n, C_n) &= 1, \quad S_n > T \; OR \; C_n > T' \\ &= 0, \quad otherwise \end{aligned} \tag{4.7}$$

$d(S_n, C_n)$ is the judgment function, the value 1 shows that attack happens, while 0 shows the normal case. T and T' are the detection thresholds. We can control the total attack detection time by setting the value of parameters T and T'. For example, if time window Δ is .2s and T'=8, an alarm is generated only when an increase in destination IP based system entropy is observed for more than 1.6s or destination IP based system entropy increases beyond value T, whichever earlier. This signifies that a sudden traffic increase in short time may be a flash crowd which is normal traffic and should be allowed. But if the network anomaly lasts for more than 1.6s, the system may be under attack.

The advantage of this improved algorithm is that it comprises implicitly a concept of process cumulating. In most of the detection algorithms, network conditions are judged according to threshold. This judgment may not be suitable in some occasions. For example, flash crowds where traffic flow in the network suddenly increases, but the flow is actually from legitimate users. The function of cumulating process is to avoid FP when the network has something abnormal just at a time point like a flash crowd.

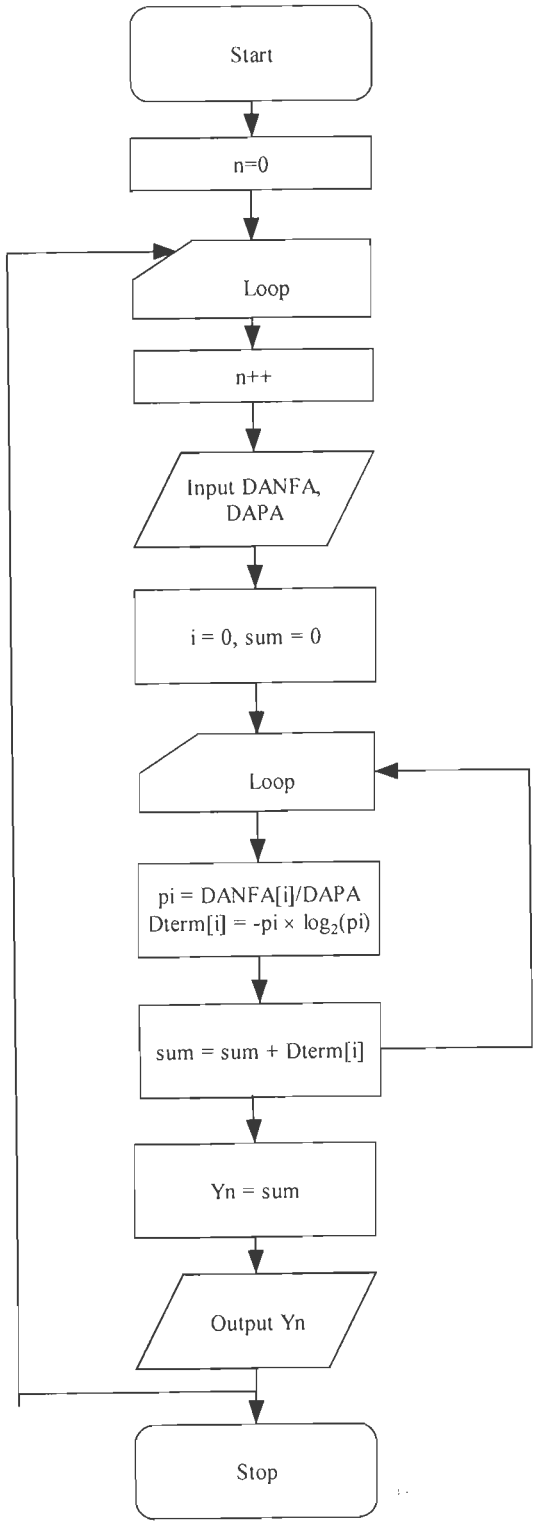
Thus the threshold based approach leads to a more real time and timely attack detection. Time based approach emphasizes on time tolerance and ignores traffic bursts in some allowable range. Network is considered under attack if threshold is reached or surge of access increases beyond tolerable limit defined by time period.

The flowcharts for microscopic-level attack detection scheme are given in Figure 4.10, Figure 4.11 and Figure 4.12.



DTNFA : Total packet arrivals in each destination IP based flow i (except honeypots)
DTPA : Total packet arrivals of all destination IP based flows (except honeypots)
t : Current time
t_w : Time window
DANFA : Packet arrivals of each destination IP based flow i in time t_w (except honeypots
DAPA : Packet arrivals of all destination IP based flows in time t_w (except honeypots)
i : Destination IP based flow id
DTempFA and DTempPA : Temperory variables

Figure 4.10. Microscopic-level attack detection: Sampling in time window t_w



DANFA : Packet arrivals of each destination IP based flow i in time t_w(except honeypots)
DAPA : Packet arrivals of all destination IP based flows in time t_w (except honeypots)
Yn : Destination IP based system entropy

Figure 4.11. Microscopic-level attack detection: Calculation of Y_n

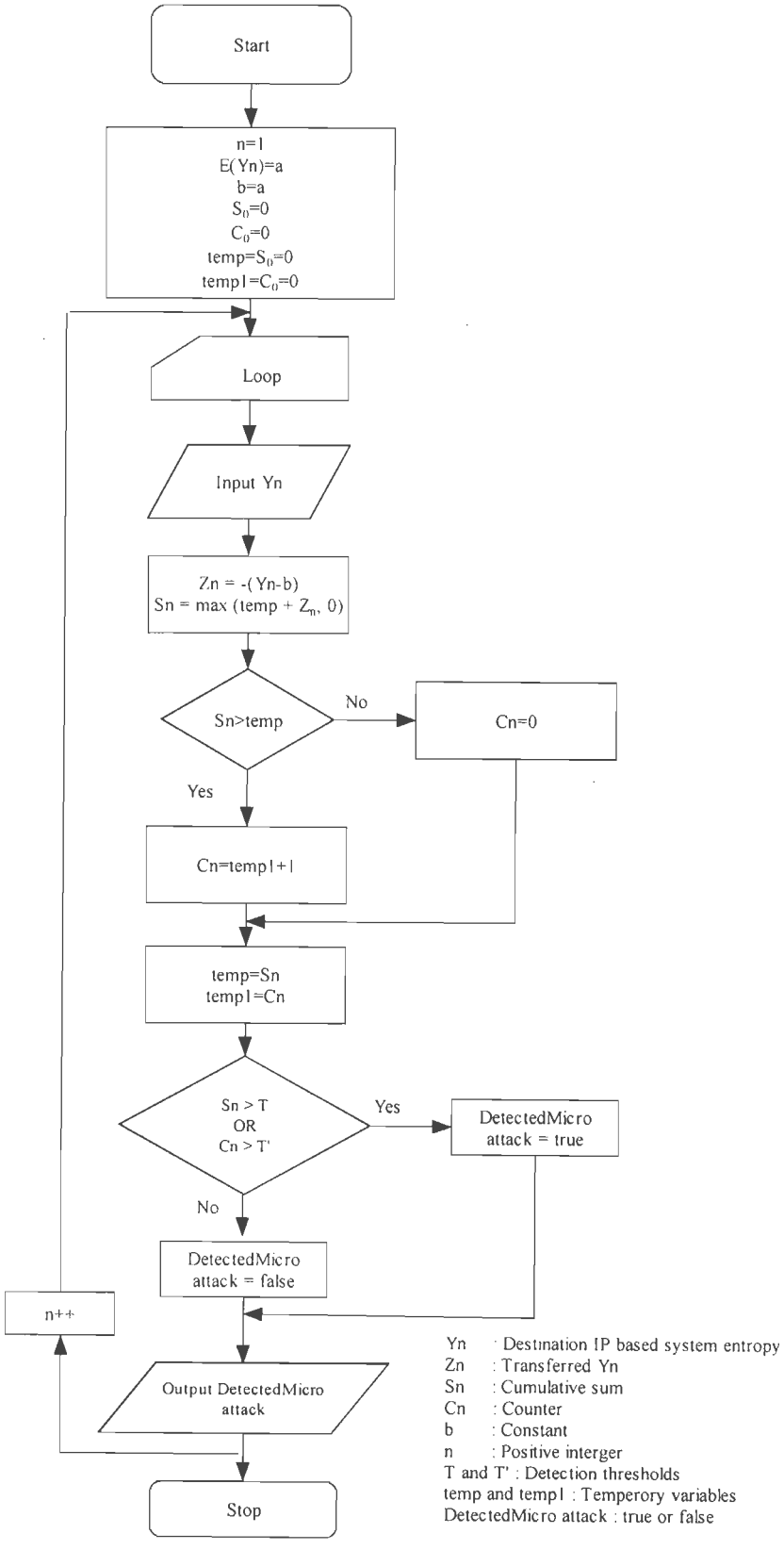


Figure 4.12. Detection of attack by Microscopic-Level Attack Detector (Mi-LAD)

Dual-Level Attack Detector (D-LAD) minimizes collateral damage due to flash crowds. Macroscopic-level detectors are tuned to detect and block only the congestion inducing traffic in the network. The threshold is optimized to minimize FP and traffic is not blocked at macroscopic-level if the network can handle it without getting congested. Therefore, flash crowds that do not induce persistent congestion remain unaffected by macroscopic-level detectors. They remain unaffected at microscopic-level because microscopic-level detectors take into account time for which observation is made before raising an alarm. And flash crowds usually fade away in small time interval. Hence, they remain unaffected by our proposed D-LAD scheme, minimizing the collateral damage.

4.4 Performance Evaluation of D-LAD

We evaluated our proposed dual-level detection technique by carrying out exhaustive simulations in network simulator-2 (ns-2) on a Linux platform. This section gives the details of the experiment design and procedure and the results obtained are discussed next.

4.4.1 Experiment Design and Procedure

We tested our scheme against an Internet type topology using network simulator 2 (ns-2) [202] as simulation testbed. For details of the simulation model, refer to Appendix A. The 156 node network shown in Figure A.1. is composed of 5 FTP servers, labeled node 117, 118, 119, 120 and 121 to be protected, 137 clients of which 48 are legitimate clients and 89 are attackers and router nodes. All FTP requests are originated randomly from different client nodes. We introduce randomness to the locations of legitimate clients and attackers. However, to simplify the analysis, we carefully place servers where they all have same access bandwidth (i.e. 3 Mbps) and have paths from clients to access.

To control the load of each run, we use Poisson process to model arrival process of the FTP clients. The inter-arrival time (IAT) of FTP client is computed by:

$$IAT = \exp(T_{ftp} / CL) \tag{4.8}$$

where T_{ftp} is the average total time for file transfer, CL is the total client load, and $exp(x)$ is the exponential distribution with mean x.

Configuration parameters, including the link rate and propagation delay are summarized in Table A.3. Each simulation experiment has 10 runs (averaged in the graphs). Legitimate clients send requests from time 0 - 30 seconds

Table 4.2. Attack detection parameters

S. No.	*Parameter*	*Value*
1.	Simulation time	30 *seconds*
2.	Attack Duration	8-20 *seconds*
3.	Window Size	.2 *seconds*
4.	Packet Size	1040 *bytes*
5.	Tolerance factor a	2-9

and attack duration is from 8-20 seconds. Attack detection parameters are as given in Table 4.2.

Simulations are carried at different values of tolerance factor a for different attack strengths. To simulate attack at different strengths, we varied attack rates and number of attackers. To observe deviations in entropy, we varied attackers from 1-80, keeping the mean attack rate constant. Then we varied the mean attack rate keeping the number of attackers constant. We also observed the deviations by keeping the total attack load (AL) constant, and generating different combinations of attack rates and number of attackers. In our experiments, we use window size of .2 seconds, for timely detection of DDoS in a manner that small scale perturbations are removed as well as computational burden is minimized.

4.4.2 Results and Discussion

We conduct experiments for the proposed dual-level entropy detection methods. The results show that our methods have better detection capability than before.

4.4.2.1 Degradation of Throughput with Attack

The aim of any DDoS attack is to congest network links so as to minimize message delivery. Throughput [203] is the measure of average rate of successful message delivery over a communication channel and is calculated as the number of bytes transmitted during specified time interval. We calculate throughput on the bottleneck link of the transit-stub network. Throughput has been calculated as sum of bytes of all flows $\sum F$ received on the bottleneck link in an observation interval (or granularity) divided by size of observation interval. We keep the size of observation interval as 1 second. Throughput values have been normalized.

The server as per its capacity planning and normal profile of legitimate clients in terms of request bytes normally have an estimate of maximum number of clients to be served at any instant of time. On this basis, since we have bottleneck of 10 Mbps, for full link utilization, we assume to serve *CL* up to 10 Mbps. We generate requests that originate randomly from 48 clients with inter-arrival time exponentially distributed (refer to Equation 4.8).

Throughput at different attack rates are shown in Figure 4.13. Here attack is conducted by a single attacker with attack strength ranging from 1 Mbps to 10 Mbps where bottleneck bandwidth is 10 Mbps. Clearly from Figure 4.13, we can observe that in the absence of attack, the throughput is nearly 1. We can say that as the attack starts at time 8 seconds, packet drops increase and hence throughput also decreases. Moreover at meek attack rates, number of packet drops is less hence they degrade throughput a little, however as attack strength increases number of legitimate as well as attack packet drops also increases. As far as high rate attacks are concerned, they almost bring the throughput to zero.

Figure 4.14 shows the decrease in throughput with a DDoS attack conducted with 80 attackers at varying attack rates. In case of highly distributed attacks, even at a meek attack rate of 0.5 Mbps per attacker, the throughput value drops to 0.

The results show that both concentrated high rate attacks and highly distributed attacks have severe impact on the throughput of the network. Throughput values become nearly 0 as soon as the attacks are launched. The results demonstrate that these attacks introduce high congestion in the network that result in abrupt failure of services.

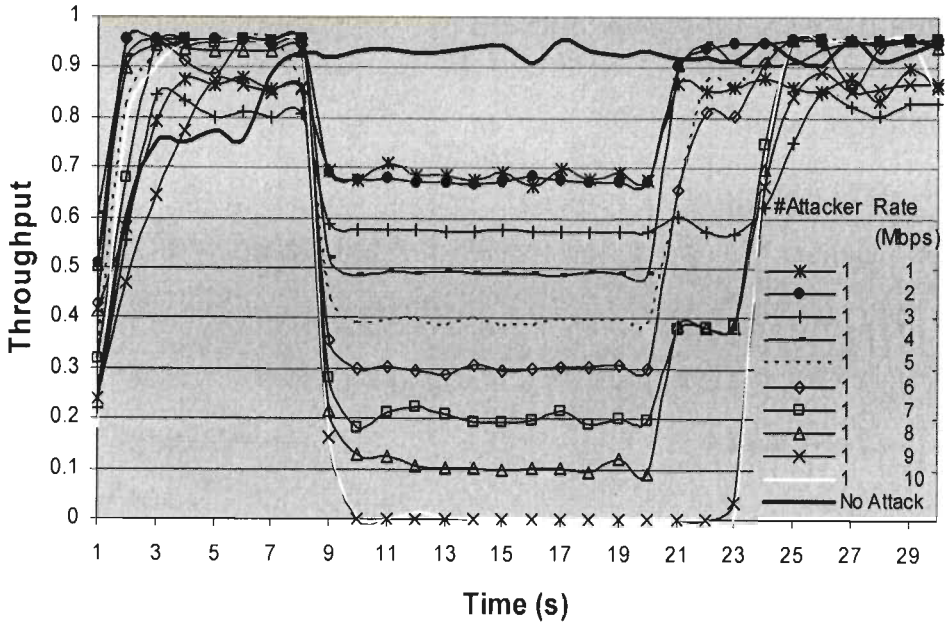


Figure 4.13. Relative degradation of throughput at different attack strengths for DoS attack

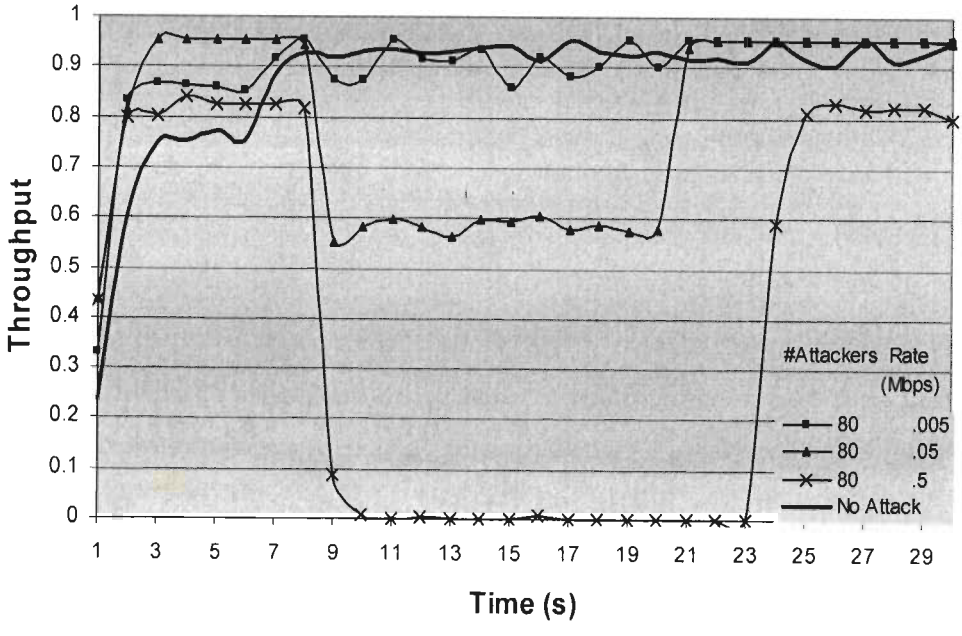


Figure 4.14. Relative degradation of throughput at different attack strengths for DDoS attack

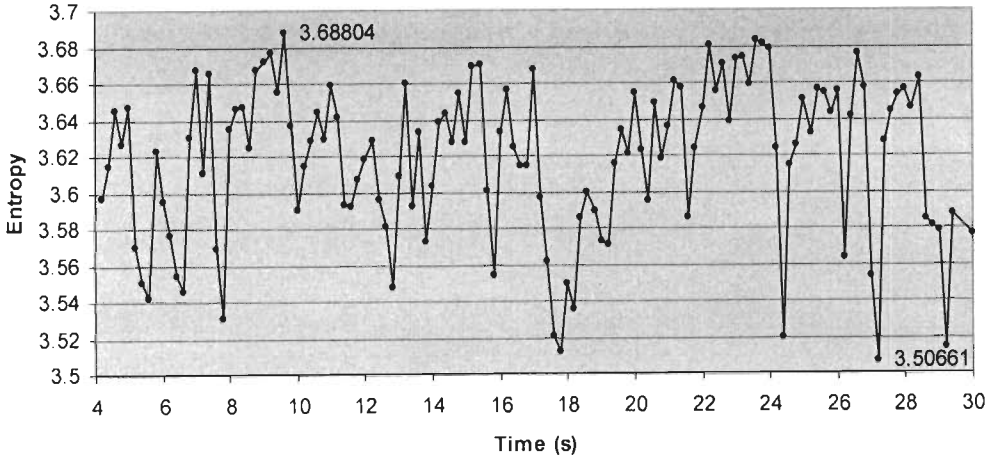


Figure 4.15. Normal entropy range without attack

It is found that normal source IP based system entropy value lies in small range as depicted in Figure 4.15. Our range is 3.50661 to 3.68804, whereas this varies depending upon network environment and type of application .The average is 3.618199, standard deviation is 0.012, and maximum absolute deviation from average is 0.043198. Finalized simulation parameters are:-

Normal Entropy Value ($H_n(X)$):-3.618199

Maximum absolute deviation from average (d):-0.043198

4.4.2.2 Detection of Attack by Ma-LAD

We performed exhaustive experimentation and looked closely at the distribution of source IP based entropy measurements for legitimate traffic and different kinds of DDoS attack traffic.

As soon as any event in Equation 4.3 triggers, attack is said to have occurred. Figure 4.16 shows time series variations in entropy when network is under distributed attack. Figure 4.17 shows time series variations in entropy for network under concentrated attack.

In case of distributed attack, attack is launched with 80 attackers with mean rate varying from .05 Mbps to 4 Mbps per attacker. Clearly in the first time window after attack is launched at 8 seconds, there is a jump in entropy value. The positive jump and persistent high value of entropy as compared to normal or no attack case reflects that it is a distributed attack. Note that mean attacker rate is low and the flows have comparatively lesser frequency.

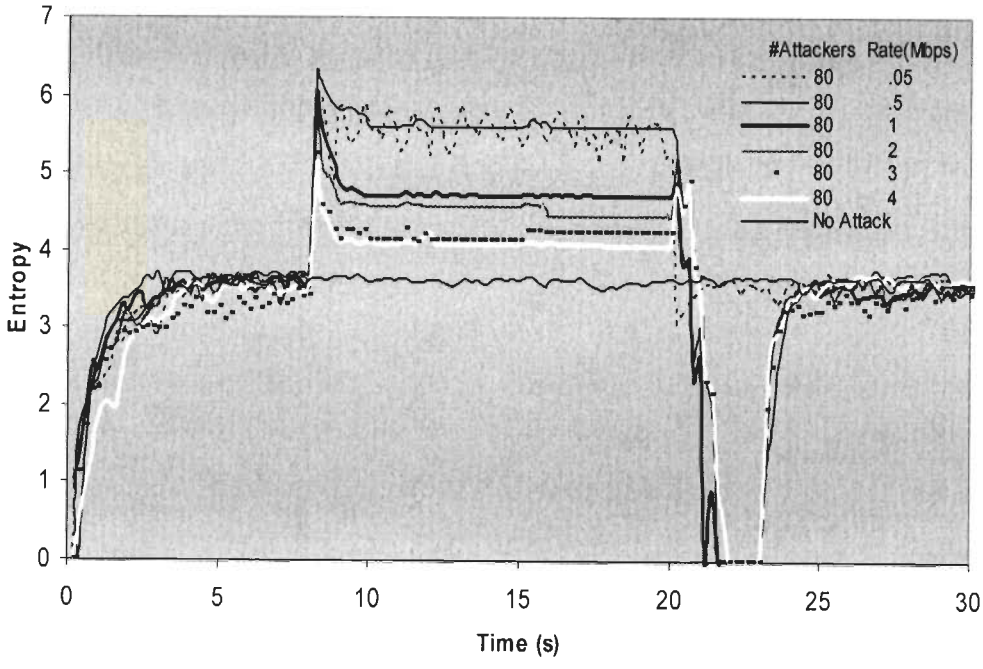


Figure 4.16. Time series entropy variations for DDoS attack

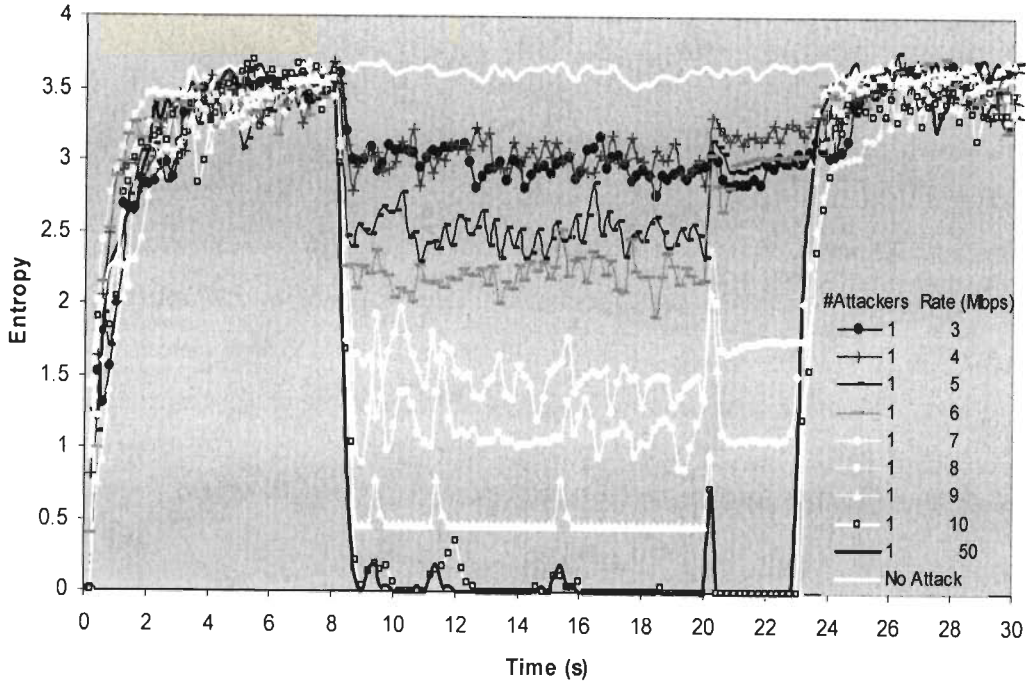


Figure 4.17. Time series entropy variations for DoS attack

As the attacker rate is very low, the traditional volume based techniques [65, 67, 71, 78] are not able to distinguish between attack and normal condition. However Figure 4.16 clearly indicates the change in entropy justifying our claim of picking even a very meek rate attack at macroscopic-level.

An attack in which source addresses were fixed or drawn from a small set produced similar dramatic results with a significant drop in entropy values. Entropy decreases in case of such concentrated high rate attack. Figure 4.17 shows entropy profile when network is put under such attack. This represents DoS attack and attack is launched with 1 attacker with mean rate varying from 3 Mbps to 50 Mbps. In the first time window after the attack is launched at 8 seconds, there is a dip in entropy value. The persistent low value of entropy as compared to normal or no attack case reflects that it is a concentrated attack. The mean attacker rate is high and flows which are causing this anomaly are highly concentrated and have comparatively high frequency.

Before the attack begins, source address entropy measurements fall entirely within the range of 3.50661–3.68804. During the attack the entropy increases or decreases depending upon the nature of attack as shown in Figure 4.16 and Figure 4.17. Normal system entropy range defined between 3.50661–3.68804 would detect these attacks without generating FP and FN. Also note that in case of distributed attacks, lower the attack rate, more the rise in entropy whereas in case of concentrated attacks, higher the attack rate, more the dip in entropy.

As shown in Figure 4.18, with meek attack rate of 0.05 Mbps per attacker, attack launched with 80 attackers has higher rise in entropy as compared to attack launched with 10 attackers. Figure 4.18 shows that at meek attack rates, there is a higher rise in entropy as attacks becomes more and more distributed. Figure 4.19 shows that at high rate there is a higher dip in entropy as attacks become more and more concentrated. As shown in Figure 4.19, with mean attack rate of 50 Mbps, entropy of the attack launched with single attacker nearly dips to zero.

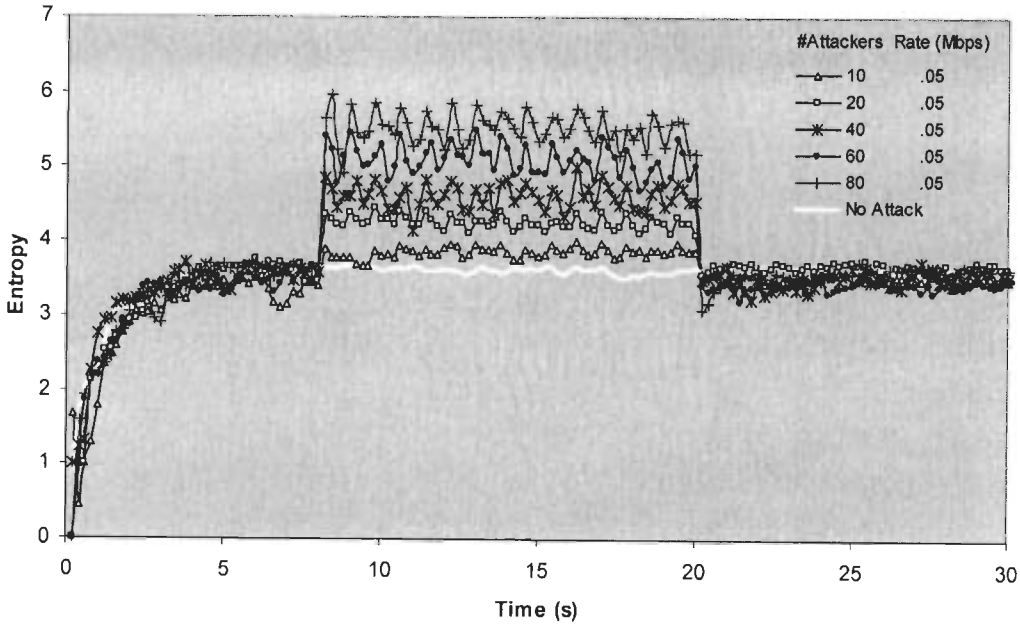


Figure 4.18. Time series entropy variations for low rate attacks

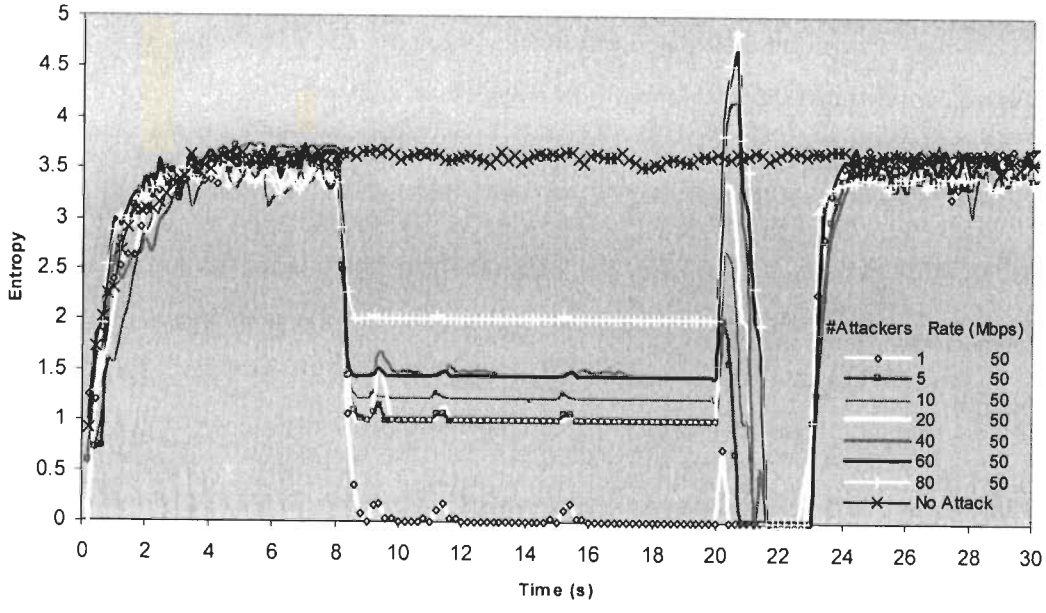


Figure 4.19. Time series entropy variations for high rate attacks

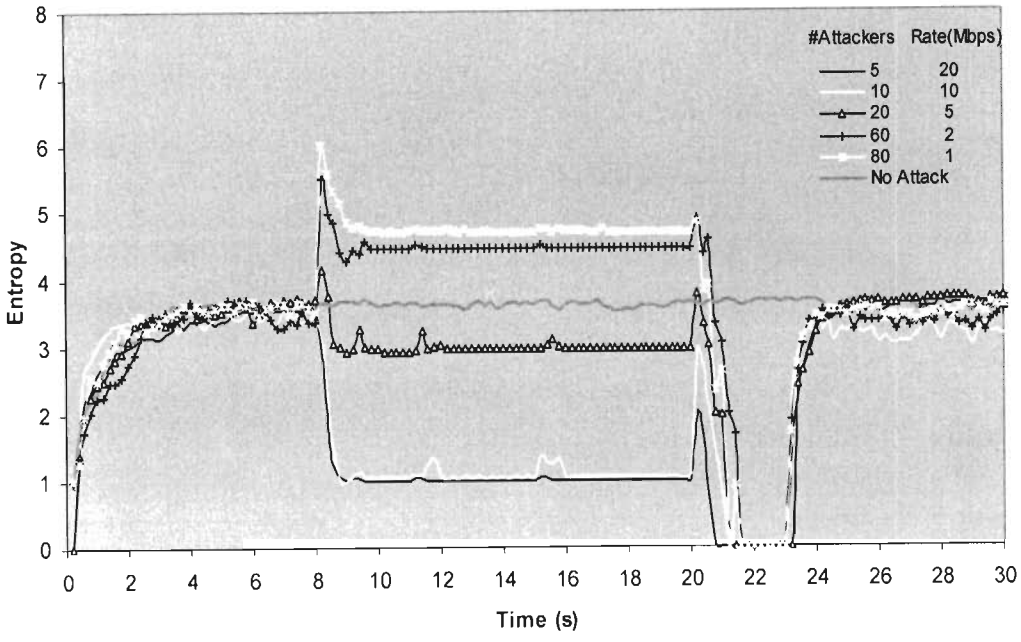


Figure 4.20. Entropy fluctuations for concentrated high rate attacks and distributed low rate attacks

Figure 4.20 shows various attacks at an attack load (*AL*) of approximately 100 Mbps. At meek attack rates, more distributed the attacks, more the rise in entropy. At high attack rates, more concentrated the attack, more the decrease in entropy.

Hence, Figures 4.16 to 4.20 show that concentrated high rate attacks and distributed low rate attacks can be easily detected by source IP based system entropy. To summarize, the results show two noteworthy effects. Distributed low rate attacks have random source addresses and concentrated high rate attacks have fixed source addresses. Legitimate traffic in almost any network contains relatively fixed set of random addresses that constitute a fraction of traffic. Hence uniform distributions usually stand out. This justifies our claim that congestion inducing macroscopic attacks can be detected early i.e. in the transit domain itself by our proposed Ma-LAD.

4.4.2.3 Threshold Optimization in Ma-LAD

According to Equation 4.3, threshold a has been defined as the tolerance factor or design parameter for Ma-LAD.

Figure 4.21 shows the entropy profile for legitimate traffic and for two different kinds of attack traffic. The first case is concentrated high rate attack, where attack is launched with single attacker at a rate of 50 Mbps. The second is the distributed low rate attack, where attack is launched with 80 attackers at mean rate of 0.5 Mbps. These represent the extreme cases of their nature of attacks (refer to Figure 4.16 and 4.17). We calculate system entropy in sliding time window. In Figure 4.21, X-axis shows the system entropy and Y- axis shows the number of time windows in which the value of system entropy was observed during the entire simulation. The histograms show that the variation in entropy statistics due to fluctuations in legitimate traffic is different when compared to deviations caused due to attacks. Simple threshold setting with system entropy of normal traffic defined in range 3.50661-3.68804, will detect these attacks consistently while yielding very few FP.

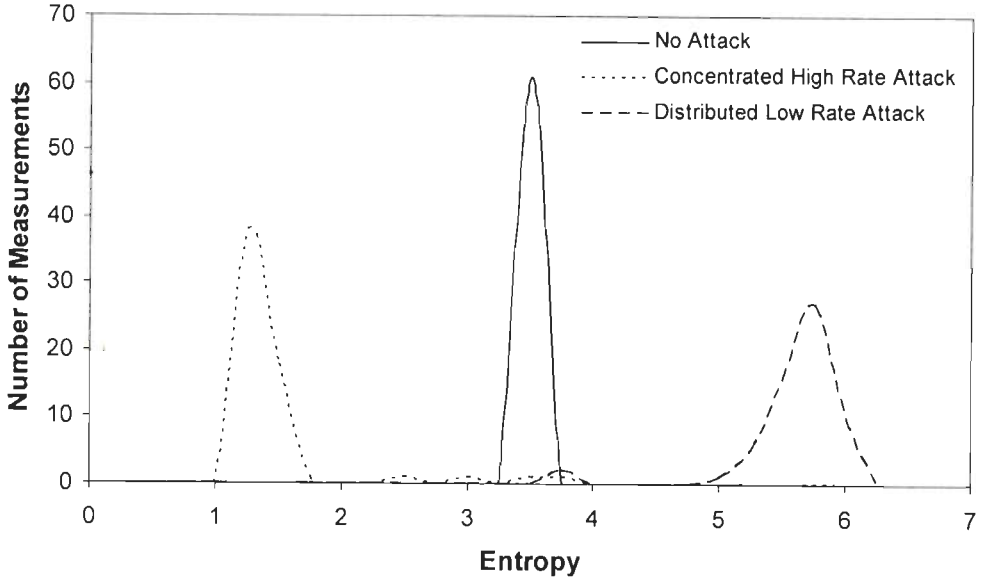


Figure 4.21. Distribution of source IP based system entropy under normal and DDoS attack conditions

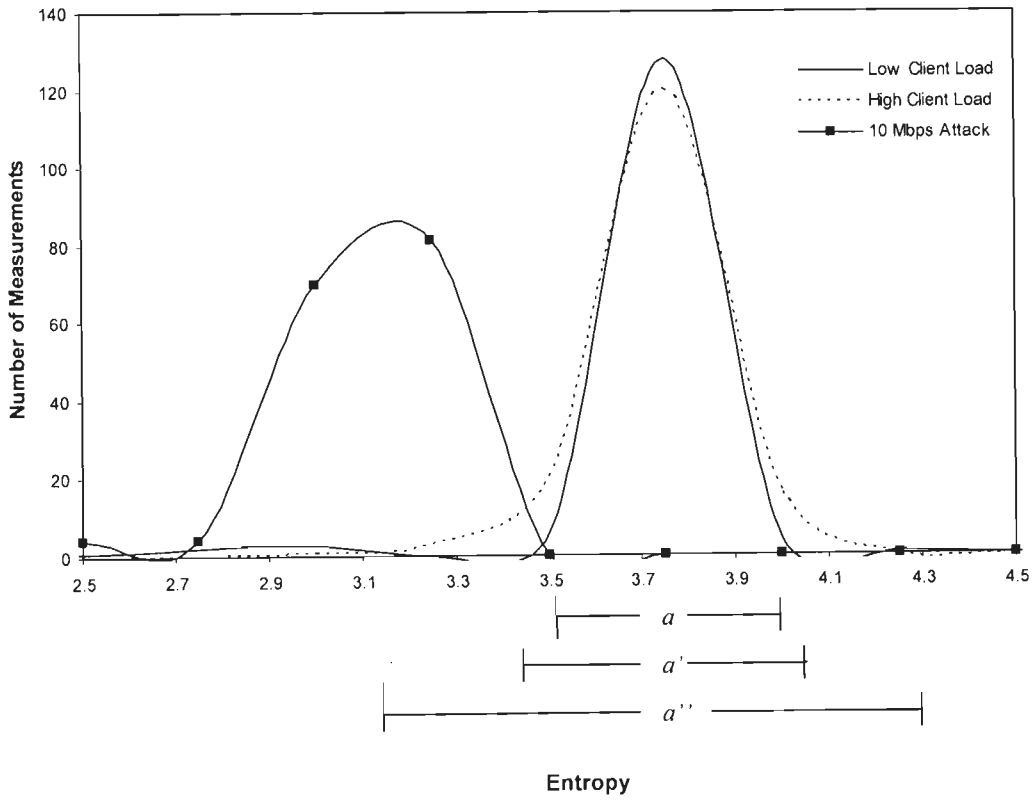


Figure 4.22. Distribution of source IP based system entropy under normal and typical-DDoS attack conditions

From Equation 4.3 and Figure 4.9 it is clear that greater the value of threshold a, greater the entropy bandwidth that classifies attacks as legitimate, lower the detection rate. On the other hand, lower the value of a, smaller the bandwidth that classifies attack as legitimate, detection rate increases and false alarms also increase. There exist a trade-off between two parameters i.e. detection rate and false alarm rate.

Figure 4.22 shows the entropy profiles of attacks simulated by 5 attackers generating traffic at the rate of 10 Mbps. The network is dynamic in nature and workload of the network is never constant. Client load may vary depending on the number of clients and traffic rate. We conduct different experiments and model two different workloads, low CL and high CL. Figure 4.22 shows that in case of low CL, a low value of tolerance factor i.e. a can easily detect all the attacks and still have low false alarms. Higher value i.e. a'' will reduce the false alarms to zero but lower the detection rate as well. However, at high CL, the parameters like detection rate and false alarm rate become very sensitive to tolerance factor a. This is explained as follows. Figure 4.22 shows that in case of high CL, at low value of tolerance factor a to maximize detection rate, a large number of false alarms are

generated. Higher value i.e. *a*" chosen to minimize false alarms reduces the detection rate to nearly 50%.

For high *CL*, a high value of tolerance factor *a* minimizes the false alarms without impacting the detection rate much if the *AL* is negligible or absent. But in the presence of high *AL*, high value of threshold *a* will reduce the detection rate and generate FN. On the other hand, low value of threshold *a* will give high detection rate but generate high number of FP.

Hence, there is a trade-off between detection rate and false alarm rate. These parameters become more sensitive at high *CL*. As discussed earlier, static or fixed value of tolerance factor *a* does not work in dynamic network environment. We therefore need to quantify and optimize the value of *a* (so as to maximize detection rate keeping false alarms low) according to the network environment and subsequently calibrate the system at optimum entropy threshold values.

We conducted experiments and for our simulation environment, we quantified the tolerance factor *a* to vary between 2 – 9 for different network scenarios. The selection of *a* and its impact on detection accuracy has been analyzed using ROC (Receiver Operating Characteristic) curve. We generate ROC curves to compare the detection performance by varying tolerance factor *a* in different network environments. We then plot sensitivity-specificity curves and determine optimum value of tolerance factor *a* for different attack conditions.

ROC Curve

For any experiment, there are four cases, (i) attack correctly detected as attack or positive (*TP* = True Positive), (ii) attack detected as legitimate or negative (*FN* = False Negative), (iii) legitimate detected as legitimate or negative (*TN* = True Negative) and (iv) legitimate detected as attack or positive (*FP* = False Positive).

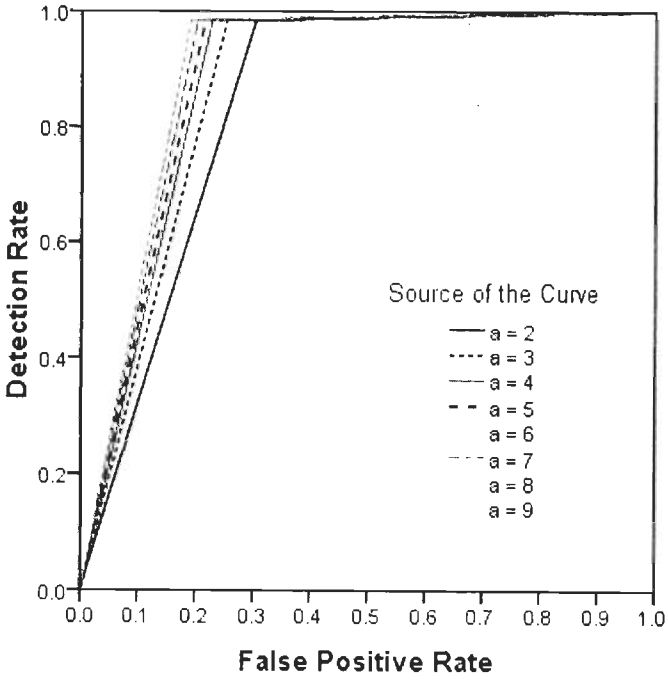
The following statistics can be defined:

Sensitivity: Probability that a test result will be positive when the attack is present (true positive rate or detection rate) i.e. $TP / (TP + FN)$.

Specificity: Probability that a test result will be negative when the attack is not present (true negative rate) i.e. $TN / (FP + TN)$.

In a ROC curve [204], the detection rate (Sensitivity) is plotted in function of the FP rate (100-Specificity) for different cut-off points. We plot ROC curve where each point represents a sensitivity/(100-specificity) (or detection rate/false alarm rate) pair corresponding to a particular value of tolerance factor a. A test with perfect discrimination has a ROC plot that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC plot is to the upper left corner, the higher the overall accuracy of the test. In other words, more the area under the curve, higher the overall accuracy of the test.

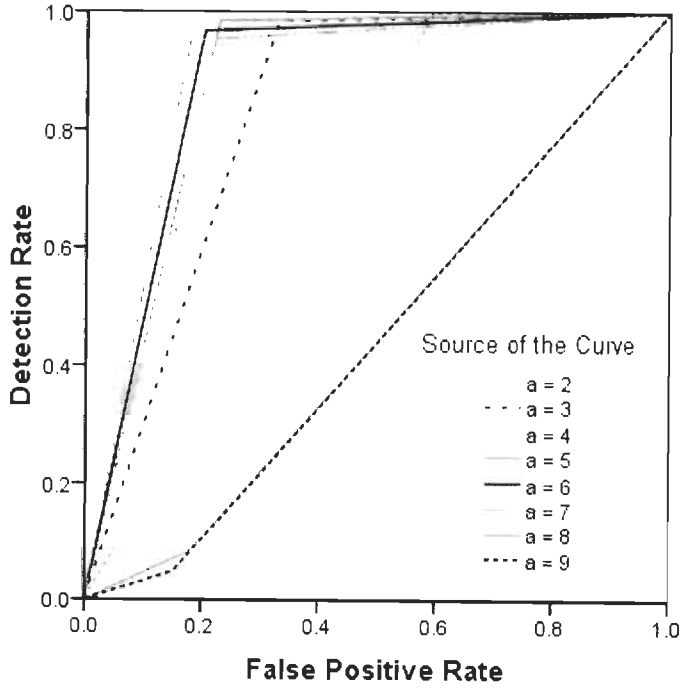
To plot the ROC curves, we vary AL in two ways, first, by varying the mean attack rate keeping the number of attackers constant (Figure 4.23 (a), (b) and (c)) and second, by varying the number of attackers keeping the mean attack rate constant (Figure 4.24 (a), (b) and (c)). In the first case, with 60 attackers and 0.5 Mbps attack rate, values of a equal to 8 and 9 give largest areas under curve (Figure 4.23. (a)), as the attack rate increases to 2 Mbps, maximum area under curve is obtained by a equal to 7 (Figure4.23. (b)) and with a further increase in AL to 5 Mbps, maximum area under curve is obtained at a still lower value of a i.e. $a = 3$ (Figure 4.23.(c)). Hence optimum value of a decreases as the AL increases. A similar trend is seen in Figure 4.24. (a), (b) and (c) where the AL increases by increasing the number of attackers, keeping the mean attack rate constant. As the AL increases, the optimum value of threshold a decreases. We next plot sensitivity-specificity curves to determine optimum value of a in varying network conditions and hence define three modes of operation.



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.840
a = 3	.865
a = 4	.878
a = 5	.884
a = 6	.891
a = 7	.891
a = 8	.897
a = 9	.897

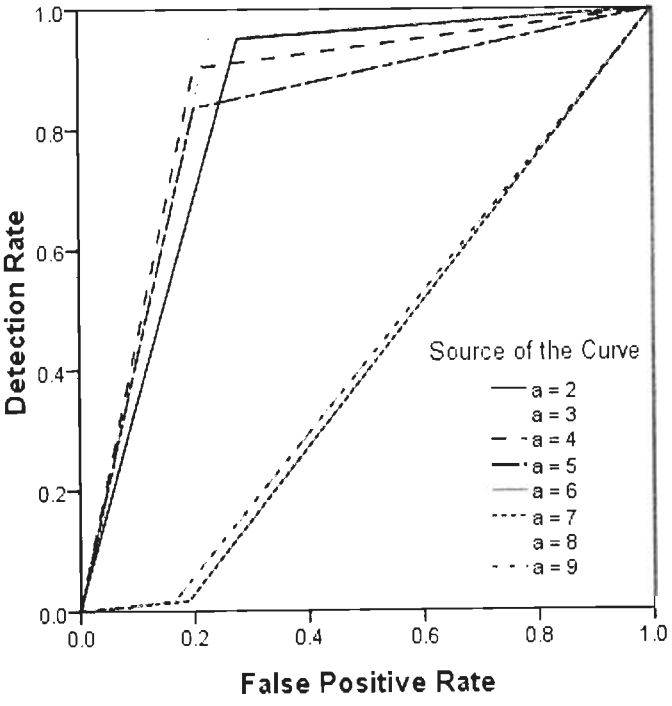
Figure 4.23. (a) ROC curve for 60 attackers; Attack Rate .5 Mbps



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.703
a = 3	.827
a = 4	.872
a = 5	.878
a = 6	.882
a = 7	.887
a = 8	.452
a = 9	.449

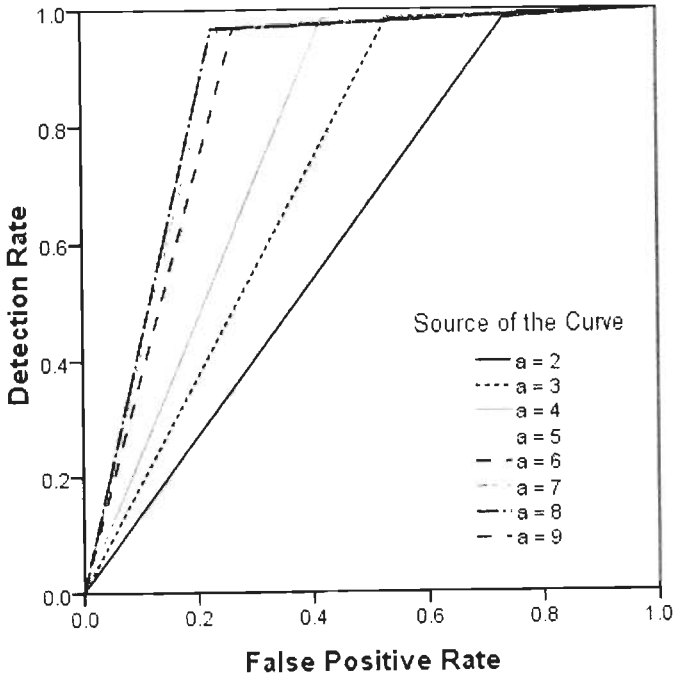
Figure 4.23. (b) ROC curve for 60 attackers; Attack Rate 2 Mbps



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.836
a = 3	.861
a = 4	.850
a = 5	.817
a = 6	.413
a = 7	.413
a = 8	.426
a = 9	.426

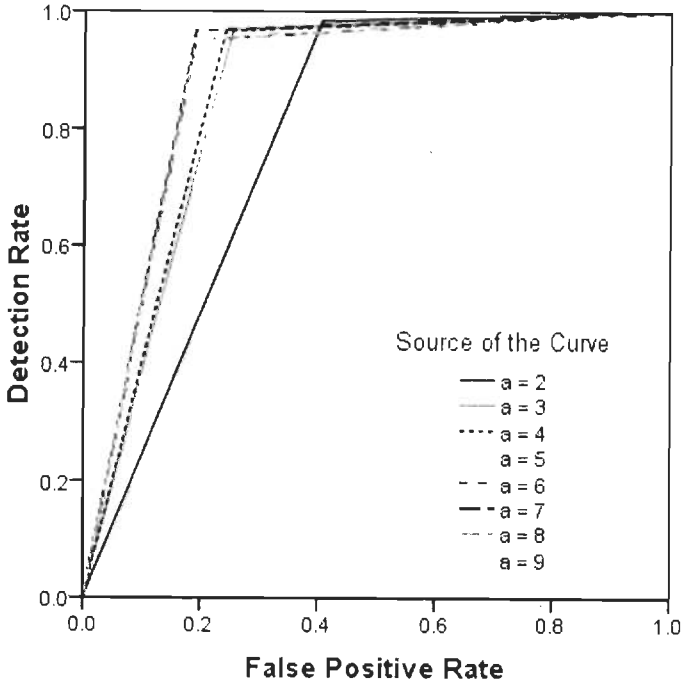
Figure 4.23. (c) ROC curve for 60 Attackers; Attack Rate 5 Mbps



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.625
a = 3	.726
a = 4	.783
a = 5	.806
a = 6	.851
a = 7	.863
a = 8	.870
a = 9	.870

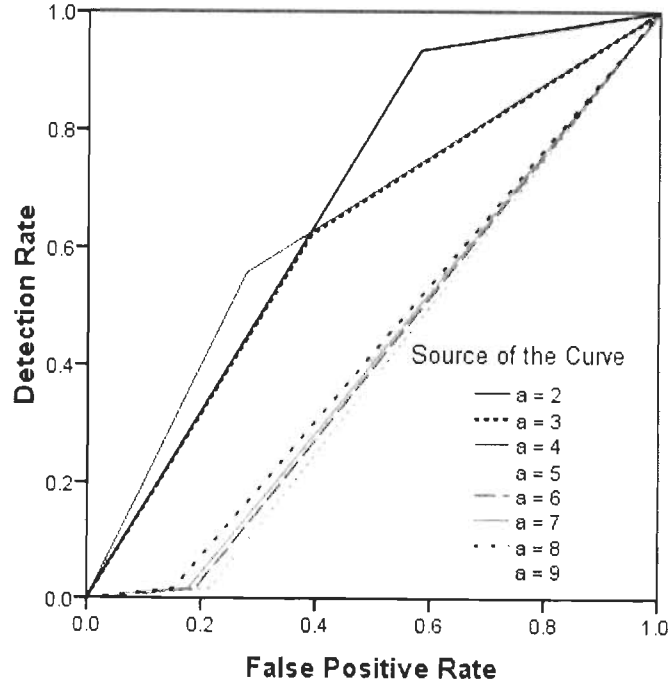
Figure 4.24. (a) ROC curve: 7 Mbps; 1 attacker



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.789
a = 3	.857
a = 4	.863
a = 5	.889
a = 6	.889
a = 7	.880
a = 8	.880
a = 9	.880

Figure 4.24. (b) ROC curve : 7 Mbps ; 10 attackers



Area Under the Curve

Test Result Variable(s)	Area
a = 2	.676
a = 3	.615
a = 4	.639
a = 5	.401
a = 6	.413
a = 7	.420
a = 8	.432
a = 9	.424

Figure 4.24. (c) ROC curve: 7 Mbps; 60 attackers

Sensitivity – Specificity Curve

ROC curves show that different values of tolerance factor give different performance results and have to be optimized according to network conditions. When a higher criterion value is selected, the FP fraction will decrease with increased specificity but on the other hand the true positive fraction and sensitivity will decrease. When a lower criterion value is selected, then the true positive fraction and sensitivity will increase. On the other hand the FP fraction will also increase, and therefore the true negative fraction and specificity will decrease. We plot the sensitivity – specificity curves against different values of tolerance factor a as shown in Figure 4.25 – Figure 4.27. The intersection of the two curves gives the optimum value of tolerance factor a.

As shown in Figure 4.25, at low AL of 0.5 Mbps, optimum value of tolerance factor a varies from 6 to 9. Note that optimum value of a decreases as AL increases as a result of increase in number of attackers. At a higher AL of 5 Mbps, optimum value of a varies from 4-6 as shown in Figure 4.26. However, at a very high AL with 80 attackers (Figure 4.27), optimum value of a varies from 2-4.

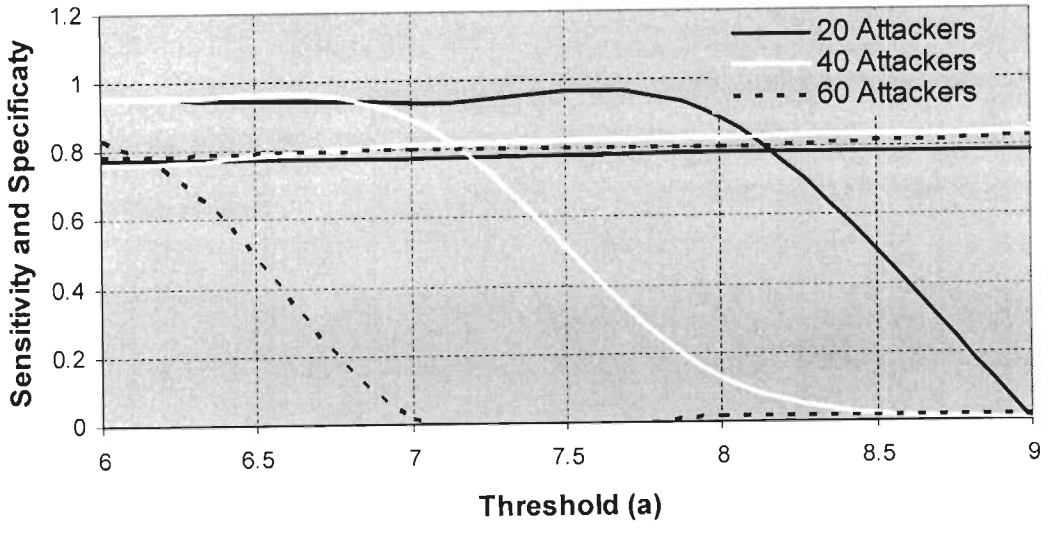


Figure 4.25. Sensitivity - Specificity Curve: AL 0.5 Mbps

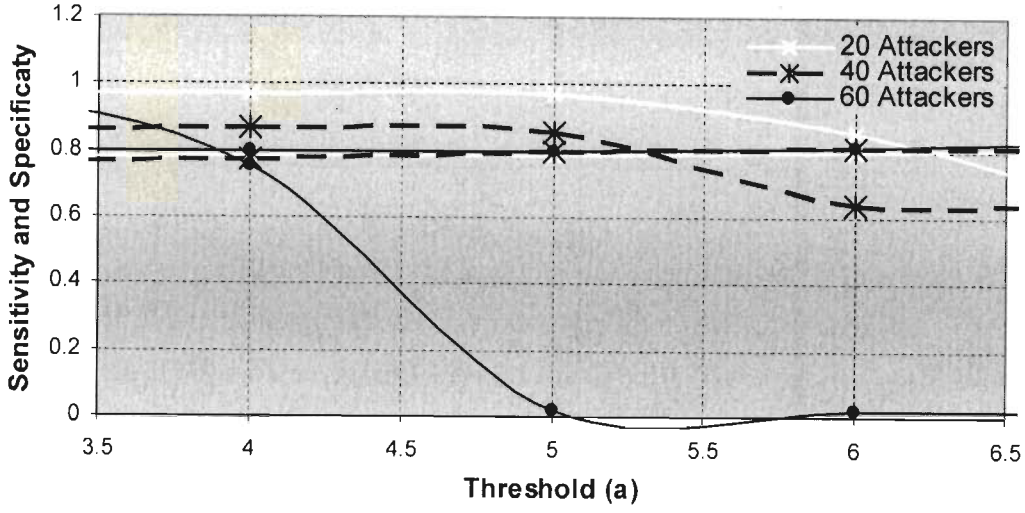


Figure 4.26. Sensitivity - Specificity Curve: *AL* 5 Mbps

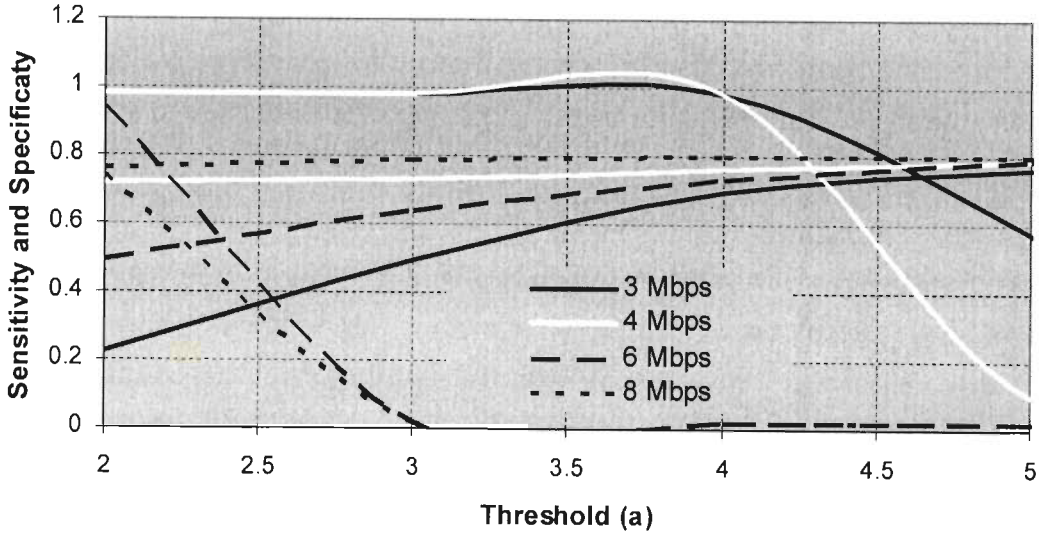


Figure 4.27. Sensitivity - Specificity Curve: 80 attackers

Table 4.3 Mapping *a* to mode of operation

Mode of Operation	*Tolerance factor* a
Naïve Defense	6 - 9
Normal Defense	4 - 6
Best Defense	2 - 4

Hence, the tradeoff between detection rate and FP rate using ROC curves provides guidelines for deciding value of *a* for a particular network environment and hence help in setting thresholds. On the basis of the above observation, we propose calibrate our Ma-LAD to operate in one of the three modes of defense, namely naïve, normal and best defense as defined in Section 4.3.2.3. Table 4.3 lists the values of tolerance factor *a* for different modes of operation.

The results obtained from ROC curves and sensitivity-specificity curves justify our proposal of using naïve defense low *AL* especially if *CL* is high and best defense under high *AL*.

4.4.2.4 Performance Shortcoming of Ma-LAD

If we make more generous assumptions about the attack tools' sophistication and attacker's knowledge, the detection becomes significantly harder. Figure 4.28 shows the results of repeating the previous experiments with stealthy DDoS attacks. The Figure shows that there are certain attacks that cannot be detected by source entropy alone. These are stealthy attacks crafted by sophisticated attackers. The attacker is assumed to know the shape of the source address distribution in legitimate traffic, but not the actual IP addresses in the frequency ranking. The simulated attack traffic thus has the same source-address frequency distribution as the legitimate traffic, but uses a different set of source addresses. There is now a significant overlap between entropy values observed under normal conditions and under attack. Another sophisticated and stealthy attack crafted as described can be more effective against source IP based entropy detection. Specifically, if the attacker knows the approximate values for average source IP entropy value observed by the detector and the fraction of traffic seen by the detector that will be attack traffic, then it can emit attack traffic with an entropy somewhat lower than average value to compensate for entropy increase resulting from the use of disjoint sets of source addresses. This way, an attacker armed with the knowledge of the detector environment could produce attack traffic that would produce little change in entropy observed at the detector.

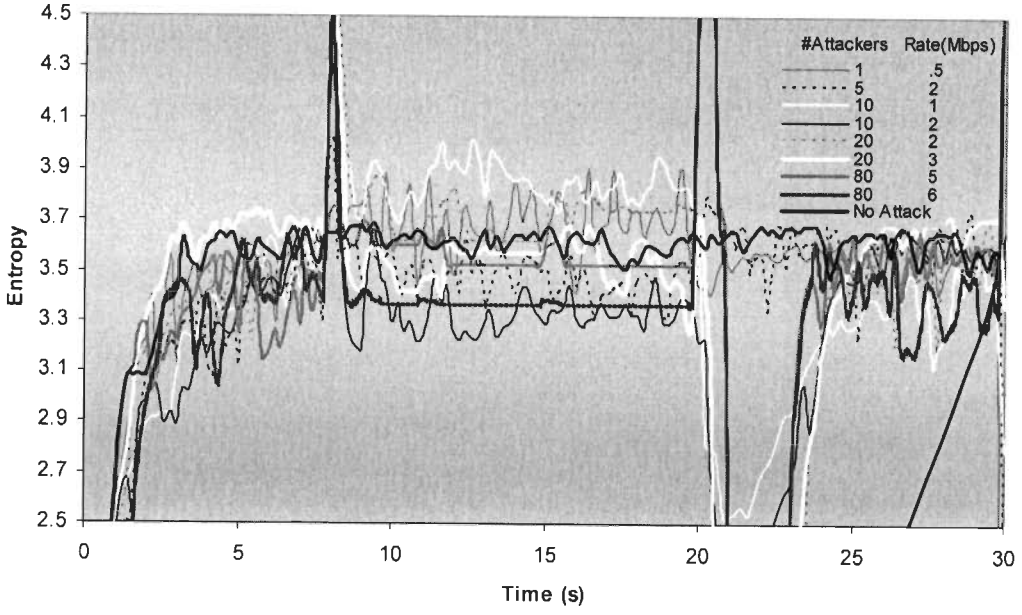


Figure 4.28. Times series entropy variations for attacks with similar statistics as normal traffic

Hence we conclude that macroscopic-level detector based on source IP based system entropy is not an adequate metrics to detect attacks. It can detect concentrated high rate attacks and highly distributed low rate attacks, but concentrated low rate attacks, certain distributed attacks and stealthy attacks by sophisticated attackers may go undetected. Our proposed Mi-LAD uses destination IP based system entropy to detect these attacks. The scatter plots in Figure 4.29 and Figure 4.30 show the efficacy of using both source and destination IP based entropy (Figure 4.30) instead of using only source IP based entropy (Figure 4.29) to detect the DDoS attacks.

The entropies of source IP aggregates may increase or decrease abruptly when the abnormal traffic is introduced in the network. Specifically, source IP based system entropy increases in case of highly distributed low rate attacks (e.g. DRDoS) and decreases in case of concentrated high rate attacks (like DoS). This is shown in Figure 4.29 where 80 attackers with 1 Mbps represent the former case whereas single attacker with 10 Mbps represents the latter case. However, as shown in Figure 4.29, the source IP based system entropy for certain crafted attacks (for e.g. attacks launched by 10 attackers) did not show remarkable distinction from source IP based system entropies of normal traffic (as shown in Figure 4.29).

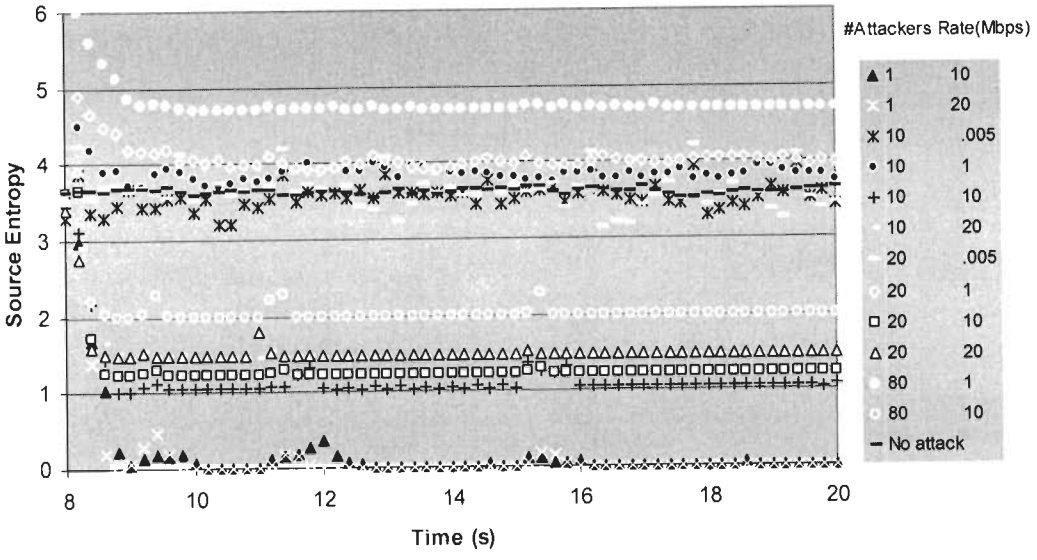


Figure 4.29. Scatter plot based on source IP based system entropy

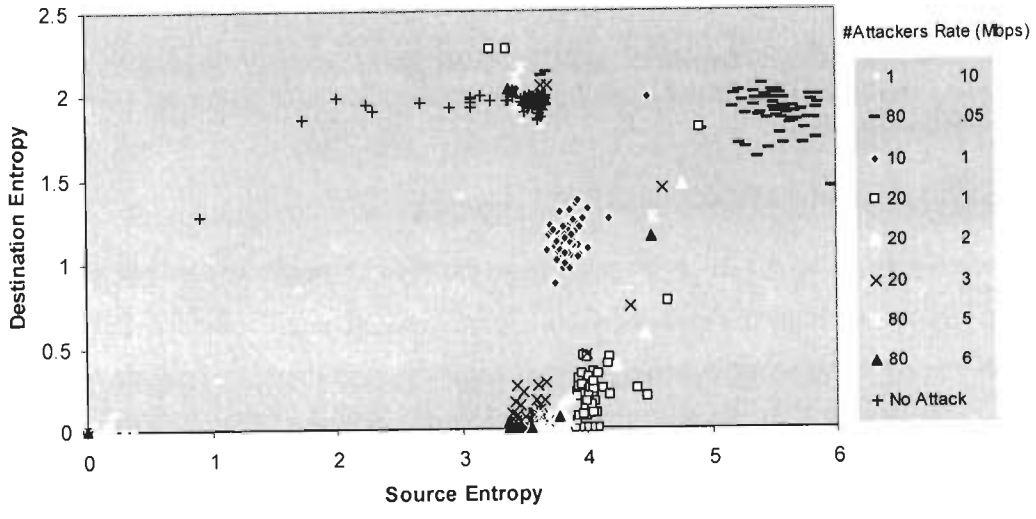


Figure 4.30. Scatter plot based on source and destination IP based system entropy

Figure 4.30 shows the time-series data of entropy transformed into the two-dimensional domains to help visually inspect the attacks. The X-axis represents the entropy of flows defined as source IP aggregates and Y-axis represent entropy of flows defined as destination IP aggregates. For each sample period we calculate the entropies and map them in the two dimensional plane.

The separation of normal traffic and attacks that have been crafted to match normal traffic characteristics, is clearly conceivable in two dimensional domain represented by source and destination as shown in Figure 4.30. This is because as the attack is concentrated on a specific destination, there is a decrease in destination IP based system entropy along with a change in source IP based system entropy. Hence normal and abnormal points are conceivably separated on basis entropy applied on both the source and destination domains.

4.4.2.5 Detection of Attack by Mi-LAD

This section documents the results of Mi-LAD obtained by applying cumulative sum over destination IP based system entropy to detect DoS and DDoS events for traffic traversing edge router of stub network. We performed exhaustive experimentation and look more closely at the distribution of destination IP address based entropy measurements for legitimate traffic and different kinds of DDoS attack traffic. We then apply CUSUM algorithm to the destination IP address based entropic measurements.

First, we conducted simulation experiments for finding out threshold for destination IP address based system entropy under normal condition as per simulation parameters given in Section 4.4.1. The normal system entropy, by using frequency distribution of number of packets per flow id for microscopic level attack detector in time windows of 0.2 seconds, lies in a small range. Simulation is also carried by taking longer window of 1.0 second. Deviations are still lesser as expected however average is almost same.

It is found that normal source IP address based system entropy value lies in small range 1.84407-2.075888. The average is 1.959979, standard deviation is 0.026708, and maximum absolute deviation from average is 0.115909. Finalized simulation parameters are:-

Normal Entropy Value ($H_n(X)$):-1.959979

Maximum absolute deviation from average (d):-0.115909

Figure 4.31 shows the time series of source IP based system entropy under normal conditions and in the presence of sophisticated DDoS attacks like 20 attackers with attack rate 2 Mbps and 3 Mbps and 80 attackers with 5 Mbps. There is a significant overlap in the time series of source IP address based system entropy for normal and attack conditions which shows that detection of such attacks is not possible based on source IP address based entropy alone. In such cases where source IP based system entropy does not show remarkable distinction, a significant drop in destination IP based system entropy clearly detects the presence of the attacks. As shown in the Figure in the absence of attack, system entropy for destination IP based flows corresponds to 1.9599 with very minor deviations, and maximum absolute deviation of 0.1159. The entropy drops to 0 as soon as the attack is launched at 8 seconds. This justifies our claim that Mi-LAD can easily detect attacks that are crafted to match statistics of normal traffic and slip past Ma-LAD.

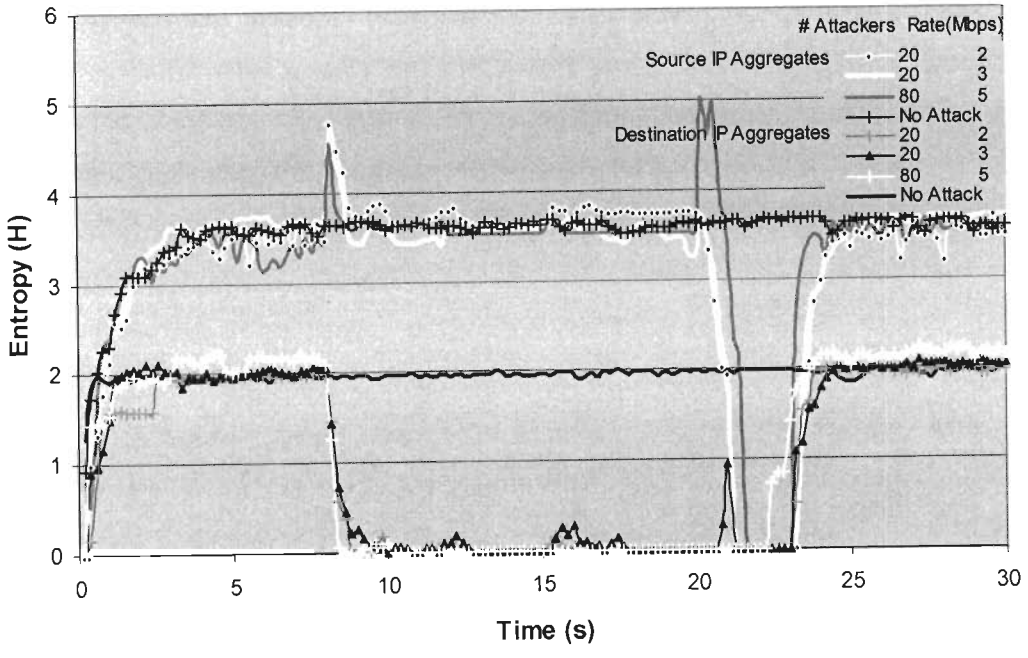


Figure 4.31 Comparison of source and destination IP based entropy time series under similar attacks

CUSUM algorithm has an assumption that in normal case, the average value of random sequence should be negative and it becomes positive after change. We apply CUSUM over destination IP based system entropy and without losing any statistics properties, we transfer the random sequence $\{Y_n\}$ (refer to Section 4.3.3.2) to another random sequence $\{Z_n\}$ with negative average value using Equation 4.4. In our experiments $\beta = \alpha = 1.959979$. Whenever the attack happens, Z_n will become large and positive. The detection threshold is the limit for the positive, which is cumulative value of Z_n. Figure 4.32 shows the times series of random sequence Z_n. In normal condition the sequence of Z_n is negative, and sometimes Z_n becomes larger than zero. But when the attack starts at 8 seconds, Z_n increases rapidly. We use the recursive formula given in Equation 4.5 for calculating S_n, cumulative sum of Z_n. The recursive formula cumulates the positive values of Z_n.

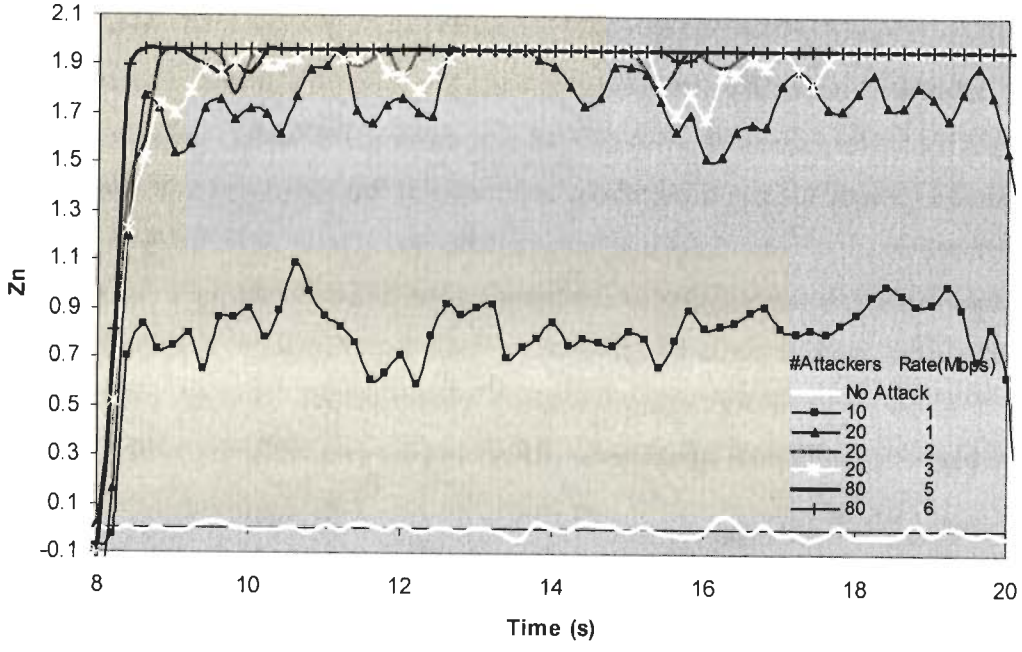


Figure 4.32. Offset statistics Z_n for destination IP based system entropy

Figure 4.33 gives the time series variation of cumulative entropy S_n under normal and varying attack conditions. As shown in Figure 4.33, in the normal case, S_n is 0 or a small positive value close to 0. When the attack happens, this cumulative value S_n increases rapidly. By setting a threshold T equal to 10 (obtained through simulations as shown in Figure 4.33) for our network environment, when $S_n > T$, the system detects the attack. Figure 4.33 shows increase in value of S_n beyond 10 after 8 seconds which detects the attack.

We next calculate C_n, counter parameter that represents a count on time for which network is experiencing a decrease in entropy. C_n is calculated according to the formula given in Equation 4.6.

Figure 4.34 shows the time series variation of counter C_n which judges the persistence of abnormal condition in the network over a time period. By setting the threshold $T' = 5$ for our network environment, when $C_n > T'$, we believe that something abnormal persisted over network tolerance limit and network is attacked. The flash crowds persist for a very small duration and are represented by small positive fluctuations that lie below threshold T' as shown in the Figure. Hence, results in Figure 4.34 justify our claim that the approach is able to differentiate between DDoS attacks and flash crowds.

The judgment function is given in Equation 4.7. The judgment function is based on two events. As soon as any one of the two events becomes true, the value of judgment function is triggered to 1 and attack is said to have occurred. For the cumulative entropy detection approach, we make use of a process to cumulate entropy. Figure 4.33 and 4.34 show the time series statistics of two events as described above. From Figure 4.33, we see that if there is a rapid increase in cumulative entropy S_n, and it reaches a value above the threshold T, the attack is detected. Moreover, a system detects an attack even if S_n lies below T but persists for more than threshold time limit T'. This is shown in figure 4.34.

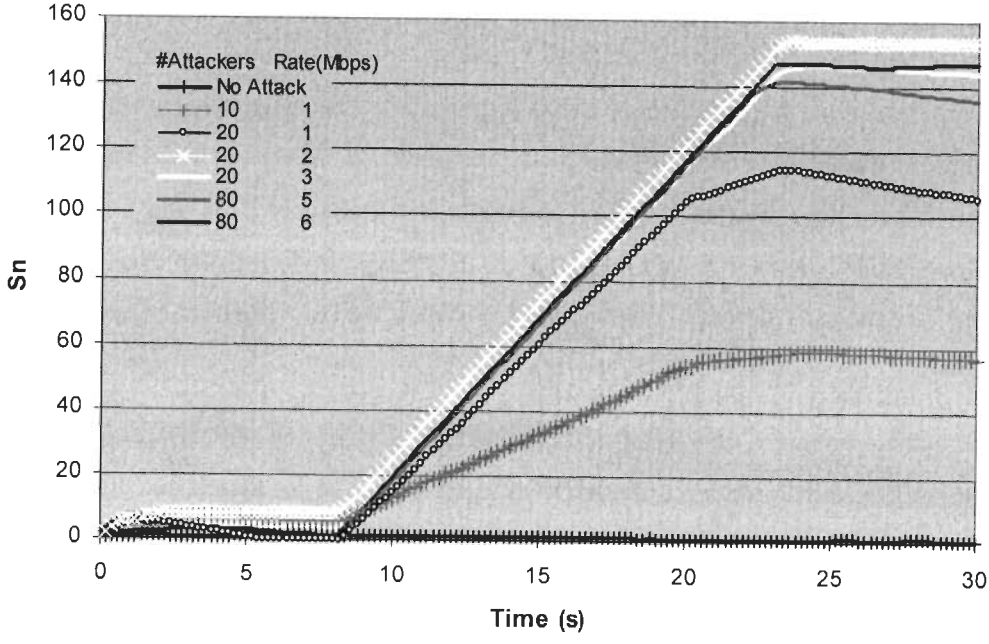


Figure 4.33. Time series variation of cumulative entropy S_n

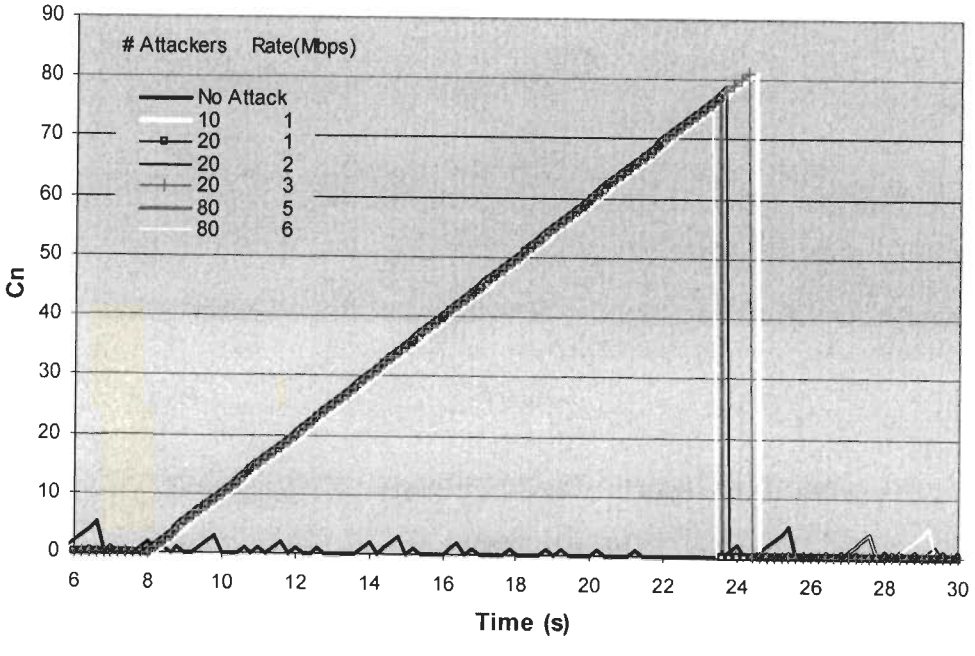


Figure 4.34. Time series variation of counter C_n

4.4.2.6 Sensitivity of Detector to Attack Detection

We document the sensitivity of entropy based approach of our detector to DoS and DDoS event detection. We simulate 10 Mbps legitimate traffic originating from 15 clients picked randomly with inter-arrival time of 0.1 seconds. *AL* is varied from 100 to .0001 Mbps representing thinning factor from 0 to 1000000 respectively. DoS and DDoS attack with 80 attackers is launched by sending attack traffic towards victim. Table 4.4 shows the attack rate in Mbps corresponding to various thinning factors. The results are given in Figure 4.35.

Our detectors provide a very high detection rate which almost equals 1 at high attack rates. Even at low attack rates, our detectors are very effective for attack detection. Figure 4.35 shows that high detection rates are possible for much lower intensities of attack. For example, a detection rate of nearly 98% is possible for DoS and DDoS events comprising only 0.90% of the total traffic. When attack traffic comprises 0.09 % of total traffic on average, the detection is still effective but to a lesser degree.

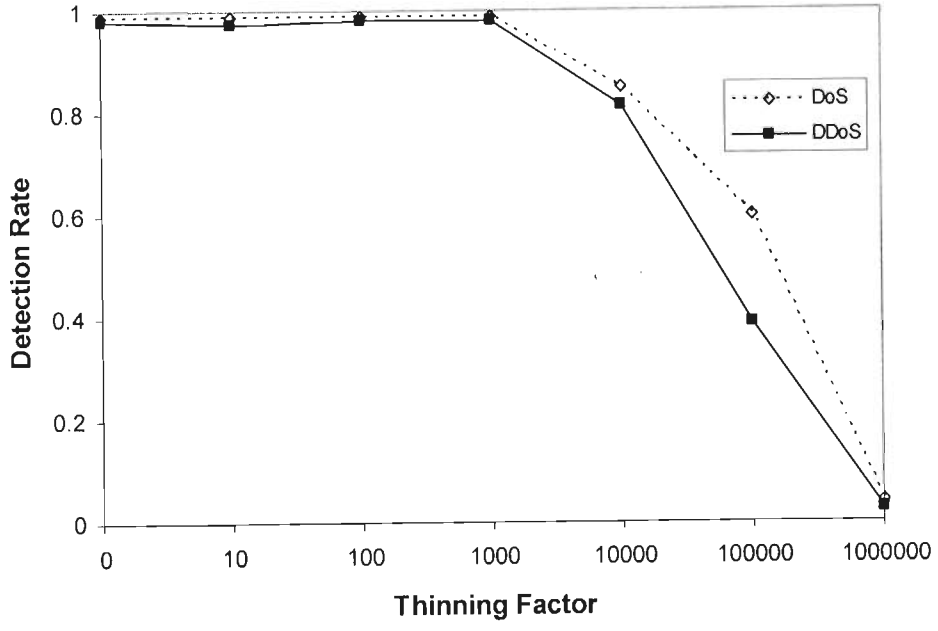


Figure 4.35. Sensitivity of detector to attack detection

Table 4.4 Intensity of DoS and DDoS attacks

S. No.	*Thinning Factor*	*Attack Rate (Mbps)*
1.	0	100
2.	10	10
3.	100	1
4.	1000	0.1
5.	10000	0.01
6.	100000	0.001
7.	1000000	0.0001

It is imperative to have low computational cost for operations carried out on each packet. Using the prototype SNORT [45] detector implementation on 1 GHz P III machine, a single feature entropy detector operates at 294,000 packets per second. If the implementation is optimized by eliminating floating point operations during per packet-handling, and the true frequency profile is kept the same, the single feature entropy detector can process 5,60,000 packets per second. This approximates to 150 Mbps that is greater than OC3 range. Moreover, the CUSUM algorithms are based on single-stage, recursive, exponentially weighted estimators. They need to do only basic operations of cumulating and comparison in each sampling period. A very little amount of memory and resource is required to store the counters. Therefore the algorithms are the least complex and have the low memory requirements. Hence our detection techniques do not have computational and memory overheads.

4.5 Conclusions

In this chapter, we present dual-level attack detection scheme. We propose simple, robust algorithms that are computationally fast and based on easily accessible information. We treat attacks as events that alter traffic feature distributions. We find that entropy is an effective metric to capture unusual changes in the traffic feature distributions. Macroscopic-level detectors operate on edge routers of transit network and are based on time series variation of source IP based system entropy whereas microscopic-level detectors operate on edge routers of stub network and are based on time series variation of destination IP based system entropy.

The results in this section are encouraging for the use of dual-level attack detection scheme. The two step approach coupled with honeypots has a considerable diagnostic power, high scope and is appropriate for detecting a wide range of DDoS attacks. It is capable of handling meek, sophisticated as well as highly distributed attacks. Most of the congestion inducing attacks are detected at macroscopic-level early in the network whereas stealthy and sophisticated attacks that remain undetected at macroscopic-level are detected at microscopic-level near the victim. The detection threshold value is adaptive and optimized according to network conditions. This minimizes FP and maximizes detection rate. Since the number of legitimate sources blocked is minimized, it reduces the collateral damage. Depending on the threshold, proposed scheme operates in one of the three modes of defense among naïve, normal and best defense. The proposed approach has a reasonable foundation to adapt according to different environments and traffic conditions. Results demonstrate that by using CUSUM algorithm over entropy for change point detection, flash crowds can be easily distinguished from the attacks at microscopic level. Honeypots compliment the entropy detection effectively to suppress the FN and improve the accuracy of detection.

The proposed detection scheme is accurate and has a high detection rate even when it comprises a small fraction of total traffic. According to simulation results, our proposed scheme can capture attacks even if they comprise just 0.09% of the total traffic. Even very meek rate DDoS attacks are detected reliably early in the network. They have modest per packet computational requirements and memory usage and this makes real time processing at high bandwidths practical. Besides being computationally fast and accurate, it adapts to varying network conditions. Hence the detection schemes discussed in this chapter can be used to estimate the presence of DDoS attack early and reliably on the Internet.

Chapter 5

Dual – Level Characterization and Response

5.1 Introduction

Characterization of traffic is of immense importance to DDoS defense framework. Once the DDoS defense system has detected that attack is really going on at a point of time, it must take immediate response decision. Accurate traffic characterization is required before any response decision can be taken. This forms the second line of defense in the proposed framework. Characterization means the process of differentiating attack traffic from legitimate traffic. The response system either drops the attack traffic in a timely fashion or renders them harmless by redirecting them into a trap for further evaluation and analysis.

The special feature of DDoS attack packets is that each individual packet is perfectly legitimate. However correlating these packets when monitored at different points can give some signs of uniqueness from legitimate traffic. Despite a large literature available, DDoS attack characterization and response problems remain unresolved. Detection scheme is always imperfect, so that both false alarms and detection failures are possible. Imperfections are possible both with regard to the detection of attack as a whole, and identification of flows that belong to attack traffic. Hence some attack packets may be missed and some legitimate packets may be incorrectly dropped.

Several schemes have been suggested to characterize attack traffic from normal traffic [45, 46, 51-55, 65-70, 72, 75, 78, 80-91, 93-98, 100-108, 111, 113-115]. As in detection, one can choose between signature based techniques [45, 46] and anomaly based [65-70, 72, 78, 81-85, 88, 103, 108]. Signature based techniques [45, 46] can identify only known attacks whereas in anomaly based techniques [65-70, 72, 78, 81-85, 88, 103, 108], it is difficult to build accurate profile of legitimate traffic and hence they generate high rate of false alarms. Moreover, the Internet traffic is noisy, which makes it difficult to extract meaningful information about attacks from any kind of traffic characteristics. Reason for limited success of attempts at characterization is that they rely on volume based metrics

like [81, 103], which do not provide sufficient information to distinguish attacks and are inaccurate. Some of the solutions have achieved impressive levels of accuracy [21, 93-98, 100], but all suffer from common weakness of themselves being exploited to give rise to DoS attacks.

Most of the techniques for response against DDoS attacks are based on isolating the attack traffic by either directing it to controlled environment or filtering out a part of the traffic [51-55, 75, 78, 80, 111, 113-117]. Filtering techniques depending on inaccurate characterization can lead to high collateral damage as they filter legitimate traffic along with attacks. Moreover, the study in this area is totally disarrayed i.e. different characterization and response methods are proposed using different topologies and attacks. No benchmarks and evaluation criteria exist which can compare different approaches. For a detailed discussion refer to Section 2.3.4.

The main objectives in this chapter are:

i. Characterization system must accurately differentiate between legitimate and attack traffic.

ii. Attack traffic should be identified early in the network and characterization must be prompt so that the action can be taken before the network resources are affected.

iii. The response system should be able to effectively stop a large portion of the attack traffic before it reaches the victim.

iv. Response should ensure good service to legitimate traffic during the attack.

v. Collateral damage due to the response must be lower than the damage suffered by legitimate clients in the absence of response.

vi. Characterization and responsive approaches must accurately work for a wide range of attacks.

To meet the above objectives, we propose a dual-level attack characterization and response technique. We show that by examining traffic feature distributions, we can characterize DDoS attacks in a systematic manner. Upon detection, steps are taken to characterize the traffic as either normal traffic or attack traffic and these characterization results are provided to the response mechanism. Our detection and characterization overlap and the method used to detect the existence of an attack provide necessary information to start responding towards traffic characterized as attack.

Similar to detection, characterization is performed at two levels. At macroscopic-level, characterization is triggered as soon as attack is detected by Ma-LAD and most of the macroscopic attack traffic is identified early at the border router of transit network. Response mechanism at this level selectively drops the congestion inducing attack traffic. The microscopic-level attack characterization is triggered as soon as attack is detected by Mi-LAD at border router of stub network and it identifies the suspicious traffic. The response mechanism then redirects the suspicious traffic of anomalous flows to honeypot for further evaluation. Legitimate traffic is subjected to servers. Our response mechanism works by implementing three rules, namely drop rule, allow rule and redirect rule for filtering attack traffic, isolating suspicious traffic from normal traffic and directing suspicious traffic to controlled environment for further monitoring. Our characterization and response techniques show a promise for complementing the action of detection with minimum collateral damage in addressing the DDoS problem.

5.2 Dual-Level Attack Characterization

The most difficult part for defending against DDoS attacks is that it is very hard to differentiate between normal traffic and attack traffic. Attack detection and hence attack characterization is easiest at the victim network as a high volume of incoming traffic can be readily used to characterize DDoS attack traffic [65, 67, 68, 71]. However, response at victim is not possible against high volume attacks as they overwhelm network resources even before they reach the defense system, leaving legitimate clients without service. Additionally, high level of attack traffic may lead to a non-selective response. Thus, while protecting the victim, the response penalizes some legitimate traffic, leading to collateral damage and denial of service. The selectiveness and effectiveness of response improves as the defense system is moved from the victim, but the detection and hence characterization accuracy deteriorates [39, 73, 77, 80, 134, 150].

To overcome the above location tradeoff, we propose dual-level characterization. The higher or macroscopic-level characterization facilitates selective and effective response away from the victim. Our algorithm for macroscopic-level characterization and response are based on the idea to allow as much traffic as possible into the network that network can tolerate, and identify and drop only the congestion inducing attack traffic. Rest of the traffic is subjected to microscopic-level characterization near the victim domain. It identifies suspicious traffic and as an initial response, redirects the suspicious traffic to honeypot to be monitored for more accurate response decision. This improves effectiveness of response and minimizes collateral damage.

As the detection and characterization overlap, a time series of the source IP based system entropy on border router of transit network (used by Ma-LAD) and a corresponding time series of the destination IP based system entropy on border router of stub network (used by Mi-LAD) calculated during detection provides necessary information for attack characterization. As soon as characterization is triggered by Ma-LAD, the edge routers of transit network, based on packet arrivals of each flow and individual entropy of each flow, calculate the contribution of entropy of the flow in total source IP based system entropy. Set of flows that have least or highest measured packet arrivals and share same or very similar entropy i.e. have minimal variations in source entropy are identified as attack flows. In case of microscopic-level characterization triggered by Mi-LAD (for details, refer to Section 4.3.1), the edge router of stub domain identifies the dominant destination IP based flow as victim of DDoS attack. It then calculates the entropy rate of individual source IP based flows destined on the victim. If a set of source IP based flows that are destined to the victim share same or very similar entropy and have minimal variations in their entropy rate, they are tagged as suspicious.

5.2.1 Macroscopic-Level Characterization

If the Ma-LAD determines that the current source IP based system entropy is below the normal range, it suggests that traffic with a single value of source IP or a relatively small number of values for source IP is dominating. Since the entropy detector tracks the value frequency, characterization can identify which values are the most common and are likely candidates for filtering. For finer targeting, the characterization watches for specific source IP values with dramatic increase in frequency and treat those as attacks (it contributes least to system entropy). Conversely, unusually high entropy source IP based system entropy suggests that low frequency source IP are the cause of attacks. Therefore, the characterization suggests that packets have high frequency values be given preferential treatment and those with low frequency values and most common frequencies be dropped. Though each of such source IP based flows contribute very less to system entropy; but they are large in number, so overall system entropy becomes very high.

In detection phase if $H_c(X)$ is more than normal $H_n(X)$, then suspected malicious flows tend to have lower frequency values of packet arrivals and the attack is termed as low rate degradation attack. While if $H_c(X)$ is less than $H_n(X)$, normal then suspected malicious flows have high values of number of packet arrivals and the attack is high rate. (Refer to Equation 4.3, Chapter 4).

Macroscopic detectors designate different flow id to each unique source IP encountered in incoming packet. At the edge router of transit network, we have aggregate of attack flows and normal flows. Let F represent set of active flows. Then,

$$F = F_n \cup F_a \qquad (F_n \cap F_a = \phi) \tag{5.1}$$

In the above Equation, F_n represent actual normal flows and F_a is set of actual attack flows. Our main task in this module is to find $F_a^* = \{f_1, f_2, ..., f_m\} \subset F$ the set of m malicious flows. Ideally,

$$(F_a^* \cap F_a = F_a) \quad AND \quad (F_a^* \cap F_n = \phi) \tag{5.2}$$

In the above Equation, F_a^* is the set of flows identified as attacks. Now the main problem is to find m:-

- For distributed low rate attacks, m numbers of least measured packet arrival flows constitute F_a^* (m number of flows that contribute least to system entropy. Since m is large, they lead to overall increase in system entropy).
- For concentrated high rate attacks, m number of highest measured packet arrival flows form F_a^* (m number of flows that contribute least to system entropy. Since m is small, there is overall decrease in system entropy).

An estimate of total attack traffic ϕ_a is used to compute m and F_a^*. The expected value of attack traffic ϕ_a can be determined as follows:

$$\phi_a = \phi_{td} - \phi_n \tag{5.3}$$

In the above Equation, ϕ_{td} is the total attack traffic received in $\{t_d - \Delta, t_d\}$ and ϕ_n is averaged total traffic. The values of ϕ_n is calculated by averaging total traffic observed from the time bottleneck link utilization is 1 up to time $t_d - \Delta$.

The number of attack flows m.can be derived from the following Equation:

$$\sum_{j=1}^{m} X_i^j (t_d + \Delta) \le \phi_a \tag{5.4}$$

In the above Equation, i is designated flow, j varying from 1 to m for least or highest measured packet arrivals, and $X(t_d + \Delta)$ represent packet arrivals for flow i in next time window after attack is detected.

The condition given in Equation 5.4 helps macroscopic-level characterization module to segregate m flows, which have either least or highest packet arrivals.

A flowchart in Figure 5.1 details characterization of attack traffic at macroscopic-level. The variables DetectedMacroattack and the array ANFA generated from detection module (Figure 4.3 – Figure 4.5, Chapter 4) together with values of ϕ_{td} and ϕ_n are used to determine m number of malicious flows. After sorting ANFA in ascending order for low rate and descending order for high rate attacks, Equation 5.4 is used to find F_a^* by employing a loop with terminating condition $s < \phi_a$.

5.2.2 Microscopic-Level Characterization

Equation 5.2 must hold for an ideal detection and characterization system. However, such a system may be subjected to high collateral damage, as in an attempt to identify all attack traffic, some normal traffic may be misclassified as attack. Though a system may detect the entire set of attack flow i.e. $F_a^* \cap F_a = F_a$ holds true, but in an attempt to do so, some normal flows will be misclassified as attacks and $F_a^* \cap F_n = \phi$ will no longer be valid.

Hence, at macroscopic-level, set of flows identified as attacks F_a^* are limited to a subset of F_a i.e. set of actual attack flows. They are essentially congestion inducing part of the entire traffic which must be responded to and filtered early in the network. A set of flows

$$F_{\setminus} = F_a - F_a^* \quad (F_a = F_s \cup F_a^* \quad AND \quad F_s \cap F_a^* = \phi) \tag{5.5}$$

remain unidentified at macroscopic-level and is identified at microscopic-level as soon as alarm is generated by Mi-LAD.

As discussed earlier, Mi-LAD designate different flow ids to each unique destination IP encountered in incoming packet for servers to be protected. If there is a decline in system entropy for the destination IP based entropy time series on edge router of stub network, attack is detected. Victim is identified and the characterization is triggered.

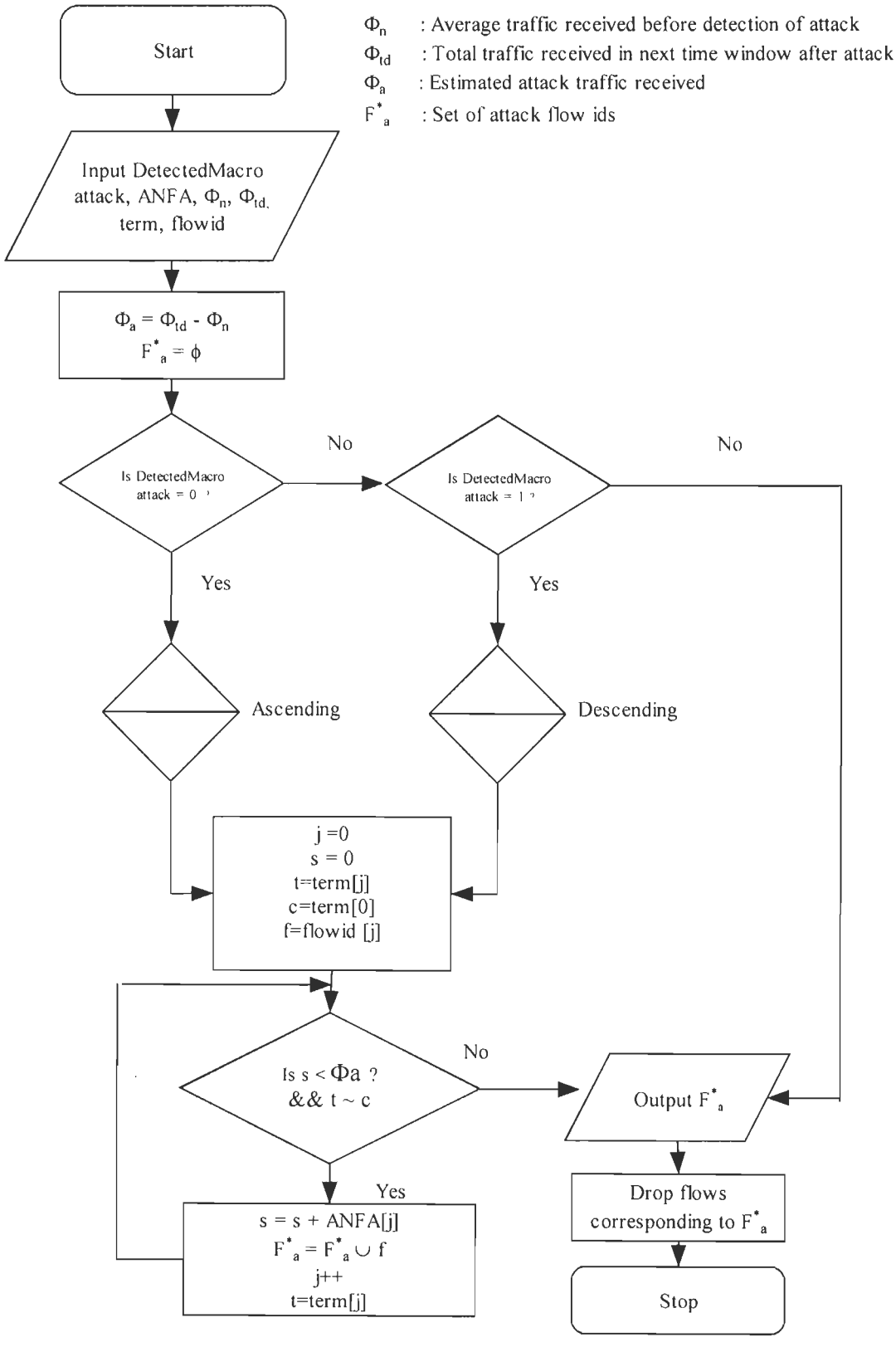


Figure 5.1. Flowchart for characterization of attack traffic at macroscopic-level

5.2.2.1 Identification of Victim

Let $S = \{S_1, S_2, ..., S_k\}$ represent a set of servers to be protected with $D = \{D_1, D_2, ..., D_k\}$ representing destination IP based flow id for each server in *S*. Let *X(t)* represents the number of packet arrivals for a flow in $\{t - \Delta, t\}$ where Δ is a constant time interval called time window. When there is an attack, the destination IP based system entropy Y_n (refer to Section 4.3.3.2) decreases dramatically, because there is one flow dominating the router. The edge router treats dominant flow as victim of DDoS attack. For the flow id having highest frequency of packet arrivals and least contribution to the destination IP based system entropy, the corresponding server is identified as victim of the attack.

$$Victim = \min(Y_n^1, Y_n^2, ..., Y_n^k) \quad and \quad \max(X^1(t), X^2(t), ..., X^k(t)) \tag{5.6}$$

In the above Equation, *k* is the number of unique destination IP based flows or the number of servers.

5.2.2.2 Technique for Microscopic-Level Characterization

Once the victim has been identified, the edge router of the stub network starts to calculate the individual entropy rate of each source IP based flow destined on the victim. Note that for microscopic-level characterization, flow is defined in similar way as for macroscopic-level characterization i.e. each unique source IP is a distinct flow. Each source IP based flow's entropy and its contribution towards total source IP based entropy on the victim is calculated on edge router of stub network. If a set of source IP flows share same or very similar entropy and there are minimal variations in their entropy rate e.g. entropy rate is zero or less than a threshold value, they are suspicious and tagged as attacks.

A flowchart in Figure 5.2 details out characterization of attack traffic at microscopic-level. The variables DetectedMicroattack and the array DANFA and Dterm generated from detection module (Figure 4.10 – Figure 4.12, Chapter 4) are used in the characterization. Dterm is sorted in ascending order to identify the victim server according to Equation 5.6. Array DSterm stores the entropy of individual source IP based flow destined on victim and is used to identify the suspicious attack flows.

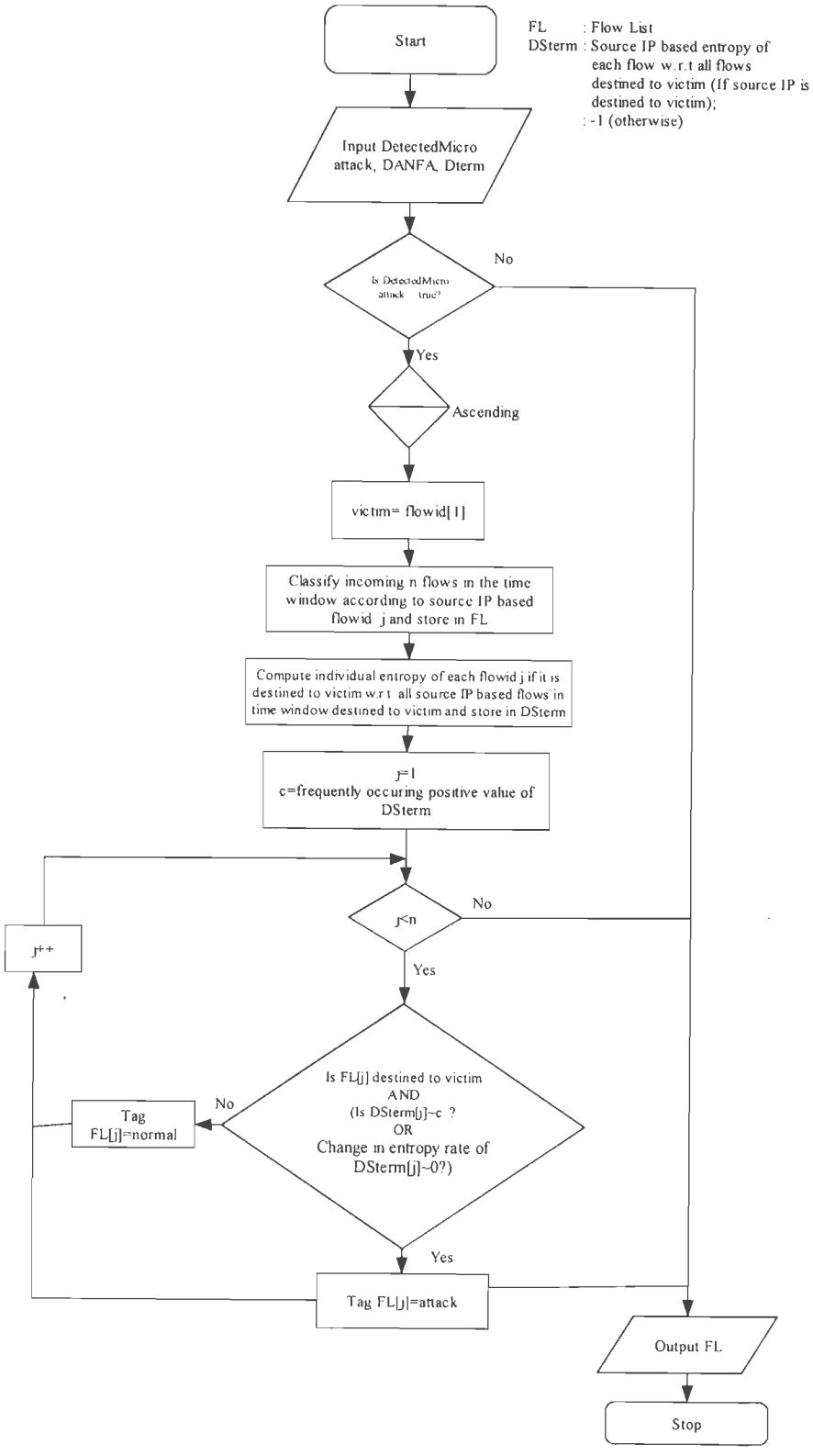


Figure 5.2. Flowchart for identification of victim and characterization of attack traffic at microscopic-level

Our proposed microscopic-level characterization technique is based on following notion. The attackers or attack tools use same mathematical functions to control the speed of attack packets sent to the victim. In an attack scenario, an attacker uses a random variable X to control the generation speed of attack packets. For example, using a constant speed to generate the packets, namely, $P\{X=C\} = 1$, and C is a constant; increasing the number of attack packets according to attack time t, $X=a.t+b$, a and b are constants; simulating the network accessing as Poisson process, $P\{X = k\} = \lambda^k e^{-\lambda}/k!$, k=0,1... and λ is a constant; and so on. Based on this observation, the different attack flows of a DDoS attack share the same regularities, which are different from normal traffic in a short time period.

We assume that attackers use same function to generate attack packets at the zombies. We employ entropy rate, which is the rate of growth of entropy, for the source IP based flows destined to victim identified to be under attack. As discussed earlier, if the flow has entropy rate zero or less than threshold, it is a suspect. Due the same regularities shared by different source IP based flows, set of flows which share same entropy and minimal variations in entropy (i.e. entropy rate is zero or less than a threshold) are considered suspects. In our analysis, we treat the attack flows as 'signal', which we expect to identify and the legitimate accessing is treated as 'noise'. To disable the above technique, attackers may use multiple attack packet generation functions in one attack, e.g. they may use random functions with different seeds. However, we assume that there are definite and finite rules behind the attack distributions. Our aim is to be able to distinguish signals of attack from noise.

We suppose there are n zombies and there is a master or an attacker that triggers the zombies to launch the attack at a point of time. The attacker uses same mathematical function with a random variable X to control the speed of attack packets. Because of the CPU differences and network delay differences to the victim, the real attack time and attack speed may differ for zombies. We prove that it is a linear relationship and a regularity exhibited by different attack flows if they use same mathematical function despite varying speed and time of attacks (based on assumption that the network is linear and stable during a short time period).

Let X_i represent the attack speed of attack flow i

$$X_i = f(X) = a_i X + b_i, \qquad a_i, b_i = C, \qquad i = 1,2,...,n$$

Following theorem is used to prove that the traffic feature distribution is not effected by delays and there is a regularity during the launch of attack provided mathematical functions to control speed of attack is same and network is linear and stable during a short time period.

Theorem: For random variable X, and $Y = f(X)$, if $f(.)$ is a linear function, then the entropy $H(X) = H(Y)$

Proof:

Suppose X is a discrete variable, and $X \in \{x_1, x_2, ..., x_n\}$, then $Y \in \{f(x_1), f(x_2), ..., f(x_n)\}$. Because of the mapping is a one-to-one mapping, therefore, the possibilities of each pair in the two domains are the same, respectively.

$$p(x_1) = p(f(x_1)), p(x_2) = p(f(x_2)), ..., p(x_n) = p(f(x_n))$$

Therefore,

$$H(Y) = -\sum_{i=1}^{n} p(y_i) \log p(y_i) = -\sum_{i=1}^{n} p(f(x_i)) \log p(f(x_i)) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) = H(X)$$

The above theorem shows clearly that the entropy of attack packet generation speed of each zombie is the same, and it exhibits regularity, if mathematical function is the same, although the CPU and the network delay may differ among zombies.

From Equation 5.1

$$F = F_n \cup F_a \qquad (F_n \cap F_a = \phi)$$

From Equation 5.5

$$F_s = F_a - F_a^* \quad (F_a = F_s \cup F_a^* \quad AND \quad F_s \cap F_a^* = \phi)$$

Substituting Equation 5.5 in Equation 5.1

$$F = F_n \cup F_s \cup F_a^* \quad (F_s \cap F_a^* = \phi \quad AND \quad F_n \cap F_s = \phi \quad AND \quad F_n \cap F_a^* = \phi) \qquad (5.7)$$

where, F_a^* is the set of flows identified as attack and filtered at macroscopic-level; F_s is a set of flows identified as suspicious attack flows at microscopic-level. Specifically, source IP based flows destined on victim that share same or very similar entropy and have minimal variations in their source entropy i.e. entropy rate is zero or less than a threshold value, are tagged as suspicious attacks and are included in set F_s. Any flow in set F_n

destined to honeypots is tagged as suspicious attack, removed from set F_n and included in set F_s.

We maintain a flow list (FL) at the edge router of stub network which is a subset of flows from set F. In a time window

$$FL(t) = F_n \cup F_s \quad (F_n \cap F_s = \phi) \tag{5.8}$$

where $t = j\Delta$; $j \in N$; Δ is a constant time interval called time window.

Hence the result of the microscopic-level characterization process is recorded in FL which contains source IP based flows, with each flow tagged as either normal or suspicious attack.

Ideally, all the flows in set F_s should be identified during the process of microscopic-level characterization. However in practical implementation, we assume that no system is perfect and only a subset F_s^* of set F_s is identified as suspicious and tagged . A set $F_s - F_s^*$ remains unidentified and results in FN. Similarly subset F_n^* of set F_n may be identified as normal, with $F_n - F_n^*$ resulting in FP.

5.2.3 Response

Our approach to DDoS defense involves response methods which are based on results obtained from characterization. The packets belonging to characterized flows as subjected to one of the three packet filter rules. As a first step to response at microscopic-level, the packets belonging to flows identified as attack are subjected to packet filtering depending upon the intensity of attack and user requirements. At microscopic-level, the response module allows the packets to enter the server or redirects them to honeypots depending on the corresponding flow being classified as normal or suspicious attack based on the results of microscopic-level characterization. Honeypots isolate those stealthy and sophisticated attacks that exploit the vulnerabilities and statistical anomaly detector fails to identify. Thus the overall goal of response module is to allow as much traffic into the network as it can tolerate without compromising with the services to the legitimate requests, and block the congestion inducing traffic early in the network.

Filter Rule: A filter rule is used to drop the packets belonging to source IP based flow aggregates that have been characterized as attacks during macroscopic-level characterization. Because the thresholds and baseline estimates established at macroscopic-level are adaptive and minimize FP, filtering is performed with minimum

collateral damage. Because the mechanism generates filtering rules immediately based on the traffic characteristics it gathers from detection and characterization at macroscopic-level, the response is immediate. Filtering reduces the impact of congestion inducing attack traffic almost instantaneously.

Redirect Rule: This rule redirects the packets to honeypots and is applied to flows that have been identified as suspicious attacks as a result of characterization at microscopic-level. Such packets belong to flows that are either destined to honeypots or the source IP address based flow that have same or similar entropy value and their entropy rate is zero or less than threshold and are destined to victim.

If the flow is found to be anomalous at microscopic-level, it is tagged as suspicious attack otherwise it is tagged as normal during characterization. Instead of just dropping the packets corresponding to suspicious attack flows enroute or resetting sessions, they are actively redirected from hostile sources to honeypot. Honeypot server responds to suspicious attack flows in exactly same manner as would the actual server to legitimate clients. Active server and honeypot both are with the same IP. Since the connection with suspicious attack flows is retained, the flows can be treated as legitimate flows and directed to active server if the corresponding flows are found to be belonging to a set of legitimate normal flows in subsequent time windows, but were momentarily misclassified as suspects. Since the packets corresponding to such flows can be transferred back to actual server in subsequent time window, FN are reduced. However if persistently, in many time windows, the flow remains tagged suspicious attack, connection remains directed at honeypot server. Also one can potentially gain more information about the attacker. Since, instead of dropping such flows, they are redirected to honeypots, they can be analyzed before a final response decision is taken. This reduces collateral damage.

A set of flows $F_n - F_n^* + F_s^*$ are subjected to redirection.

Allow Rule: During DDoS attack, there is a need to allow a particular kind of traffic that belongs to legitimate clients to pass through the network and reach the server or destination. The allow rule is used to allow the packets belonging to flows identified as legitimate during the characterization step.

A set of flows $F_n^* + F_s - F_s^*$ are permitted access to the server.

The production rules for response are as follows:

Production Rules for Response

If
{Flow identified as attack at macroscopic-level}
FILTER
else if
{Flow destined to honeypot OR identified as suspicious attack at microscopic-level}
REDIRECT
else
ALLOW

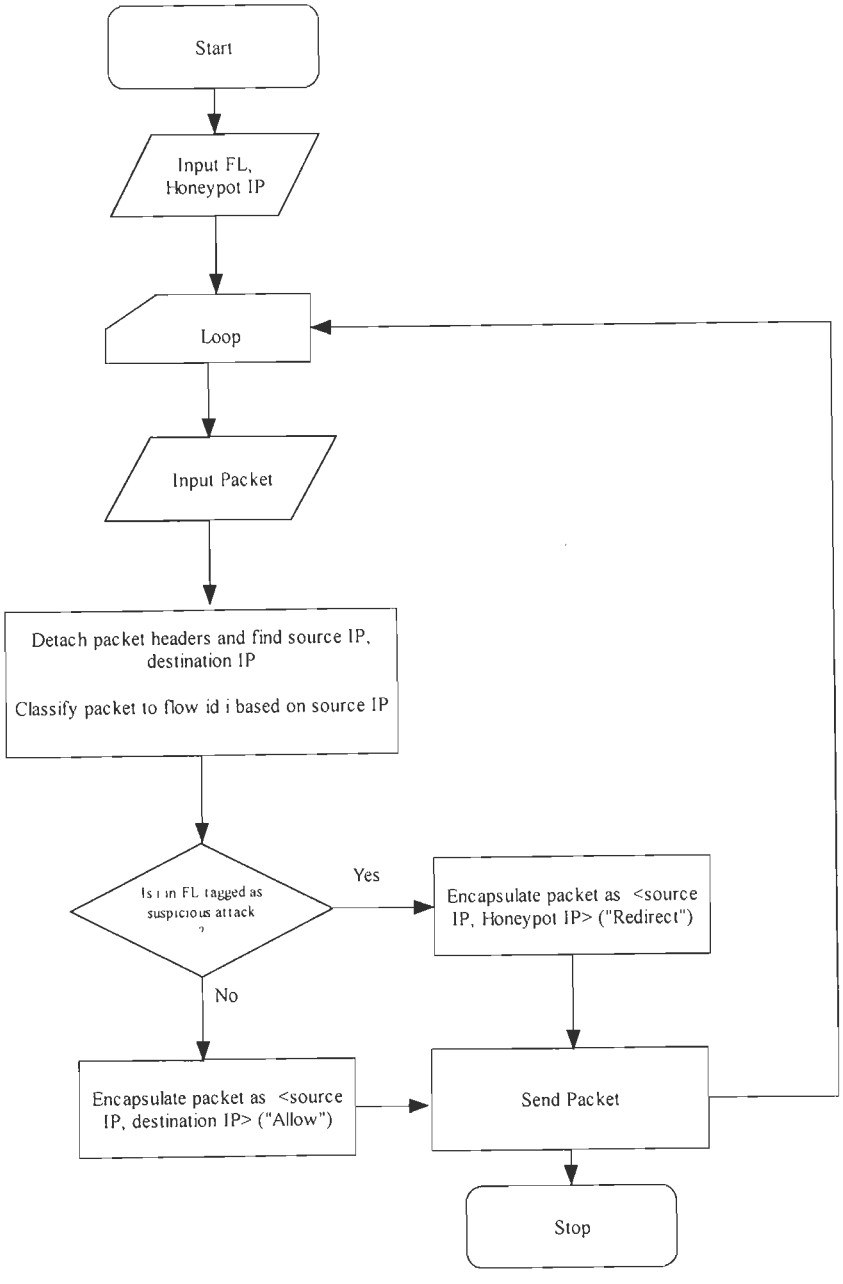


Figure 5.3. Flowchart for microscopic-level response

Filtering effort is immediate and reduces the impact of attack downstream almost instantly. Concise recommended rules are generated for responders to impose using the individual classification decisions. As the response rules generated are concise and imposed according to individual classification decision, it enables focused filtering and reduces impact of responses on legitimate traffic.

A flowchart in Figure 5.3 details out the response to flows tagged suspicious attacks and normal at microscopic-level. It takes input *FL* and honeypot IP. *FL* is obtained as a result of characterization at microscopic-level and contains source IP based flows tagged as normal or suspicious attack (refer to Figure 5.2). Packets corresponding to suspicious attack flows are redirected to honeypots whereas packets corresponding to normal flows are allowed to servers.

5.3 Overall Detection, Characterization and Response Algorithms

Overall algorithm for macroscopic-level detection, characterization and response at the edge router of transit network is as follows:

Algorithm

---

i. For a time window, initialize the parameters and size of time window.

ii. Count the number of packets from different sources in a time window.

iii. Calculate the system entropy H(X) based on source IP aggregates (refer to Equation 4.2). Determine the optimum threshold; calibrate the system; detect the attack (refer to Equation 4.3).

iv. If Attack = true, trigger characterization and start monitoring the individual contribution of entropy of each source IP based flows towards total system entropy.

v. Determine m flows that have highest or least packet arrival rates and contribute least to source IP aggregate based system entropy.

vi. Drop the identified attack flows (refer to Equation 5.4).

Overall algorithm for microscopic-level detection, characterization and response at the edge router of stub network is as follows:

Algorithm

i. For a time window, initialize the parameters and size of time window.

ii. Count the number of packets to different destinations.

iii. Calculate the system entropy H(X) based on destination IP aggregates (refer to Equation 4.2). Apply CUSUM and judgment function (refer to Equation 4.6 and Equation 4.7).

iv. If Attack = true, identify the victim (refer to Equation 5.6) and start monitoring the source IP flow based entropy for flows destined to victim.

v. Set of source IP based flows that share same or very similar entropy and minimal variations in their entropy such that their entropy rate is zero or less than threshold value are suspects and tagged as suspicious attacks. Moreover flows destined to honeypots are also tagged as suspicious attacks (refer to Equation 5.7 and Equation 5.8).

vi.. The flows tagged as suspicious attacks are directed to honeypots whereas the flows tagged as normal or legitimate are allowed access to servers.

Overall scheme is as follows:

i. Raising DDoS attack alarm at edge routers in transit stub network: This corresponds to macroscopic-level detection. We collect source IP based flow samples on edge router of transit network for a time window. If the entropy of system changes dramatically or flows are destined to honeypots or both, a DDoS attack is confirmed and alarm is raised at the router.

ii. Handling the DDoS suspects at microscopic-level: We collect destination IP based flow samples on edge router of stub network for a time window. We monitor system entropy as well as contribution of entropy of each destination towards the total system entropy. If the entropy of system decreases dramatically, the victim is identified and alarm is generated.

iii. Discriminating DDoS attacks from legitimate traffic: In case of macroscopic-level characterization, packet arrivals and entropy rate for each source IP based flows are calculated. Set of flows having least or highest measured packet arrivals which share same or very similar entropy and minimal variations in their entropy are attack flows. In case of microscopic-level detection, source IP based flows destined to victim that share same or very similar entropy and minimal variations in their source entropy are suspects and tagged as suspicious attacks. Flows destined to honeypots are also tagged as suspicious attacks.

iv. Eliminating DDoS attack packets before they reach the target: Packets corresponding to the attack flows identified at macroscopic-level on the edge router of transit network are discarded. In case of microscopic-level, the packets corresponding to flows either tagged as attacks or destined to honeypots are isolated from legitimate traffic and redirected to honeypots. Hence the detection, identification, elimination and isolation of attacks are done before the attack packets reach the target.

5.4 Performance Evaluation

5.4.1 Experiment Design and Procedure

We tested our scheme against an Internet type topology using network simulator 2 (ns-2) [202] as simulation testbed with same parameters as used for detection phase using D-LAD scheme. For details on simulation model, refer to Appendix A. The 156 node network shown in Figure A.1. is composed of five FTP servers, labeled node 117, 118, 119, 120 and 121 to be protected, 137 clients of which 48 are legitimate clients and 89 are attackers and router nodes (which represent point of presence). All FTP requests are originated randomly from different client nodes. We introduce randomness to the locations of legitimate clients and attackers. However, to simplify the analysis, we carefully place servers where they all have same access bandwidth (i.e. 3 Mbps) and where they all have paths from clients to access.

To control the load of each run, we use Poisson process to model arrival process of the FTP clients. The inter-arrival time of FTP client (IAT) is computed by:

$$IAT = \exp(T_{ftp} / CL) \tag{5.9}$$

where T_{ftp} is the average total time for file transfer, CL is the total client load, and $exp(x)$ is the exponential distribution with mean x.

Table 5.1. Attack characterization parameters

S. No.	*Parameter*	*Value*
1.	Simulation time	30 *seconds*
2.	Attack Duration	8-20 *seconds*
3.	Window Size	.2 *seconds*
4.	Packet Size	1040 *bytes*
5.	Tolerance factor *a*	2-9

Configuration parameters, including the link rate and propagation delay are summarized in table A.3. Each simulation experiment has 10 runs (averaged in the graphs). Legitimate clients send requests from time 0-30 seconds and attack duration is from 8-20 seconds. Attack characterization parameters are as given in Table 5.1.

5.4.2 Results and Discussion

We conduct experiments for dual-level attack characterization method. Characterization process is triggered as soon as D-LAD detects an attack. Characterization requires monitoring the contribution of each flow's entropy towards total system entropy. We treat the attack flows as 'signal', which we expect to identify and the legitimate flows are treated as 'noise'. The test results show that our methods have accurate characterization capability.

5.4.2.1 Macroscopic-Level Characterization

Two kinds of attacks are simulated. The first attack is simulated with a single attacker attacking at a rate of 1 Mbps. Second attack is simulated with 80 attackers with a mean attack rate of .5 Mbps. Figure 5.4 shows time series entropy of each distinct source IP based flow monitored on edge router of transit network. The Figure shows that flow 5, 6 and 7 are easily distinguishable from other flows. These flows correspond to attack signals. Flows 1 to 4 are noise and correspond to legitimate flows.

In the first case, flow 5 with the least contribution to source IP based system entropy and most frequent packet arrivals was identified as attack flows, with value of m (refer to Equation 5.4) corresponding to 1.

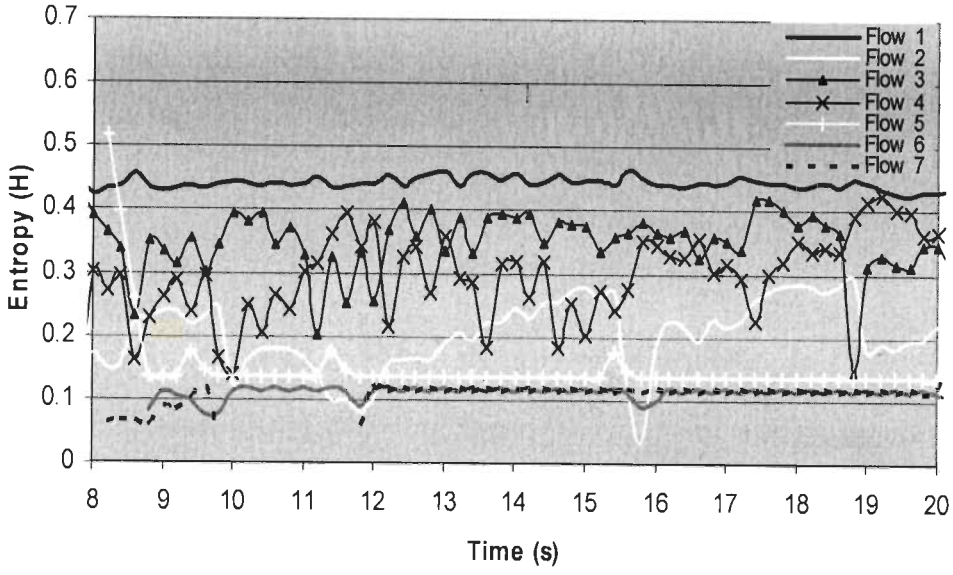


Figure 5.4. Time series entropy variation of each distinct source IP based flow

In the second case, the characterization process triggered as a result of detection, on simulating 80 attackers with 0.5 Mbps attack rate after 8 seconds, alarmed distributed low rate attack and identified flow 6 and 7 (up to flow 85 not shown in the Figure) as the attack flows. These attack flows are signals with less frequent packet arrivals and hence contribute least to the source IP based total system entropy as shown in Figure 5.4. However, such flows being large in number increase the total system entropy at edge router of transit network.

5.4.2.2 Microscopic-Level Characterization

As soon as the microscopic-level characterization is triggered by Mi-LAD; victim or server under attack is identified. We monitor the contribution of each destination IP based flow's entropy towards total system entropy at edge router of stub network where each distinct destination IP corresponds to a server to be protected. Since, the overall decrease in the destination IP based total system entropy triggers the microscopic-level characterization, there is one destination IP based flow that dominates the rest of the flows due to heavy rate of packet arrivals, and therefore flow with highest frequency of packet arrivals or that contributes least to overall destination IP based system entropy is identified as victim.

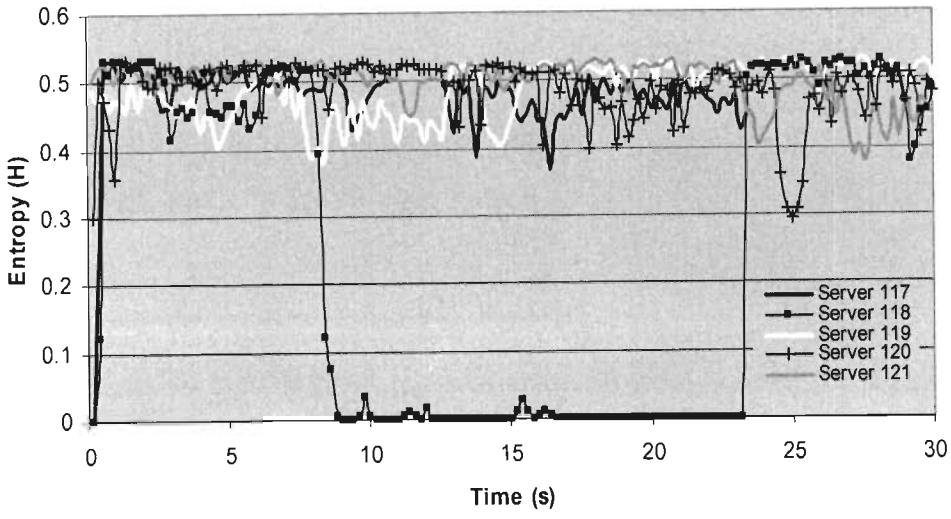


Figure 5.5. Time series entropy variations of distinct destination IP based flow corresponding to each server

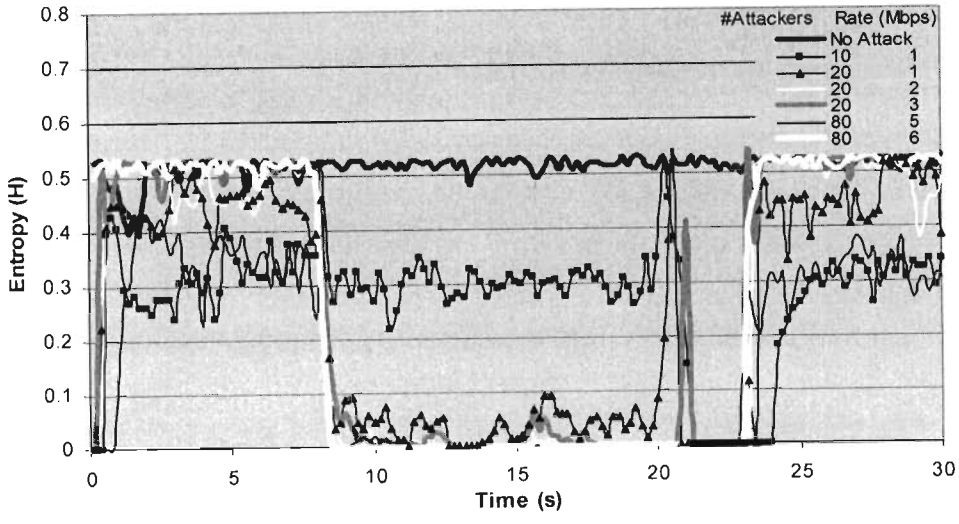


Figure 5.6. Time series entropy variations showing contribution of victim flow towards total destination IP based system entropy under different attack strengths

Identification of the victim

Figure 5.5 shows time series entropy variation of each destination IP based flow which corresponds to each server in the server pool of five servers to be protected. The attack is simulated with 20 attackers at the rate of 2 Mbps, and starts at 8 seconds, with attackers randomly selecting one of the five servers from the server pool to launch their attack. Figure 5.5 clearly shows that server 118 has the least measured entropy value during the attack period and contributes least to the total system entropy; and hence is the cause of decrease in total system entropy. This implies that server 118 has highest measured frequency of packet arrivals. The attack packets targeted to server 118 are captured in time series entropy variations identifying it has victim of the attack.

Figure 5.6 shows the relationship between different attack strengths and decrease in entropy of the flow identified as victim. The entropy of server 118 decreases in presence of different attack strengths. The higher the strength of attack more is the decrease in the entropy of the victim. It is so because higher the strength of attack more is the frequency of packet arrivals, and lesser the contribution of the flow's entropy in total system entropy. Destination can be identified easily in all cases of attack strengths.

Characterization of suspicious attacks

Characterization of attacks at microscopic-level requires monitoring entropy of each distinct source IP based flows that are destined to victim identified in the above step.

We simulated the attack with 10 attackers at rate of 1 Mbps and monitored time series entropy of each distinct Source IP based flow destined to server 118. Figure 5.7 gives the results of simulation. The set of source IP based flows from 1 to 10 share same entropy space and there are minimal variations in their entropy rate. These dominant signals are effortlessly identified tagged as suspicious attacks.

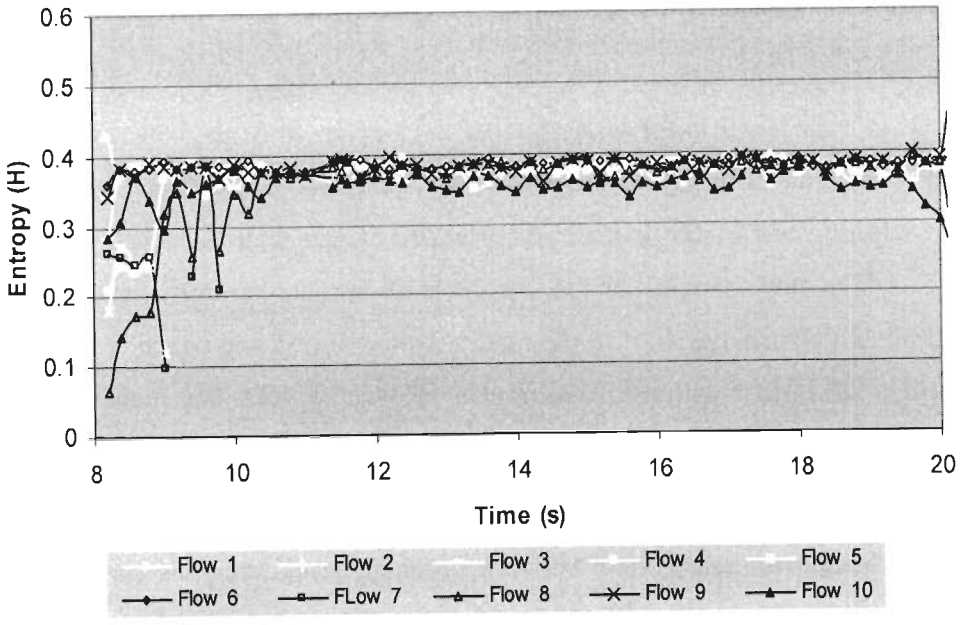


Figure 5.7. Time series entropy variations of each distinct source IP based flow destined to the victim (server 118) monitored on edge router of stub network ; 10 attackers ;1 Mbps

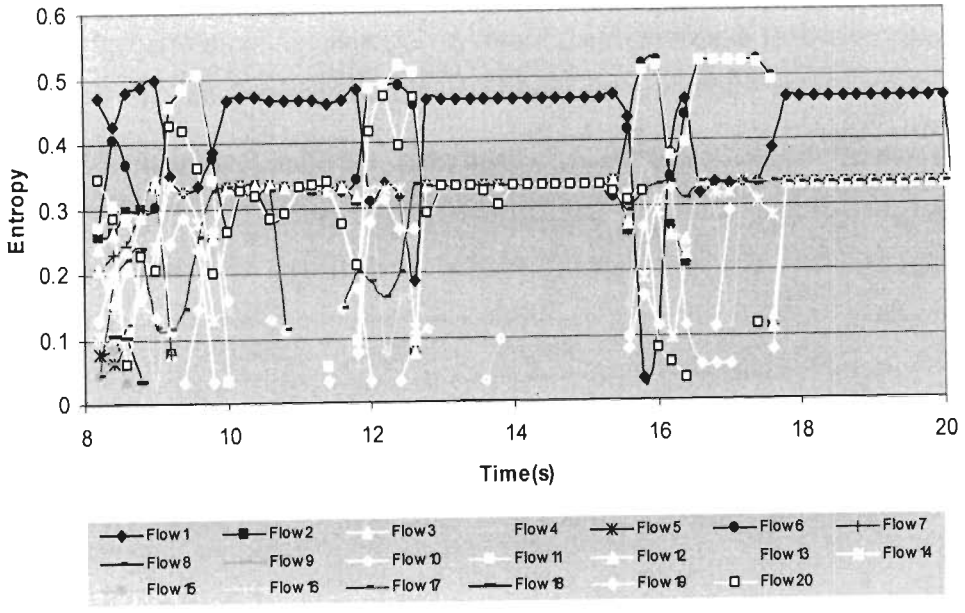


Figure 5.8 Time series entropy variations of each distinct source IP based flow destined to the victim (server 118) monitored on edge router of stub network; 20 attackers; 2 Mbps

Figure 5.8 shows the results when previous simulation was repeated with 20 attackers attacking at rate of 2 Mbps. The Figure shows two distinct sets of signals for the 20 flows with no variations in the entropy for each set of flows. The entropy rate for each flow in both sets becomes zero soon after the attack is launched at 8 seconds. This justifies the fact that attacks generated by sophisticated attackers using different attack functions can be easily identified.

Figure 5.9 shows the attacks simulated with 72 attackers. The Figure shows two distinct sets of signals for 72 flows with no variations in the entropy for each set of flows. The entropy rate for each flow in both sets becomes zero soon after the attack is launched at 8 seconds. This justifies the fact that attack signals are easily distinguished from background noise or the legitimate flows and attack flows are identified even if they are generated by sophisticated attackers using different attack functions at high attack strength.

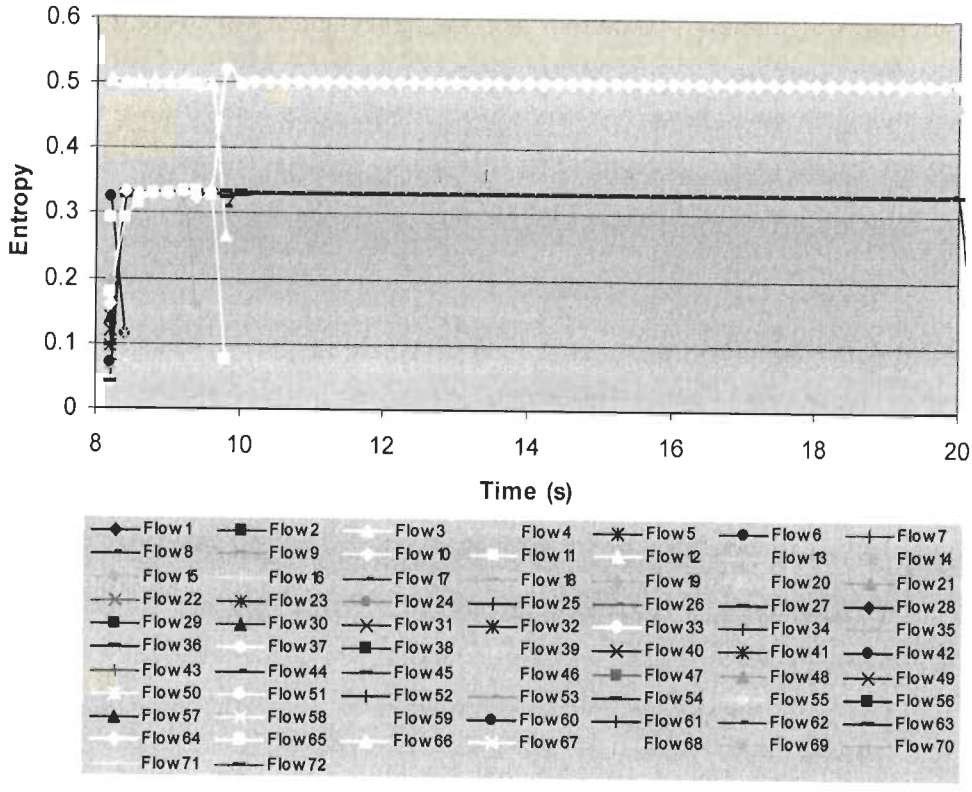


Figure 5.9. Time series entropy variations of each distinct source IP based flow destined to the victim (server 118) monitored on edge router of stub network; 72 attackers

5.4.2.3 Performance of the Overall Scheme

Figure 5.10 shows the ratio of legitimate and attack packets accepted under different strengths of attack. The strength of attack was increased by increasing the number of attackers from 1 to 80. The Figure shows that more than 90% of the good packets were identified and accepted irrespective of the mode of defense (namely, naïve, normal and best defense, refer to Section 4.3.2.3). This clearly justifies our claim that collateral damage caused in the presence of the proposed responsive measure is much less than the damage suffered by legitimate clients in the absence of response, which reduced the throughput to zero as soon as attack was launched (refer to Section 4.4.2.1, Chapter 4). There is no perceivable decrease in the acceptance rate of legitimate packets even if the magnitude of attack increases. Legitimate packets remain unaffected and are not dropped because, before congestion inducing attack packets flood the network resources and cause legitimate packets to drop, they are filtered at macroscopic-level i.e. much early before reaching the victim.

However, naïve defense has the highest packet acceptance ratio because the thresholds are such that they favor legitimate flows so as to minimize FP. Best defense, on other hand has slightly decreased packet acceptance ratio because system is calibrated so as to minimize FN and hence increases FP causing good packets to suffer during characterization and response.

Though, the packet acceptance ratio for good packets remains same with increasing number of attackers, the bad packet acceptance ratio increases with increase in number of attackers and attack strength. The increase is highest for naïve defense, because the system is calibrated to reduce false alarms and hence reduces the detection rate. Hence some attack flows remain undetected (refer to Figure 4.8) and go uncharacterized at macroscopic-level. In case of best defense (refer to Figure 4.6), the system is calibrated for high detection rate and minimum FN. Therefore bad packet acceptance ratio decreases almost to zero in case of best defense.

Hence, the simulation results indicate that response prototype blocks substantial congestion inducing attack traffic, allowing some of the suspicious traffic into the network, that can be further analyzed before taking any decision, and hence reduces collateral damage.

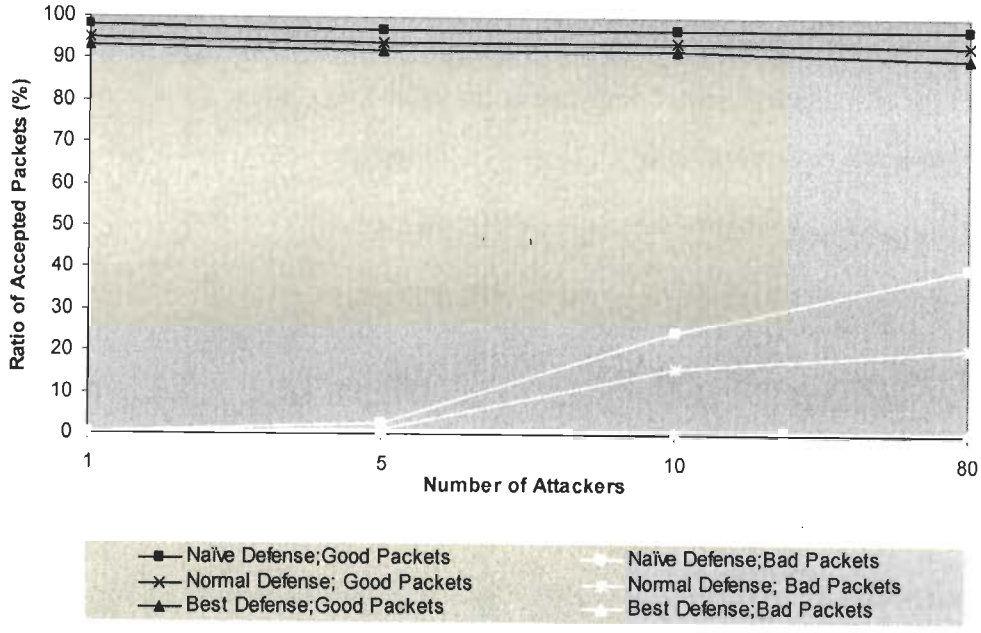


Figure 5.10. Ratio of accepted packets vs. different number of attackers for three modes of defense

5.5 Conclusions

Our proposed D-LAD has been extended to provide attack characterization of network traffic. In our proposed dual-level attack characterization technique, macroscopic-level identifies congestion inducing attack flows whereas suspicious attack flows are identified at microscopic-level. Presence of honeypots identifies the attack flows exploiting the vulnerabilities that otherwise remain undetected. Our proposed technique is applicable to a variety of attacks. We are able to characterize attack from legitimate traffic even if it is from same site as legitimate traffic, destined for nodes on same network or victim, or shares characteristics like application protocol etc.

Our proposed response technique enables more focused filtering and redirection, and reduces the impact on legitimate traffic. As a response to attack flows identified at macroscopic-level, filtering is performed. It blocks substantial proportion of generated DDoS attack traffic at macroscopic-level. With the two-fold protection, packets with higher probability of being valid are offered preferential service, while packets which have been marginally classified as invalid or suspicious attack at microscopic-level will still be allowed and directed to honeypots. It drills down to investigate suspicious DDoS flows more closely at honeypots at microscopic-level. Therefore, more certain and refined response is triggered for suspicious attack flows at microscopic-level. Results show that

collateral damage caused in the presence of our proposed responsive techniques is much less than the damage suffered by legitimate clients in the absence of response. Therefore, the techniques for characterization and response proposed in this chapter show a promise for complementing detection and characterization with minimum collateral damage.

Chapter 6

Dynamic Honeypot Based Attack Mitigation

6.1 Introduction

A complete DDoS defense system requires not only detecting presence of attack and identifying attack traffic but also mitigating the attack. Mitigation is the process of minimizing the effects of an ongoing attack. Mitigation approaches to DDoS attacks have reached to a level of development of survivable systems that prevent the failure of network operations. However, aim of successful mitigation technique should be to provide stable network functionality as well as service continuity to legitimate clients, with guaranteed QoS under dynamically changing network conditions. Mitigation of attacks to a level of maintaining service continuity with guaranteed QoS is very challenging problem that has not yet been solved.

Depending upon the location of target, DDoS attacks can be network-level or service-level. Various link-based defenses have been proposed to mitigate the effect of network-level attacks that saturate critical links in path of legitimate clients. Filtering [51, 52, 75, 87, 88, 116, 126, 128] is the simplest, easiest and most effective way to mitigate the effects of a DDoS attack, if the attack characterization is accurate. Most of the IP traceback schemes [39, 142] based on packet marking [39, 76, 132-141, 143, 146, 172] suffer from IP packet marking overheads. Link testing schemes for IP traceback [111, 113, 144] cannot detect the degrading attacks which do not cause congestion at links. Other link based defenses [78-80, 153-155] cause collateral damage which in itself turns out to be DoS. Special routing schemes [111, 116, 117, 156, 173] suffer from potentially enormous resource requirements, require changes to global routing tables and have high overheads of storage, processing and secure communication. Scheduling algorithms [158-160] are too expensive in terms of delays. Other techniques [7, 77, 114, 129, 161-167, 169] are either inefficient or require global cooperation.

Node-based defense for service-level attacks include resource pricing [98-100, 121], changing IP address [55], QoS regulation [122], over provisioning [57] etc. These schemes either require excessive state monitoring, specialized client programs, have delays or are costly.

Therefore, each of the above mitigation technique has its own set of limitations. Most of the mitigation techniques are inadequate, inefficient or overly restrictive. The above solutions neither eliminate the mitigation problem, nor deter the potential attackers. Therefore, with reactive approaches in place, network under DDoS attack often becomes functionally unstable and legitimate users in the network suffer in terms of poor response times and frequent network failures. Compounding the problem is to maintain an acceptable level of QoS. Also, the Internet is dynamic in nature and the topic of automated responses to DDoS attacks has not received much attention. To develop an effective, scalable, and resilient mitigation scheme is a big challenge. Refer to Section 2.3.5 for further details.

Our proposed mitigation scheme has the following objectives:

i. Proactively mitigate effects of DDoS attacks.
ii. Respond autonomously to ensure legitimate clients are served efficiently and not affected by the attacks.
iii. Ensure that system is not mistakenly filtering legitimate traffic.
iv. Ensure stable network functionality, resource availability and service continuity with guaranteed QoS to legitimate clients.
v. Ensure that too many resources are not sacrificed during mitigation.
vi. Optimize use of resources in dynamic network environment to ensure that benefit of the scheme is manifold as compared to the cost incurred.
vii. Deter the potential attackers during mitigation process.

In the previous chapters, we proposed dual-level attack detection and characterization and identified the congestion inducing attack flows. The identified attack flows were filtered at macroscopic-level. For the rest of the traffic analyzed in stub domain at microscopic-level, suspicious flows were identified from normal ones.

In this chapter, we propose a service-level proactive mitigation approach for the suspicious attack flows identified at microscopic-level. Our scheme uses autonomous dynamic honeypot [205] and forms the third line of defense of the proposed framework.

Our approach to service-level attack mitigation is as follows:

i. Isolating the suspicious attack flows and redirecting it to honeypots present inside the victim network for treatment.

ii. Forwarding the good traffic of legitimate flows to active servers in the network to maintain service continuity.

iii. Replicating the servers and changing their location, number and duration of service.

The total budget (total number of machines) gets partitioned into two groups: active servers and honeypots. The traffic is handled through honeypots or active servers contingent on the input derived from characterization at microscopic-level. The good traffic of normal legitimate flows is routed to one of the active servers, while the attack is diffused across honeypots. By "dynamic" we mean that the number of servers and honeypots is adaptive and changing. By "autonomous" we mean that change in number and locations of servers and honeypots is triggered independently with changing network conditions.

We present exhaustive cost-benefit analysis of our proposed mitigation scheme. We analyze our proposed honeypot framework for different scenarios of attacked network under three modes of defense among naïve, normal and best.

6.2 Building Blocks of the Proposed Scheme

This section gives an overview of our mitigation scheme. We identify the major issues to be addressed and focus on the building blocks of our scheme.

At macroscopic level, i.e. on border of transit network, congestion inducing attack flows were identified, that, if permitted into the network, may cause severe network performance degradation. This serves to mitigate link-based attacks at network-level.

At microscopic-level, i.e. on border of stub network, we consider mitigation as the problem of estimating total attack traffic targeted towards the victim and isolating it. Characterization scheme at microscopic-level tags all the arriving flows as either normal legitimate flows or suspicious attack flows in moving time window. This information is maintained in the *FL*. Refer to Section 5.2.2.2 for details. Our proposed mitigation scheme makes use of the information derived from this characterization. Legitimate flows are directed to one of the active servers. Services are replicated on all the servers. Traffic belonging to the suspicious attack flows is processed by a honeypot.

Whenever a suspicious attack flow arrives, instead of just dropping the flow enroute or sending it to active server or resetting sessions, they are redirected to a honeypot. Honeypot server responds to these flows in contained manner as would the actual server to legitimate clients. It logs information about the flows either for a fixed time interval equal to the characterization time window (or a multiple thereof); or till the expiry of current eon (variable time duration, discussed later); or till the existence of flow, whichever less. Therefore connections with the suspicious attack flows are retained by honeypots.

If the suspicious attack flow directed to honeypot is tagged as normal is subsequent characterization time window, it regains its status as legitimate client, and is redirected to active server. Therefore, legitimate traffic mistakenly handled by honeypot is processed successfully, albeit at higher latency. However if the flow persistently remains tagged as suspicious attack, connection remains directed at honeypot server, before it is dropped. Honeypot logs the information and therefore one can potentially gain more information about the attacker.

Active servers and honeypots both are in the same domain. Active servers and honeypots continuously change their number and location among a pool of servers. Number and location of active servers and honeypots depend on current *CL* and attack *AL*. The duration between successive change in number and location of active servers and honeypots is called "eon".

Each server in the server pool alternates between providing the service and acting as a honeypot. This is done in a manner unpredictable to attackers. Each time the location of active server changes, service roams and the roaming time and the address of the new server are kept secret except from legitimate clients tagged as normal. Legitimate clients, depending upon their trust level, can track the active-servers for certain time duration. Therefore, each client has to prove itself as being legitimate by getting corresponding flow tagged as normal before perceiving knowledge of the secret location of the active server and being granted access. Once a client has this knowledge, it will be able to track down the active server. As the active server changes its location, all legitimate connections are migrated to the new server. Duration for which each legitimate client is able to track the location of active servers, depends upon the trust it has built with the system.

Honeypots are instrumented to detect potential attacks. Hence any flow destined to honeypots loses its status as a legitimate client, is considered suspicious attack and handled by honeypot.

The active servers change their location either reactively in response to the detection and identification of an attack, or proactively to defend against unpredictable and undetectable attacks. The reactive component allows the scheme to benefit from characterization step. It generates judicious mixture of active server and honeypots in reaction to current *CL* and *AL*. The proactiveness of the scheme makes it difficult for the attackers to guess where and when the active server roams. The proactive component also allows the scheme to tolerate undetected attacks that may attempt to bypass the system. Since, legitimate flows can track location of active servers; any traffic destined to honeypot is an attack. The server state is proactively flushed each time the server roams, causing all illegitimate connections to be dropped. Since each time server drops all its current (attack) connections as it switches from honeypot to active server, it opens a window of opportunity for legitimate requests before the attack re-builds up. FN are thus, weeded off when active servers and honeypots change their location.

6.2.1 Server Replication

In a node based defense for server-oriented attacks, if the server can accurately distinguish between normal and attack traffic, DDoS attacks can be stopped by simply dropping the illegitimate attack packets and restricting the access to server to normal packets coming from known legitimate source addresses. Accurate characterization requires time and computation. Efficient filtering implementations at line speed are not easy to design. Delay in characterization and response results in performance issues. Overprovisioning of resources like deploying load balancers and having hardware like servers and bandwidth available, to increase the capacity in an emergency, increases resistance against DDoS attacks.

Server replication makes available servers in excess so that they can accommodate both the attack and the legitimate traffic, thus avoiding denial of service. It strengthens the victim to withstand the attack. The server is randomly selected for each arriving request and share the load equally at all times. This is by far the most widely used approach for service-level DDoS defense.

In order to withstand service level DDoS attacks, we propose to replicate the servers in our scheme. However, it is insufficient to prepare for DDoS attacks by server replication alone. Though, replication can diffuse the load of link-oriented DDoS attacks over a number of simultaneous replicas, replication is not DDoS attack-tolerant in the same degree as it is fault-tolerant. It raises the bar for the attacker who must generate a

sufficiently strong attack to overwhelm abundant resources, but replication cannot tolerate large-volume attacks that can clog all the replicas. For server-oriented attacks, even with a smaller number of packets, attacks can exploit vulnerabilities in all the replicas and knock them down almost simultaneously. Another disadvantage is that cost of server replication is prohibitive for small systems. If a system does not usually experience high traffic volume, it needs modest servers for service continuity. In such case, replicating a lot more servers will be wasteful, as they rarely get used, whereas replicating just a bit more will not be able to withstand DDoS attacks. Finally, automated tools can be used to launch an attack even against a well-provisioned network.

Hence, server replication alone is an expensive solution and does not guarantee to withstand the DDoS attacks. It fails in a scenario where servers promise QoS guarantees to clients. Replication can make a system tolerate attacks; it does not effort to mitigate the attacks on the servers.

6.2.2 Dynamic Roaming Honeypots

Replication by itself is insufficient, as while being able to tolerate DDoS attacks, it fails to mitigate them. All the replicas are active and accepting attack connections. Our goal is to mitigate the effects of DDoS attacks and not to scarify too many resources. Replication becomes costly during time periods of low network load, when only few resources are enough to service legitimate clients. It does not guarantee service continuity during high attack loads because it does not effort to mitigate attacks. To overcome these disadvantages of replication, we propose to dynamically change the degree of replication. In addition, we also propose to isolate the suspicious attack flows.

Our scheme [193] uses honeypots [174-176] to isolate the suspicious attack flows. During mitigation, identified suspicious attack flows are directed to isolated zone of honeypots where they cannot cause any further damage to the network [193]. Honeypot system sets up a controlled environment similar to the service system, deceiving attackers. Honeypots generate automated response against attacks. This approach leads the attacker to believe that they are succeeding in their attack, whereas in reality they are simply wasting time and resources. Hence, honeypot based attack isolation [193] along with replication helps it to withstand high *AL*. Advantage of using honeypots is that they trap the attackers and gain information about attacks. They can be taken offline for analysis. Any data retrieved from honeypot is most likely related to the attacker. Thus, they give in depth information that is needed to rapidly and effectively respond to attack.

However, static honeypots [193, 206], because of their deployment at fixed (thus detectable) locations, can be avoided by sophisticated attacks. Moreover, if an attacker comes to know the presence of honeypot in network, honeypot may be compromised. A compromised static honeypot can be used to attack other servers in the network. In our scheme, instead of being statically placed, we propose to change the locations of both active servers and honeypots.

Many schemes have been proposed in literature where server roams or changes its IP address to mitigate effects of attack. IP hopping [207] protects a public server; as the server changes its IP address after detecting it is under attack without changing its location. Only changing the IP address of the server without physically moving the service retains malicious state entries in the server implanted during the attack. Illegitimate or attack packets will still go to the same network after the address is changed. IP hopping does not block a persistent attacker which looks up the new IP address of server using DNS. By physically moving the service from one server to another and cleaning the state of the old server avoids the limitation of IP hopping. Figure 6.1 illustrates logical roaming used by IP hopping [207] and the proposed physical roaming. Consider two machines, machine 1 has IP address A.B.C.D, machine 2 has IP address W.X.Y.Z and destination IP is A.B.C.D i.e. machine 1 at time T1. In case of logical roaming, at time T2, the machines change their IP addresses as the destination IP changes to W.X.Y.Z with the packets still destined to machine 1. In case of physical roaming, at time T2, the IP address of machines are not changed as the destination IP changes to W.X.Y.Z, so that packets are now destined to machine 2.

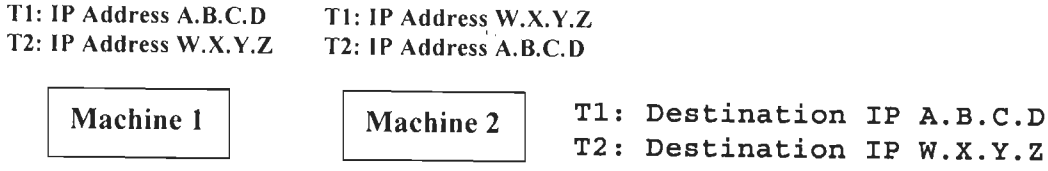


Figure 6.1 (a) Logical roaming

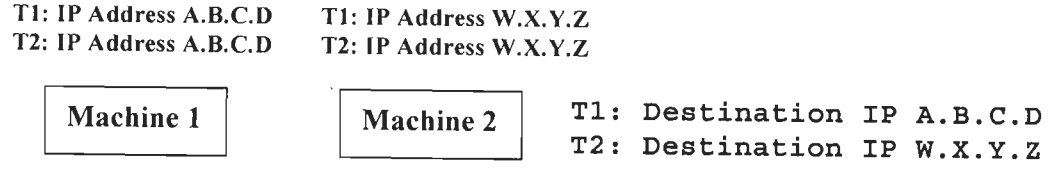


Figure 6.1 (b) Physical roaming

Mutable Services [130] is a framework to allow for reactively relocating service front-ends and informing only pre-registered clients of the new location through a secure DNS-like service. In [186], hybrid architecture has been suggested for defense against DoS attacks, where a passive honeypot has been used for protection against relatively static attacks. In [124], a secure and light-weight mechanism to proactively change the location of the active server within a server pool was proposed to mitigate the attacks. However, the scheme incurred an overhead that caused performance degradation both in the absence of attacks and under low attack loads. In [208], because of sacrificing some servers to act as honeypots, distributing the load on all the servers outperformed the scheme in the case of a high legitimate client load combined with a low attack load. Both these schemes [124, 208] also suffered from the disadvantage of the cost of replication in absence of attacks.

To minimize the cost of replication and maximize the performance, limited network resources must be judiciously used during an attack. In our scheme, we propose to dynamically change the degree of replication depending on network conditions. Degree of replication of active servers is increased if CL is high, whereas it is reduced if there is low CL and high AL, thus increasing number of honeypots and keeping ahead of even rapidly increasing volumes of DDoS requests. We also propose to change the physical location of active servers and honeypots.

To change the degree of replication of active servers and to change the physical location of active servers and honeypots, following issues must be addressed: firstly, what should a honeypot look like, secondly, where to deploy servers and honeypots and thirdly, how many. As our approach requires servers and honeypots to continuously change their number and physical location, it deals with in-process connections. How to migrate the connections to the new server as the active server migrates from one location to another is another issue. Moreover, physical roaming has overhead due to connection reestablishments that result in performance degradation. Connection reestablishment incurs an overhead as every time old connections are broken and new connections established, there is TCP slow start that pulls down the response time. Therefore, roaming should not be triggered too late that attack builds up, and it should not be too early that time spent in TCP connection reestablishment causes legitimate clients to suffer and cause performance degradation. When should roaming be triggered is an important question that is addressed. Finally, each legitimate client with normal traffic should be able to perceive

the secret location of the active server and its duration. A client with higher trust should be able to track down the active server for a longer duration.

File Transfer Protocol (FTP) services are based on TCP and have been replicated on all nodes to make honeypot act like FTP server, responding in contained manner to attack flows. Server and honeypot roaming mechanism adds a layer of DDoS attack-tolerance above plain replication. Dynamic roaming honeypots and changing degree of server replication do away with cost of replication and provide guaranteed QoS besides attack-tolerance. The effectiveness of our framework relies on when and how do the numbers and locations of active servers and honeypots change, how the legitimate clients know where the active servers are and how we migrate the in-process connections. Next, we describe the building blocks of our scheme which address the issues discussed above.

6.2.2.1 Dynamic Honeypots

The total server budget gets partitioned into two groups, active servers and honeypots. Active servers change their location within the pool of servers, and the rest of the nodes in server pool act as honeypots. Number of active servers and honeypots depends on the current *CL* and threat level. *AL* represents the threat level in the network. *CL* and *AL* are determined from the characterization scheme described in Chapter 5 and depend upon the number of normal and suspicious flows identified during microscopic level characterization in the *FL* (refer to Section 5.2.2.2). We grade *CL* and *AL* into three categories, namely, low, moderate and high. Similarly, both the number of honeypots and the number of active servers are divided into three categories namely low, moderate and high. For each combination of *CL* and *AL*, we use low, moderate and high to determine a range on the number of honeypots and active servers. A combination from the range is chosen probabilistically in a way that total budget remains same. Our scheme is such that in case of high *CL* and low *AL*, adequate number of active servers is generated to serve client requests. Where as in case of high *AL* and low *CL*, modest number of active servers is generated with enough number of honeypots to fend off the attack. The design details of above proposed technique are discussed later.

6.2.2.2 Proactive Roaming

To allow for undetectable, dynamic and proactive roaming behavior, the algorithm is based on backward hash chain for calculation of roaming time and location of destinations. We use backward hash chains, the members of the chain are generated using one-way hash functions like SHA-1 and MD5 [209-211]. For this class of functions, it is

computationally infeasible to reverse the direction of the function application; that is, given the output of such a function, it is computationally infeasible for an adversary to know the input. One way hash functions have been used in reverse direction of their generation. The backward usage of hash functions is inspired by the work done in [124, 212-214]. Hash functions have a lightweight computational overhead and hence simple and efficient implementation.

To be able to know the active server location, a client needs to have at least two information: the server address and the time that the server will be active. Service time is divided into *eons*; at the end of each eon, number of active server changes; and the active servers and honeypots migrate in the server pool. To calculate the duration of service eon and location of active servers in the service eon, a long hash chain is generated using a one-way hash function $H(\cdot)$, and used in a backward fashion. The last key in the chain, K_n, is randomly generated and each key, K_i $(0 < i < n)$, in the chain is computed as $H(K_{i+1})$ and used to calculate both the length, R_i, of service eon E_i and the location, S_i, of the active server during E_i as follows:

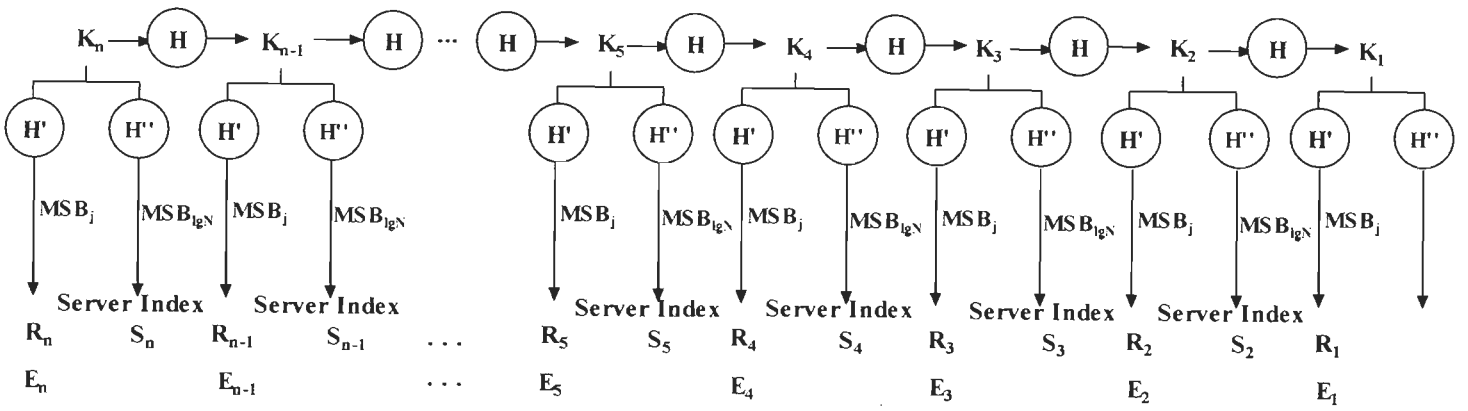
$$R_i = \text{MSB}_j(H'(K_i)) \text{ and} \tag{6.1}$$

$$S_i = servers[\text{MSB}_{\lg N}(H''(K_i))], \tag{6.2}$$

where $\text{MSB}_j(x)$ are the j most significant bits of x, 2^j represents an upper bound on eon length, N is the total number of servers, and the array *servers* contains an <IP address, TCP port> pair for each server in the server pool. H' and H'' *are* public one-way hash functions, such as MD5 [211]. Hence, the client is able to track the active server within a pool of servers and determine the time for which it would be active.

Based on trust levels, clients can be classified into distinct classes each with different privileges for accessing the servers [215]. Here privilege is defined by the duration (i.e. number of eons) for which a client can access the server.

Each legitimate client is assigned a roaming key, K_t, from the hash chain, with a varying value of t according to each client's level of trust which it builds during characterization step. K_t allows the client to track the servers to and including eon E_t. For instance, in Figure 6.2, suppose if a client is assigned K_3, it will be able to generate K_2 and K_1 and therefore track the service until eon E_3 whereas a client assigned K_5 can follow service until E_5 starting from E_1.



H, H', H'' : One way hash functions
K_n : Last key in the chain
N : Total number of servers in the pool
E_i : i^{th} Service eon
R_i : Length of the service eon E_i
S_i : Location of server in service eon E_i
$MSB_x(y)$:x most significant bit of y

Figure 6.2. Calculation of server location and eon duration using backward hash chains

Our proposed scheme defines two autonomous updates to serve two purposes:

(i) To allow for changing the eon length's upper bound, 2^j, to reflect the current threat level. We defined an upper bound on the time interval between consecutive server roaming instances. This upper bound is adaptively changed to react to the current threat level. For instance, in a high threat period this upper bound is set to be small, while it is set to a large value in normal conditions. Effect of this upper bound and hence length of eon R_i (referred to as migration interval or miv) is investigated and its value is decided according to CL and AL.

(ii) Since there are multiple active servers, and therefore the scheme is extended to determine location of a subset of active servers instead of a single active server depending upon number of active servers.

More details on design of the scheme are discussed later.

6.2.3 Service Migration

This section discusses the mechanism used for migrating the active connections. We assume our service to be a TCP-based service.

Two well-known TCP-connection migration mechanisms are the TCP-Migrate [216], developed at MIT, and the Migratory-TCP [217], developed at Rutgers University. Both provide a framework for moving one end point of a live TCP connection from one location and establishing it at another location having a different IP address and/or a different port number. These mechanisms are used for mobility support and fault or attack tolerance. They have been used for roaming servers and require some modifications to suit our requirements.

Both mechanisms deal with following issues in a slightly different way: (i) how the TCP connection is continued between the new end points; (ii) how to recover both TCP and application states; and (iii) when to trigger the migration mechanism.

In MIT's TCP-Migrate [216], during connection establishment, the migration feature is requested through a TCP option (Migrate-Enabled). By the means of a handshaking protocol, a shared key is established between the two connection end points. As per a migration request from one end point, represented by another TCP option (Migrate), the TCP control block at the fixed end point is updated to reflect the new location of its peer. To protect against connection hijacking, the secret key agreed upon during the connection establishment should accompany the migration request. State recovery in the new server is

achieved via periodic state updates from the old server to the server pool. The migration request is issued by a new server and triggered by an overload at the old server, detected by a health monitor. Implementations at both the transport and session layers are available. The TCP-layer in both the server and the client needs to be changed. However, no application layer updates are necessary.

During connection establishment between two Migratory-TCP-enabled peers [217], a list of available servers, along with a certificate for each server, is passed from the server to the client. A migration request, also implemented as a TCP option, consists of both the certificate of the new server and the connection id (client IP address, client port, old server IP address, old server port) of the migrating connection. However, no security measures are implemented to protect the migration process. State recovery at the new server is achieved either on-demand, that is, when the client sends the migration request to the new server, or through periodic state updates. Triggered by an internal QoS monitor in its kernel, a client can issue a migration request to any server in the server list which the client receives in the connection establishment phase. Both the server and the client TCP layers should be changed and the server application layer should also be modified to allow for application-layer state snapshots and state recovery at the new server. One limitation of the on-demand state update is in the case of old server's crash or failure due to an attack.

Current TCP connection migration mechanisms require some modifications to defend against DDoS attacks. In TCP connection migration schemes, both the application and TCP states should be recovered at the new server. A TCP connection migration scheme suitable for roaming mechanism should assume very little or no dependence at all on the old server in the state recovery process, as it can be already blocked due to the DoS attack. It also should be lightweight (i.e., with as small an overhead as possible). Periodic state updates instead of lazy, on-demand updates is used because there is no guarantee on the operating state of the old server to be alive and available for state requests.

Therefore instead of using periodic server-to-server updates, clients themselves participate in state recovery [124] due to two main reasons. First, server-to-server updates are expensive (e.g., all the servers will have to be updated in order not to reveal the identity of honeypots). Second, the state of the old server might already be faulty, malicious, and/or overloaded, and migrating it completely would render the new server vulnerable as well. Pushing the state to the clients filters out malicious state entries at the new server because *only* subscribed legitimate clients know the address of the next

roaming server and the roaming time and only these clients will be able to create state entries for their connections at the new server.

It is proposed to use the clients for storing periodic state checkpoints of both TCP and per-connection application state of the server. The same period is used for all connections, though starting from the time each connection was established. State update messages are sent using the client's secure channel and to the same port as normal traffic to avoid the possibility of an attacker faking erroneous state updates and/or trying to block state update packets.

6.3 System Model

This section discusses the system components and various models used by the system components.

6.3.1 System Components

Our system consists of the following components:

6.3.1.1 Server Pool and Firewalls

A pool of N homogeneous servers is physically deployed. Each server is connected to the outside network through a firewall. Geographically dispersing the servers and/or deploying them over a resilient overlay network [218] provide for better path independence among the servers. We assume a firewall at each server's entry point that can perform adaptive filtering. Refer to Appendix A, Figure A.1 for corresponding simulation model. Nodes numbered 117-121 represent servers.

6.3.1.2 Clients

Two classes of clients are assumed: (i) legitimate clients, which can establish connections with active servers. For each legitimate client we assume a QoS contract with the service. According to this contract, the current server allocates a certain amount of resources to legitimate client requests and responds to legitimate requests in a bounded time period; (ii) illegitimate clients or attackers trying to degrade the service responsiveness by DDoS attacks, which are stealth and sophisticated attacks or undetectable attacks that impersonate legitimate client's traffic characteristics. In the corresponding simulation model (refer to Figure A.1) 137 clients are randomly distributed among which 48 are legitimate and 89 are attackers.

6.3.1.3 The Service

Our service is a generic TCP-based service. The service can be replicated. k out of N servers are active and providing the service at any point of time. Depending on the granularity of clock synchronization, the active interval of one server can be interleaved with the active interval of another one while the service is being roamed.

6.3.2 Attack Model

In addition to undetected network level attacks, attackers we are defending against are interested in launching a DDoS attack against our services. Service-level attacks are attractive to attackers because they are usually low-bandwidth attacks that go undetected at network level. They can result in consuming both server and network resources. Service-level attack may be launched using a spoofed or non-spoofed source addresses. Service-level attacks are possible in both public and private service models.

An attacker can impersonate the statistics of normal traffic. Attackers can launch attack packets at a slow rate that gracefully degrades the services or high rate. All these actions result in node-based or link-based DoS attack. The attack can be launched from a large number zombies [179] .We assume that attackers request service from servers selected randomly from pool of N servers.

6.3.3 Service Model

Our service network consists of a pool of N homogenous servers in a server pool with k active FTP servers and (N-k) honeypot servers. Each destination in the network should be able to behave according to whether the corresponding node acting as an active server or honeypot. Thus the services have been replicated on all servers.

We assume servers providing FTP service which is a TCP based service that can be replicated. An FTP connection remains alive until either the legitimate FTP request is fulfilled completely; or the tag of the flow in the *FL* (refer to Section 5.2.2.2) changes at the start of next characterization time window; or if the number and location of servers and honeypots change (eon expires) before the request is complete. An attack flow is directed to honeypot. Honeypot retains the connection and responds to attacker in contained manner for either a fixed time interval equal to the time window of characterization; or till the current eon terminates; or till the existence of flow, whichever is less. If the flow is retagged as attack in subsequent characterization window before termination of current eon, then the connection with honeypot is retained. In case of FP, the connection is migrated back to active FTP server in subsequent characterization time window.

6.4 Design Details

Our approach [219] for dynamic honeypot and active server generation is in response to flows identified as normal and suspicious attack in FL (refer to chapter 5, Equation 5.8) maintained at edge router of stub network. A Honeypot Controller (HC) has been modeled that performs three functions. First, using FL as the input, it determines the values of CL and AL. Second, it then triggers Dymanic Honeypot Engine (DHE) that performs three tasks, decides the number of honeypots and active servers, location of honeypots and active servers and the length of eon. Third, the legitimate traffic is routed to one of the randomly selected active servers in the server pool. The controller establishes a dedicated connection and adjusts routing information so that the attack traffic is forwarded to the randomly selected honeypot for interaction.

6.4.1 Design of Honeypot Controller (HC)

Figure 6.3 shows the functional diagram of HC. It takes the input from FL (output of microscopic-level characterization) and triggers DHE. It imposes either "allow" rule or "redirect" rule for packets of flows in FL in each characterization time window. It calculates CL and AL after each eon and triggers the generation of honeypot and servers.

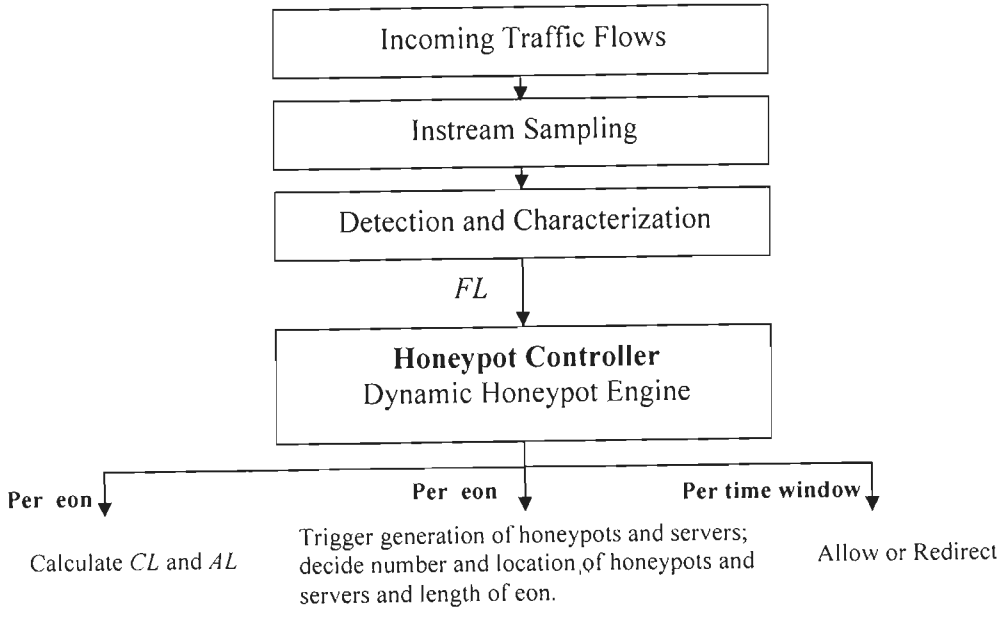


Figure 6.3. Functional diagram of honeypot controller (HC)

6.4.2 Computational Algorithms for Dynamic Honeypot Engine (DHE)

HC keeps a track of FL and state of flows over moving time windows. Using FL as the input it determines the values of CL and AL. DHE maps the current network load to determine number of servers (N_S) and number of honeypots (N_H), decides their location and length of eon. This is explained below.

6.4.2.1 Deciding the Number of Honeypots and Active Servers

Let N_S and N_H represent the number of servers and number of honeypots respectively. Our aim is to find out the optimum values of N_S and N_H depending upon the network load. We define an array *vector* of size *lengvec* such that $lengvec=N_S+N_H$; whose elements are in the form of ordered pair set of IP address and port number of the honeypots and active servers i.e. $vector[i] = \langle IP\ address; port\ no. \rangle$. *lengvec* represents the total budget N i.e.total number of servers in the server pool. We further define two arrays *subvecNS* and *subvecNH* whose elements are indices of the array *vector* such that at any time t following holds true:

$$lengvec = Length(subvecNS) + Length(subvecNH)\ AND\ subvecNS \cap subvecNH = \phi \quad (6.3)$$

where, *Length* is an operation that computes the size of an array.

Let N_{fl} represent the number of legitimate flows (tagged as normal in FL) and N_{fa} the number of attack flows (tagged as suspicious attacks in FL).

Figure 6.4 gives steps for computation of N_S and N_H by DHE.

DHE uses CL and AL to generate a judicious mixture of active servers N_S and honeypots N_H from the server pool at the start of each eon for replication and load balancing.

Table 6.1. shows the mapping from CL and AL to optimum combination of number of servers and honeypots [220], N_S and N_H respectively.

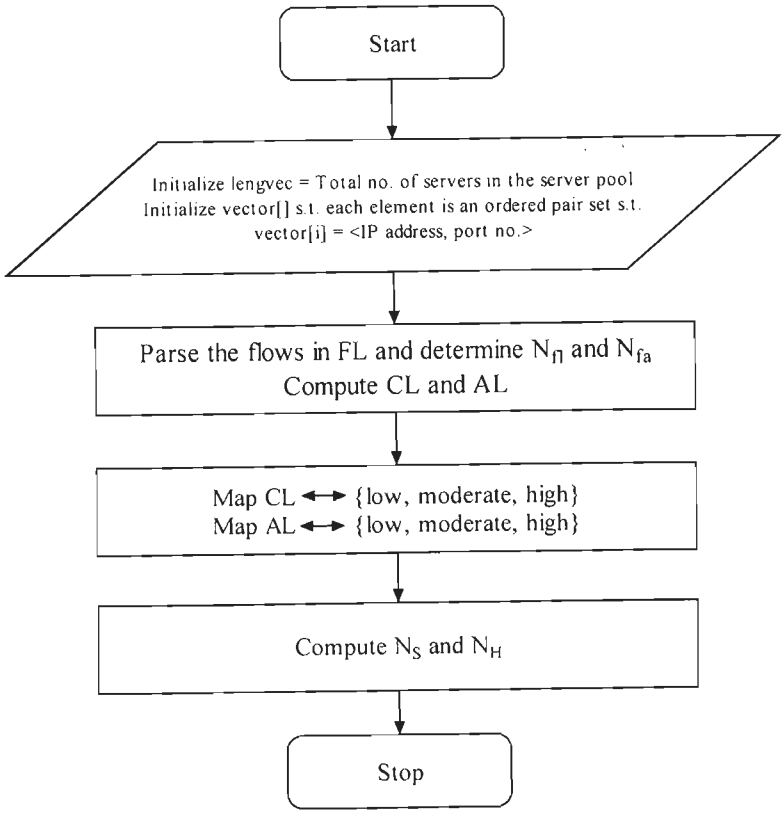


Figure 6.4. Steps for computation of N_S and N_H at DHE

Table 6.1. Mapping load to honeypots and servers

Attack Load (AL)	**Client Load** (CL)	**Number of Honeypots** (N_H)	**Number of Servers** (N_S)
Low	Low	Low	(2*Moderate) - Low
Low	Moderate	Low	(2*Moderate) - Low
Low	High	Low	(2* Moderate) - Low
Moderate	Low	Moderate	Moderate
Moderate	Moderate	Moderate	Moderate
Moderate	High	(Moderate – Low* ; Moderate)	(Moderate + Low*; Moderate)
High	Low	2 Moderate - Low	Low
High	Moderate	(Moderate; Moderate + Low**)	(Moderate; Moderate - Low**)
High	High	Moderate	Moderate

Table 6.2. Defining low, moderate and high loads

	Low	Moderate	High
CL	< Bottleneck Link BW (link utilization <1)	≅ Bottleneck Link BW (link utilization ≅ 1)	≥ Bottleneck Link BW
AL	< 30 % BW	30% - 65 % BW	> 65 % BW

Defining low, moderate and high for CL and AL

For *CL* and *AL*, the approximations for *low*, *moderate* and *high* are defined according to the bottleneck bandwidth (BW) resource according to Table 6.2.

Note that the test values are set to validate the proposed technique under network load extremities, considering high link utilization as moderate *CL*. A modest percentage of bandwidth comprising attack traffic is considered low and a high percentage of bandwidth comprising attack traffic is considered intolerable.

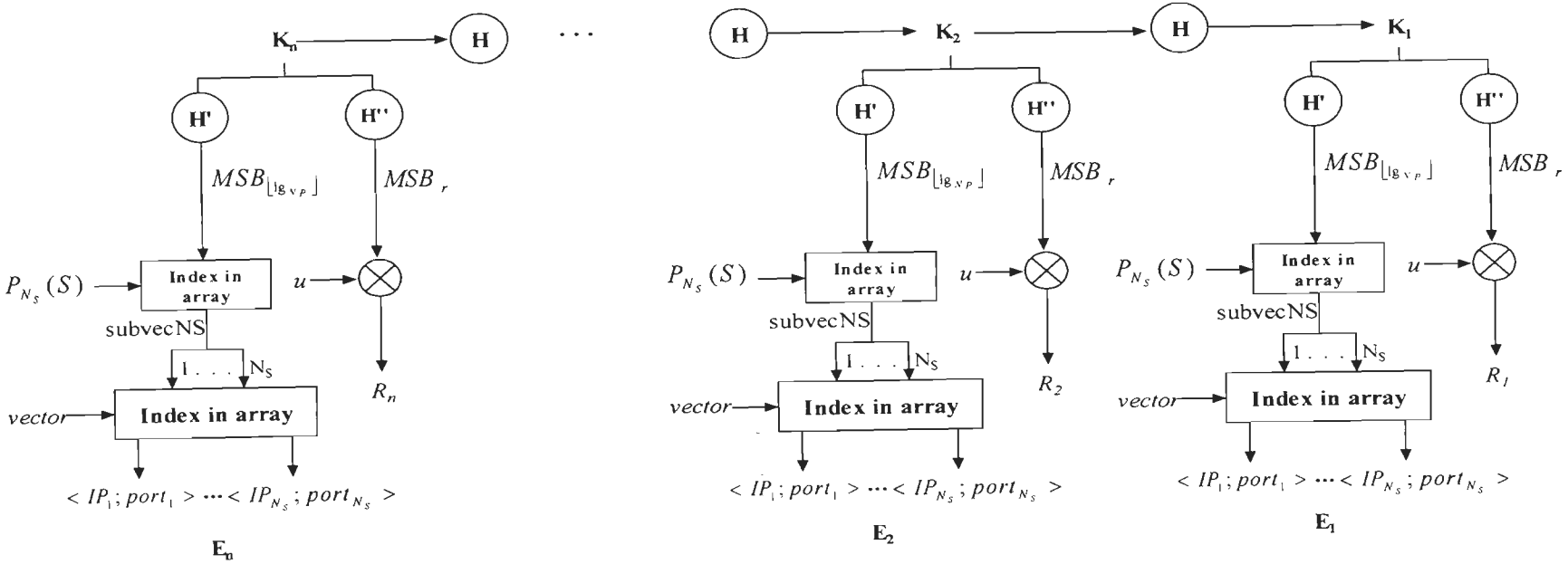
Defining low and moderate for N_S and N_H

The parametric values of *low* and *moderate* in computing N_S and N_H in Table 6.1 depends on the total budget N *which gets partitioned into* N_S and N_H ($N = N_S + N_H$) such that $moderate = N/2$; $low = \lceil \leq 30\%N \rceil$; $low^* = \lfloor \leq 30\%N \rfloor$; low^{**} alternates between $\lfloor < 30\%N \rfloor$ and $\lceil < 30\%N \rceil$ to give priority to client requests.

6.4.2.2 Deciding the Location of Honeypots and Active Servers

Having identified N_S and N_H, next we utilize backward hash chain to calculate the locations and achieve roaming as shown in Figure 6.5.

A track of long backward hash chain (discussed in Section 6.2.2.2) is kept. It is used to change the current active servers as follows: Let K_i be a key in the backward hash chain. Let an array *servers* contains an <IP address, TCP port> pair for each server in the server pool. Let S represent the set of indexes of the array *vector*. Also, let $P_{N_S}(S)$ represent an ordered set of all possible subsets of S with cardinality N_S.



H, H', H'' : One way hash functions
K_n : Last key in the chain
N : Total number of servers in the pool
E_i : i^{th} Service eon
R_i : Length of the service eon E_i
$MSB_x(y)$: x most significant bit of y
vector : set containing ordered pair set of <IP,port> for all servers in server pool
S : Set of indices of *vector*
N_S : Number of servers
$P_{N_S}(S)$: Ordered set of all possible subsets of S with cardinality N_S
N_P :Cardinality of $P_{N_S}(S)$
subvecNS :Array containing pointers to (indices of) active servers in *vector*

Figure 6.5. Calculation of server location and eon duration for dynamic honeypots and active servers

The cardinality N_P of $P_{N_S}(S)$ is

$$N_P = \begin{pmatrix} N_S \\ \\ N \end{pmatrix} \quad \Rightarrow \quad N_P = \frac{N_S!}{(N!)*(N-N_S)!} \tag{6.4}$$

Then, for service eon E_i, the set of current active servers is

$$\text{The set of current active servers} = P_{N_S}(S)[MSB_{\lfloor \lg N_P \rfloor} H'(K_i)] \tag{6.5}$$

In the above Equation, where $H'(\cdot)$ is a one-way hash function and $\text{MSB}_j(x)$ are the j most significant bits of x. The elements of the set so derived are elements of array *subvecNS*. All the elements of set S not in *subvecNS* form the elements of array *subvecNH*. The elements at indices of array *vector* that correspond to elements of *subvecNS* are addresses of active servers and elements at indices of array *vector* that correspond to elements of *subvecNH* are addresseses of honeypots.

6.4.2.3 Deciding the Length of Eon

As discussed earlier, the length of eon should be as large as possible so that file requests are handled in same eon to avoid TCP migration, connection reestablishment and TCP slow start phase. However, in case of attacked network, smaller eon is favored to quickly change the location of servers and clean their state. As shown in Figure 6.5 the length R_i, of the service eon E_i is uniformly distributed in the interval $[u, u+2^r]$ seconds and calculated as follows:

$$R_i = u + MSB_r(H''(K_i)) \tag{6.6}$$

In the above Equation, r is the dynamic parameter that is changed adaptively. r represents the current threat level signified by N_H and changes according to change in N_S and N_H. Value of r is inversely proportional to the threat level, i.e. inversely proportional to N_H and is derived from Table 6.3. These values are specific to the discussed design environment. The value u represents a lower bound on the idle time of a server and should be long enough to analyze attacks.

Table 6.3. Mapping N_S and N_H to r

r		N_S				
		L	**M-L**	**M**	**M+L**	**H**
N_H	**L**	5	5	5	5	5
	M – L	4	4	3	3	4
	M	3	4	3	4	3
	M + L	2	1	1	2	2
	H	1	1	1	1	1

L = Low; M = Moderate; H = (2*Moderate) – Low.

HC keeps account of N, N_S, u, r, servers list *vector*, and $P_{N_S}(S)$ that form the roaming updates. Values of key K in backward hash chain are distributed among clients based on their trust levels over multiple eons. The attack traffic is isolated from legitimate traffic and diffused over honeypots instead of interfering with active servers in the pool.. A higher-level responsibility of HC is to manage and prioritize the legitimate clients.

6.4.3 The Mathematical Model for Three Operating Points

At any active ftp server node i, the arriving traffic is the aggregate of several legitimate flows and possibly of some suspicious attack flows. Let $n = (n_1, n_2, ..., n_{N_n})$ be the set of normal flows and $d = (d_1, d_2, ..., d_{N_{fa}})$ be set of attack flows such that $FL = n \cup d$ and $n \cap d = \phi$. The total traffic rate λ_i arriving to any node i is composed of two parts:

$$\lambda_i = \sum_{\in n} \lambda_i^n, n + \sum_{\in d} \lambda_i^d, d \tag{6.7}$$

where λ_i^n, n is the legitimate incoming traffic rate which belongs to normal flow n, and λ_i^d, d is the arrival rate of attack packets belonging to flow d.

Ideally, if i is an active server, $\sum_{\in d} \lambda_i^d, d = 0$, such that $\lambda_i = \sum_{\in n} \lambda_i^n, n$

If i is a honeypot, $\sum_{\in n} \lambda_i^n, n = 0$ such that $\lambda_i = \sum_{\in d} \lambda_i^d, d$,

Some attack traffic may be mistakenly taken to be normal traffic while some normal traffic will be mistakenly thought to be attack traffic. Any traffic characterized as attack is redirected at one of the randomly selected honeypot servers. Thus, a fraction f_i, n of normal traffic (the probability of false alarms) and a major fraction of attack traffic d_i, d (the probability of correct detection) will be redirected to honeypot by the HC. If the

Table 6.4. Traffic fractions on honeypot and active server for different operating points

i	**Best Defense (Min false Negatives)**	**Naïve Defense (Min False Alarms)**
Active server	$1-d_i, d=0;\ 1-f_i, n$	$1-f_i, n=1;\ 1-d_i, d$
Honeypot	$d_i, d=1;\ f_i, n>0$	$d_i, d<1;\ f_i, n=0$

attack detection and characterization mechanism were perfect, at any honeypot node i, we would have $f_i, n=0$ and $d_i, d=1$.

In the best defense mode of operation (refer to Section 4.3.2.3), the probability of FN is zero i.e. practically $d_i, d=1$ at honeypot node. In the naïve defense mode of operation, probability of FP is zero i.e. practically $f_i, n=0$ at honeypot node. Normal defense is the intermediate operating point where $1 > f_i, n > 0$. Table 6.4 shows the traffic fraction at active server and honeypot for naïve and best mode of operation.

6.5 Parametric Dependencies

There exist exhaustive parametric dependencies between various phases in our defense framework and their regulation in real time makes the service network DDoS attack tolerant and insensitive to AL.

The tolerance factor a (refer to Section 4.3.2.1) tunes the framework to operate among one of the best, normal and naïve defense mode. Depending on the detection rate and false alarm rate for the selected mode, number of normal flows and suspicious attack flows vary and so do the calculated CL and AL. This effects the number of active servers N_S, number of honeypots N_H and the length of eon or migration interval (miv) which determine the network's performance. These dependencies are shown in Figure 6.6. Therefore, to analyze our framework, we study the three network performance parameters, namely, goodput, Average Response Time (ART) and Mean Time Between Failures (MTBF) for the different modes of defense under varying network conditions.

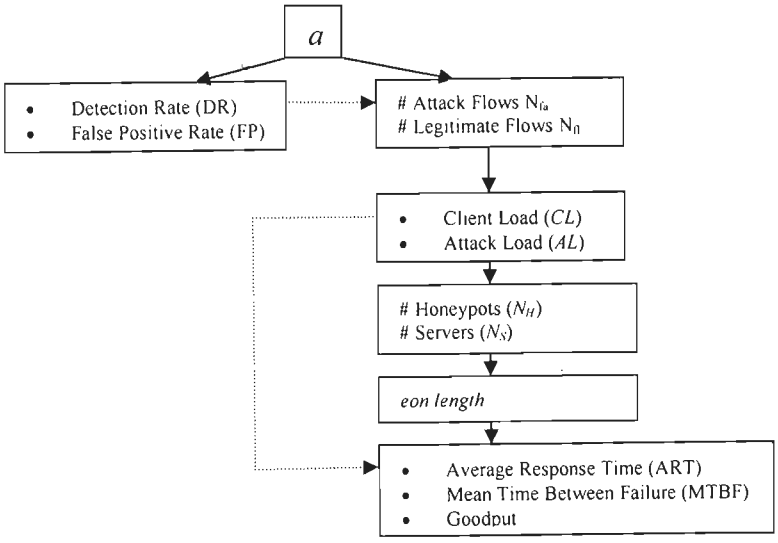


Figure 6.6. Parametric dependencies between various defense phases of the framework

6.6 Performance Evaluation

The first objective of our simulation study is to perform the cost and benefit analysis of the proposed dynamic roaming honeypot scheme for attack mitigation in several environments. Second, we evaluate our proposed framework by exhaustive analysis of three key network performance parameters namely goodput, ART and MTBF for the different modes of defense under varying network conditions.

6.6.1 Experiment Design and Procedure

We tested our scheme against an Internet type topology using network simulator 2 (ns-2) [202] as simulation testbed The parameters are same as used for evaluation of detection and characterization phase. For details on simulation model, refer to Appendix A. The 156 node network shown in Figure A.1 is composed of five FTP servers, labeled node 117, 118, 119, 120 and 121 to be protected, 137 clients of which 48 are legitimate clients and 89 are attackers and router nodes. All FTP requests are originated randomly from different client nodes. We introduce randomness to the locations of legitimate clients and attackers. However, to simplify the analysis, we carefully place servers where they all have same access bandwidth (i.e. 3 Mbps) and where they all have paths from clients to access.

To control the load of each run, we use Poisson process to model arrival process of the FTP clients. The file size of each transfer is fixed to 1 megabits. The inter-arrival time (IAT) of FTP client is computed by:

$$IAT = exp(T_{ftp}/CL), \tag{6.8}$$

In the above Equation, T_{ftp} is the average total time for file transfer, CL is the total client load, and $exp(x)$ is the exponential distribution with mean x.

To simulate sophisticated attacks crafted to match the behavior of legitimate clients, AL is computed in terms of arrival rate of attackers similar to IAT formula in Equation 6.8. Configuration parameters, including the link rate and propagation delay are summarized in Table A.3.

We use FTP client-server model for the experiment. HC maintains the FL, and distributes the dynamic roaming information to the legitimate requesting clients corresponding to normal flows in FL. Hence, a legitimate client with flow tagged as normal receives the addresses of the servers, pointers to the active servers and length of eon from HC. These clients can follow the active servers by their own computation. The simulation flowcharts for the scheme are given in Appendix B.

For the cost-benefit analysis, we carefully consider the length of eon or miv. The small interval will cause unnecessary migration, whereas the large interval will have higher chance for being attacked. We select the miv that is big enough to allow at least one client to finish a transfer within at most one migration to the highly loaded environment. According to the simulation setup a client took about 1.5 seconds to finish a transfer. Therefore miv were varied among 2, 4, 6, 8, 10, 20 and 30 seconds. We vary CL from 1-10 Mbps and total AL from 1-16 Mbps.

To evaluate our proposed framework, we vary CL among 1, 2.5, 5, 7.5 and 10 Mbps and vary AL among 1, 2.9, 6, 8.5 and 16 Mbps to represent different combinations of low, moderate and high as given in Table 6.2.

Each simulation experiment has 10 runs (averaged in the graphs). Legitimate clients send requests from time 0 to time 300 seconds and attack duration is from 50-150 seconds. A simulation run ends when all legitimate clients finish their requests. Attack mitigation parameters are as given in Table 6.5.

Table 6.5. Attack mitigation parameters

S.No.	Parameter	Value
1.	Simulation time	300 *seconds*
2.	Attack Duration	50-150 *seconds*

6.6.2 Results and Discussion

6.6.2.1 Cost – Benefit Analysis of Dynamic Roaming Honeypots

The cost is measured in terms of increased response times of legitimate clients and total number of connection migrations caused, while the benefit is measured in terms of improved response times while the server is being attacked.

Cost –Benefit Analysis of dynamic honeypot based attack isolation

We simulate classic replication technique using 5 servers. We also simulate honeypot based isolation technique using 3 servers and 2 honeypots.

We split the experiment into five cases (i) Attack and no defense, (ii) no attack and replication, (iii) attack and replication, (iv) no attack and honeypot based isolation, and (iv) attack and honeypot based isolation. First case gives the cost of leaving the network undefended, (ii) gives the cost of replication, (iii) gives the benefit of replication, (iv) gives the cost of honeypot based isolation whereas (v) gives the benefit of honeypot based isolation coupled with replication and advantage over classic replication. We also analyze the effect of varying number of honeypots for different network scenarios.

Figure 6.7 shows that attacked network left alone with no defense has a very high ART (case i). This response time increases with increase in CL. Server replication with no attacks (case ii) gives minimum ART values with increase in CL and is the best case scenario. However, in case of attacked network with replication (case iii), ART is higher than (ii) but lower than (i). This shows that replication alone is somewhat able to tolerate DDoS attacks as compared to (i), but it cannot weed them off from the path of legitimate clients and therefore have high ART values than (iii).

Replication coupled with honeypot based attack isolation gives the benefit over classic replication. Figure 6.7 shows the increase in values of ART with CL in case of honeypot based isolation with no attack (case iv) and attack (case v). There is only a small increase in ART in attacked network as compared to non-attacked network. This is because the

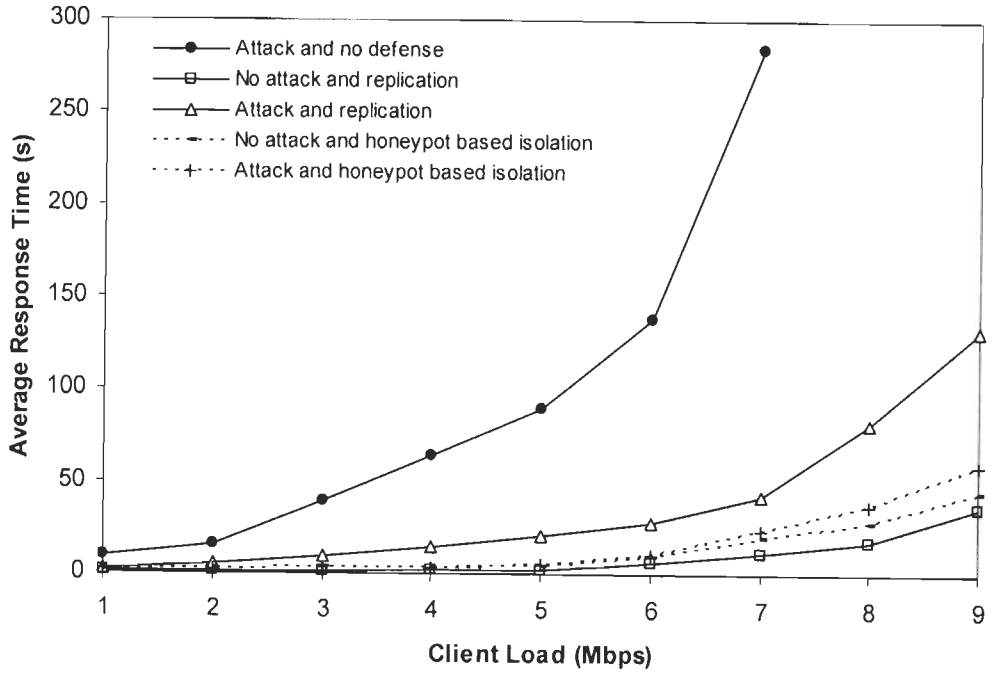


Figure 6.7 Cost benefit analysis of (i) classic replication; (ii) replication coupled with honeypot based isolation

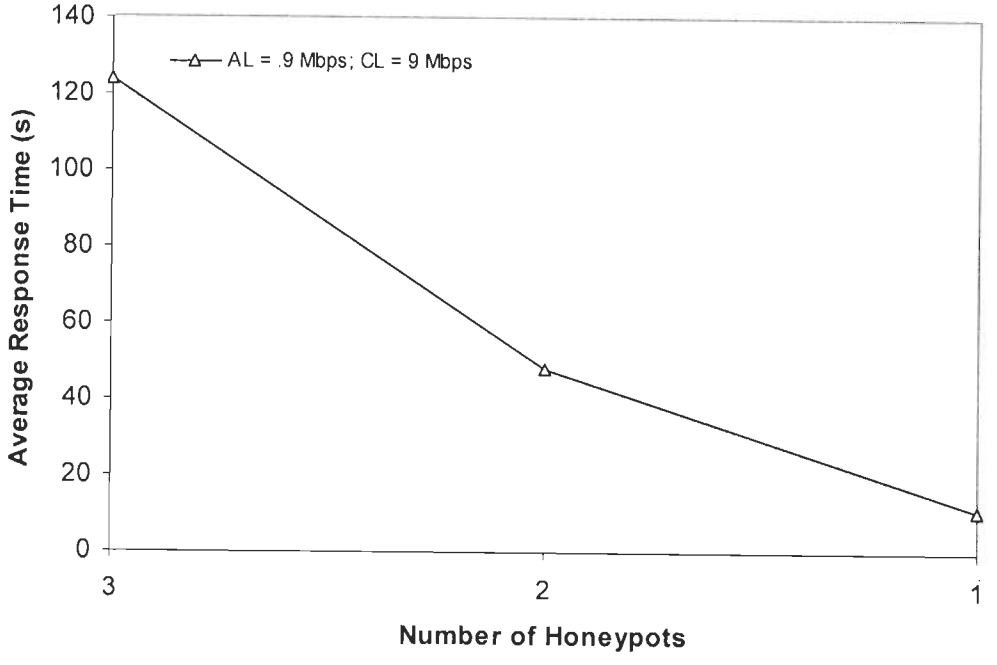


Figure 6.8. Effect of number of honeypots on ART

attacks have been weeded off from path of legitimate clients and the network becomes DDoS tolerant.

The graph also shows the cost incurred by honeypot based attack isolation in case of no attack. In the absence of attacks, honeypot based isolation (case iv) gives higher ART than classic replication scheme (case ii). In the absence of attack, at higher CL, the server replicas (case ii) are able to serve the high CL keeping ART low; whereas sacrificing some of the servers as honeypots when there is no attack increases the ART because the number of active servers which could have otherwise furnished client requests take up the role of honeypots even when there are no attacks. Therefore the degree of replication of servers and number of honeypots in a network should adapt themselves to varying network conditions.

In case of absence of attack or low AL, during high CL, static honeypot incurs a cost in terms of increased ART. Hence we propose to dynamically vary the number of honeypots. As shown in Figure 6.8, with a very low AL of 0.9 Mbps, ART decreases with decrease in number of honeypots. In a pool of 5 servers, with 3 honeypots, only 2 active servers are available to server legitimate clients whereas with 1 honeypot, number of active servers increase to 4. With the decrease in number of honeypots, the replication degree of active servers increases and more active servers are available to serve the high rate of legitimate clients requests, hence decreasing the ART.

Cost –Benefit Analysis of honeypot and server migration

For the cost benefit analysis of migration (or roaming), we split the experiment into four cases: (i) No attack and no migration, (ii) migration without being attacked, (iii) being attacked without migration and (iv) being attacked with migration. The first case gives us the cost of FTP transfer, while the comparison of second case with first gives the cost of migration. The cost incurred by attack will be shown in third case. Lastly, the fourth case will draw the benefit of deployment of our proposed scheme to mitigate the attack. Table 6.6. gives the parameter settings for the cost-benefit analysis of honeypot and server migration.

Table 6.6. Parameter settings for cost benefit analysis

Case of Simulation	Parameter Settings		
	CL (Mbps)	miv (*seconds*)	*AL(Mbps)*
1. No Migration and no attack	1-10	n/a	n/a
2. Migration and no attack	1-10	2,4,6,8,10,20,30	n/a
3. No migration and attack	1-10	n/a	1-16
4. Migration and Attack	1-10	2,4,6,8,10,20,30	1-16

Cost of Migration

We measure the cost of the roaming in terms of the increased response time (ART), the average number of migrations for a transfer and the total number of migrations in the absence of attacks. These are shown in Figures 6.9, 6.10 and 6.11 respectively. As the miv decreases, the *ART*, average number of migrations per transfer and total number of migrations increase.

The results show that miv of 2 seconds has higher ART, number of migrations per transfer and total number of migrations as compared to the case of no migration. As the miv decreases, each file transfer has to undergo higher number of migrations before the file transfer is complete. Every time migration takes place, there is a delay introduced due to TCP connection reestablishment. Therefore, as the chance for connection migration before finishing a transfer increases, it takes longer time to finish a transfer leading to increased ART. The no migration case outperforms all other cases in the absence of attacks. Overhead of migration is small in terms of response time, in absence of attack at low and moderate *CL*. As the *CL* increases, overhead of migration becomes significant.

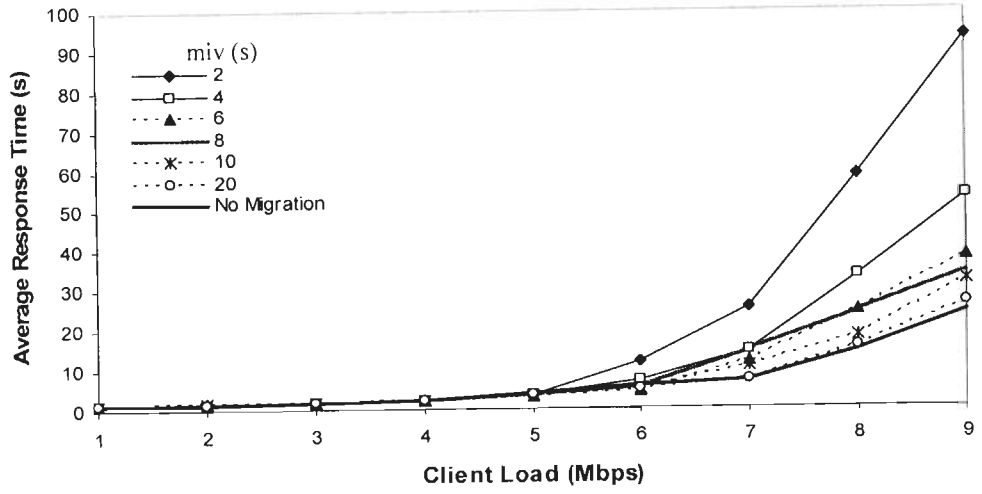


Figure 6.9. Cost of migration : Increased ART

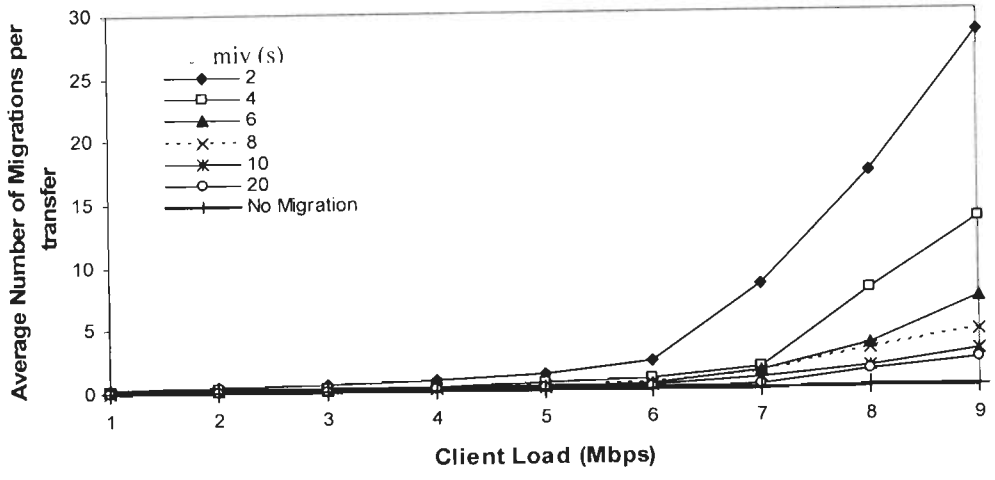


Figure 6.10. Cost of migration : Increased average number of migrations per transfer

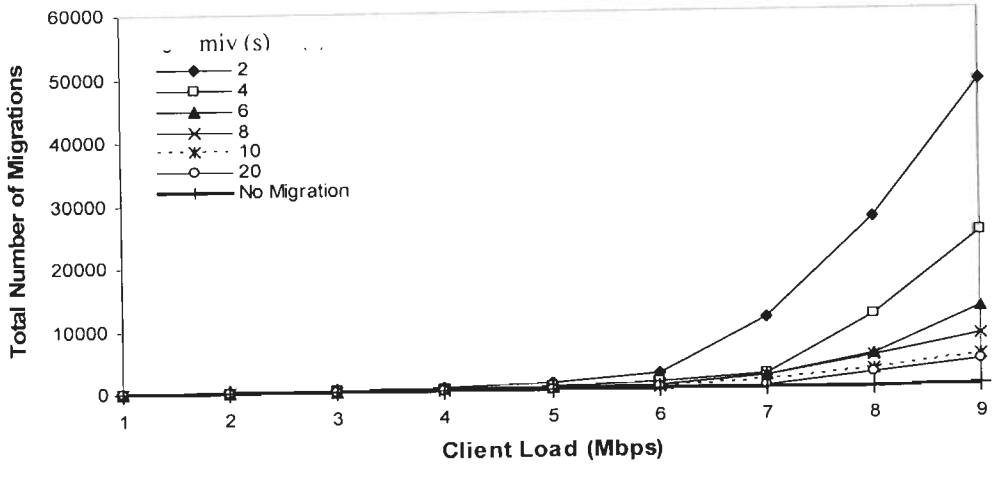


Figure 6.11. Cost of migration : Increased total number of migrations

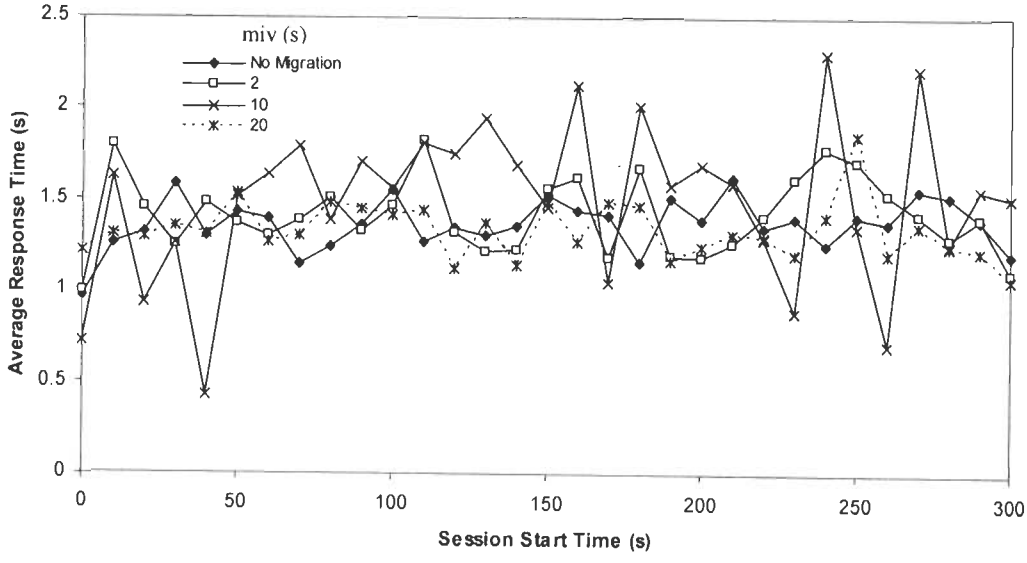


Figure 6.12. Cost of migration : $CL = 2\ Mbps$

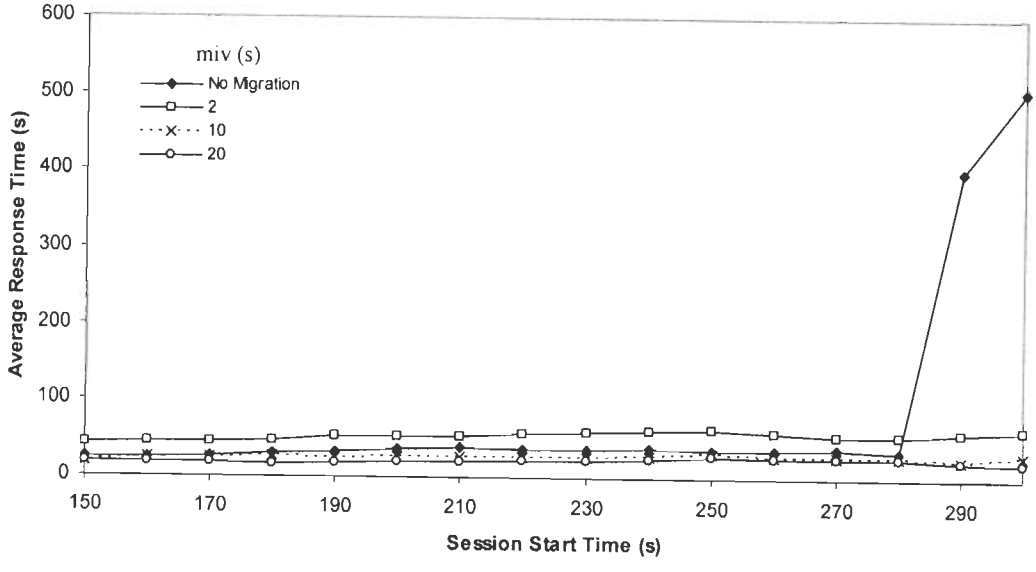


Figure 6.13. Cost of migration : $CL = 10\ Mbps$

Figure 6.12 and Figure 6.13 show the statistics in the aggregate level for simulation from 0-300 seconds for *CL* of 2 Mbps and 10 Mbps respectively. The FTP requests are grouped based on their start times in the granularity of 10 seconds and the average ART is taken for each group. At high *CL* of 10 Mbps, the ART remains stable in case of migration and increases significantly in the absence of migration, showing that cost of migration changes into profit at high *CL*. At low *CL*, the migration introduces the unnecessary cost of reestablishment. However, at a high *CL* of 10 Mbps migration reduces the effect of TCP congestion as it moves the congested TCP connection to a new path.

The results show that at high *CL*, cost incurred in the presence of the proposed mitigation scheme is less than the damage suffered by the legitimate clients in terms of increased response times in the absence of mitigation.

Cost of Attack

We measure the cost incurred by the attacks in terms of the increased ART and the number of dropped packets. The results are shown in Figures 6.14, 6.15 and 6.16 respectively. Figure 6.14 and Figure 6.15 show that all *AL* affect the ART of the FTP clients. The effect is negligible at small *AL* and gains significance as the *AL* increases. This is because as the attack flows are identified and directed towards honeypots, attack has no impact. The ART increases significantly at high *AL*.

We further validate the results by calculating the number of dropped packets for each case. Figure 6.16 shows effect of attack on the number of packets dropped. As the *AL* increases, the number of packets dropped increase.

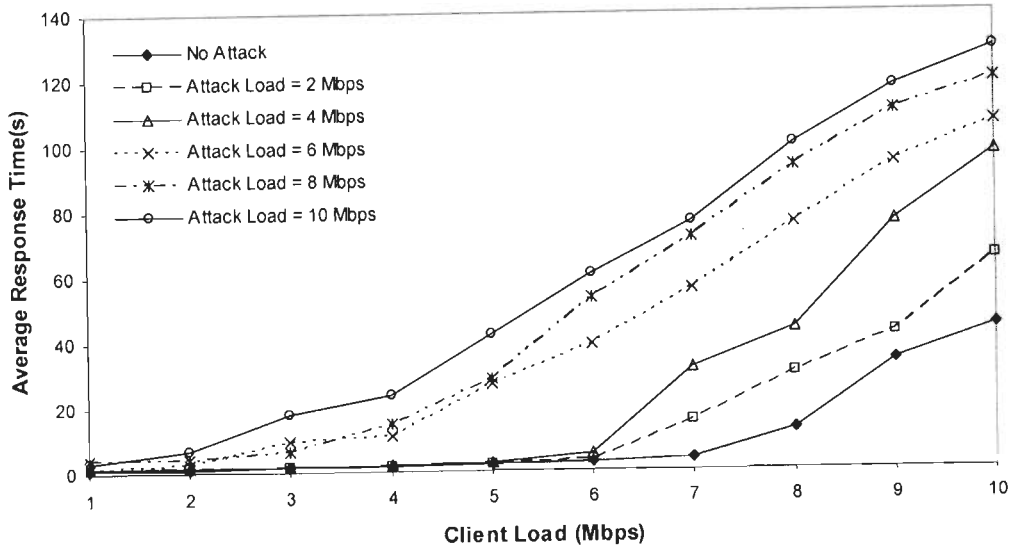


Figure 6.14. Effect of TCP Attacks on ART: Total *AL*

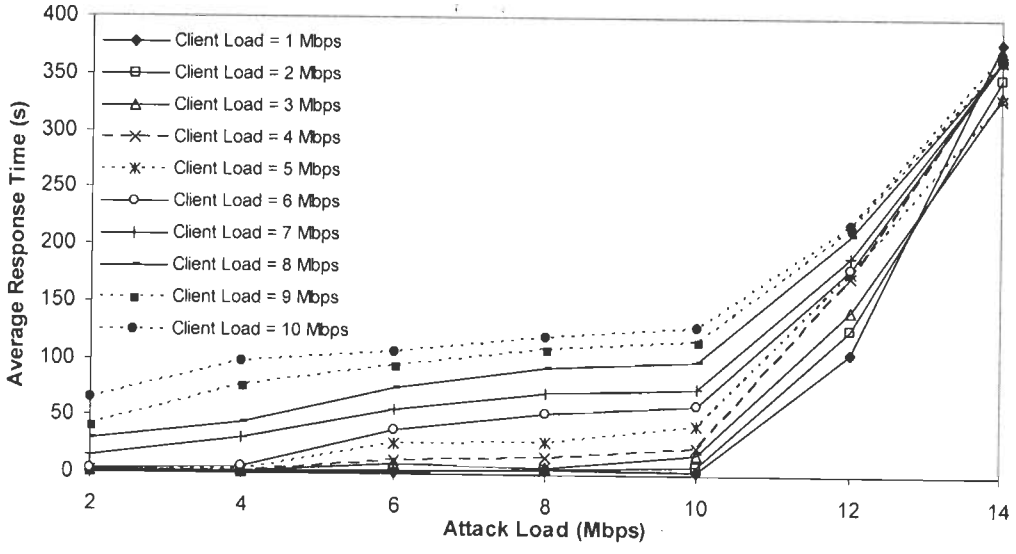


Figure 6.15. Effect of TCP attacks on ART : Total *CL*

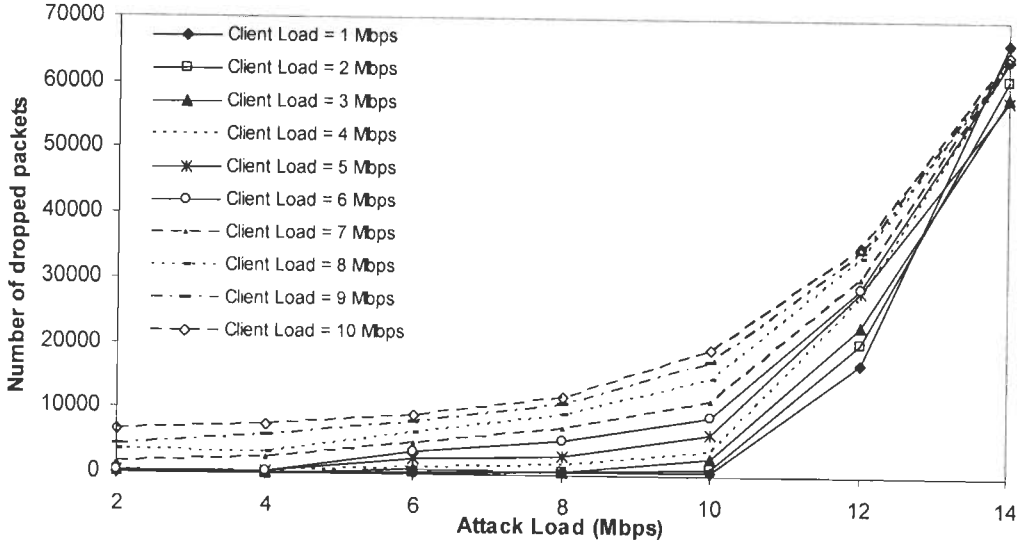


Figure 6.16. Effect of TCP attacks on number of packets dropped : Total *CL*

Benefit of Migration

In the case of the TCP attack, the migration improves the ART in all simulated cases as shown in Figures 6.17 – 6.20. The reason is simply that migrating connections to a non-attacked server decreases the effect of TCP congestion and increase opportunity for the transfers to be completed a lot quicker than leaving them with the stalled server.

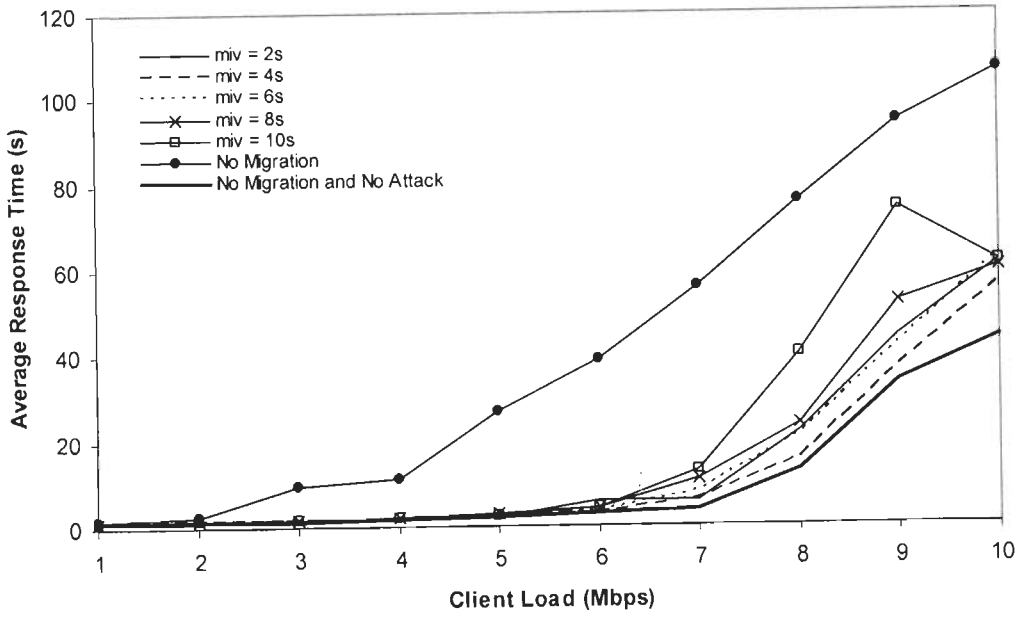


Figure 6.17. Benefit of migration : 4 Mbps *AL*

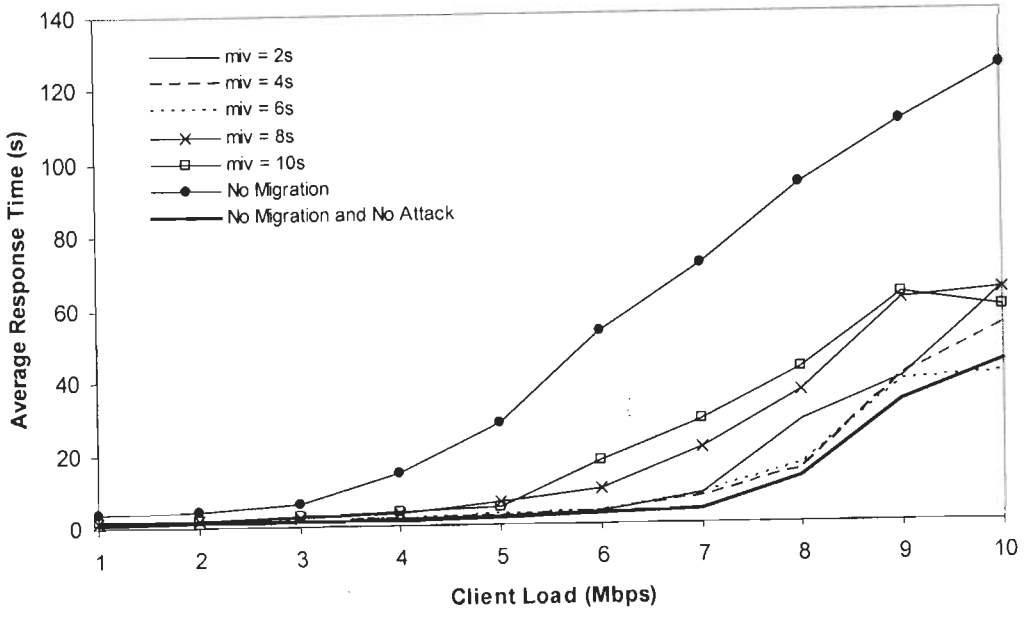


Figure 6.18. Benefit of migration : 6 Mbps *AL*

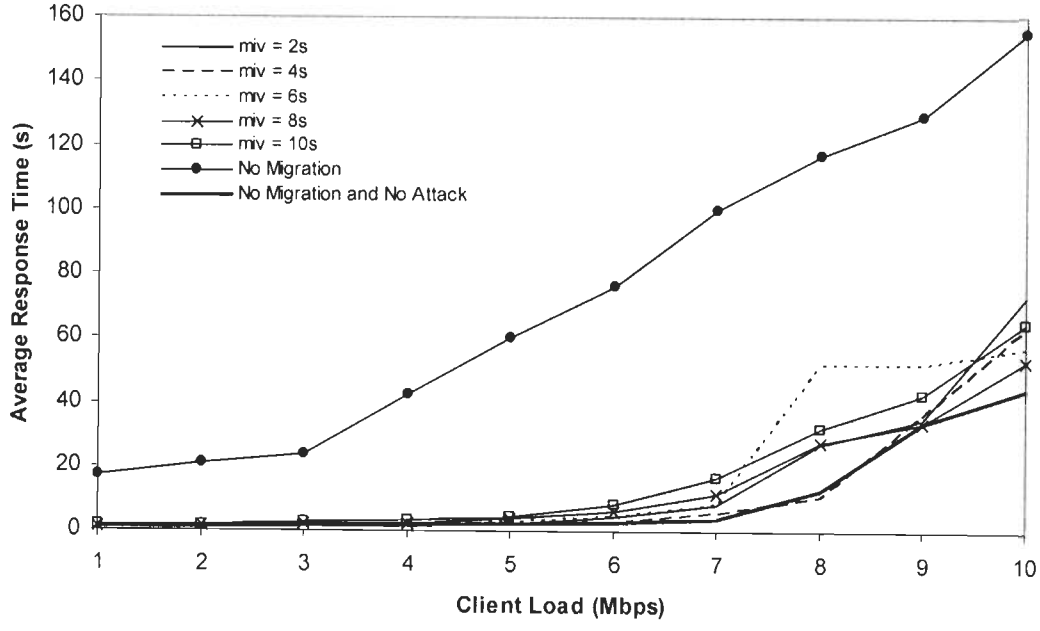


Figure 6.19. Benefit of migration : 8 Mbps *AL*

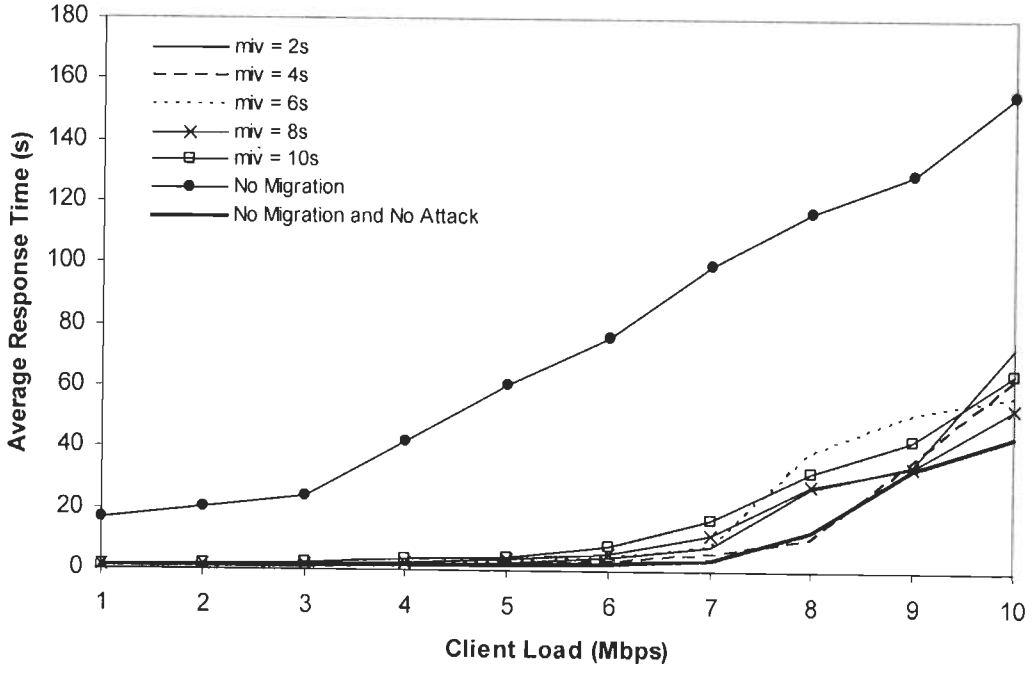


Figure 6.20. Benefit of migration : 10 Mbps *AL*

Figure 6.21 and Figure 6.22 show the statistics in the aggregate level for simulation from 0-300 seconds for *CL* of 2 Mbps and 8 Mbps respectively. The FTP requests are grouped based on their start times in the granularity of 10 seconds and the average of ART is taken for each group. In the cases of the attacks, the result shows that the migration improves the response time in all cases. As shown in Figure 6.21, all the migration cases (i.e. miv 2, 10, and 20 seconds) are below the non-migration cases, where they perceive the same *AL*. In addition, as shown in the high *CL* case in Figure 6.22, the lower the miv, the lower the ART of the clients. This can be explained as in the previous section that in the case of high *CL*, the frequent migration refreshes the TCP connections and let them start with the new fresh uncongested paths. Therefore the migrating connections are likely to progress more than the non-migrating ones.

The results show that at high *CL* and in the presence of attacks, benefit of the proposed mitigation scheme is more than the damage suffered by the legitimate clients in terms of increased response times in the absence of mitigation.

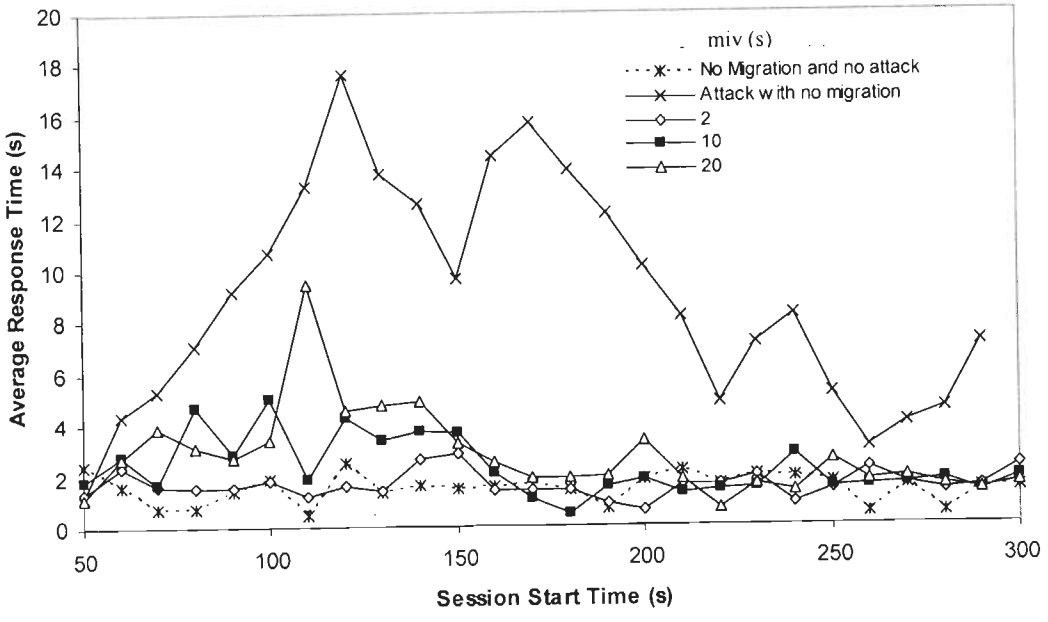


Figure 6.21. Benefit of migration : 10 Mbps *AL* ; 2 Mbps *CL*

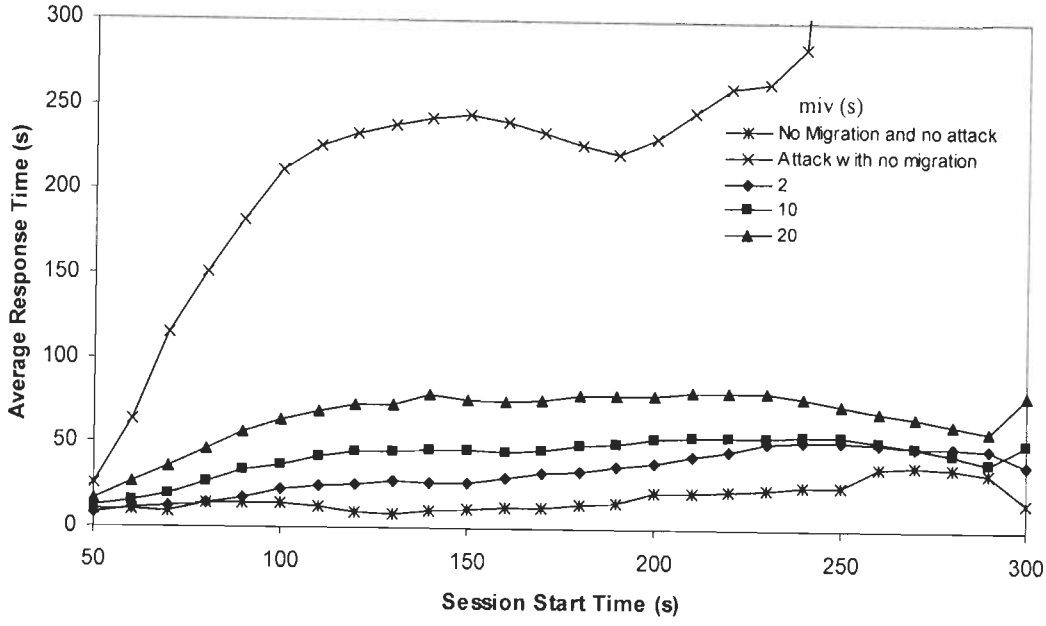


Figure 6.22. Benefit of migration : 10 Mbps *AL* ; 8 Mbps *CL*

Effect of CL

As mentioned in Section 6.2, our scheme has both reactive and proactive components. Proactive component causes all the illegitimate connections to be dropped as the server changes its location. In addition to connection dropping, we simulate filtering in response to anomalous flows for the reactive component. Whenever a honeypot server receives a flow, it records the source address into a list of attackers and makes it available to all the servers. Servers filter out flows with source addresses in the attackers list.

We analyze our scheme for three cases, (i) migration with filter enabled, (ii) migration with filter disabled and (iii) no migration. Figure 6.23 shows that ART increases with increasing *CL* for all cases. The variation in ART for the filtering and non-filtering cases is very meek. This suggests that the anomalous flows once identified and redirected to honeypots do not appear on active servers and in the path of legitimate requests. Hence filtering at active servers does not have any significant effect on ART. Therefore, filtering results validate our flow identification and redirection schemes at microscopic-level.

Effect of AL

Figure 6.24 shows that there is a very minor increase in ART with increasing *AL*. Once the attack flows are detected and redirected to honeypot servers, the attack has no impact. In case of no migration, ART increases significantly with increase in *AL*. The variation in ART for filtering and non-filtering cases is very meek due to reasons described above.

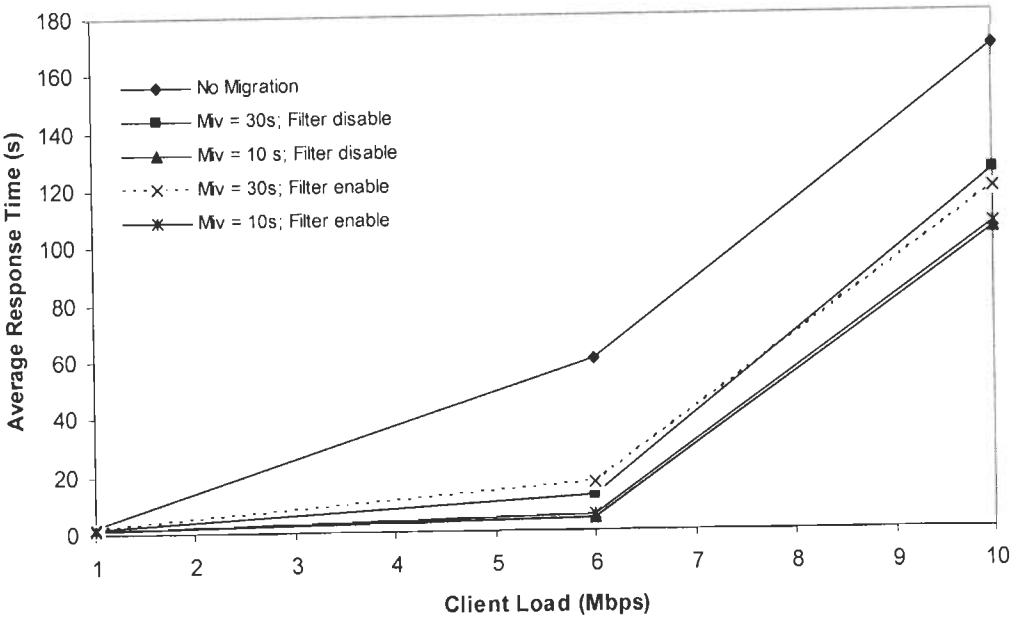


Figure 6.23. Effect of *CL* for different miv; $AL = 7Mbps$

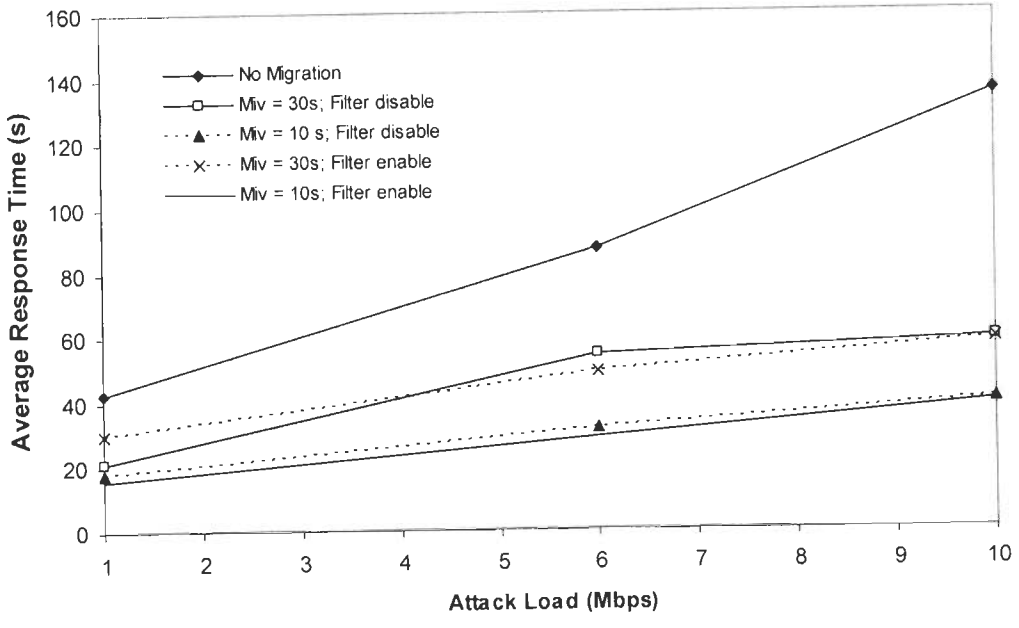


Figure 6.24. Effect of *AL* for different miv; $CL = 7Mbps$

Effect of migration interval (miv)

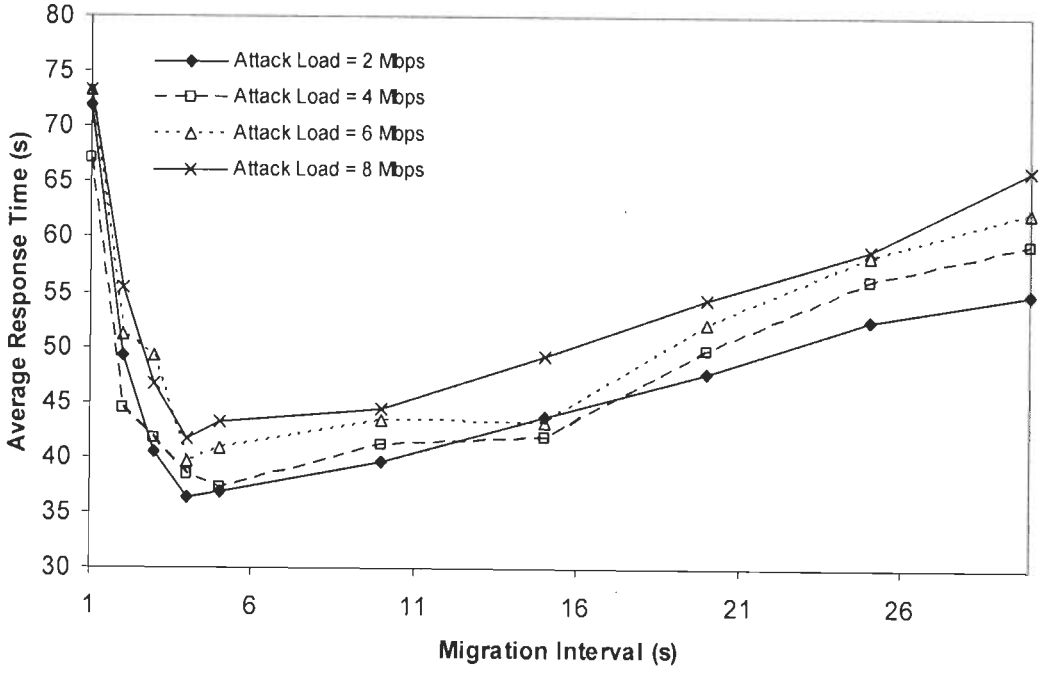


Figure 6.25. Effect of miv for different AL ; $CL = 5Mbps$

Figure 6.25 shows that for a very small miv, overhead of migration is dominant; as miv increases, the frequency of connection re-establishment and TCP slow start decreases, resulting in a decreasing ART. As miv increases beyond a particular value, the ART increases because the connection-dropping effect of migration that decongests the servers occurs less frequently with high miv and reduces the no-attack-time window developed due to migration. There exists a critical value of miv that strikes a balance between migration benefit and its overhead. From the above, it is clear that the exact value of the optimum miv depends on network characteristics, such as CL and AL. We derive these values of miv from Table 6.1 and Table 6.3 using Equation 6.6 and investigate our proposed framework in the next section.

Table 6.7 Simulation parameters for different network scenarios

Network Load	Parameter Values (Mbps)		
	Low	Moderate	High
CL	1.0	6.0	9.0
AL	3.0	6.0	12.0

6.6.2.2 Analysis of the Proposed Framework

To analyze the proposed framework, we investigate the three network performance parameters namely goodput, *ART* and MTBF for the three defense mechanisms, naïve, normal and best defense under varying network scenarios. We derive the best-case performance for each parameter and the benefits of each defense mechanism. To simulate various network scenarios, we vary *CL* and *AL* according to Table 6.7, values of which are in accordance with Table 6.2.

Goodput

The aim of any DDoS attack is to minimize legitimate traffic reaching the server. Goodput is a measure of legitimate traffic reaching at servers and is calculated as sum of bytes received per flow of all normal $\sum F_n$ divided by size of time window. Goodput gives the measure of effectiveness of the system. Goodput under different network scenarios for the three modes of defense is shown in Figure 6.26. Five cases have been simulated, (i) no attacks, (ii) best defense (iii) normal defense, (iv) naïve defense, and (v) no defense.

Figure 6.26 shows that for our proposed framework variation in goodput is independent of *AL* in most of the cases. This can be explained as follows. The flows identified suspicious as a result of microscopic-level characterization are directed to honeypots. Since, $d_1, d \cong 1$ at honeypot server (refer to Table 6.4) and due to the proactive component that drops all the illegitimate connections, the probability of FN and therefore attack flows reaching active servers is very low. Hence, the presence of attack does not cause legitimate packets to drop, and the goodput remains unaffected.

In case of low *CL*, best, normal and naïve defense schemes give maximum goodput which is almost equal to ideal goodput i.e. goodput with no DDoS. Value of N_S triggered by honeypot controller (HC) is such that there are adequate numbers of active FTP servers to fulfill all legitimate requests at low *CL*. In case of no defense, as the *AL* increases goodput decreases slightly.

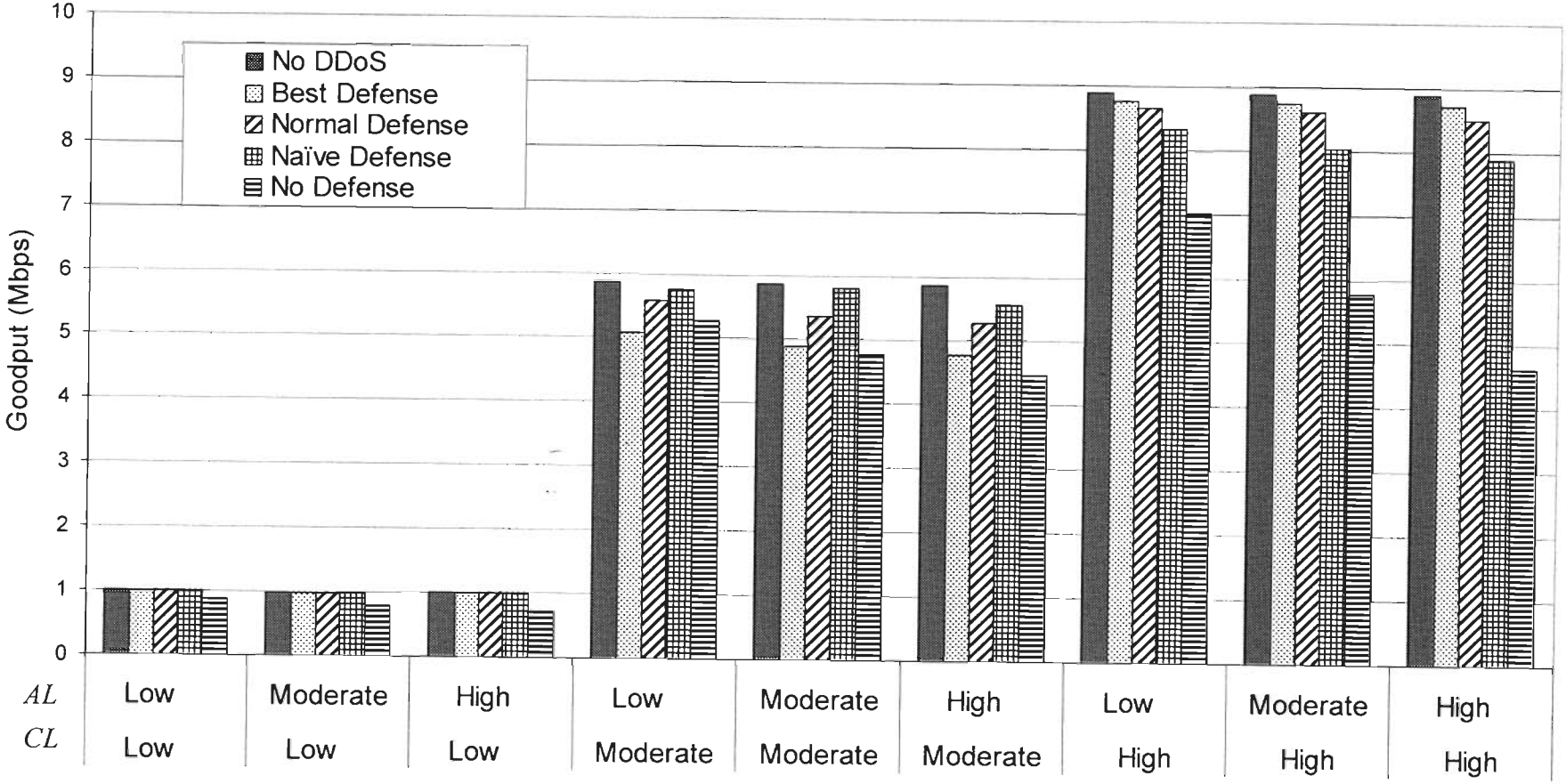


Figure 6.26. Variation in goodput under different network scenarios for the three modes of defense

In case of moderate CL and no defense, goodput under no DDoS is almost equal to ideal goodput and falls with increase in AL in absence of defense. For moderate CL and particular value of attack, say moderate AL, naive defense gives higher goodput than best defense. This can be explained as follows. In case of naïve defense, the number of legitimate clients being sent directly to active FTP servers is high as compared to best defense, where some legitimate flows may be detected as attacks and directed to honeypots. Since the fraction of legitimate traffic going to honeypot servers is nearly zero $(f_l, n=0)$ in naïve defense, the rate of FP reduce and high fraction of legitimate clients reach the active server. Hence the goodput is high as compared to best defense. With moderate CL and a particular defense, say naïve, increase in AL decreases the goodput slightly. As the AL increases, HC triggers new values of N_S and N_H, there is a probability that more servers are reserved as honeypots, resulting in lesser servers to be available to service client requests, causing higher drop of legitimate packets due to increased incoming traffic at limited active FTP servers. Decrease in goodput cannot go beyond a limit because there is a lower threshold bound on the number active servers (refer to Table.6.1). Hence the mapping scheme maintains stable network functionality.

In case of high CL, goodput in absence of attack is slightly lesser than ideal goodput due to loss of packets caused by buffer overflow as high legitimate flows flock active FTP servers. For high CL and a particular value of AL, say moderate AL, best defense gives higher goodput than naïve defense. This behavior is exactly opposite to that of moderate CL and can be explained as follows. As the CL is high, best defense sends a selected set of CL to limited active FTP servers to be processed efficiently without loss while dropping the attack flows. However, in case of naïve defense, more number of client requests to be sent to active FTP servers along with a higher attack traffic entering the network. It increases the processing load on active FTP servers causing high packet drops and losses thus reducing goodput. With high CL and a particular defense, say naïve, increase in AL decreases the goodput slightly. This behavior is same as shown in case of moderate CL as explained. In case of no defense, goodput decreases significantly with increase in AL.

Figures 6.27 and 6.28 summarize the results of the goodput. Figure 6.27 illustrates that for DDoS attacked network, percentage reduction in goodput in case of no defense is much higher than any other mode of defense. Any one of the three modes of defense improves the network performance by lowering the percentage reduction in goodput. From Figures 6.27 and 6.28, we also conclude that tuning the system for best defense at high CL and naïve defense at low CL gives the best performance in terms of goodput.

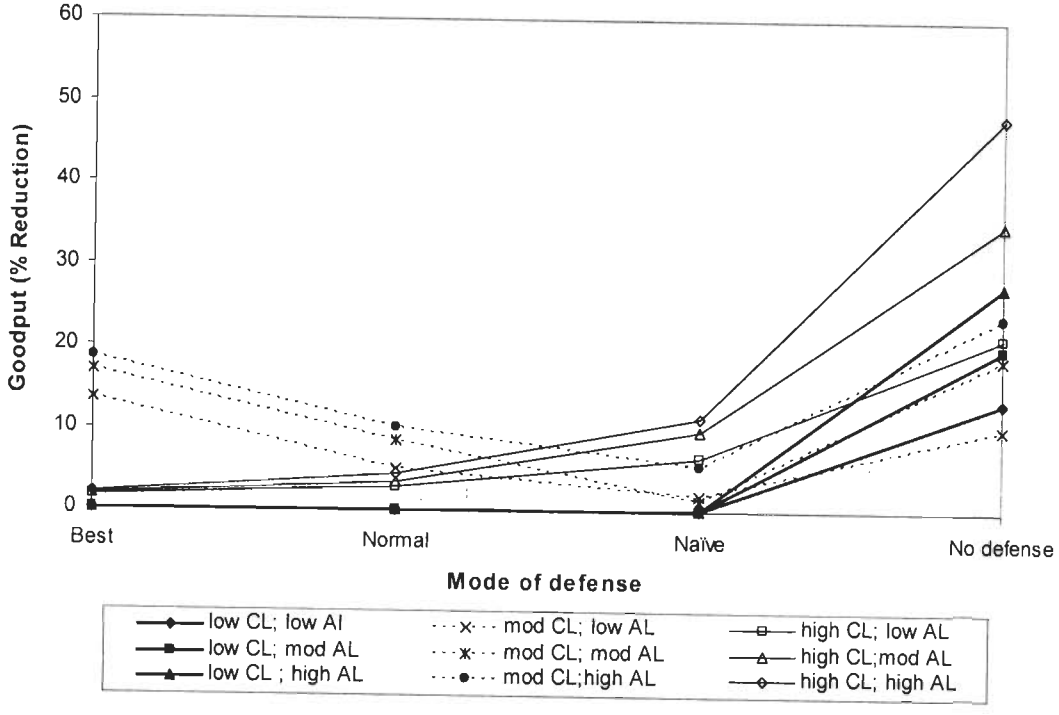


Figure 6.27. Percentage reduction in goodput for various modes of defense w.r.t. goodput in case of no attack

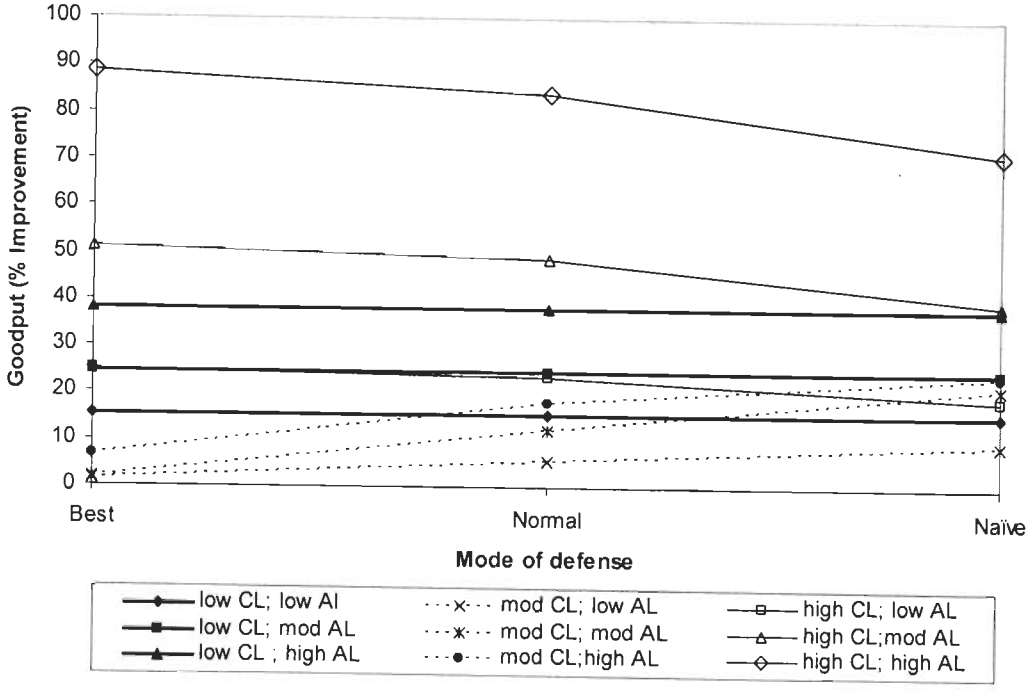


Figure 6.28. Percentage improvement in goodput for various modes of defense w.r.t. goodput in case of no defense

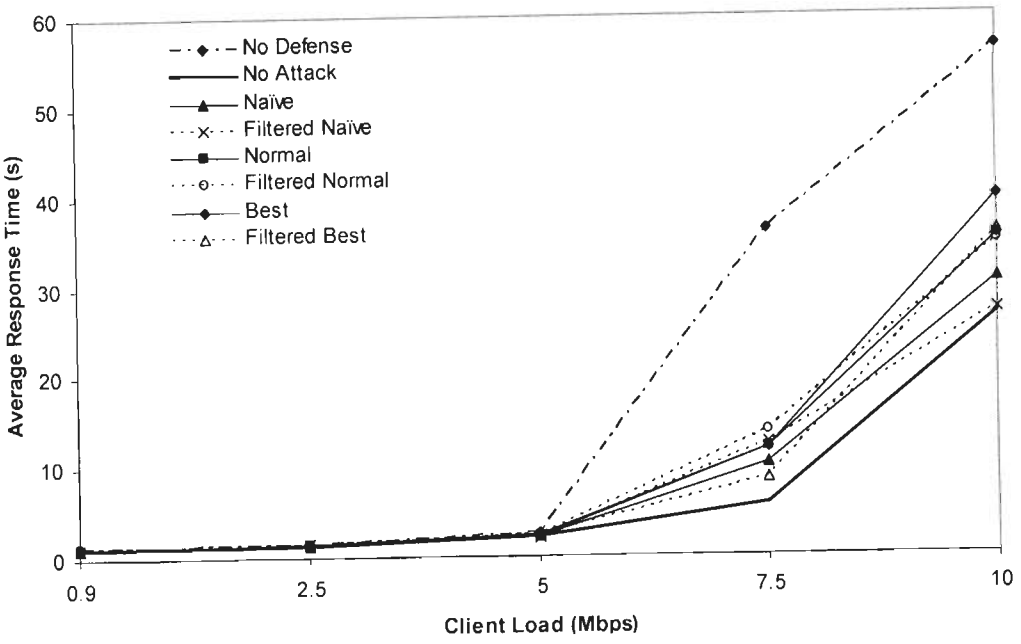


Figure 6.29. Variation of ART with CL (Low AL)

Average Response Time (ART)

ART signifies the efficiency of the system and gives a measure of system functionality in the presence of AL. We study the ART in varying network scenarios for the three modes of defense. We compare the ART for legitimate clients when filtering scheme is implemented (refer to Section 6.6.2.1.) with respect to scheme using no filtering for all three modes of defense. To simulate different network scenarios, we use the low, moderate and high values of CL and AL as given in Table 6.7.

Figure 6.29 shows that in the absence of defense, the ART increases with an increase in CL even in the presence of small AL. On the other hand, in the absence of attacks, the scheme gives a stable ART up to limited CL due to the generation of adequate number of servers to service the legitimate requests. In case of low AL, the naïve defense gives lowest ART values. In case of low AL, ART is higher for best defense as compared to naïve defense. Naive defense has high FN and low FP. As AL is less, FN become insignificant and do not interfere with normal legitimate flows. Most of the legitimate flows are identified and serviced, hence ART decreases. Best defense gives high FP and low FN. As the CL increases, the numbers of FP increase. These flows are first directed to the honeypots before being redirected to active servers. This leads to an increase in values of ART with increasing CL in case of best defense. Filtering scheme performs almost the same as its non-filtering counterpart due to the reasons described earlier (refer to Section 6.6.2.1).

Figure 6.30 shows the variation in ART with increasing CL at high AL. At high AL, best defense gives better ART values as compared to naive defense. It is so because best defense gives high FP and low FN. Due to negligible FN, best defense is able to identify all attacks even at high AL and this isolates the active servers from high AL, thus maintaining the stable ART. On the other hand, naïve defense gives low FP but high FN. So at higher AL, more attackers remain unidentified and flock the server, giving higher values of ART. At a very high CL, the ART curve shows abrupt and chaotic behavior due to the reasons explained later in this section. Filtering scheme performs almost the same as its non-filtering counterpart due to the reasons described earlier (refer to Section 6.6.2.1).

Figure 6.31 shows in case of no DDoS defense, ART increases significantly with a slight increase in AL. In the presence of dynamic honeypots, in case of naïve defense, the ART increases linearly with increase in AL. At high AL, naïve defense gives higher FN. Hence AL may go to active servers causing congestion and increased response times. For normal and best defense, ART remains almost consistent even with the increase in AL. This is so because there are almost negligible FN in case of normal defense and best defense. All the attacks are either filtered or redirected to honeypots as soon as they are detected, isolating the active servers from effect of attacks, hence giving a stable ART.

Figure 6.32 shows variation in ART with AL in the presence of high CL. In the case of no defense, ART is very sensitive to AL and increases even with a slight presence of AL for high CL. It is so because, with a high CL, presence of even small AL would cause the disruption of services and increase in ART if no defense is present. However, in the presence of dynamic honeypots, all attack traffic is diverted to honeypot relieving the active servers from the attack. For high CL and low AL, naïve defense has lower ART as compared to best defense. It is so because, with a high CL, naive defense generates negligible FP and due to low AL, does not give too many FN and only little attack traffic is diverted to active server. While in case of best defense, large legitimate population is detected as attack and goes to honeypot before being directed to active server, thus increasing the values of ART. At high AL and high CL, for best defense, all the attacks are either detected and filtered or redirected to honeypots, with minimum FN. Hence, active server remains isolated from attacks giving a consistent ART. But due to large number of FP due to high CL and best defense, legitimate clients may be directed towards honeypots before being redirected

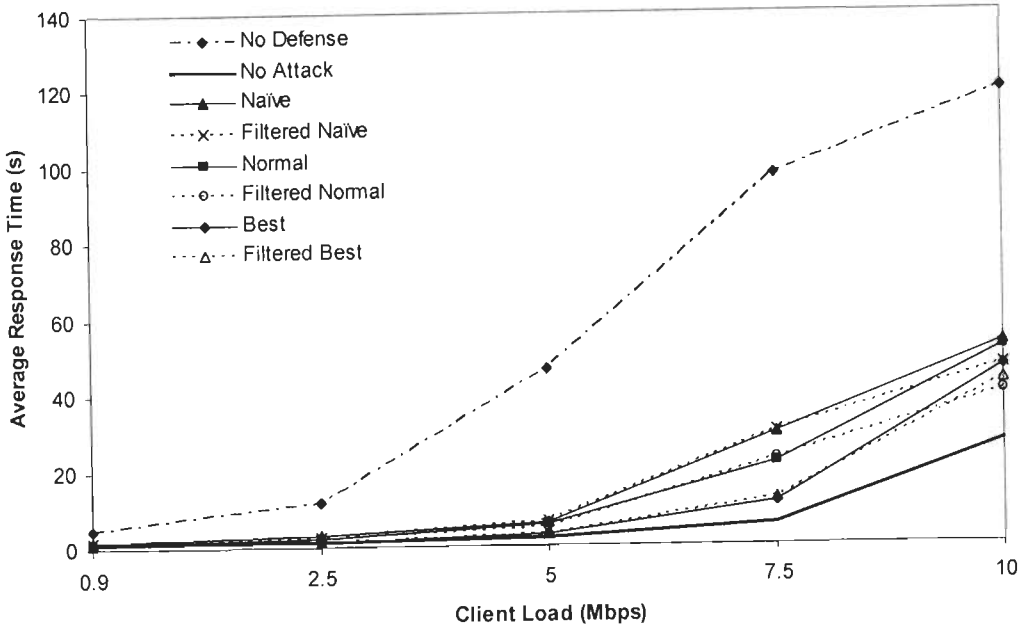


Figure 6.30. Variation of ART with CL (High AL)

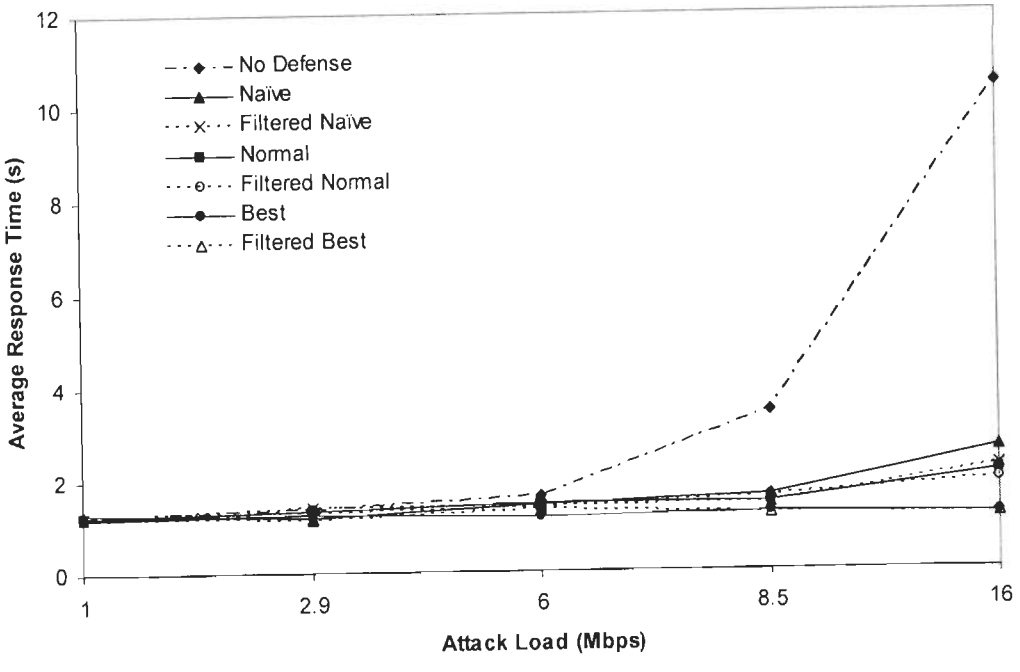


Figure 6.31. Variation of ART with AL (Low CL)

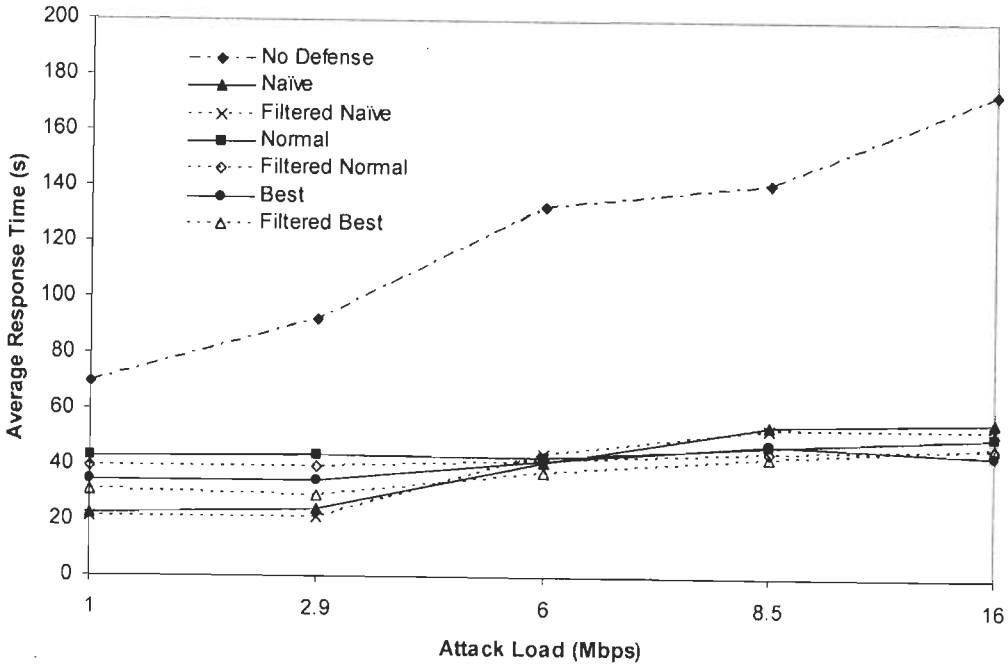


Figure 6.32. Variation of ART with AL (High CL)

towards active servers, thus increasing ART. On the other hand, naïve defense may decrease FP but will increase FN at high AL, thus clogging the active servers leading to increases ART. Therefore, high AL and high CL gives chaotic behavior because high AL favors best defense whereas high CL favors naïve defense.

The graphs in Figure 6.31 and Figure 6.32 demonstrate that our framework with optimum parameters show AL independent behavior up to a certain amount of attack traffic and is capable of giving stable network functionality with even in the presence of attacks. Also, filtering schemes perform almost the same as their non-filtering counterpart due to the reasons described earlier (refer to Section 6.6.2.1).

Figures 6.33 to 6.36 summarize the results of the goodput. Figures show that network with high CL goes in favor of naïve defense and network with high AL goes in favor of best defense for optimum ART.

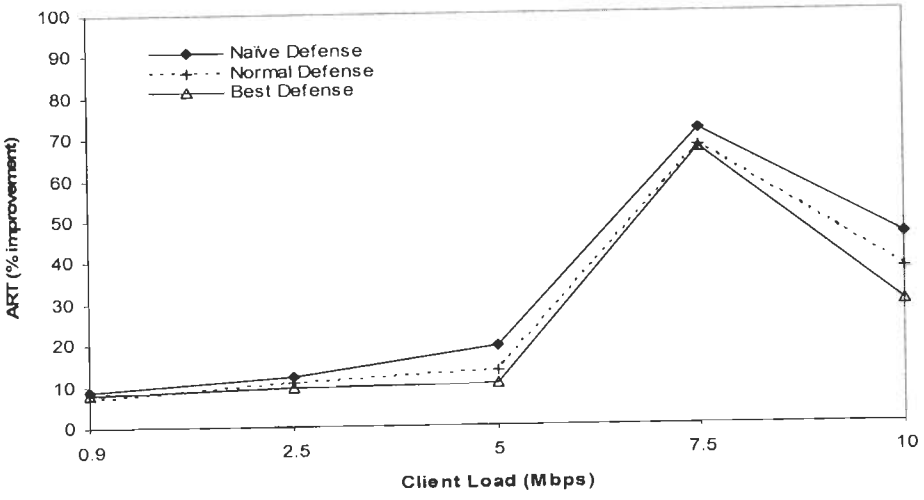


Figure 6.33. Percentage improvement in ART with CL (Low AL)

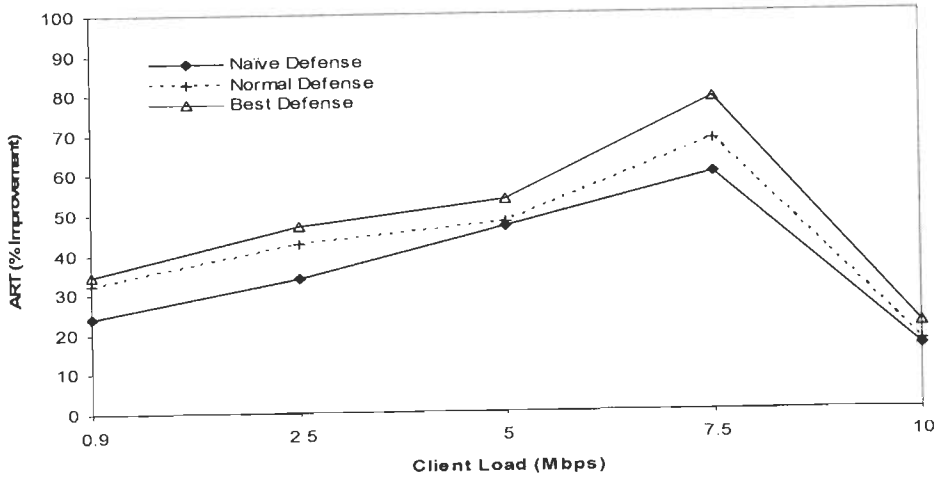


Figure 6.34. Percentage improvement in ART with CL (High AL)

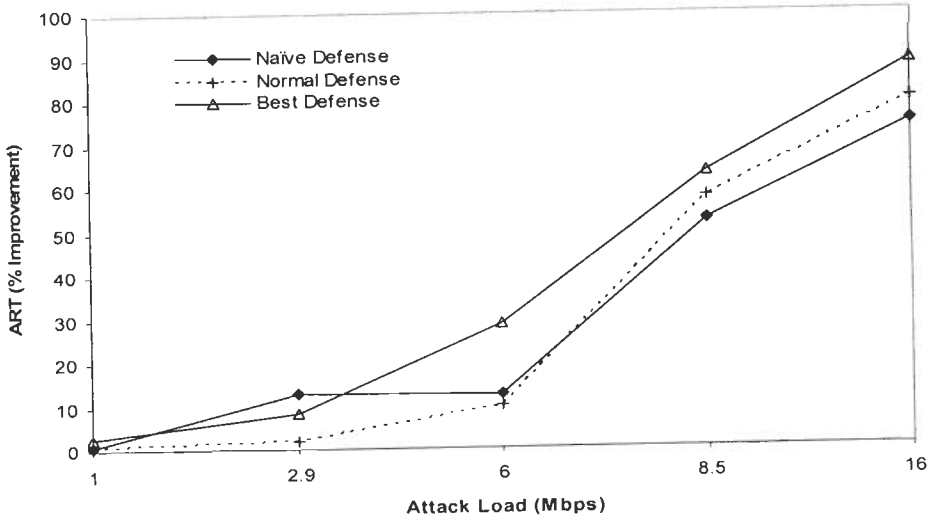


Figure 6.35. Percentage improvement in ART with AL (Low CL)

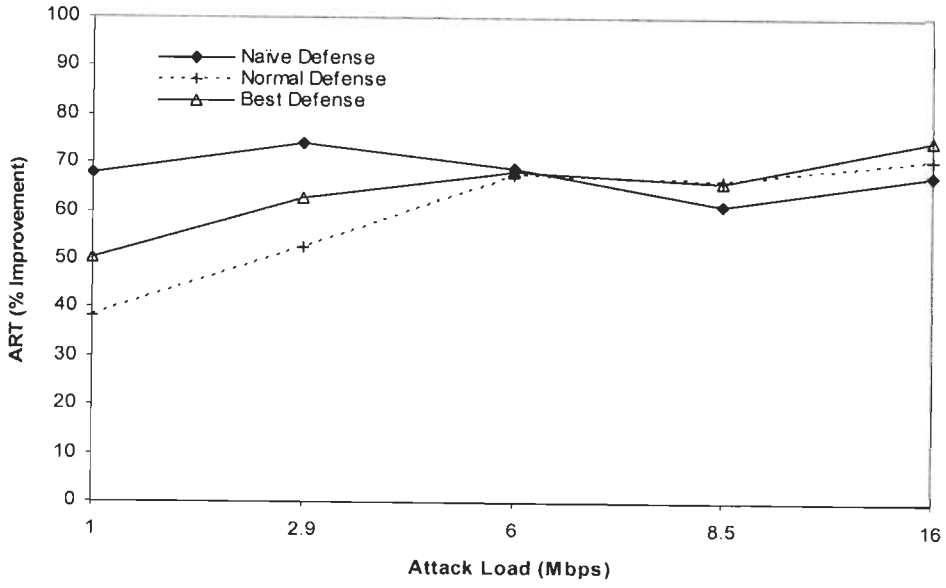


Figure 6.36. Percentage improvement in ART with *AL* (High *CL*)

Mean Time Between Failures (MTBF)

Mean time between failures signifies the reliability. It is an attribute than quantifies attack tolerance of the network. Consider the network as a system whose input is the client request and output is the response to the request. Ideally, system is said to have failed if there is no response to the client request or when the ART becomes infinite. However, a system must also guarantee a promised QoS where the response to a request should reach the client within a predefined interval or there is an upper limit on the ART. For the purpose of simulation, we assume the system to have failed if the ART becomes greater than the sum of duration of last five eons. Figure 6.37 show variation of MTBF with increasing *AL*. We overloaded the network with extremely high *CL* and *AL* (varying from 20 Mbps to order of 100 Mbps) and analyzed MTBF to determine behavior of network for three modes of defense, namely, best, normal and naïve defense.

In Figure 6.37, angular distance represents the *AL* whereas radial distance represents the MTBF. The end point of radar plot curves if concentrated at center (i.e. radial distance becomes zero) show zero MTBF or total network failure (Small angular distance and radial distance of the series with series converging at center shows failure). Greater angular distance and high radial distance shows the capability of framework to survive under high *AL*. In the presence of high *CL*, network survives with increase in *AL* but increase in *AL* beyond a limit causes abrupt decrease in stability. The network is comparatively more stable with respect to moderate and low *CL*. In case of high *CL*, best

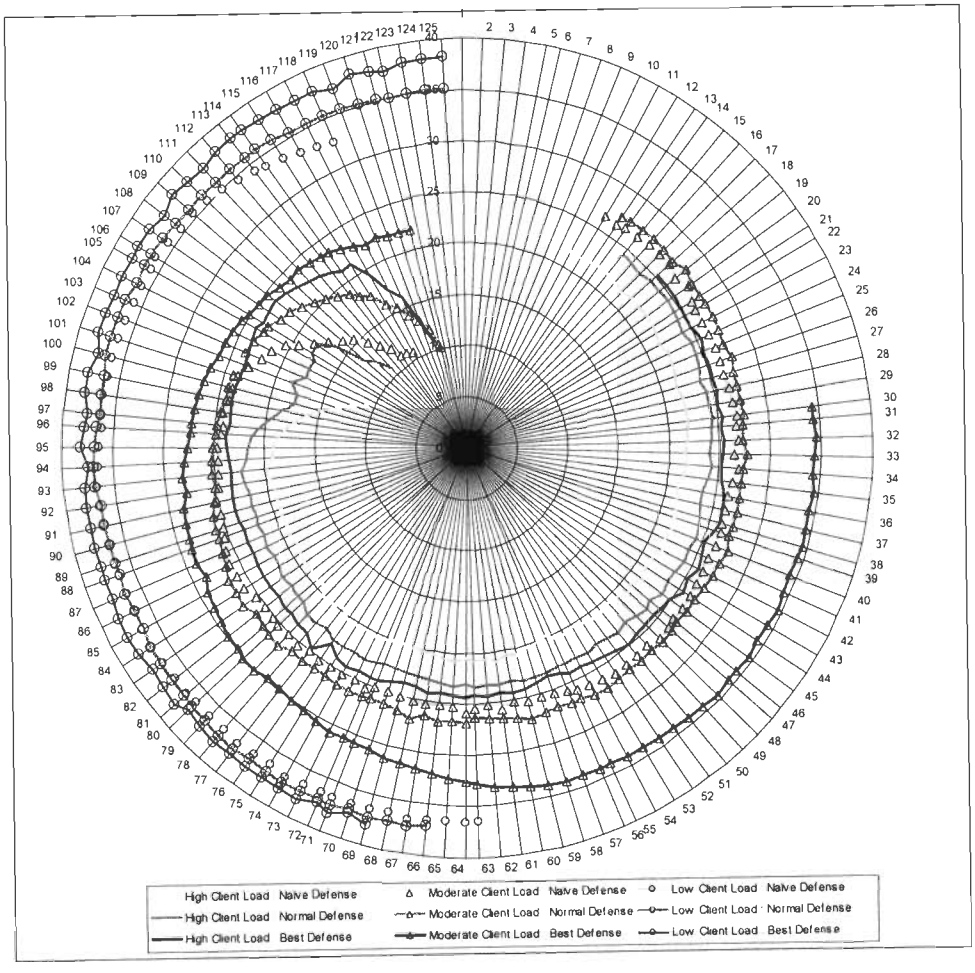


Figure 6.37. Variation of MTBF with *AL*: Naïve, Normal and Best defense; The Radar Plot

defense gives better stability as compared to naïve defense. It is so because in best defense, FN are nearly zero and all the attack flows are detected and directed to honeypots. Whereas in case of naïve defense, at high *AL*, many attack flows will go undetected and will be directed to active server causing disruption in services and failure of the network with increase in *AL*.

At moderate and low *CL*, best defense has greater MTBF than naïve defense due to similar reasons as described. It is notable that no abrupt failure is reported in case of moderate and low *CL* in the presence of high *AL*. However, at low *CL*, the network becomes much stable with no total network failure reported even at very high *AL*. Moreover stability remains consistent even if the *AL* increases to very high extremes in case of low *CL*, normal and naïve defense. This demonstrates that the proposed framework has the potential to give stable network functionality even in the presence of attacks without service disruptions. Best defense gives best performance for MTBF under any amount of network load.

Major observations from the performance evaluation are summarized in the following table:

Table 6.8. Modes of operation for Goodput, MTBF and ART

	Goodput			MTBF			ART		
AL/CL	Low	Mod	High	Low	Mod	High	Low	Mod	High
Low	Na, No, B	Na	B	B	B	B	Na	Na	Na
Mod	Na, No, B	Na	B	B	B	B	B	No	Na
High	Na, No, B	Na	B	B	B	B	B	Chaotic	Chaotic

Na: Naïve Defense; No: Normal Defense; B: Best Defense

The entire results for the three modes of operation are tabulated in Table 6.8. In general, high *AL* favors best defense whereas high *CL* tunes the network to operate in naïve defense mode. From the above discussion, it is clear that our proposed framework shows significant improvement in important network performance parameters with a minor tradeoff at high *AL* and high *CL*. The priority of network parameters also determines the mode of defense. For e.g., at low *AL* and high *CL*, ART goes in favor of naïve defense whereas goodput goes in favor of best defense. The framework is auto responsive to changes in network for resource allocation and load balancing. An increase in *AL* triggers best defense mode in subsequent detection window to maintain stable network functionality. Hence the delays and disruptions are handled proactively, i.e. before they occur.

6.7 Conclusions

We propose an autonomous mitigation scheme against DDoS attacks. The proposed scheme aims to restrict the suspicious attack flow reach the server and isolates them. The judicious mixture and self organization of active servers and honeypots at different time intervals provide continued client services with guaranteed QoS even in attacked network and under dynamically changing network conditions. It protects both the servers and the victim network. Our scheme provides a proactive approach to mitigation against the attack. The location of active server changes before the attack builds up. Overhead of migration is small in terms of response time, in absence of attack at low and moderate *CL*.

At high CL, even in absence of attacks, migration outperforms the other cases as the benefits of the proposed scheme are more than the damage suffered by legitimate clients in absence of the scheme.

In the overall framework, we presented the exhaustive parametric dependencies at various phases of attack and their regulation in real time to make the service network DDoS attack tolerant and insensitive to AL. We evaluated framework in a comprehensive way, including both benefits and disadvantages (worst-case performance). The results are very promising and show that the proposed framework is robust, resilient and can withstand high levels of DDoS attacks even in peak client hours, with client services remaining unaffected. Hence the framework provides stable network functionality with load balancing and resource management. Our framework shows a favorable AL independent behavior, for AL below a threshold. The framework acknowledges situation dependency of defense modes, and is able to choose the most suitable defense from available defense modes against a specific network scenario and user requirements.

Chapter 7

Conclusions and Scope for Future Work

7.1 Conclusions

Existing challenges in current approaches for defending against DDoS attacks are: to detect the attacks early in the network, to identify the attack traffic accurately and efficiently, to respond to congestion inducing attack traffic before they reach the victim without causing collateral damage, and to mitigate the effects of the attack while providing a guaranteed Quality of Service (QoS). In this thesis, a honeypot framework has been proposed that provides an integrated solution for networks under DDoS attacks and is based on the three lines of defense, namely detection; characterization and response; and mitigation. Efficient techniques have been proposed for each of the above lines of defense. The major contributions of our work can be summarized as follows:

i. Novel techniques for detecting DDoS attacks have been devised. The Dual-Level Attack Detector (D-LAD) operates at two levels i.e. macroscopic-level and microscopic level. Using the macroscopic-level attack detectors (Ma-LAD), the congestion inducing attack is detected early in the network, before it converges at the victim.

ii. The proposed detection techniques are fast and have low computational and memory overheads. The D-LAD exploits the concepts entropy to measure the traffic feature distributions at both the levels. Since a single attribute entropy detector can process packets in OC3 range, it has high processing speed and low computational costs. The microscopic-level detectors (Mi-LAD) are based on CUSUM algorithms that are single-stage, recursive, exponentially weighted estimators. They are the low complexity and low memory use. Hence the proposed techniques are fast and do not have computational and memory overheads.

iii. Besides being computationally fast, D-LAD is simple to implement as it is based on easily accessible information of source IP at the edge of transit domain (required by

Ma-LAD) and both destination IP and source IP at the edge of stub domain (required by Mi-LAD).

iv. The proposed D-LAD technique provides a considerable diagnostic power, is effective and can detect a wide range of attacks. Attacks can be captured even if they comprise just 0.09% of the total traffic. D-LAD is a hybrid of honeypots and anomaly detectors. Ma-LAD can detect concentrated high rate and highly distributed meek attacks. The attacks that are crafted to match statistics of normal traffic; cause graceful degradation of services and slip past Ma-LAD are detected at microscopic-level by Microscopic-Level Attack Detectors (Mi-LAD). The attacks that exploit the vulnerabilities of the system and other unpredictable attacks are detected in presence of honeypots. Therefore the proposed detection schemes have high scope.

v. The devised detection mechanism is accurate and adaptive to dynamic network environment. It can operate in different modes of defense among naïve, normal and best defense according to network load and user requirements. As the mode of defense is based on optimized thresholds that are updated with dynamically changing conditions of traffic in the Internet, it minimizes false positives and negatives. Presence of honeypots further suppresses the false negatives.

vi. Effective techniques for characterization of attacks at the macroscopic-level and microscopic-level have been proposed that are fast and accurate. The dual-level characterization techniques are based input derived from detection. The flows belonging to congestion inducing traffic which are a strong indication of attack traffic are identified early and accurately in the network. The suspicious attack flows are identified at microscopic-level near the victim where they are further analyzed in presence of honeypots. Hence, the characterization at dual-level facilitates efficient, accurate and fast identification of attacks as well as suspects.

vii. The proposed characterization techniques can distinguish between flash crowds and normal traffic. At macroscopic-level, the detection and hence characterization is triggered by voluminous and congestion inducing attacks. Hence, flash crowds which are a momentary surge of legitimate traffic are untouched. Microscopic-level characterization utilizes cumulative sum and change point detection techniques that use time of persistence of abnormal condition as the judgment function. Since flash crowds are momentary, they are distinguished from DDoS events. This reduces the collateral damage due to flash crowds.

viii. In the proposed mechanism, quick analysis and coarse response to attack flows is produced at macroscopic-level whereas more certain and refined response is for suspicious flows is triggered at microscopic-level.

ix. Rules suggested for responding to identified attack flows are concise, fast and accurate. Collateral damage caused in the presence of our proposed response rules is much less than the damage suffered by legitimate clients in the absence of response. Macroscopic-level characterization occurs early in network. It imposes "filter" rule for focused filtering that blocks substantial portion of DDoS attack and enables immediate response to DDoS without effecting legitimate traffic. Microscopic-level characterization approach drills down to investigate suspicious DDoS flows more closely and as a response, "redirect" rule diverts the identified suspicious attacks to honeypots. With the two fold mechanism, packets with higher probability of being valid and harmless are offered preferential service with the "allow" rule, while packets which have been marginally classified as invalid will still be allowed and directed to honeypots. They may later receive service if there is available bandwidth, so as to minimize collateral damage inflicted by false detection. More than 90% of the legitimate traffic is accepted irrespective of the modes of defense.

x. The proposed mitigation scheme is efficient and allocates limited resources where they are most needed to provide resilience to node-based attacks. It provides continued client services with guaranteed QoS even in attacked network and under dynamically changing network conditions. The judicious mixture of active servers and honeypots, with the ability of legitimate clients to track the active servers, and isolation of suspicious attack flows provides maintains stable network functionality even under attacked network. Since the algorithms for changing the number, location and duration of active servers and honeypots and those used by legitimate clients to track the locations of active servers are based on lightweight backward hash chains, they have low computational, memory and processing overheads.

xi. Our scheme provides a proactive approach to mitigation against the attack because the active server is isolated from attack traffic and bandwidth of the links with active server will not be exhausted by the attack traffic. Also, the location of active server changes before the attack builds up. Overhead and cost of the scheme is small in absence of attack at low and moderate client loads. At high client load, our scheme outperforms the normal case i.e. gives better results even in the absence of attacks.

xii. Our framework shows a favorable attack-load-independent behavior with respect to network performance parameters like average response time, mean time between failure and goodput, for attack load below a threshold. We evaluated framework in a comprehensive way, including both benefits and disadvantages (worst-case performance) for various defense mechanisms. The framework acknowledges situation dependency of defense modes, and is able to choose the most suitable defense from available defense modes against a specific network scenario and user requirements.

Table 7.1 Features of defense techniques in the proposed framework

Parameters	Proposed Scheme
Basis of Defense	Traffic Feature Distribution (D-LAD and Characterization) Dynamic Roaming (Mitigation)
Detection and Characterization Location	Intermediate Network (Ma-LAD); Victim Network (Mi-LAD and Honeypots)
Filtering and Response Location	Intermediate Network (At macroscopic-level for most congestion inducing attacks identified with confidence); Victim Network (At microscopic-level for stealth and crafted suspicious attacks, attacks exploiting vulnerabilities)
Attack Signatures	Change in traffic feature distribution (based on source and destination IP)
False Positive Rate	Minimized by optimizing thresholds and honeypots
False Negative Rate	Suppressed by optimizing thresholds and honeypots
Detection Accuracy	High
Detection Sensitivity	High
Scope of Detection	High
Effective Against Spoofing	Yes
Collateral Damage	No
Computational Requirement	Minimum (Lightweight Hash and Entropy Computations)
Memory Overheads	Minimum (for storing cumulative sums and FL during mitigation)
Communication Overheads	Overheads of secure communication for sending dynamic roaming update messages
Deployment Difficulty	Medium and localized

Table 7.1 shows the features of the various defense techniques in the proposed framework. Comparing the techniques proposed in this thesis with the key existing techniques given in Table 2.2 and 2.3 on the same set of parameters shows that our proposed framework has the potential to defend against DDoS and is more efficient and effective than most of the existing techniques.

The contribution of the thesis provides a range of defenses that can severely limit the damage caused by DDoS attacks. We highlight how the three lines of defense complement each other and are integrated into a resilient solution for DDoS attack problem. This is a significant step in providing robust Internet services that can be used for electronic commerce and on-line services. Our study gives a rich understanding of DDoS attack problem and opens the way for long term solutions.

7.2 Scope for future Work

This study opens up a number of avenues for future work. A number of research issues need to be addressed. Some of them are as follows:

i. Though the dual-level detection and characterization schemes proposed in Chapter 4 and Chapter 5 have improved the DDoS defense to a great extent, but at the same time it induces flow level state monitoring overheads. Sampling in time windows has reduced the state monitoring overheads, but size of sampling time window induces a bias with fast monitoring and computational requirements. Optimization of sampling time window on the basis of peak client hours and using data structures like bloom filters can minimize the computational and storage complexity. Computational cost can be further reduced by implementing optimizations that approximate the true frequency profile while reducing or eliminating floating point operations in the packet-handling code.

ii. The DDoS defense proposed in this thesis does not identify the upstream ingress edges of malicious flows. Packet marking techniques can used to identify the ingress edges further upstream. This can be used for implementing filtering further upstream.

iii. Dynamic roaming honeypot mitigation scheme proposed in Chapter 6 are not able to protect the dynamic roaming update messages to be exchanged from being intercepted by the hackers. Secure group communication or current security mechanisms like IPSec, PKI, CA and authentication using public and symmetric keys can be used to meet the requirements. The authentication, confidentiality and

integrity of sources of information are also an issue in the mitigation scheme proposed in Chapter 6. A layer of indirection between the server pool and legitimate clients in the form of a network overlay of access gateways or virtual overlay node can be utilized to meet the functionality of HC to decouple client authorization from service provision. Also physical roaming has been proposed and implemented. However, benefits of logical roaming cannot be ignored.

iv. There can be other criteria for dynamically changing number of servers and honeypots, for e.g. increasing the granularity in the decision by a higher resolution grading of loads into more categories, or classification to facilitate different types of service depending on temporal and spatial requirements. Also, keys are distributed to legitimate clients depending upon their trust level, with level of trust specific to an organization and according to certain policies. These policies are unique to network environment and user requirements and will differ for different networks.

v. The lightweight computational techniques proposed for detection and roaming scheme for mitigation can be further explored in wireless domains for jamming attacks etc.

vi. Simulation Experiments in ns-2 testbed have been used for validation, as a proof of concept and for evaluation of the proposed schemes, but deployment and investigations using real time test beds or real attack traces will be more useful.

Appendix A

Simulation Model

A.1 Topology

We simulate a network representative of the structure of the Internet. For the study in this thesis, we model the Internet as transit-stub network. We choose AT&T networking environment and generate its transit-stub model. The 156 node AT&T topology given in Figure A.1. is quite famous and often used for simulations [221].

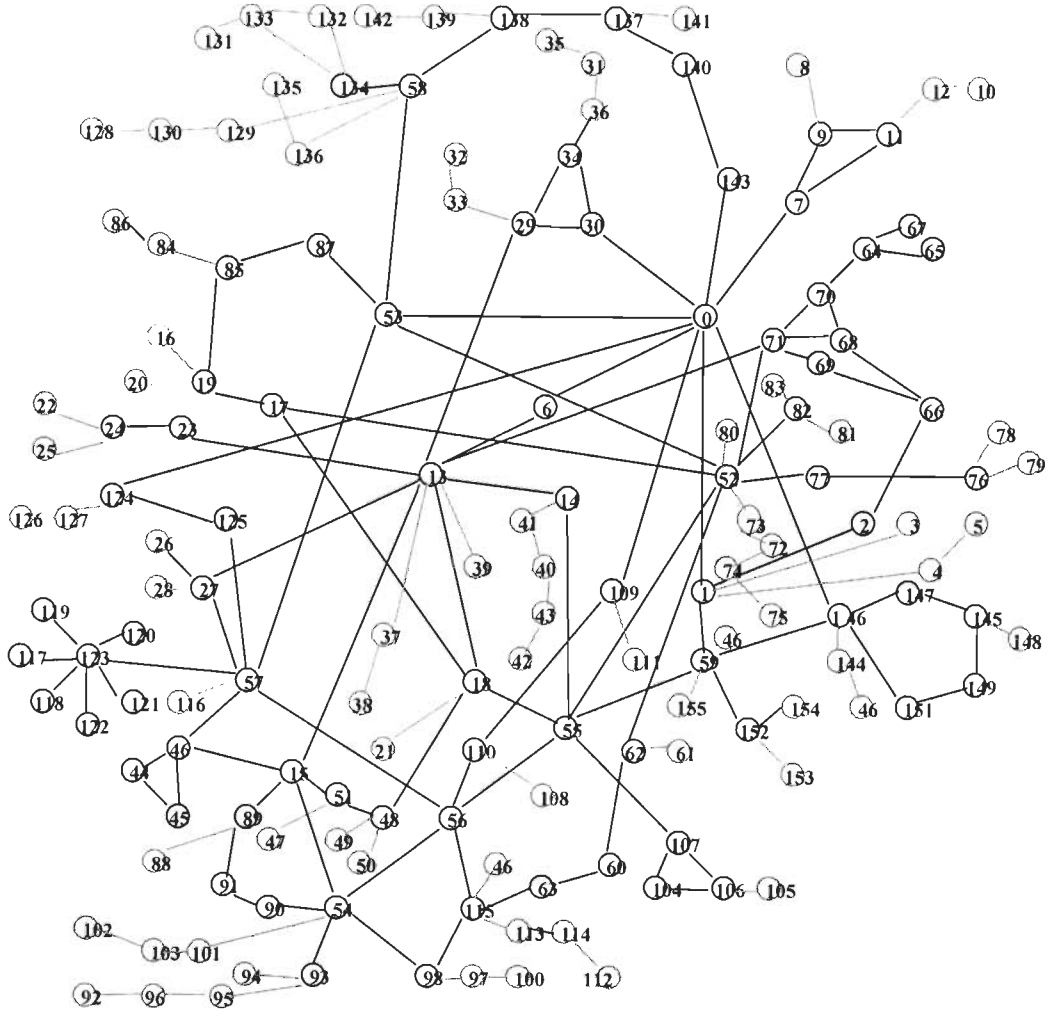


Figure A.1. Simulation topology of AT&T transit-stub network used in our experiments

It is composed of interconnected transit and stub domains. The transit domain comprises a set of highly connected backbone nodes. Backbone node is either connected to several stub domains or other transit domains. Stub domain usually has one or more router nodes, which have links to transit domains.

NS2 [202] (Network Simulator 2) topology for AT&T transit-stub model has been generated using Georgia Tech Internet Topology Generator (GT-ITM) [222-224] and extended nam [225] (network animator) with the parameters given in Table A.1. [221]. Edge Method is the method for generating edges and 3 signifies pure random method. Alpha is the random graph parameter which signifies the probability with which edge is added between pair of vertices. Our specific model has 3 core routers, 4 transit nodes with 3 stubs per transit and 4 stub nodes. The topology considered is similar to the one used traditionally to depict a typical client-server scenario in the Internet. We simulate 5 servers and 137 clients including legitimate clients as well as attackers. Transit domain edge routers are point of presence (POPs) of the Internet Service Providers (ISP) and stub domains are customer domains attached to POPs.

For visualization of the topology, there are 3 parameters to tune the automatic layout process : (i) Ca, the attractive force constant, (ii) Cr, the repulsive force constant and (iii) number of iterations i.e. how many times to run the auto layout procedure. A method to layout a 100 node or greater random transit-stub topology generated by Georgia Tech's ITM internet topology modeled in Network Animator (nam) is presented in [226]. First, Ca and Cr are set to 0.2, and 30 iterations are performed, then Cr is set to 1.0, Ca to about 0.01, and 10 iterations are performed, then Ca is ser to 0.5, Cr to 1.0, 6 iterations are performed. These parameters are summarized in Table A.2.

Table A.1. Topology generation parameters

Method	Average Stubs/transit	Extra t-s links	Extra s-s links
Transit Stub	3	12	12
Top Nodes	Edge Method	Alpha	
3	3	0.3	
Transit Nodes	Edge Method	Alpha	
4	3	0.5	
Stub Nodes	Edge Method	Alpha	
4	3	0.2	
Total Number of nodes = 3 * 4 * (1 + 4 * 3) = 156			

Table A.2. Parameters for layout and visualization of the simulated AT&T network

C_a	C_r	*Iterations*
.2	.2	30
.01	1	10
.5	1	6

A.2 Parameters of Simulation

Table A.3. provides the basic parameter set for simulation. The links are assigned as recommended in [227] with the following bandwidths and delays: 10 Mbps bandwidth and 1 ms delay for all inter-stub links (1st level links) and 1 Mbps bandwidth and 10 ms delay for intra-stub links (2nd level links). For the sake of fast simulations, we do not use realistic link capacities nor does realistic network load (although their relative values correspond to realistic cases).

Table A.3. Basic parameters for simulation

S. No.	*Parameter*	*Value*
1.	Number of legal sources	15-48
2.	Number of attackers	1-89
3.	Backbone link bandwidth	100 *Mbps*
4.	Backbone link delay	0 *seconds*
5.	Bottleneck link bandwidth	10 *Mbps*
6.	Bottleneck link delay	1 *ms*
7.	Access link bandwidth for legitimate clients	1 *Mbps*
8.	Access link delay for legitimate clients	10 *ms*
9.	Server link bandwidth	3 *Mbps*
10.	Server link delay	1 *ms*
11.	Mean attack rate	<3.0 *Mbps* (low rate) 3.0 – 6.5 *Mbps* (moderate rate) > 6.5 *Mbps* (high rate)
12.	Mean client load	<7.0 *Mbps* (low rate) 7.0-9.0 *Mbps* (moderate rate) >9.0 *Mbps* (high rate)

Table A.4. Traffic parameters

S.No	*Parameter*	*Value*
1.	Traffic arrival process	Poisson
2.	Connection startup time	Random 1-4 *seconds*

Table A.4. gives the traffic parameters. Previous studies [228] have shown that request inter-arrival times follow an exponential distribution. Thus, the request arrival process corresponds to a Poisson process, where users arrive independently of one another. We vary the number of clients and mean client request inter-arrival time to impose different workloads on the network. Number of attackers, attack rate and attack inter-arrival time are varied to simulate different attack loads.

A.3 Components of Simulation Model

Client Model: Two types of clients are considered: legitimate clients and attackers. The legitimate clients obey the TCP protocol, whereas it is not expected of the attackers to adhere to the TCP congestion avoidance protocols. The legitimate clients are modeled by FTP applications that run on TCP protocol. They obey the constraints imposed by the TCP protocol. Attackers use the attack model described next.

Attack Model: The attacks have been modeled in two ways. The first type of the attack deploys Constant Bit Rate (CBR) traffic generators as the attackers, while the second one utilizes FTP requests to launch an attack. The protocols used in transport layers and the attacked resources are different for both types of attacks. The CBR generator uses UDP, while the FTP deploys TCP. In addition, the UDP generator generates only one way traffic; therefore, the UDP attackers deplete only the bandwidth of the links from the source to the destination. In contrast, the TCP generator generates two-way traffic; thus, the TCP attackers target the bandwidth in both directions. The path that is for the acknowledgment, however, has a much lighter attacking traffic than the path used to carry the data.

In the first case, the attacks are modeled by CBR traffic on UDP. This choice is done as a UDP sender does not need to wait for any acknowledgement from the receiver before sending out further outstanding packets. This property is apt to model an attacker as an attacker would normally send out large volume or bursts of packets continuously with the aim of flooding the links leading to the server under attack. It is also used to model attacks

that are crafted to match statistics of normal traffic by variation in the number of attackers and attack rate.

In the second case, FTP requests are utilized to launch an attack. The reason behind this choice is to model attacks which are crafted to behave and appear similar to legitimate clients. Hence these attackers have been modeled as requesting files over TCP in the same manner as legitimate clients.

Service Model: The service provided by the FTP server is a generic TCP-based service. The legitimate clients connect to the server with the aim of achieving file downloads, whereas the attackers aim at clogging the resources at bottleneck link and servers in order to make the service unavailable to the legitimate clients. The servers are modeled by a simple TCPSink which send out ACK packets for packets they receive.

Appendix B Simulation Flowcharts

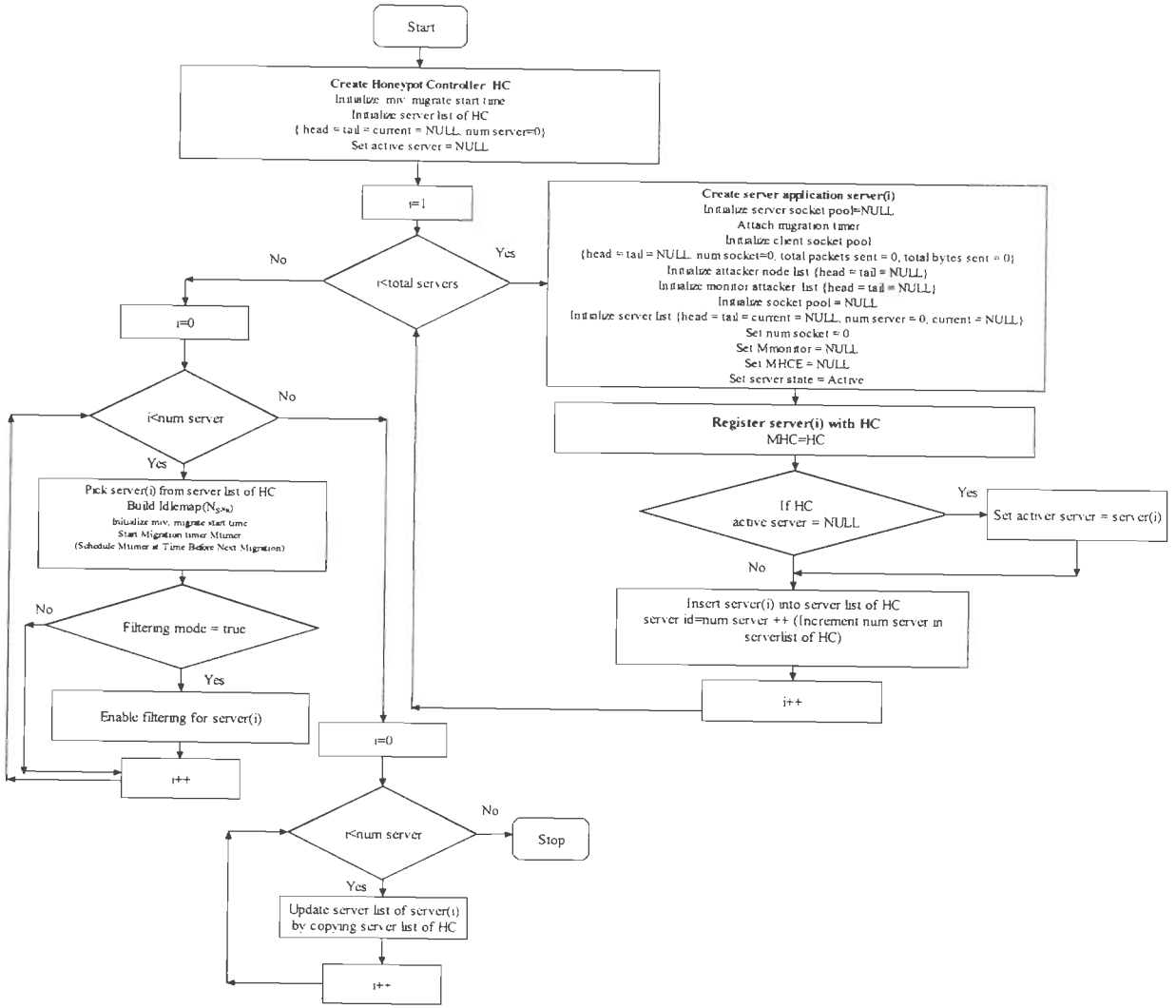


Figure B.1. Initialize servers and Honeypot Controller (HC)

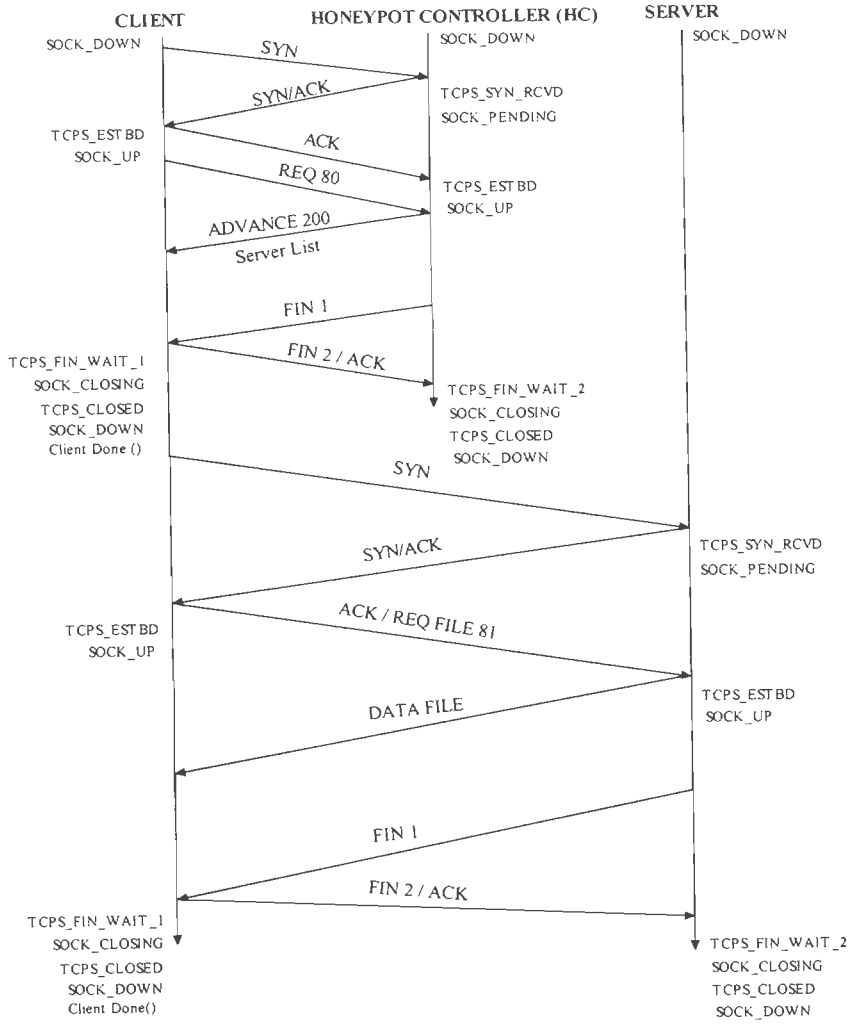


Figure B.2. Messages exchanged and socket states at different times between client, HC and server for a file transfer

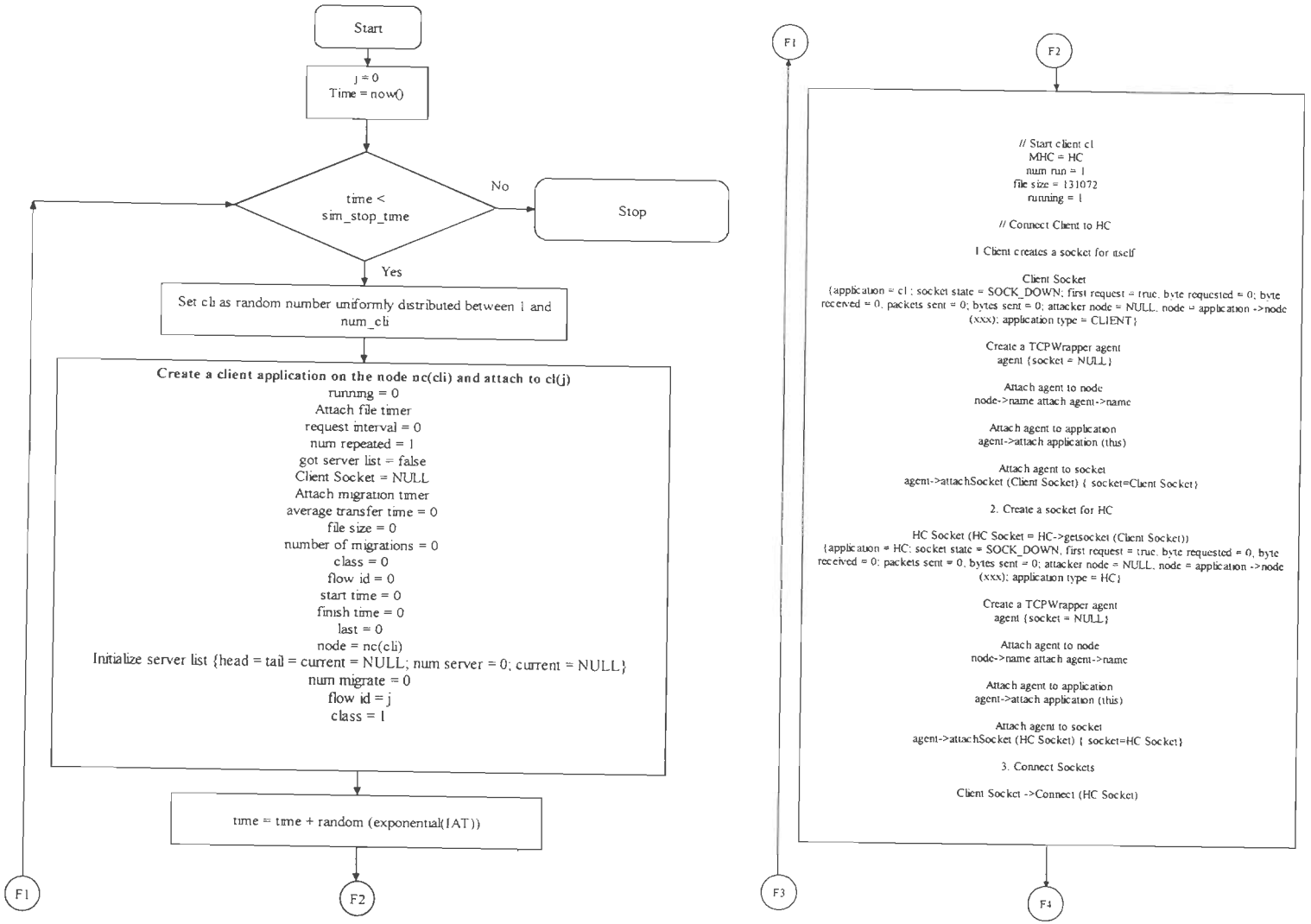


Figure B.3(a). Start client application and obtain server list

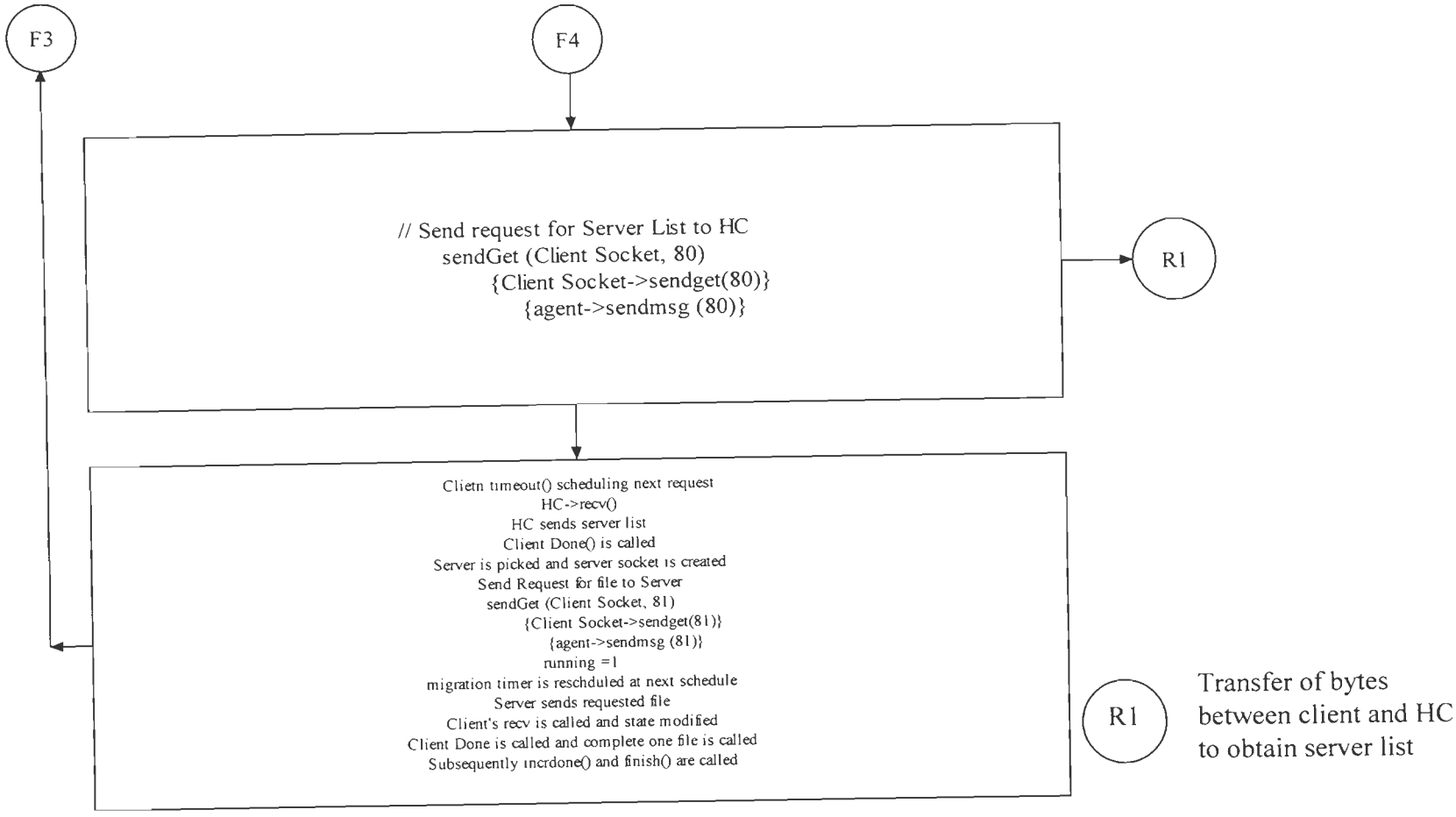


Figure B.3(b). Start client application and obtain server list (contd.)

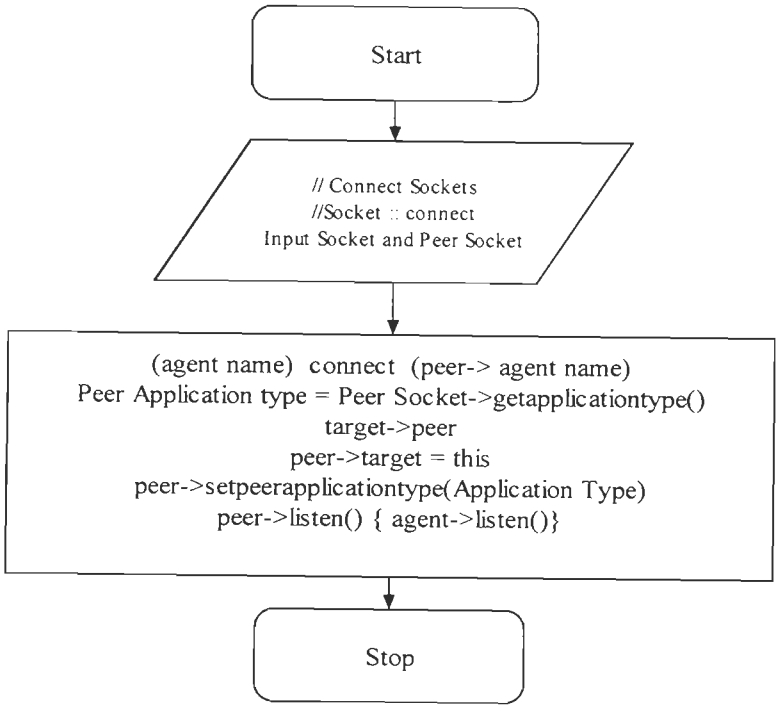


Figure B.4. Connect sockets

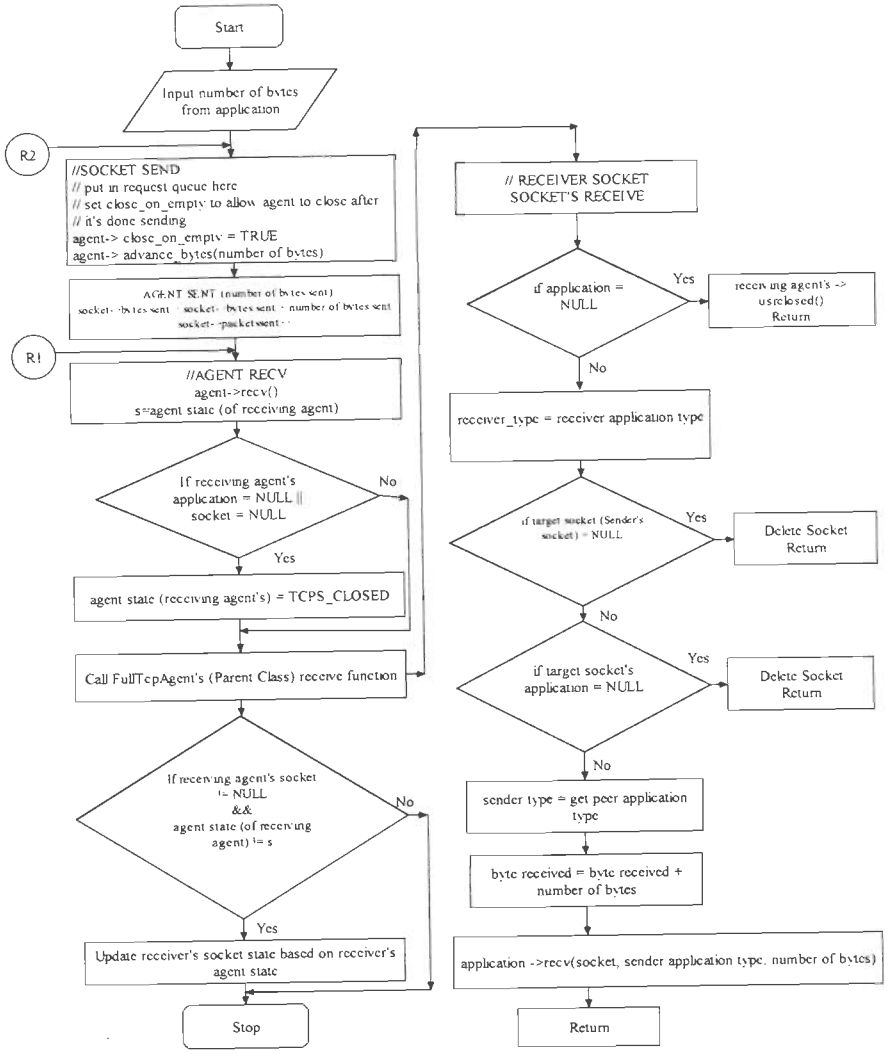


Figure B.5. Transfer of bytes between sender and receiver application for send() and recv()

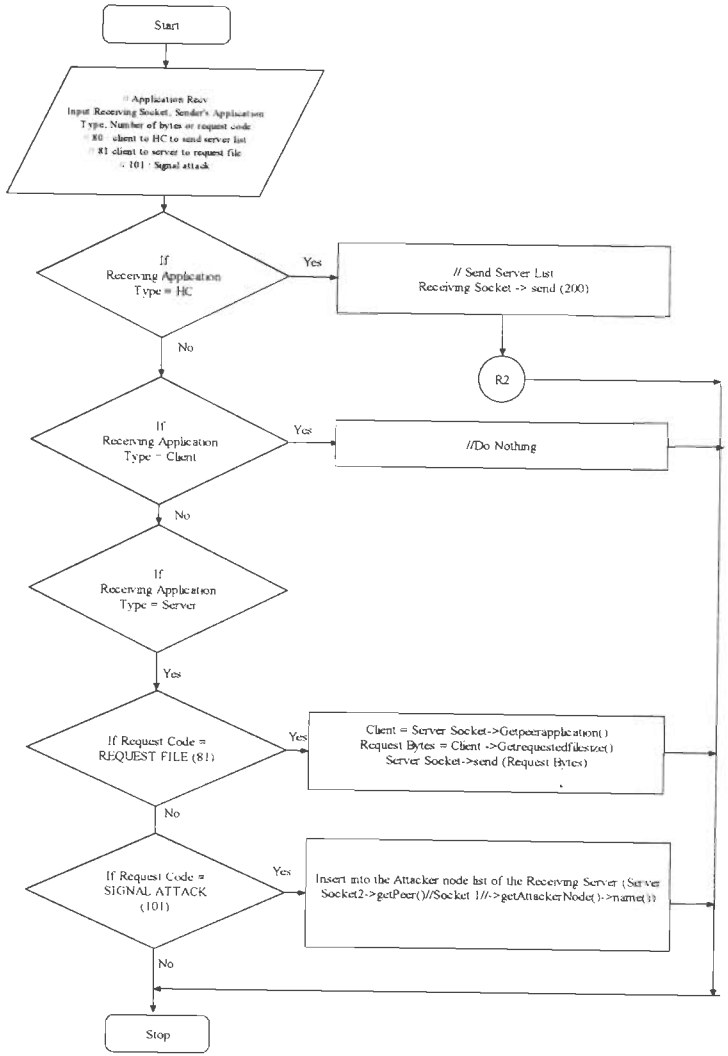


Figure B.6. Receive function of an application

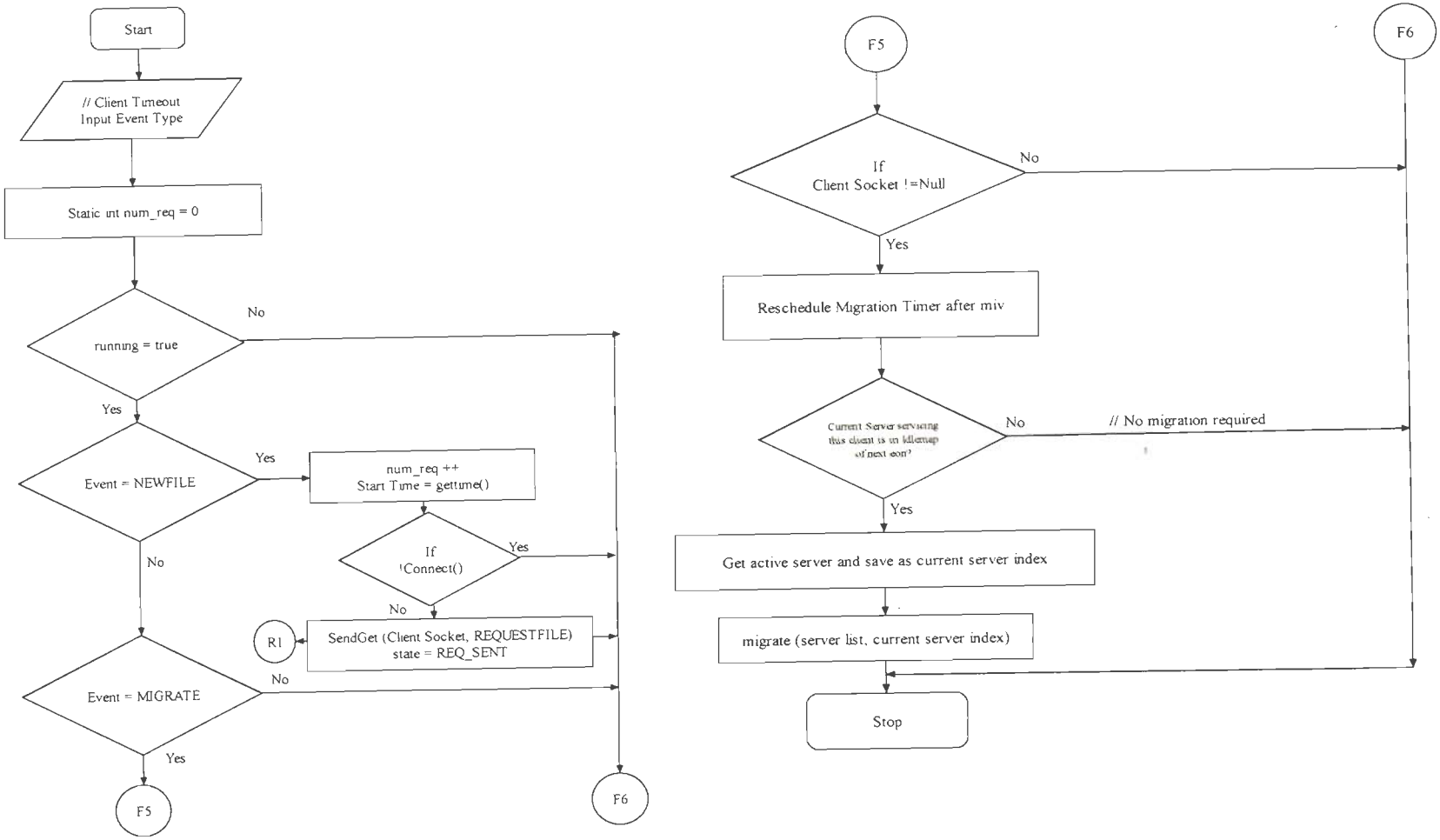


Figure B.7. Client timeout

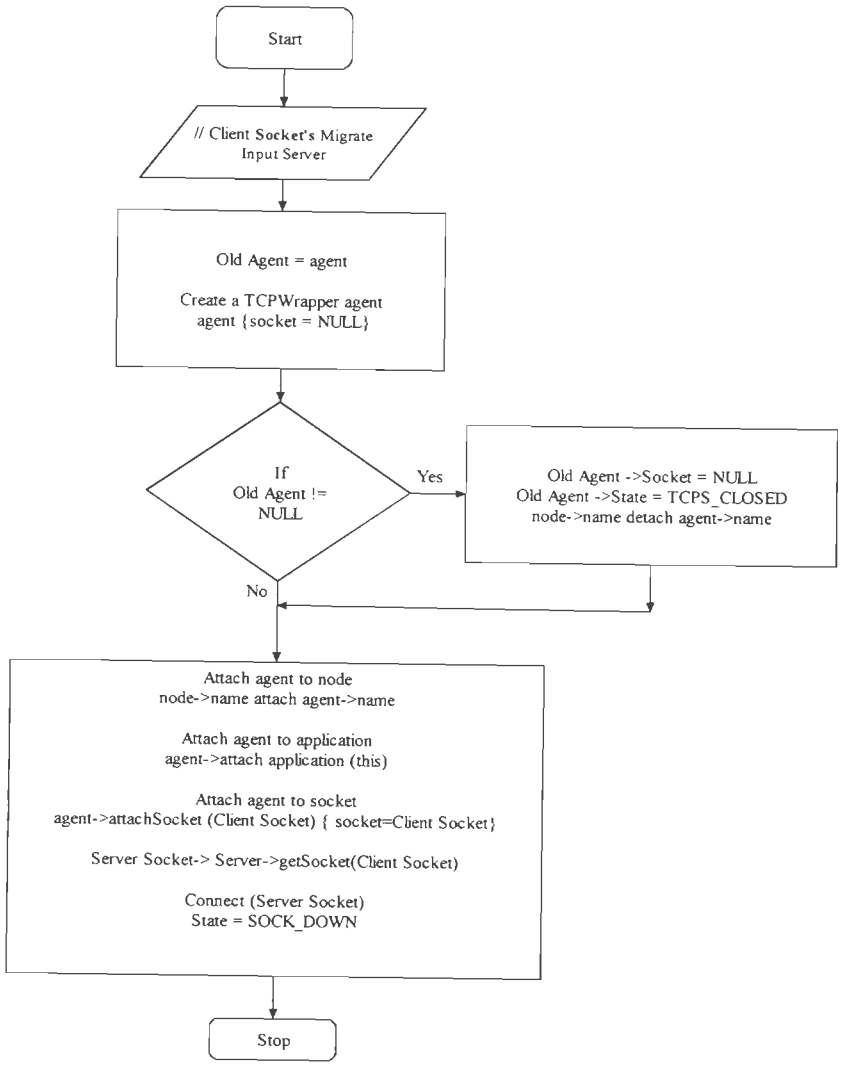
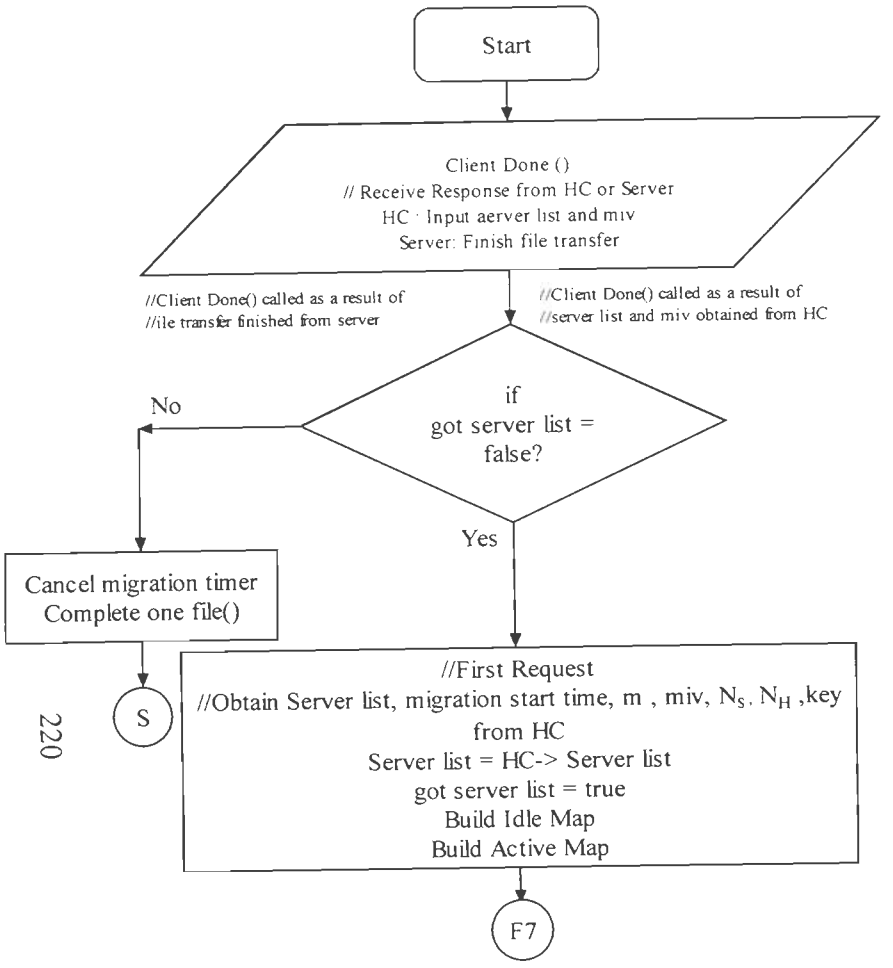


Figure B.8. Client migrate



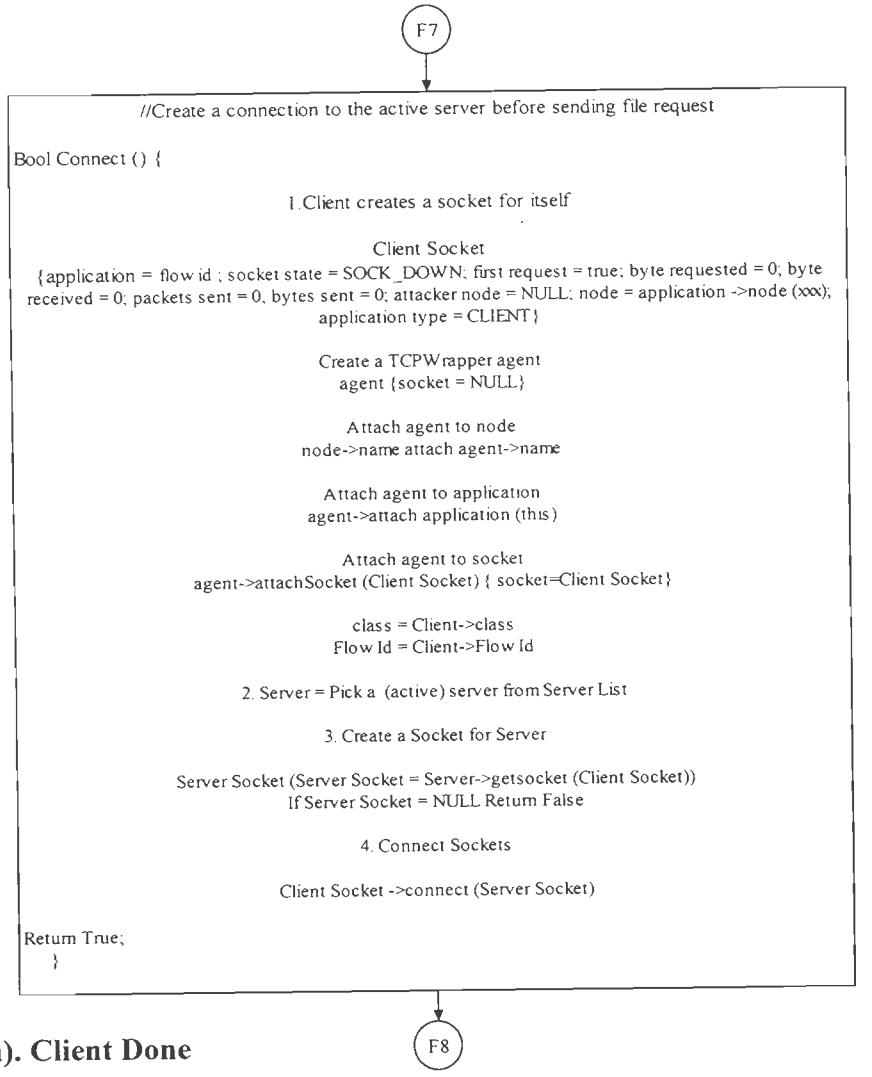


Figure B.9 (a). Client Done

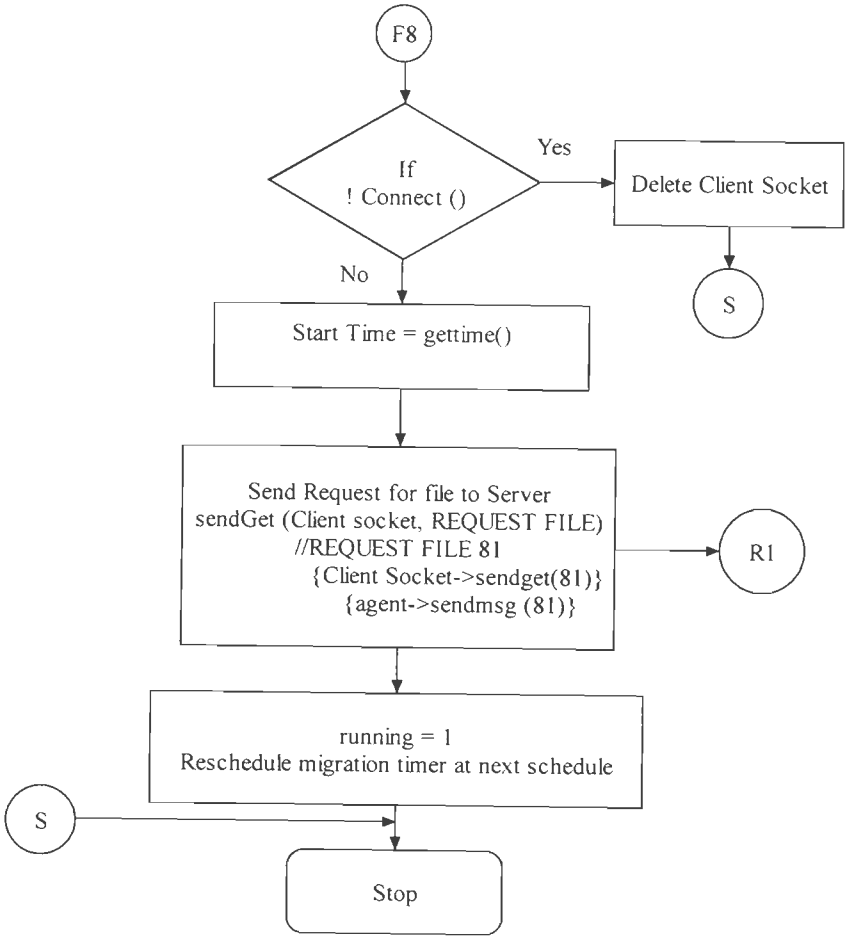


Figure B.9 (b). Client Done (contd.)

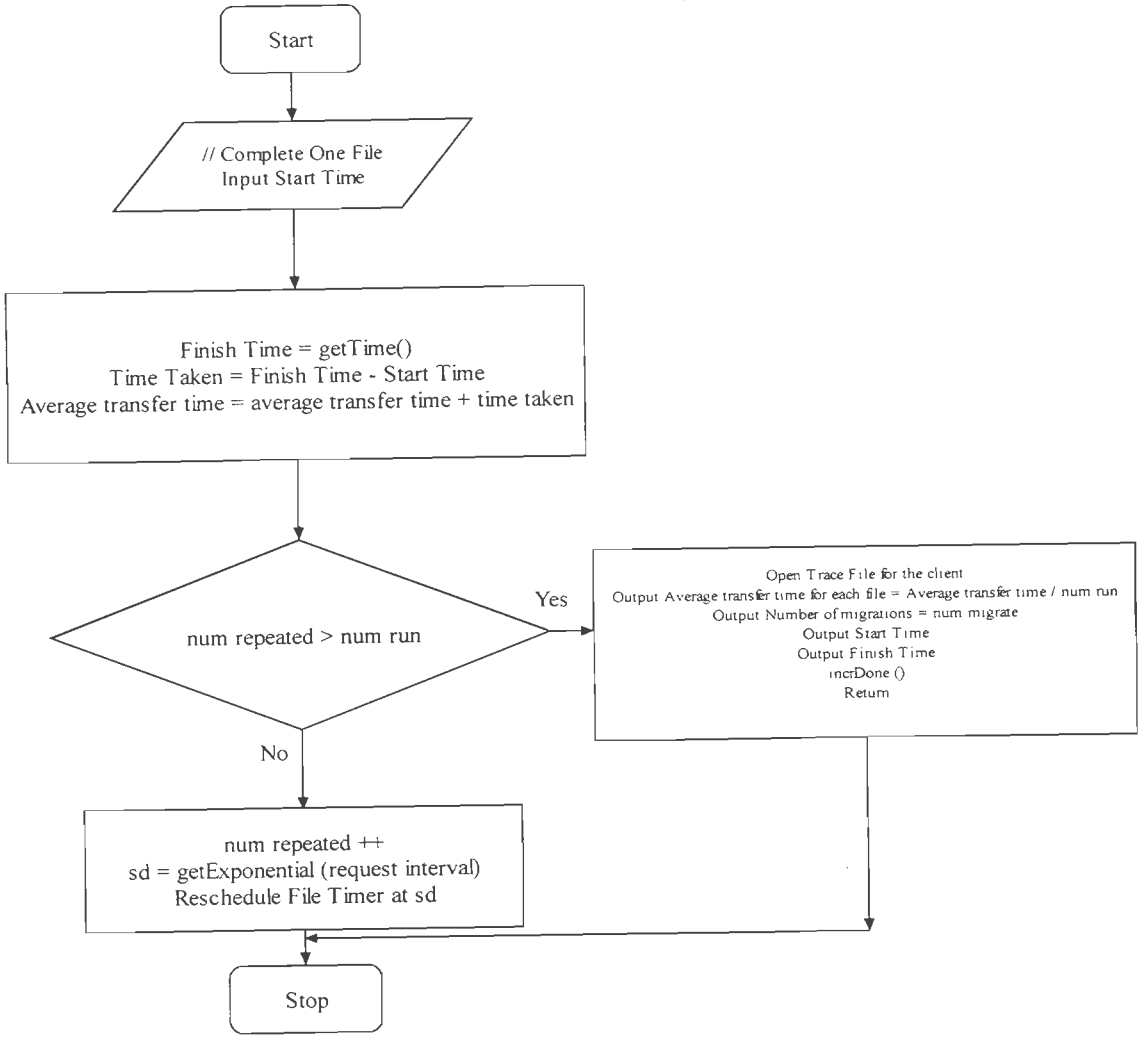


Figure B.10. Complete one file

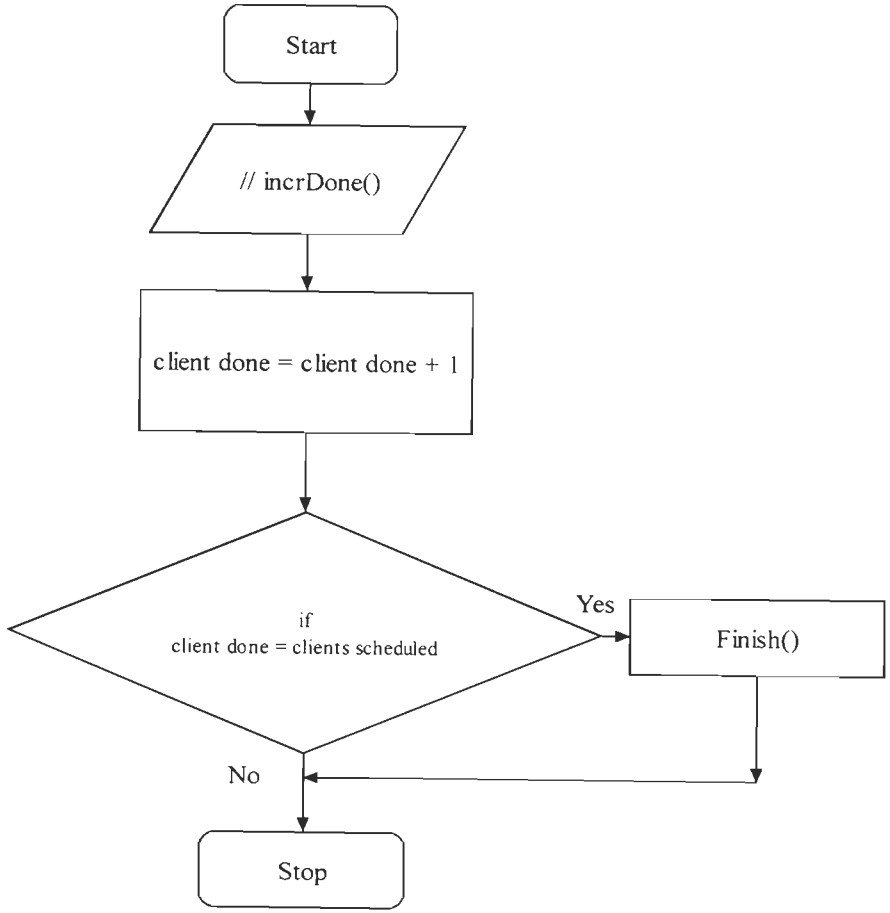


Figure B.11. Increment the number of clients done (file transfer complete)

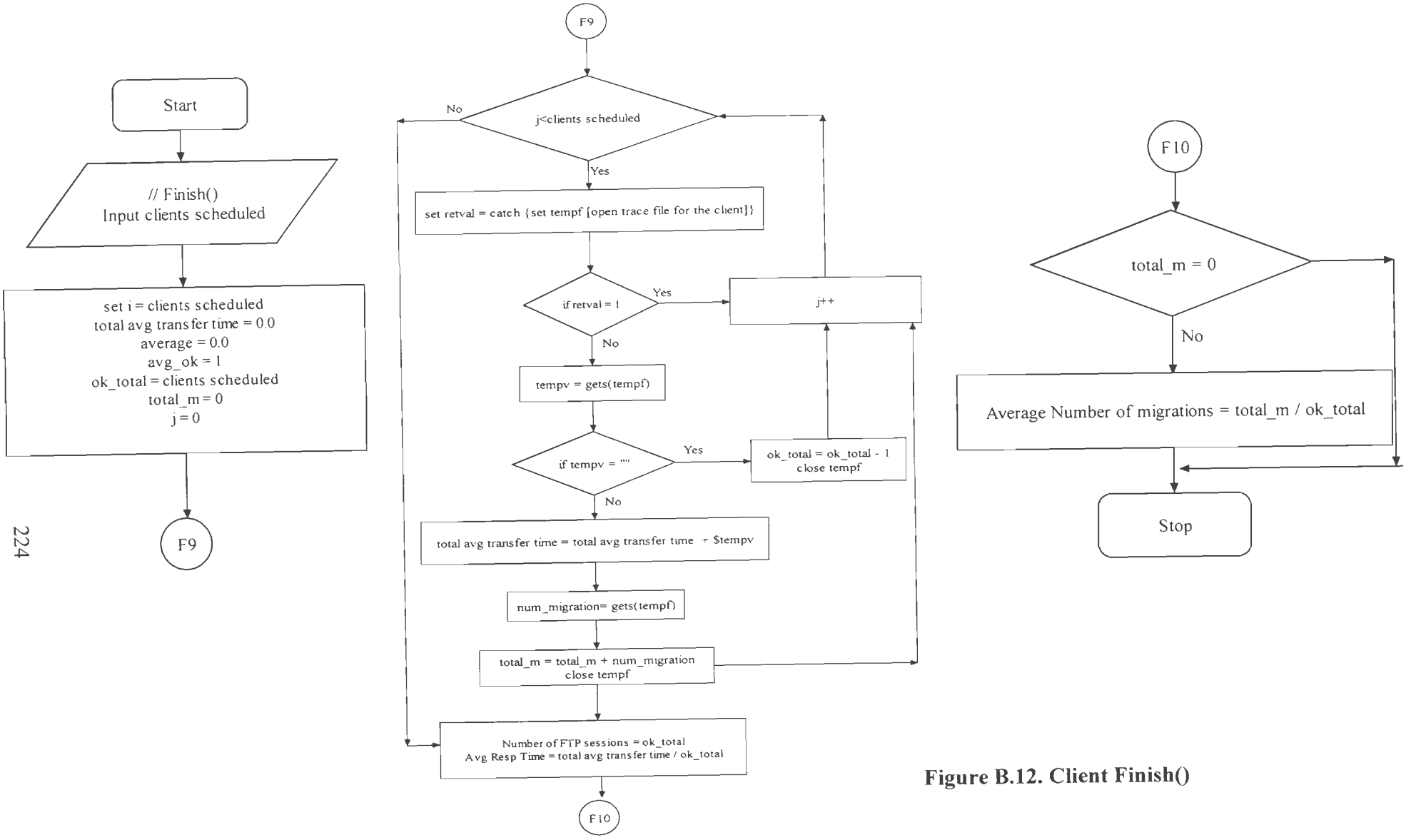


Figure B.12. Client Finish()

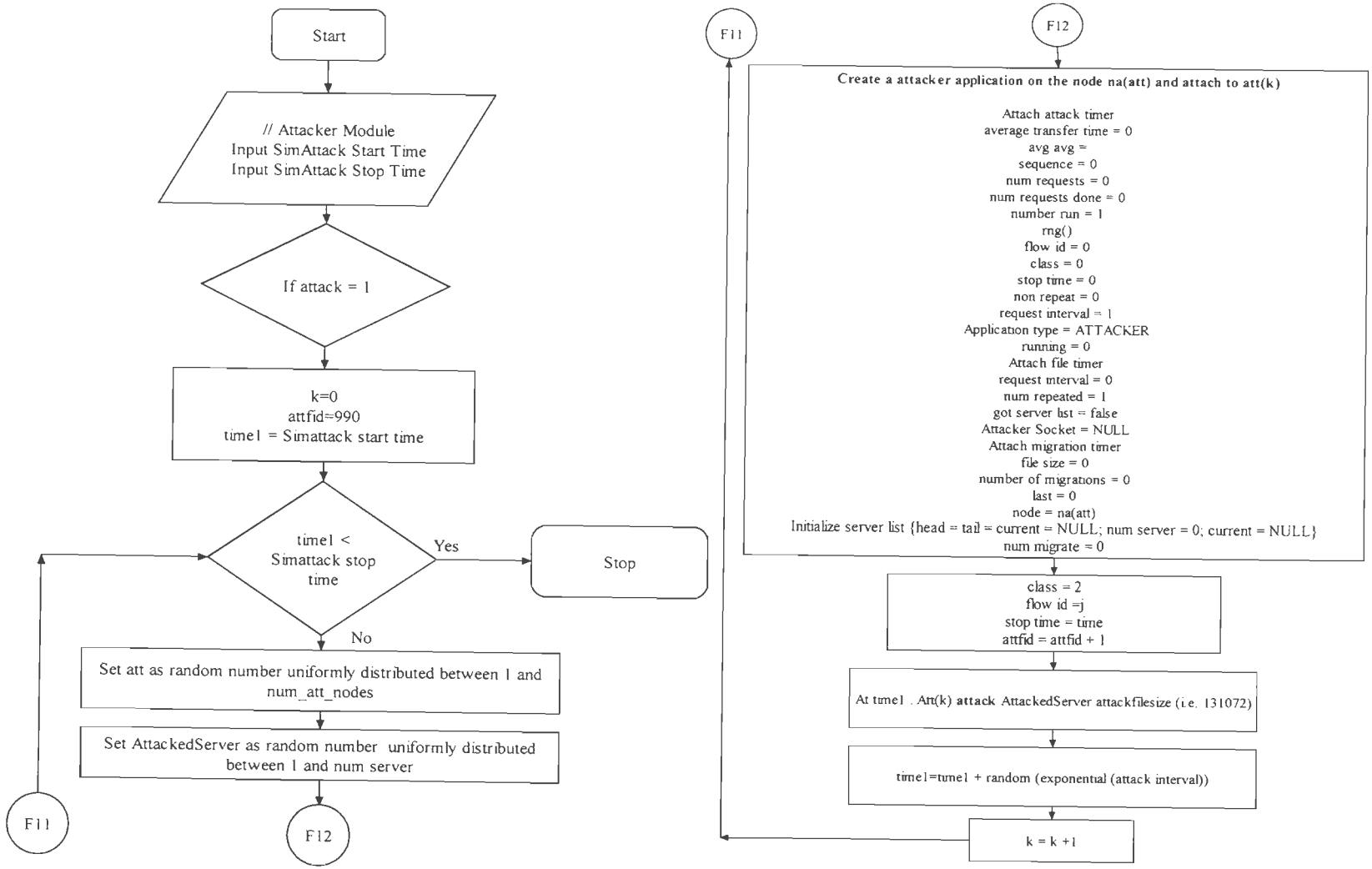


Figure B.13. Initialize attacker

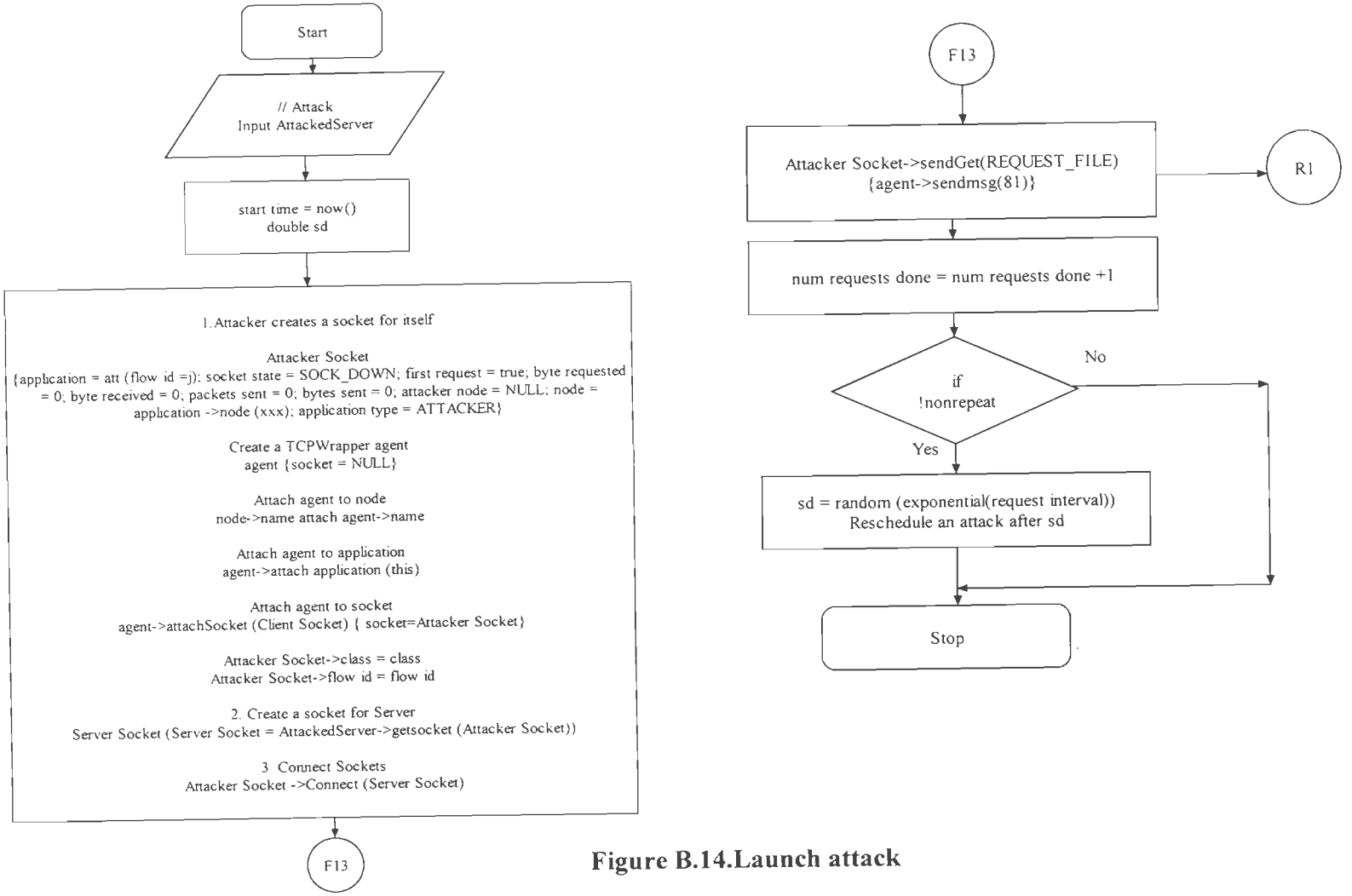


Figure B.14.Launch attack

Figure B.15. Attacker timeout

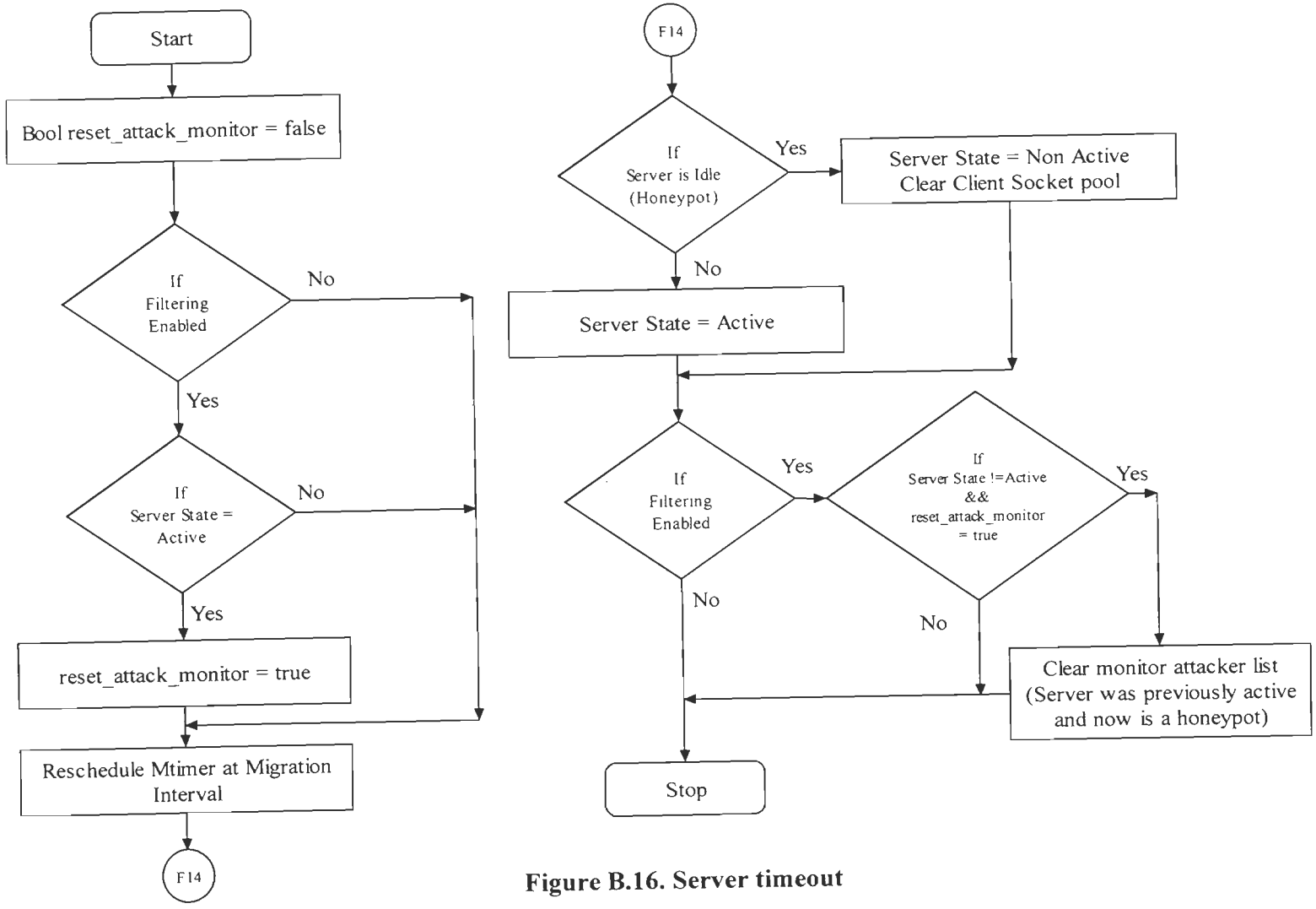


Figure B.16. Server timeout

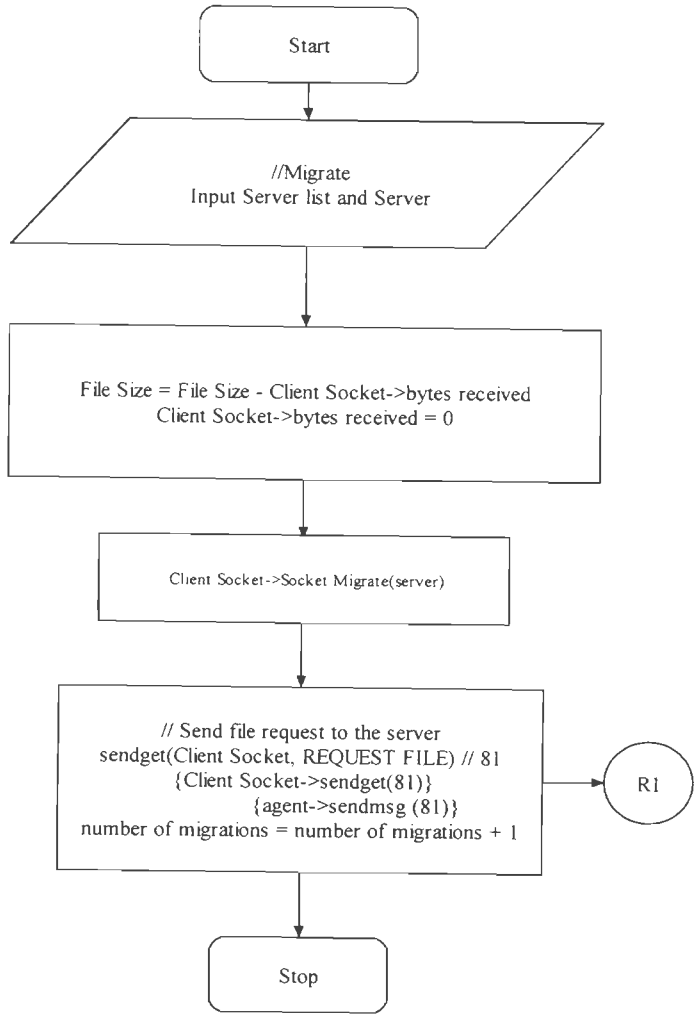


Figure B.17. Server migrate

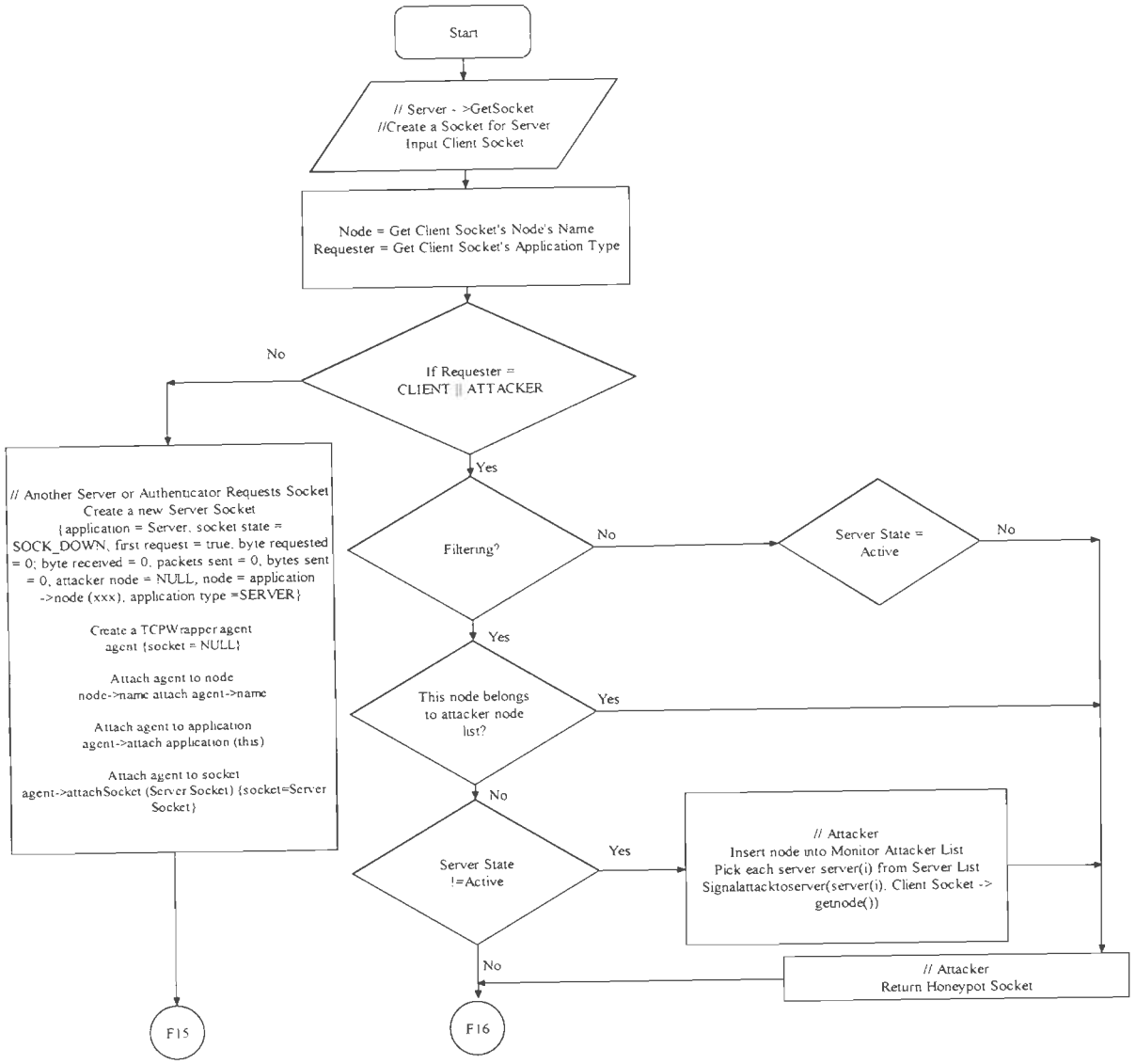


Figure B.18 (a). Create a server socket

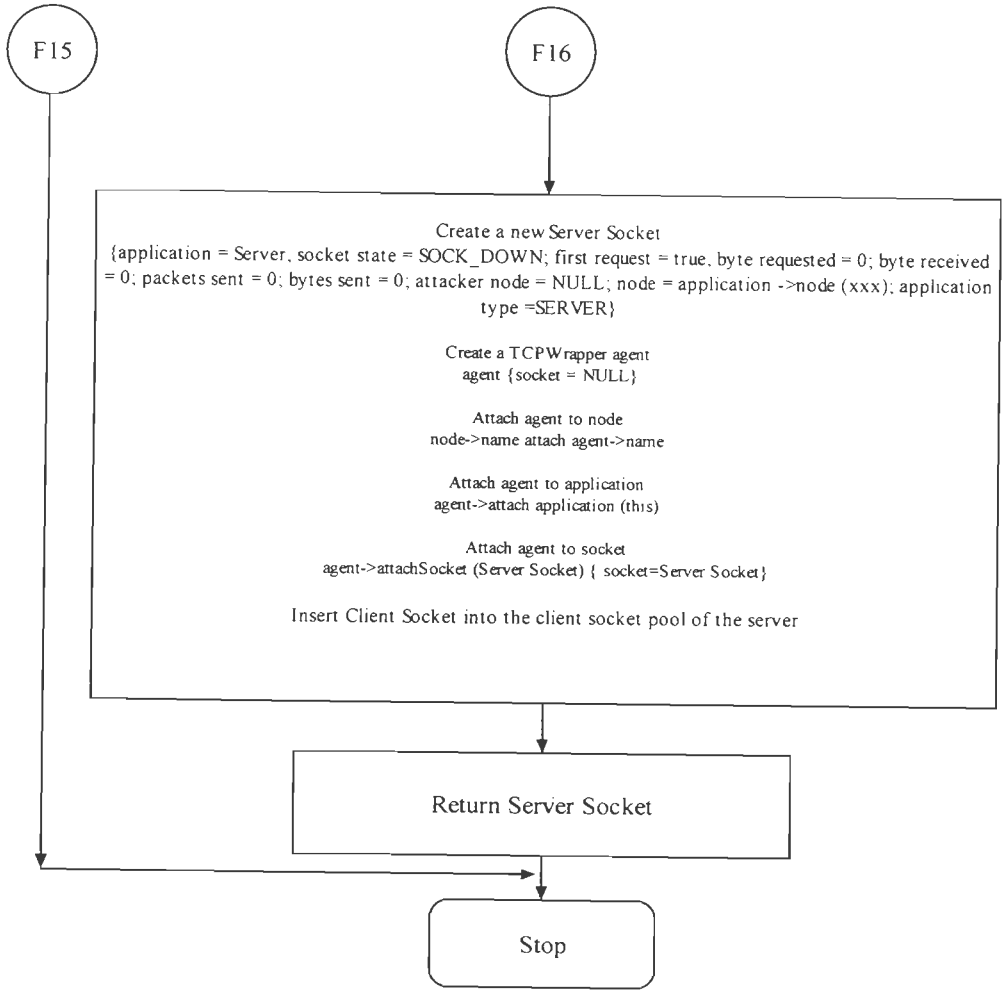


Figure B.18 (b). Create a server socket (contd.)

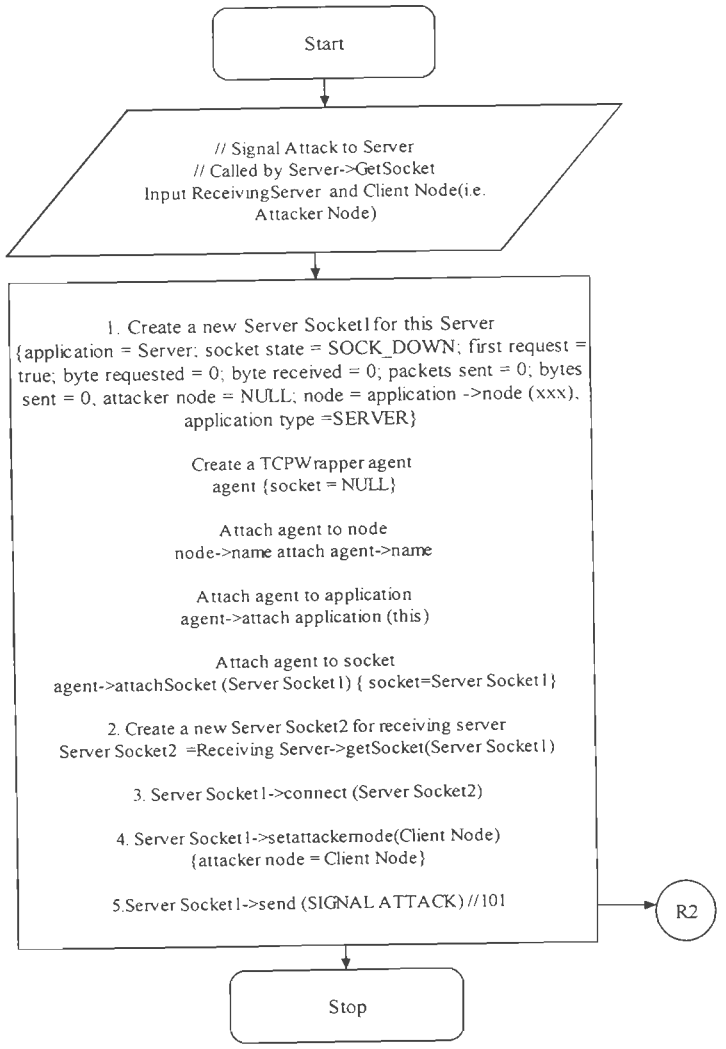


Figure B.19. Signal attack to server (in case of filtering)

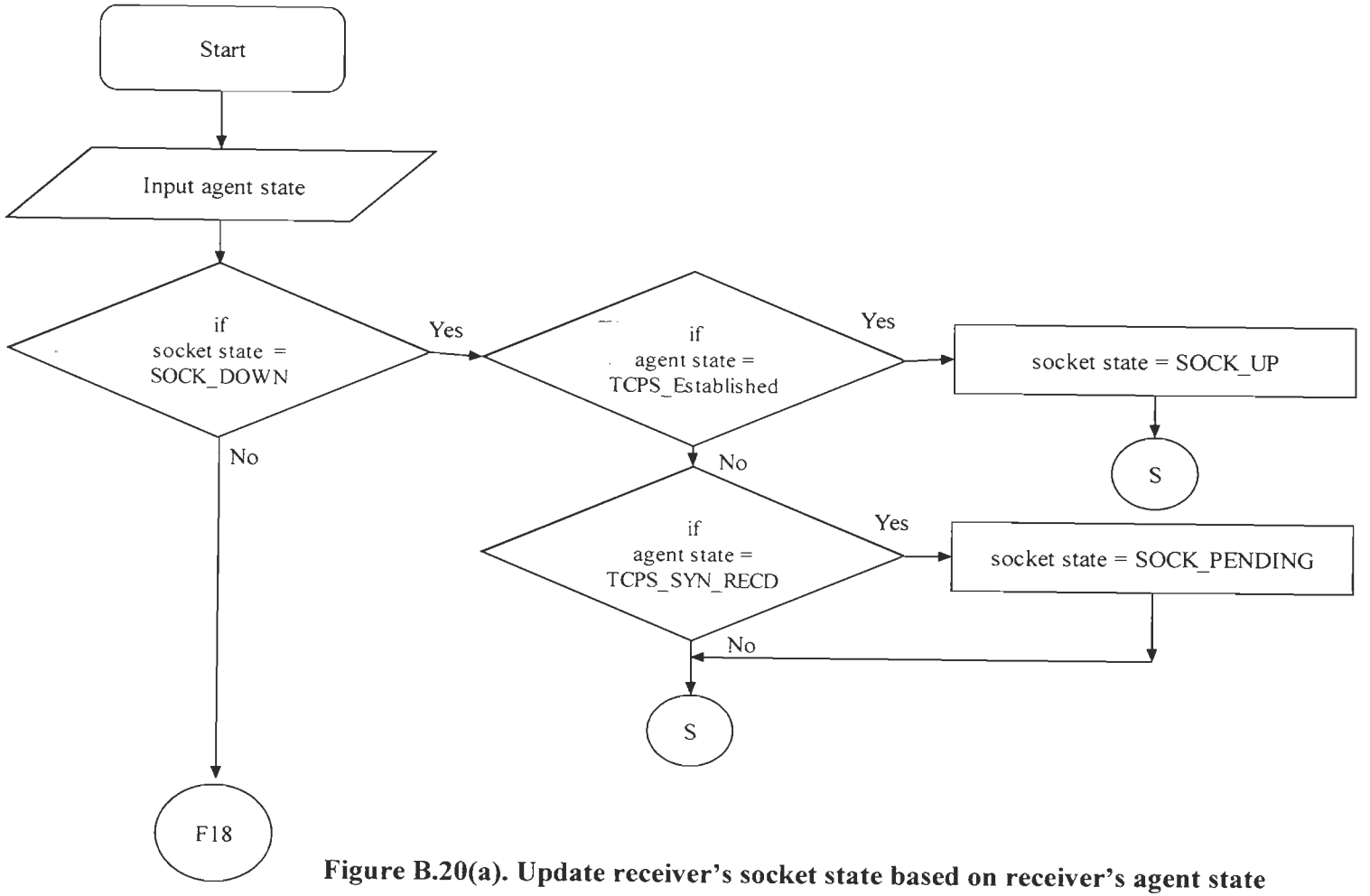


Figure B.20(a). Update receiver's socket state based on receiver's agent state

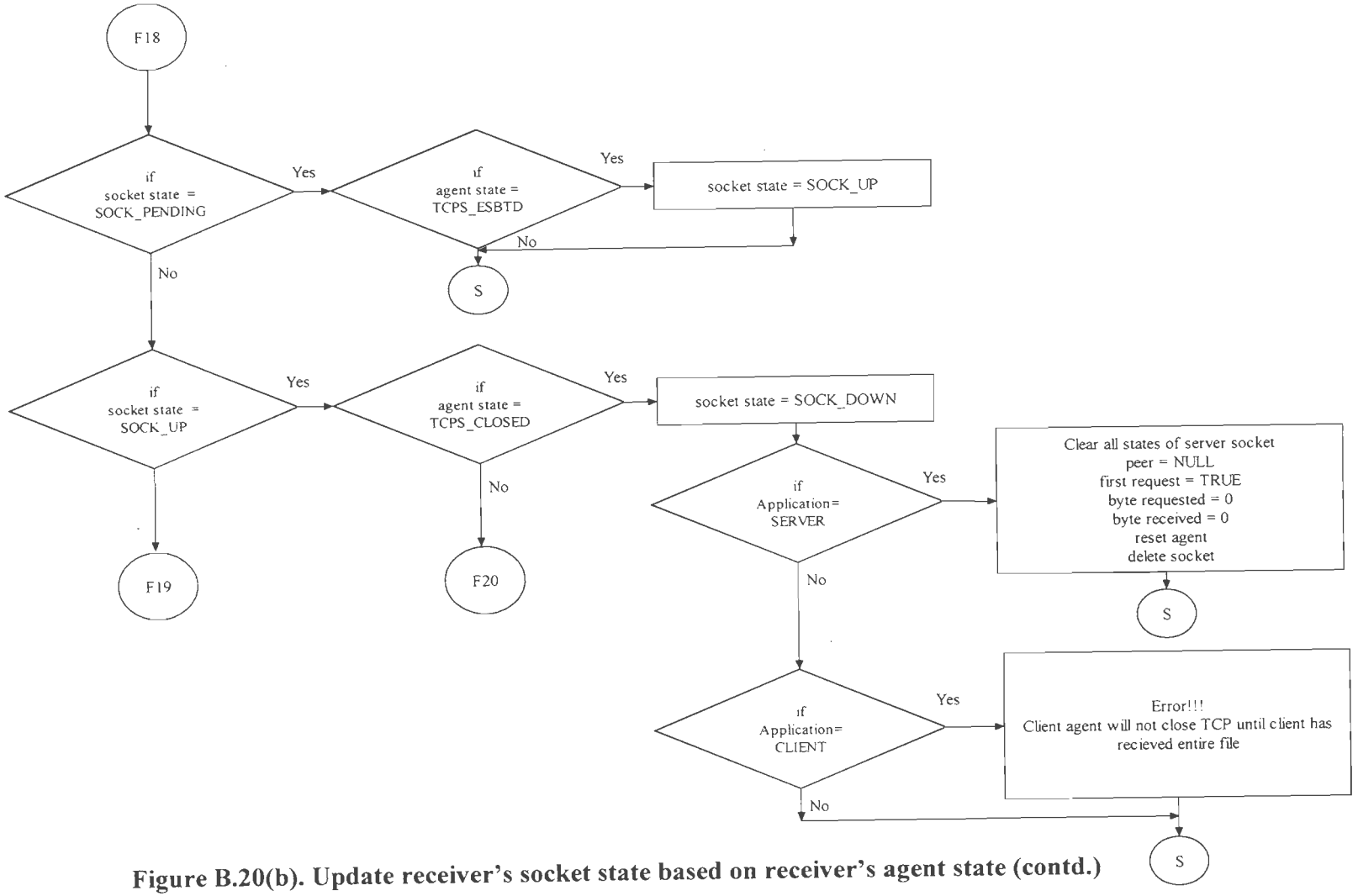


Figure B.20(b). Update receiver's socket state based on receiver's agent state (contd.)

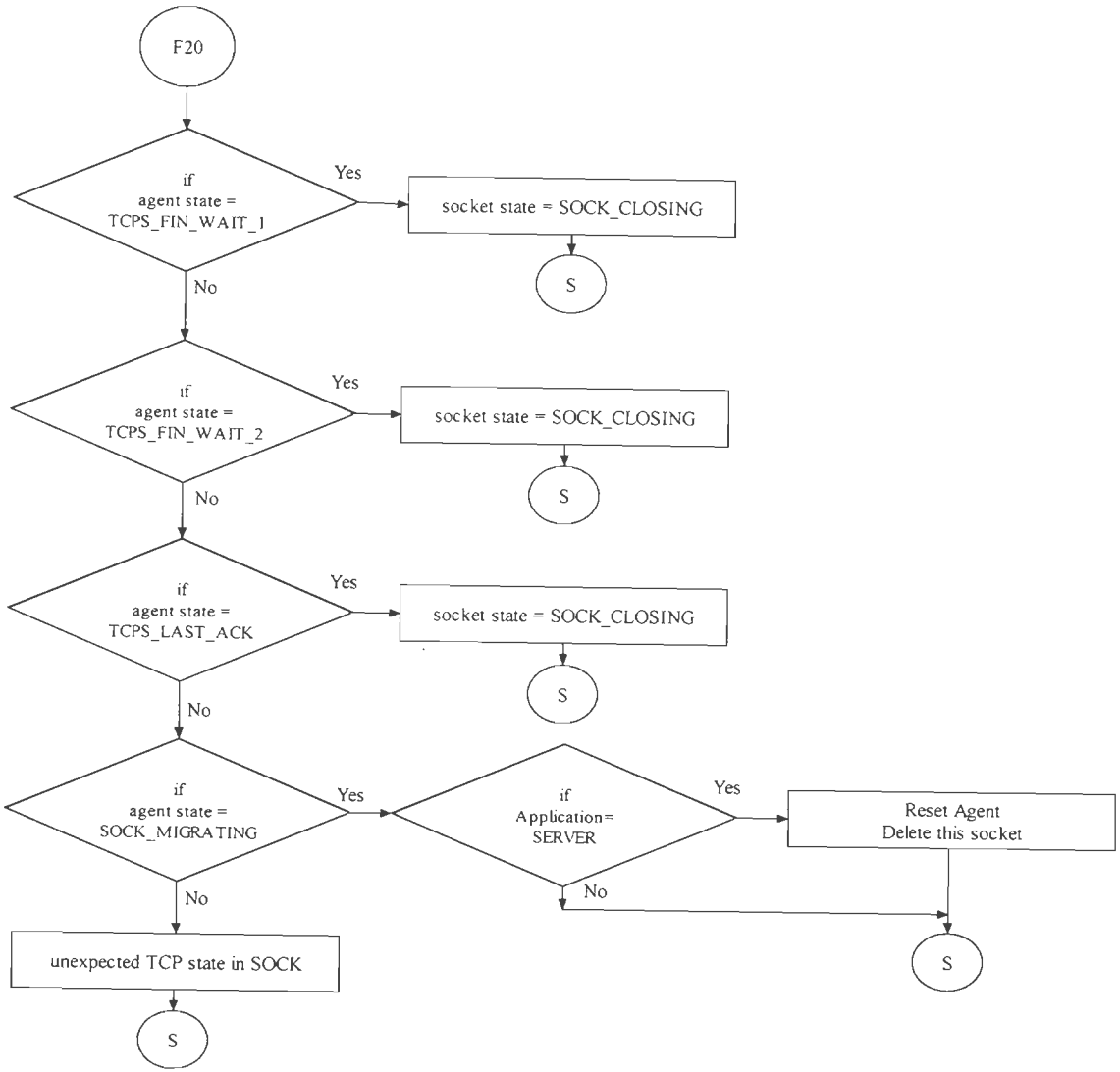


Figure B.20(c). Update receiver's socket state based on receiver's agent state (contd.)

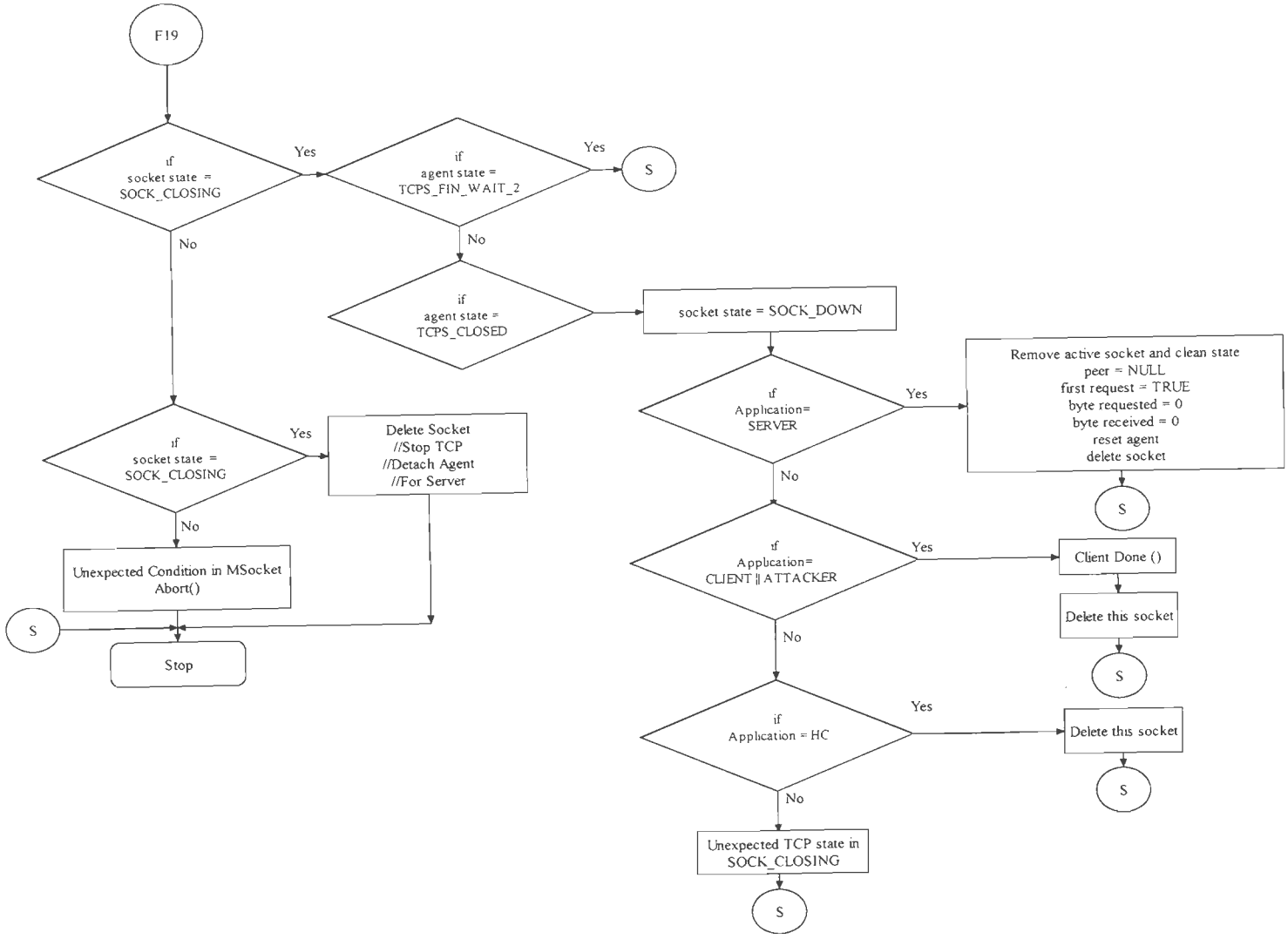


Figure B.20(d). Update receiver's socket state based on receiver's agent state (contd.)

References

[1] H. F. Lipson, "Tracking and tracing cyber-attacks: Technical challenges and global policy issues," Special Report CMU/SEI-2002-SR-009. CERT Coordination Center, 2002.

[2] A. Bharati, V. Chaitanya, and R. Sangal, "Information Revolution," in *Information Revolution and Indian Languages IRIL99, Arxiv preprint cs/0308017*, Internet:http://arxiv.org/abs/cs/0308017, 1999.

[3] CERT/CC Statistics. http://www.cert.org/stats/cert_stats.html.

[4] L. Garber, "Denial-of-service attacks rip the Internet," *Computer,* vol. 33, pp. 12-17, 2000.

[5] CSI/FBI Computer Crime and Security Survey. http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2004.pdf.

[6] CERT/CC Denial of Service. http://www.cert.org/tech_tips/denial_of_service.html.

[7] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic, "Distributed denial of service attacks," in *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2275-2280, 2000.

[8] V. D. Gligor, "A note on the denial-of-service problem," in *Proc. IEEE Symposium on Security and Privacy*, p. 139, 1983.

[9] R. T. Morris, "A weakness in the 4.2 BSD Unix TCP/IP software," in *Computing science technical report, AT&T Bell Labs*. vol. 117, 1985.

[10] CERT. CA-1996-21: TCP SYN flooding and IP spoofing attacks. http://www.cert.org/advisories/CA-1996-21.html. Sept. 1996.

[11] ISS, Inc.: RealSecure intrusion detection system. http://www.iss.net.

[12] J. Glave, "WebCom Security Software Failed in Server Attack," *Wired News*, Internet: http://www.wired.com/news/technology/0,1282,1052,00.html, Dec 1996.

[13] J. Mirkovic, "D-WARD: source-end defense against distributed denial-of-service attacks," Ph.D. Thesis, University of California Los Angeles, 2003.

[14] K. Coale, "Romanian Cracker Takes Down the Undernet," *Wired News*, Internet: http://www.wired.com/news/technology/0,1282,1052,00.html, Dec 1996.

[15] CERT.CA-1996-26: Denial-of-service attack via ping. http://www.cert.org/advisories/CA-1996-26.html, 1996.

[16] J. Glave, "Smurfing Cripples ISPs," *Wired News*,

Internet: http://www.wired.com/news/technology/0,1282,9506,00.html, Dec 1996.

[17] CERT. CA-1998-01: Smurf IP Denial-of-Service Attacks. http://www.cert.org/advisories/CA-1998-01.html, 1998.

[18] J. Glave, "Pentagon Hacker Exposed by Justice Department," *Wired News*, Internet: http://www.wired.com/news/technology/0,1282,11030,00.html, 1998.

[19] "'Mafiaboy' hacker jailed," Internet: http://www.news.bbc.co.uk/1/hi/sci/tech/1541252.stm.

[20] K. McCarthy, "Wanna know how BT.com was hacked?," Internet:http://www.theregister.co.uk/2000/07/25/wanna_know_how_bt_com/, Sept 2001.

[21] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *ACM SIGCOMM Computer Communication Review,* vol. 31, pp. 38-47, 2001.

[22] S. Gibson, "The strange tale of the denial of service attacks against grc. com," *Gibson Research Corporation,* vol. 31, pp. 1-28, 2001.

[23] R. Stenger, "New 'Code Red' worm entices Web hijackers," Internet: http://edition.cnn.com/2001/TECH/internet/08/06/code.red.two/ , 2001.

[24] "Teenager cleared of hacking," Internet: news.bbc.co.uk/1/hi/england/hampshire/dorset/3197446.stm, 2001.

[25] B. McWilliams, "Yaha Worm Takes Out Pakistan Government's Site," http://www.securityfocus.com/news/501, 2002.

[26] C. D. Marsan, "DDoS attack highlights Net problems," Internet: http://www.networkworld.com/news/2002/1028ddos.html, 2002.

[27] B. Chaffin, "Spam King Inundated By Junk Mail, Fails To See The Irony," Internet: http://www.macobserver.com/article/2002/12/06.11.shtml, 2002.

[28] S. Byers, A. D. Rubin, and D. Kormann, "Defending against an Internet-based attack on the physical world," *ACM Transactions on Internet Technology,* vol. 4, pp. 239-254, 2004.

[29] CERT. CA-2003-04. MS SQL Server Worm. http://www.cert.org/advisories/CA-2003-04.html, 2003.

[30] "South Korean markets hit by net worm," Internet: http://news.bbc.co.uk/2/hi/business/2698385.stm, 2003.

[31] K. Poulsen, "Slammer worm crashed Ohio nuke plant network," Internet: http://www.securityfocus.com/news/6767, 2003.

[32] "Hackers cripple al-Jazeera sites,"

Internet: http://news.bbc.co.uk/2/hi/technology/2893993.stm, 2003.

[33] "Akamai Provides Insight into Internet Denial of Service Attack," Internet: http://www.akamai.com/html/about/press/releases/2004/press_061604.html, 2004.

[34] K. Poulsen, "FBI busts alleged DDoS Mafia," Internet: http://www.securityfocus.com/news/9411, 2004.

[35] "Blackmailers target $1m website," Internet: http://news.bbc.co.uk/2/hi/technology/4621158.stm, 2006.

[36] P. Galli, "DoS Attack Brings Down Sun Grid Demo," Internet:http://www.eweek.com/c/a/Database/DoS-Attack-Brings-Down-Sun-Grid-Demo/, 2006.

[37] "DNS DDoS Attack Takes Down China Internet," Internet:http://www.darknet.org.uk/2009/05/dns-ddos-attack-takes-down-china-internet/, 2009.

[38] U. Javaid, T. Rasheed, S. Khan, and A. Sahibzada, "A Pragmatic Analysis of Distributed Denial of Service (DDoS) Attacks: Countermeasures and Mitigation," in *Proc. IEEE International Conference on Emerging Technologies*, pp. 1-6, 2006.

[39] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," *ACM SIGCOMM Computer Communication Review,* vol. 30, pp. 295-306, 2000.

[40] R. S. Sandhu and P. Samarati, "Access control: principle and practice," *IEEE Communications Magazine,* vol. 32, pp. 40-48, 1994.

[41] W. R. Cheswick, S. M. Bellovin, and A. D. Rubin, *Firewalls and Internet security: repelling the wily hacker*: Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[42] R. Oppliger, "Internet security: firewalls and beyond," *Communications of the ACM,* vol. 40, pp. 92-102, 1997.

[43] P. E. Proctor, *Practical intrusion detection handbook*: Prentice Hall PTR Upper Saddle River, NJ, USA, 2000.

[44] S. Reddy and S. Nandi, "Enhanced Network Traffic Anomaly Detector," in *Lecture Notes in Computer Science*. vol. 3816: Springer, pp. 397-403, 2005.

[45] M. Roesch and C. Green, "Snort: The open source network intrusion detection system," Internet: http://www.snort.org.

[46] V. Paxson, "Bro: A system for detecting network intruders in real-time," *Computer Networks,* pp. 2435-2463, 1999.

[47] H. Debar, M. Dacier, and A. Wespi, "Revised taxonomy for intrusion-detection systems," *Annales des Télécommunications,* vol. 55, pp. 361-378, 2000.

[48] D. Janakiram, A. Gunnam, N. Suneetha, V. Rajani, and K. V. K. Reddy, "Object-oriented wrappers for the Linux kernel," *Software: Practice and Experience,* vol. 38, pp. 1411-1427, 2008.

[49] D. Dal, S. Abraham, A. Abraham, S. Sanyal, and M. Sanglikar, "Evolution Induced Secondary Immunity: An Artificial Immune System Based Intrusion Detection System," in *Proc. Computer Information Systems and Industrial Management Applications, CISIM'08*, pp. 65-70, 2008.

[50] C. Iheagwara, A. Blyth, and M. Singhal, "A comparative experimental evaluation study of intrusion detection system performance in a gigabit environment," *Journal of Computer Security,* vol. 11, pp. 1-33, 2003.

[51] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing," RFC2267, Internet: www.isi.edu/in-notes/rfc2267.txt, 1998.

[52] K. Park and H. Lee, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets," *ACM SIGCOMM Computer Communication Review,* vol. 31, pp. 15-26, 2001.

[53] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," RFC 1771, Interent: http://www.ietf.org/rfc/rfc1771.txt, 1995.

[54] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang, "SAVE: Source address validity enforcement protocol," in *Proc. INFOCOM*, pp. 1557-1566, 2002.

[55] X. Geng and A. B. Whinston, "Defeating distributed denial of service attacks," *IT Professional,* vol. 2, pp. 36-42, 2000.

[56] T. H. Kim and S. Lee, "Intelligent Method for Building Security Countermeasures," in *Lecture Notes in Computer Science*. vol. 4252/2006: Springer, pp. 745-750, 2005.

[57] R. Bush, D. Karrenberg, M. Kosters, and R. Plzak, "Root name server operational requirements," *RFC2870,* Internet: http://www.ietf.org/rfc/rfc2870.txt, 2000.

[58] Akamai Corporation. http://www.akamai.com.

[59] A. A. Bertossi, M. C. Pinotti, R. Rizzi, and P. Gupta, "Allocating servers in infostations for on-demand communications," in *Proc. 17th International Symposium on Parallel and Distributed Processing* p. 24.1, 2003.

[60] G. Singh and J. M. Conrad, "Easy-to-use communication interfaces for data acquisition," in *Proc. IEEE Southeastcon*, pp. 111-116, 2008.

[61] S. K. Gupta, V. Goyal, B. Patra, S. Dubey, and A. Gupta, "Design and Development of Malafide Intension Based Privacy Violation Detection System (An Ongoing Research Report)," in *Lecture Notes in Computer Science*. vol. 4332: Springer, p. 369, 2006.

[62] J. Wu and D. P. Agrawal, "Guest editors' introduction: challenges in designing fault-tolerant routing in networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 10, pp. 961-963, 1999.

[63] M. Mandviwalla and N. F. Tzeng, "DRA: A dependable architecture for high-performance routers," in *Proc. International Conference on Parallel Processing Workshops*, pp. 265-272, 2004.

[64] K. Argyraki and D. Cheriton, "Network capabilities: The good, the bad and the ugly," in *Proc. ACM Workshop on Hot Topics in Networks (HotNets-IV)*, 2005.

[65] T. M. Gil and M. Poletto, "MULTOPS: a data-structure for bandwidth attack detection," in *Proc. 10th conference on USENIX Security Symposium*, p. 3, 2001.

[66] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN flooding attacks," in *Proc. IEEE INFOCOM*, pp. 1530- 1539, 2002.

[67] R. B. Blazek, H. Kim, B. Rozovskii, and A. Tartakovsky, "A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods," in *Proc. IEEE Workshop Information Assurance and Security*, pp. 220-226, 2001.

[68] C. M. Cheng, H. T. Kung, and K. S. Tan, "Use of spectral analysis in defense against DoS attacks," in *Proc. IEEE GLOBECOM*, pp. 2143-2148, 2002.

[69] A. Kulkarni and S. Bush, "Detecting distributed denial-of-service attacks using kolmogorov complexity metrics," *Journal of Network and Systems Management*, vol. 14, pp. 69-80, 2006.

[70] J. B. D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran, R. K. Mehra, S. S. Co, and M. A. Woburn, "Proactive detection of distributed denial of service attacks usingMIB traffic variables-a feasibility study," in *Proc. IEEE/IFIP International Symposium on Integrated Network Management*, pp. 609-622, 2001.

[71] B. Bencsath and I. Vajda, "Protection against DDoS attacks based on traffic level measurements," in *Proc. International Symposium on Collaborative Technologies*, pp. 22-28, 2004.

[72] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," in *Proc. DARPA Information Survivability Conference and Exposition*, pp. 303-314, 2003.

[73] G. Carl, G. Kesidis, R. R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *IEEE Internet Computing,* vol. 10, pp. 82-89, 2006.

[74] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *Proc. Conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 217-228, 2005.

[75] T. Peng, C. Leckie, and K. Ramamohanarao, "Protection from distributed denial of service attacks using history-based IP filtering," in *IEEE International Conference on Communications*, pp. 482-486, 2003.

[76] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM,* vol. 13, pp. 422-426, 1970.

[77] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling high-bandwidth flows at the congested router," in *Proc. 9th International Conference on Network Protocols*, p. 192, 2001.

[78] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *ACM SIGCOMM Computer Communication Review,* vol. 32, pp. 62–73, 2002.

[79] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Aggregate-based congestion control," *ACM SIGCOMM Computer Communication Review,* vol. 32, p. 69, 2002.

[80] J. Ioannidis and S. M. Bellovin, "Implementing pushback: Router-based defense against DDoS attacks," in *Proc. Network and Distributed System Security Symposium*, pp. 1-12, 2002.

[81] M. S. Kim, H. J. Kang, S. C. Hung, S. H. Chung, and J. W. Hong, "A flow-based method for abnormal network traffic detection," in *IEEE/IFIP Network Operations and Management Symposium*, pp. 599-612, 2004.

[82] J. Kong, M. Mirza, J. Shu, C. Yoedhana, M. Gerla, and S. Lu, "Random flow network modeling and simulations for DDoS attack mitigation," in *IEEE International Conference on Communications*, pp. 487- 491, 2003.

[83] C. Manikopoulos and S. Papavassiliou, "Network intrusion and fault detection: a statistical anomaly approach," *IEEE Communications Magazine,* vol. 40, pp. 76-82, 2002.

[84] S. Jin and D. S. Yeung, "A covariance analysis model for DDoS attack detection," in *Proc. IEEE International Conference on Communications*, pp. 1882- 1886, 2004.

[85] A. Hussain, J. Heidemann, and C. Papadopoulos, "A framework for classifying denial of service attacks," in *Proc. Conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 99-110, 2003.

[86] E. Gelenbe, R. Lent, and A. Nunez, "Self-aware networks and QoS," *Proceedings of the IEEE*, vol. 92, pp. 1478-1489, 2004.

[87] C. Jin, H. Wang, and K. G. Shin, "Hop-count filtering: an effective defense against spoofed DDoS traffic," in *Proc. 10th ACM conference on Computer and communications security*, pp. 30-41, 2003.

[88] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the source," in *Proc. 10th IEEE International Conference on Network Protocols*, pp. 312-321, 2002.

[89] J. Mirkovic and P. Reiher, "D-WARD: a source-end defense against flooding denial-of-service attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, pp. 216-232, 2005.

[90] Y. Kim, W. C. Lau, M. C. Chuah, and H. J. Chao, "Packetscore: A statistics-based packet filtering scheme against distributed denial-of-service attacks," *IEEE Transactions on Dependable and Secure Computing*, pp. 141-155, 2006.

[91] P. E. Ayres, H. Sun, H. J. Chao, and W. C. Lau, "ALPi: A DDoS defense system for high-speed networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 1864-1876, 2006.

[92] Y. Cheng and R. L. Kashyap, "An axiomatic approach for combining evidence from a variety of sources," *Journal of Intelligent and Robotic Systems*, vol. 1, pp. 17-33, 1988.

[93] W. G. Morein, A. Stavrou, D. L. Cook, A. D. Keromytis, V. Misra, and D. Rubenstein, "Using graphic turing tests to counter automated ddos attacks against web servers," in *Proc. 10th ACM International Conference on Computer and Communications Security*, pp. 8-19, 2003.

[94] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: Breaking a visual captcha," in *Proc. Computer Vision and Pattern Recognition*, pp. 134-141, 2003.

[95] S. Kandula, D. Katabi, M. Jacob, and A. Berger, "Botz-4-sale: Surviving organized DDoS attacks that mimic flash crowds," in *Proc. 2nd Symposium on Networked Systems Design and Implementation (NSDI)*, pp. 287-300, 2005.

[96] Z. Gao and N. Ansari, "Differentiating Malicious DDoS Attack Traffic from Normal TCP Flows by Proactive Tests," *IEEE Communications Letters*, vol. 10, pp. 793-795, 2006.

[97] R. Thomas, B. Mark, T. Johnson, and J. Croall, "NetBouncer: client-legitimacy-based high-performance DDoS filtering," in *Proc. DARPA Information Survivability Conference and Exposition*, pp. 14-25, 2003.

[98] T. Aura, P. Nikander, and J. Leiwo, "DOS-resistant authentication with client puzzles," in *Lecture Notes in Computer Science*. vol. 2133, B. Christianson, B. Crispo, J. A. Malcolm, and M. Roe, Eds.: Springer, pp. 170-177, 2001.

[99] A. Juels and J. Brainard, "Client puzzles: A cryptographic countermeasure against connection depletion attacks," in *Proc. NDSS '99*, pp. 151–165, 1999.

[100] X. F. Wang and M. K. Reiter, "Defending against denial-of-service attacks with puzzle auctions," in *Proc. Symposium onSecurity and Privacy*, pp. 78-92, 2003.

[101] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proc. 2nd ACM SIGCOMM Workshop on Internet measurment*, pp. 71-82, 2002.

[102] Z. L. Zhang, V. J. Ribeiro, S. Moon, and C. Diot, "Small-time scaling behaviors of Internet backbone traffic: An empirical study," in *Proc. IEEE INFOCOM*, pp. 1826-1836, 2003.

[103] M. Thottan and C. Ji, "Anomaly detection in IP networks," *IEEE Transactions on signal processing*, vol. 51, pp. 2191-2204, 2003.

[104] P. Mutaf, "Defending against a Denial-of-Service Attack on TCP," in *Proc. Recent Advances in Intrusion Detection*, pp. 1-15, 1999.

[105] J. Haggerty, T. Berry, Q. Shi, and M. Merabti, "DiDDeM: a system for early detection of TCP SYN flood attacks," Proc. IEEE Global Telecommunications Conference, pp. 2037- 2042, 2004.

[106] J. Xu and W. Lee, "Sustaining availability of web services under distributed denial of service attacks," *IEEE Transactions on Computers*, vol. 52, pp. 195-208, 2003.

[107] S. S. Kim and A. L. N. Reddy, "Statistical techniques for detecting traffic anomalies through packet header data," *IEEE/ACM Transactions on Networking*, vol. 16, pp. 562-575, 2008.

[108] J. Yan, S. Early, and R. Anderson, "The xenoservice-a distributed defeat for distributed denial of service," *Proceedings of ISW 2000*, 2000.

[109] J. K. Kishore, L. M. Patnaik, V. Mani, and V. K. Agrawal, "Application of genetic programming for multicategory pattern classification," *IEEE Transactions on Evolutionary Computation*, vol. 4, pp. 242-258, 2000.

[110] D. M. Divakaran, H. A. Murthy, and T. A. Gonsalves, "Traffic Modeling and Classification Using Packet Train Length and Packet Train Size," in *Lecture Notes in Computer Science*. vol. 4268: Springer, pp. 1-12, 2006.

[111] R. Stone, "CenterTrack: An IP overlay network for tracking DoS floods," in *Proc. 9th conference on USENIX Security Symposium* p. 15, 2000.

[112] Internet: www.cisco.com/debugging.htm.

[113] H. Burch, "Tracing Anonymous Packets to Their Approximate Source," in *Proc. 14th USENIX conference on System administration*, pp. 319-328, 2000.

[114] Y. Xiong, S. Liu, and P. Sun, "On the defense of the distributed denial of service attacks: an on-off feedback control approach," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 31, pp. 282-293, 2001.

[115] D. K. Y. Yau, J. C. S. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 29-42, 2005.

[116] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure overlay services," in *Proc. ACM SIGCOMM*, pp. 61-72, 2002.

[117] D. G. Andersen, "Mayday: Distributed filtering for internet services," in *Proc. 4th conference on USENIX Symposium on Internet Technologies and Systems*, pp. 31-42, 2003.

[118] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker, "DDoS defense by offense," in *Proc. 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 303-314, 2006.

[119] B. Zhao, C. Chi, W. Gao, S. Zhu, and G. Cao, "A Chain Reaction DoS Attack on 3G Networks: Analysis and Defenses," in *Proc. IEEE INFOCOM (to appear)*, 2009.

[120] O. Spatscheck and L. L. Peterson, "Defending against denial of service attacks in Scout," *Operating Systems Review*, vol. 33, pp. 59-72, 1998.

[121] D. Mankins, R. Krishnan, C. Boyd, J. Zao, M. Frentz, and B. B. N. Technologies, "Mitigating distributed denial of service attacks with dynamic resource pricing," in *Proc. 17th Annual Computer Security Applications Conference*, pp. 411-421, 2001.

[122] A. Garg and A. L. N. Reddy, "Mitigation of DoS attacks through QoS regulation," *Microprocessors and Microsystems*, vol. 28, pp. 521-530, 2004.

[123] B. G. Chun, P. Mehra, and R. Fonseca, "Dam: a dos attack mitigation infrastructure," Class Paper CS261-f02: Computer Security, UC Berkeley, 2003.

[124] S. M. Khattab, C. Sangpachatanaruk, R. Melhem, D. Mosse, and T. Znati, "Proactive server roaming for mitigating denial-of-service attacks," in *Proc. International Conference on Information Technology: Research and Education*, pp. 286-290, 2003.

[125] C. Sangpachatanaruk, S. M. Khattab, T. Znati, R. Melhem, and D. Mossé, "Design and analysis of a replicated elusive server scheme for mitigating denial of service attacks," *The Journal of Systems & Software*, vol. 73, pp. 15-29, 2004.

[126] P. Dewan, P. Dasgupta, and V. Karamcheti, "Defending against Denial of Service attacks using Secure Name resolution," in *Proc. International Conference on Security and Management*, pp. 675-681, 2003.

[127] F. Kargl, J. Maier, and M. Weber, "Protecting web servers from distributed denial of service attacks," in *Proc. 10th international conference on World Wide Web*, pp. 514-524, 2001.

[128] J. Brustoloni, "Protecting electronic commerce from distributed denial-of-service attacks," in *Proc. 11th international conference on World Wide Web*, pp. 553-561, 2002.

[129] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC 2475, Internet: http://www.ietf.org/rfc/rfc2474.txt, 1998.

[130] A. A. Ivan, J. Harman, M. Allen, and V. Karamcheti, "Partitionable services: A framework for seamlessly adapting distributed applications to heterogeneous environments," in *Proc. International Conference on High Performance Distributed Computing (HPDC)*, pp. 103-112, 2002.

[131] U. K. Tupakula and V. Varadharajan, "Analysis of traceback techniques," in *Proc. Australasian workshops on Grid computing and e-research*, pp. 115-124, 2006.

[132] A. Belenky and N. Ansari, "IP traceback with deterministic packet marking," *IEEE Communications Letters*, vol. 7, pp. 162-164, 2003.

[133] K. Park and H. Lee, "On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack," in *Proc. IEEE INFOCOM*, pp. 338-347, 2001.

[134] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," in *Proc. IEEE INFOCOM*, pp. 878-886, 2001.

[135] T. Peng, C. Leckie, and K. Ramamohanarao, "Adjusted probabilistic packet marking for IP traceback," in *Lecture Notes in Computer Science*, G.Goos, J.Hartmanis, and J. V. Leeuwen, Eds.: Springer, pp. 697-708, 2002.

[136] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback," *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, pp. 119-137, 2002.

[137] M. Sung and J. Xu, "IP traceback-based intelligent packet filtering: A novel technique for defending against Internet DDoS attacks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, pp. 861-872, 2003.

[138] S. M. Bellovin, M. Leech, and T. Taylor, "ICMP traceback messages," Internet: http://search.ietf.org/internet-drafts/draft-ietf-itrace-01.txt, 2001.

[139] C. D. Marsan, "Denial-of-Service threat gets IETF's attention," Internet: http://www.networkworld.com/news/2000/0724itrace.html, 2000.

[140] S. F. Wu, L. Zhang, D. Massey, and A. Mankin, "Intention-Driven ICMP Trace-Back," Internet: http://tools.ietf.org/html/draft-ietf-itrace-intention-00, 2001.

[141] A. C. Snoeren, "Hash-based IP traceback," in *Proc. 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 3-14, 2001.

[142] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent, and W. T. Strayer, "Single-packet IP traceback," *IEEE/ACM Transactions on Networking (TON)*, vol. 10, pp. 721-734, 2002.

[143] L. A. Sanchez, W. C. Milliken, A. C. Snoeren, F. Tchakountio, C. E. Jones, S. T. Kent, C. Partridge, W. T. Strayer, B. B. N. Technol, and M. A. Cambridge, "Hardware support for a hash-based IP traceback," in *Proc. 2nd DARPA Information Survivability Conference and Exposition* pp. 146-152, 2001.

[144] G. Sager, "Security Fun with OCxmon and cflowd," Internet: http://www.caida.org/funding/ngi1998/content/security/1198/, 1998.

[145] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *Proc. IEEE Symposium on Security and Privacy*, pp. 93-107, 2003.

[146] A. Mankin, D. Massey, C. L. Wu, S. F. Wu, and L. Zhang, "On design and evaluation of "intention-driven" ICMP traceback," in *Proc. 10th International Conference on Computer Communications and Networks*, pp. 159-165, 2001.

[147] A. Yaar, A. Perrig, and D. Song, "StackPi: New packet marking and filtering mechanisms for DDoS and IP spoofing defense," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 1853-1863, 2006.

[148] S. K. Rayanchu and G. Barua, "Tracing attackers with deterministic edge router marking (DERM)," in *Lecture Notes in Computer Science*. vol. 3347, R. K. Ghosh and H. Mohanty, Eds.: Springer, pp. 400-409, 2004.

[149] H. Y. Chang, R. Narayan, S. F. Wu, B. M. Vetter, X. Wang, M. Brown, J. J. Yuill, C. Sargor, F. Jou, and F. Gong, "DECIDUOUS: decentralized source identification for network-basedintrusions," in *Proc. 6th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 701-714, 1999.

[150] U. K. Tupakula and V. Varadharajan, "A practical method to counteract denial of service attacks," in *Proc. 26th Australasian computer science conference* pp. 275-284, 2003.

[151] U. K. Tupakula and V. Varadharajan, "A controller agent model to counteract DoS attacks in multiple domains," in *Proc. IFIP/IEEE Eighth International Symposium on Integrated Network Management*, pp. 113-116, 2003.

[152] J. Lee and G. de Veciana, "Scalable multicast based filtering and tracing framework for defeating distributed DoS attacks," *International Journal of Network Management*, vol. 14, pp. 43-60, 2005.

[153] T. Peng, C. Leckie, and K. Ramamohanarao, "Defending against distributed denial of service attack using selective pushback," in *Proc. 9th IEEE International Conference on Telecommunications*, pp. 411-429, 2002.

[154] S. Floyd, V. Paxson, and S. Shenker, "Aggregate-Based Congestion Control and Pushback," *ACIRI Annual Review*,
Internet: http://www.aciri.org/floyd/talks/ACIRI-Dec00.pdf, 1999.

[155] S. Floyd, S. M. Bellovin, J. Ioannidis, K. Kompella, R. Manajan, and V. Paxson, "Pushback messages for controlling aggregates in the network," Internet: http://bgp.potaroo.net/ietf/idref/draft-floyd-pushback-messages/, 2001.

[156] K. Lakshminarayanan, D. Adkins, A. Perrig, and I. Stoica, "Taming IP packet flooding attacks," *ACM SIGCOMM Computer Communication Review*, vol. 34, pp. 45-50, 2004.

[157] A. K. Bhattacharjee, K. Ravindranath, and A. Pal, *DDSCHED: A DISTRIBUTED DYNAMIC REAL-TIME SCHEDULING ALGORITHM*: Nova Science Publishers, Inc.,USA, 2001.

[158] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," *Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 1-12, 1989.

[159] P. E. McKenney, S. R. I. Int, and M. Park, "Stochastic fairness queueing," in *Proc. INFOCOM*, pp. 733-740, 1990.

[160] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: a scalable architecture to approximate fair bandwidth allocations in high-speed networks," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 33-46, 2003.

[161] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397-413, 1993.

[162] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking (TON)*, vol. 7, pp. 458-472, 1999.

[163] D. Lin and R. Morris, "Dynamics of random early detection," in *Proc. ACM SIGCOMM*, pp. 127-137, 1997.

[164] T. J. Ott, T. V. Lakshman, and L. H. Wong, "Sred: stabilized red," in *Proc. INFOCOM*, pp. 1346-1355, 1999.

[165] S. Floyd and K. Fall, "Router mechanisms to support end-to-end congestion control," Internet: http://www-nrg.ee.lbl.gov/floyd/papers.html, 1997.

[166] W. Zhao, D. Olshefski, and H. Schulzrinne, "Internet quality of service: An overview," *Columbia University, New York, New York, Technical Report CUCS-003-00*, Internet: http://www.research.ibm.com/people/o/olshef/qos.shtml, 2000.

[167] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers," RFC 2474, Internet: www.ietf.org/rfc/rfc2474.txt ,1998.

[168] S. D. Crocker, "Protecting the internet from distributed denial-of-service attacks: a proposal," *Proceedings of the IEEE*, vol. 92, pp. 1375-1381, 2004.

[169] H. Wang and K. G. Shin, "Transport-aware IP routers: A built-in protection mechanism to counter DDoS attacks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, pp. 873- 884, 2003.

[170] L. Santhanam, A. Kumar, and D. P. Agrawal, "Taxonomy of IP Traceback," *Journal of Information Assurance and Security*, vol. 1, pp. 79-94, 2006.

[171] S. P. Pal, R. R. Suman, G. S. A. Kumar, and R. Malhotra, "Virtual video caching: A scalable and generic technique for improved quality of video service," *Journal of High Speed Networks*, vol. 13, pp. 249-263, 2004.

[172] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring Internet denial-of-service activity," *ACM Transactions on Computer Systems (TOCS)*, vol. 24, pp. 115-139, 2006.

[173] D. Schnackenberg, K. Djahandari, and D. Sterne, "Infrastructure for intrusion detection and response," in *DARPA Information Survivability Conference and Exposition*, pp. 3-11, 2000.

[174] L. Spitzner, "Honeypots: definitions and value of honeypots," Internet: http://www.tracking-hackers.com/papers/honeypots.html, 2003.

[175] L. Spitzner, "Honeypots: simple, cost-effective detection," *SecurityFocus InFocus Article, Internet: http://www.securityfocus.com/infocus/1690,* 2003.

[176] L. Spitzner, *Honeypots: tracking hackers*: Addison-Wesley Professional, 2003.

[177] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites," in *Proc. 11th Word Wide Web conference*, pp. 293-304, 2002.

[178] C. Douligeris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: classification and state-of-the-art," *Computer Networks,* vol. 44, pp. 643-666, 2004.

[179] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review,* vol. 34, pp. 39-53, 2004.

[180] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni, "Analysis of a Denial of Service Attack on TCP," in *Proc. 1997 IEEE Symposium on Security and Privacy*, pp. 208-223, 1997.

[181] CERT CA-1996-01: UDP Port Denial-of-Service Attack. Internet: http://www.cert.org/advisories/CA-1996-01.html, 1996.

[182] "DoS using nameservers," CERT Coordination Center, Internet: http://www.cert.org/incidentnotes/IN-2000-04.html, 2000.

[183] S. Dietrich, N. Long, and D. Dittrich, "Analyzing distributed denial of service tools: The shaft case," in *Proc. 14th Systems Administration Conference*, pp. 329–339, 2000.

[184] S. Krishnamurthy, "IP Layer Packet Redirection," Master's Report, Stony Brook University, 2004.

[185] A. Sardana, B. Gandhi, and R. Joshi, "A Novel Online Technique to Characterize and Mitigate DoS Attacks using EPSD and Honeypots," in *Innovative Algorithms and Techniques in Automation, Industrial Electronics and Telecommunications*, T. Sobh, K. Elleithy, A. Mahmood, and M. Karim, Eds.: Springer, pp. 49-54, 2007.

[186] K. G. Anagnostakis, S. Sidiroglou, P. Akritidis, K. Xinidis, E. Markatos, and A. D. Keromytis, "Detecting targeted attacks using shadow honeypots," in *Proc. 14th USENIX Security Symposium*, pp. 129–144, 2005.

[187] T. B. Reddy, B. S. Manoj, and C. S. R. Murthy, "Multimedia traffic support for asynchronous ad hoc wireless networks," in *Proc. 1st International Conference on Broadband Networks*, pp. 569-578, 2004.

[188] R. Ingle and G. Sivakumar, "Tunable group key agreement," in *Proc. 32nd IEEE Conference on Local Computer Networks*, pp. 1017-1024, 2007.

[189] S. Yeldi, S. Gupta, T. Ganacharya, S. Doshi, D. Bahirat, R. Ingle, and A. Roychowdhary, "Enhancing network intrusion detection system with honeypot," in *Proc. TENCON 2003*, pp. 1521-1526, 2003.

[190] A. Sardana and R. Joshi, "An Integrated Honeypot Framework for Proactive Detection, Characterization and Redirection of DDoS Attacks at ISP level," *Journal of Information Assurance and Security*, vol. 1, pp. 1-15, 2008.

[191] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, pp. 3-55, 2001.

[192] C. Beck and F. Schlögl, *Thermodynamics of chaotic systems: An introduction*: Cambridge University Press, 1993.

[193] A. Sardana, K. Kumar, and R. C. Joshi, "Detection and Honeypot Based Redirection to Counter DDoS Attacks in ISP Domain," in *Proc. 3rd International Symposium on Information Assurance and Security*, pp. 191-196, 2007.

[194] A. Sardana and R. Joshi, "Dual-Level Attack Detection for Networks under DDoS Attacks," *IEEE transactions on Systems, Man, and Cybernetics (SMC) Part C: Applications & Reviews, Special Issue on Availability, Reliability and Security*, 2009 (communicated).

[195] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. IEEE INFOCOM*, pp. 594--602, 1996.

[196] E. W. Zegura, K. L. Calvert, and M. J. Donahoo, "A quantitative comparison of graph-based models for Internet topology," *IEEE/ACM Transactions on Networking (TON)*, vol. 5, pp. 770-783, 1997.

[197] A. Sardana, R. Joshi, and T. Kim, "Deciding Optimal Entropic Thresholds to Calibrate the Detection Mechanism for Variable Rate DDoS Attacks in ISP Domain," in *Proc. International Conference on Information Security and Assurance*, pp. 270-275, 2008.

[198] A. Sardana, R. C. Joshi, T. Kim, and S. Jang, "Deciding optimal entropic thresholds to calibrate the detection mechanism for variable rate DDoS attacks in ISP domain: honeypot based approach," *Journal of Intelligent Manufacturing*, pp. 1-12, 2008.

[199] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and application*. NJ: Prentice Hall Englewood Cliffs, 1993.

[200] T. Peng, C. Leckie, and K. Ramamohanarao, "Detecting distributed denial of service attacks by sharing distributed beliefs," in *Lecture Notes in Computer Science*. vol. 2727, R. Safavi-Naini and J. Seberry, Eds.: Springer, pp. 214-225, 2003.

[201] T. Peng, C. Leckie, and K. Ramamohanarao, "Proactively detecting distributed denial of service attacks using source IP address monitoring," in *Lecture Notes in Computer Science*, N. Mitrou, Ed.: Springer, pp. 771-782, 2004.

[202] *Network Simulator NS-2. Internet: http://www.isi.edu/nsnam.*

[203] S. Srivastava, S. Tripathi, D. Sanghi, and A. K. Chaturvedi, "A code allocation protocol for maximizing throughput in CDMA based ad hoc networks," in *IEEE Wireless Communicatons and Networking Conference (WCNC)*, pp. 1385-1390, 2003.

[204] M. H. Zweig and G. Campbell, "Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine," *Clinical Chemistry*, vol. 39, pp. 561-77, 1993.

[205] A. Sardana and R. C. Joshi, "Autonomous dynamic honeypot routing mechanism for mitigating DDoS attacks in DMZ," in *Proc.16th IEEE International Conference on Networks*, pp. 1-7, 2008.

[206] N. Weiler, "Honeypots for distributed denial-of-service attacks," in *Proc. 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 109-114, 2002.

[207] J. Jones, D. F. Cohen, D. Dittrich, and C. Huegen, "Distributed denial of service attacks: defenses," *Global Integrity Corporation*, pp. 1-14, 2000.

[208] S. M. Khattab, C. Sangpachatanaruk, D. Mosse, R. Melhem, and T. Znati, "Roaming honeypots for mitigating service-level denial-of-service attacks," in *Proc. 24th International Conference on Distributed Computing Systems*, pp. 328-337, 2004.

[209] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *Journal of the ACM*, vol. 33, pp. 792-807, 1986.

[210] O. Goldreich, L. Levin, and N. Nisan, "On constructing 1-1 one-way functions," in *Proc. Electronic Colloquium on Computational Complexity*, pp. 1–11, 1995.

[211] R. Rivest, "The MD5 message-digest algorithm," RFC1321, Internet: http://www.ietf.org/rfc/rfc1321.txt, 1992.

[212] R. L. Rivest and A. Shamir, "PayWord and MicroMint: Two simple micropayment schemes," in *Lecture Notes in Computer Science*. vol. 1189, M. Lomas, Ed.: Springer, pp. 69-88, 1997.

[213] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, pp. 770 - 772, 1981.

[214] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless networks*, vol. 8, pp. 521-534, 2002.

[215] M. L. Das, A. Saxena, V. P. Gulati, and D. B. Phatak, "Hierarchical key management scheme using polynomial interpolation," *ACM SIGOPS Operating Systems Review*, vol. 39, pp. 40-47, 2005.

[216] A. C. Snoeren, H. Balakrishnan, and M. F. Kaashoek, "The migrate approach to internet mobility," in *Proc. Oxygen Student Workshop*, Interent: http://nms.lcs.mit.edu/~snoeren/papers/migrate-sow.pdf, 2001.

[217] F. Sultan, K. Srinivasan, D. Iyer, and L. Iftode, "Migratory TCP: Connection migration for service continuity in the Internet," in *Proc. 22nd International Conference on Distributed Computing Systems*, pp. 469-470, 2002.

[218] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. 18th ACM symposium on Operating systems principles*, pp. 131-145, 2001.

[219] A. Sardana and R. Joshi, "An auto-responsive honeypot architecture for dynamic resource allocation and QoS adaptation in DDoS attacked networks," *Computer Communications, Elsevier*, vol. 32, pp. 1384-1399, 2009.

[220] A. Sardana and R. C. Joshi, "Simulation of Dynamic Honeypot Based Redirection to Counter Service Level DDoS Attacks," in *Lecture Notes in Computer Science*, vol. 4812, P. McDaniel and G. Shyam K, Eds., pp. 259-262, 2007.

[221] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz, "On realistic network topologies for simulation," in *Proc. ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pp. 28-32, 2003.

[222] K. Calvert and E. Zegura, "GT internetwork topology models (GT-ITM)," Internet: http://www.cc.gatech.edu/projects/gtitm/. 1997.

[223] GT-ITM. Georgia Tech Internetwork Topology Models.

Internet: http://www.cc.gatech.edu/projects/gtitm/.

[224] K. I. Calvert, M. B. Doar, and E. W. Zegura, "Modeling internet topology," *IEEE Communications Magazine*, vol. 35, pp. 160-163, 1997.

[225] R. Canonico, D. Emma, G. Ventre, and V. Claudio, "Extended NAM: An ns2 Compatible Network Topology Editor for Simulation of Web Caching Systems on Large Network Topologies,"
Internet: http://www.grid.unina.it/grid/ExtendedNamEditor/index.html, 2003.

[226] "The ns manual-the VINT project," K. Fall and K. Varadhan, Eds. Univ. California, Berkeley, CA, 2001.

[227] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *Proc. IEEE INFOCOM*, pp. 1190-1199, 2002.

[228] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*: Prentice Hall, 1982.

Publications out of the work

Journals

1. A. Sardana and R.C. Joshi, "An Auto Responsive Honeypot Architecture for Dynamic Resource Allocation and QoS Adaptation in DDoS Attacked Networks," *Computer Communications*, Elsevier, vol. 32, pp. 1384-1399, April 2009.
2. A. Sardana, R.C. Joshi, T. Kim and S. Jang, "Deciding Optimal Entropic Thresholds to Calibrate the Detection Mechanism for Variable Rate DDoS Attacks in ISP Domain: Honeypot based approach," *Journal of Intelligent Manufacturing*, Springer. DOI: 10.1007/s10845-008-0204-3, December 2008.
3. A. Sardana and R.C. Joshi, "An Integrated Honeypot Framework for Proactive Detection, Characterization and Redirection of DDoS Attacks at ISP level," *International Journal of Information Assurance and Security (JIAS)*, vol. 3 issue 1, ISSN 1554-1010, Dynamic Publishers Inc., USA, pp. 1-15, March 2008.

Book Chapters

1. A. Sardana, B. Gandhi, and R. Joshi, "A Novel Online Technique to Characterize and Mitigate DoS Attacks using EPSD and Honeypots," in *Innovative Algorithms and Techniques in Automation, Industrial Electronics and Telecommunications*, T. Sobh, K. Elleithy, A. Mahmood, and M. Karim, Eds.: Springer, pp. 49-54, 2007.
2. A. Sardana and R. C. Joshi, "Simulation of Dynamic Honeypot Based Redirection to Counter Service Level DDoS Attacks," in *Lecture Notes in Computer Science*, vol. 4812, P. McDaniel and G. Shyam K, Eds.: Springer, pp. 259-262, 2007.

International Conferences

1. A. Sardana and R. C. Joshi, "Autonomous Dynamic Honeypot Routing Mechanism for Mitigating DDoS Attacks in DMZ," in *Proc. 16th IEEE International Conference on Networking*, India, pp. 1-7, Dec 2008.

2. A. Sardana and R. C. Joshi, "Honeypot Based Routing to Mitigate DDoS Attacks on Servers at ISP Level," in *Proc. International Symposium on Information Processing (ISIP 2008),* Moscow, pp. 505-509, May 2008.

3. A. Sardana and R. C. Joshi, "Deciding Optimal Entropic Thresholds to Calibrate the Detection Mechanism for Variable Rate DDoS Attacks in ISP Domain," in *Proc. 2nd International Conference on Information Security and Assurance,* Korea, pp. 270-275, April 2008.

4. A. Sardana, R. C. Joshi, "Simulation of Dynamic Honeypot Based Redirection to Counter Service Level DDoS Attacks," in *Proc. International Conference on Information Systems and Security ICISS 2007*, Springer LNCS, New Delhi, India, pp. 259-262, Dec 2007.

5. A. Sardana, K. Kumar and R. C. Joshi, "Detection and Honeypot based Redirection to Counter DDoS Attacks in ISP Domain," in *Proc. IEEE The Third International Symposium on Information Assurance and Security*, Manchester, U. K., pp. 191-196, Aug 2007.

6. A. Sardana, B. Gandhi and R. C. Joshi, "A Novel Online Technique to Characterize and Mitigate DoS Attacks using EPSD and Honeypots," in *Proc. International Conference on Telecommunications and Networking, TeNe 06,* Springer LNCS, Bridgeport U.S., pp. 49-54, Dec 2006.

7. A. Sardana, B. Gandhi and R. C. Joshi, "A Novel Framework for Characterization, Source Identification and Mitigation of DoS Attacks," in *Proc. International Conference on Information Security and Computer Forensics ISCF 06*, pp. 99-107, Dec 2006.

National Conferences

1. A. Sardana and R. C. Joshi, "Security Policies in Different Scenarios for Networks under DDoS Attacks," in *Proc. 2nd National Conference on Competitive Technology and Network Dynamics,* Ghaziabad, India , pp. 1-6, 2009.

2. S. Singh, A. Sardana and R. C. Joshi, "Detection and Mitigation of TCP SYN Flood Attacks Using Virtual Blackhole Honeypots," in *Proc. National Conference On Emerging Trends In Electrical, Electonics And Computer Technologies*, New Delhi, pp. 300-306, September 2008.

3. R. C. Joshi and A. Sardana, "Secure Honeypot Architecture: New Information Warfare Deception Technique" in *Proc. DefCom'06, National Conference on Communications in Tactical Battle Area,* New Delhi, India, pp. 99-104, 2006.

Communicated

1. A. Sardana and R. C. Joshi, "Dual-Level Attack Detection for Networks under DDoS Attacks," in *IEEE transactions on Systems, Man, and Cybernetics (SMC) Part C: Applications & Reviews, Special Issue on Availability, Reliability and Security.*
2. A. Sardana and R. C. Joshi, "Three – Tier Honeypot Framework for Networks under DDoS Attacks," in *12th International Symposium on Recent Advancements in Intrusion Detection (RAID 2009).*