ABSTRACT

ON MODELING AND MITIGATING NEW BREED OF DOS ATTACKS

by
Amey Bhaskar Shevtekar

Denial of Service (DoS) attacks pose serious threats to the Internet, exerting in tremendous impact on our daily lives that are heavily dependent on the good health of the Internet. This dissertation aims to achieve two objectives: 1) to model new possibilities of the low rate DoS attacks; 2) to develop effective mitigation mechanisms to counter the threat from low rate DoS attacks.

A new stealthy DDoS attack model referred to as the "quiet" attack is proposed in this dissertation. The attack traffic consists of TCP traffic only. Widely used botnets in today's various attacks and newly introduced network feedback control are integral part of the quiet attack model. The quiet attack shows that short-lived TCP flows used as attack flows can be intentionally misused. This dissertation proposes another attack model referred to as the perfect storm which uses a combination of UDP and TCP. Better CAPTCHAs are highlighted as current defense against botnets to mitigate the quiet attack and the perfect storm.

A novel time domain technique is proposed that relies on the time difference between subsequent packets of each flow to detect periodicity of the low rate DoS attack flow. An attacker can easily use different IP address spoofing techniques or botnets to launch a low rate DoS attack and fool the detection system. To mitigate such a threat, this dissertation proposes a second detection algorithm that detects the sudden increase in the traffic load of all the expired flows within a short period. In a network rate DoS attacks, it is shown that the traffic load of all the expired flows is less than certain thresholds, which are derived from real Internet traffic analysis. A novel filtering scheme is proposed to drop the low rate DoS attack packets. The simulation results confirm attack mitigation by using proposed technique. Future research directions will be briefly discussed.

ON MODELING AND MITIGATING NEW BREED OF DOS ATTACKS

by
Amey Bhaskar Shevtekar

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Engineering

Department of Electrical and Computer Engineering

May 2009

APPROVAL PAGE

ON MODELING AND MITIGATING NEW BREED OF DOS ATTACKS

Amey Bhaskar Shevtekar

3/4/09

Dr. Nirwan Ansari, Dissertation Advisor Date
Professor of Electrical and Computer Engineering, NJIT

3-4-09

Dr. Edwin Hou, Committee Member Date
Associate Professor of Electrical and Computer Engineering, NJIT

03/04/09

Dr. Roberto Rojas-Cessa, Committee Member Date
Associate Professor of Electrical and Computer Engineering, NJIT

3/4/09

Dr. Yanchao Zhang, Committee Member Date
Assistant Professor of Electrical and Computer Engineering, NJIT

03/04/2009

Dr. Cristian Borcea, Committee Member Date
Assistant Professor of Computer Science, NJIT

BIOGRAPHICAL SKETCH

Author: Amey Bhaskar Shevtekar

Degree: Doctor of Philosophy

Date: May 2009

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2009

- Master of Science in Telecommunications,
  New Jersey Institute of Technology, Newark, NJ, 2004

- Bachelor of Science in Electronics and Telecommunications Engineering,
  University of Mumbai, Mumbai, India, 2002

Major: Computer Engineering

Presentations and Publications:

Amey B. Shevtekar and Nirwan Ansari,
"A router based technique to mitigate reduction of quality (RoQ) attacks,"
Elsevier Computer Networks Journal, Vol. 52, No. 5, pp. 957-970, April 2008.

Amey B. Shevtekar, John C. Stille, and Nirwan Ansari,,
"On the impacts of Low Rate DoS Attacks on VoIP Traffic,"
Invited paper in Wiley Security and Communication Networks Journal, Vol. 1, No. 1, pp. 45-56, Feb 2008.

Amey B. Shevtekar, Karunakar Anantharam, and Nirwan Ansari,
"Low Rate TCP Denial-of-Service Attack Detection at Edge Routers,"
IEEE Communication Letters, Vol. 9, No. 4, pp. 363-365, April 2005.

Amey B. Shevtekar and Nirwan Ansari,
"Do Low Rate DoS Attacks Affect QoS Sensitive VoIP Traffic?,"
Proceedings of the IEEE ICC, Istanbul, Turkey, pp. 2153-2158, June 2006.

Amey B. Shevtekar and Nirwan Ansari,
"Method and System to mitigate Low Rate Denial of Service attacks,"
Pending patent application no 12/127246, filed May 2008.

Amey B. Shevtekar and Nirwan Ansari,
"Proactive Test Based Differentiation Method and System to Mitigate Low Rate DoS Attacks,"
Pending patent application no 12/127235, filed May 2008.

Amey B. Shevtekar and Nirwan Ansari,
"System and Method For a Completely Automated Public Turing-Test to Tell Computers and Humans Apart (CAPTCHA),"
Provisional patent application NJIT no 09-017, filed Feb 2009.

Amey B. Shevtekar, Nirwan Ansari, and Roger Karrer,
"The Perfect Storm: towards a Perfect DDoS Attack,"
Proceedings of the IEEE Sarnoff Symposium, Princeton, NJ, April 2009.

Amey B. Shevtekar and Nirwan Ansari,
"A Proactive Test Based Differentiation Technique to Mitigate Low Rate DoS Attacks,"
Invited paper in proceedings of the IEEE ICCCN, Honolulu, Hawaii, pp. 639-644, August 2007.

Amey B. Shevtekar and Nirwan Ansari,
"Is it Congestion or a DDoS Attack?,"
Submitted IEEE Communication Letters, March 2009.

To my beloved Grandmother.

ACKNOWLEDGMENT

I would like to express my deepest appreciation to my advisor, Dr. Nirwan Ansari, who not only served as my dissertation advisor providing necessary expert guidance, but also gave me freedom for pursuing independent research. His valuable critique improved my subject knowledge as well as writing skills. Special thanks are given to Dr. Edwin Hou, Dr. Roberto Rojas-Cessa, Dr. Yanchao Zhang, and Dr. Cristian Borcea for participating in my committee.

I also wish to thank Dr. Roger Karrer for providing an internship opportunity at Deustche Telekom Laboratories, Germany which led to some new ideas and works. I would like to acknowledge NSF (Grant Numbers's # 0435250, 0726549, 0552483, and 0726549) and NJIT for providing financial support during my PhD studies. Many of my fellow graduate students in the Advanced Networking Laboratory and friends are deserving recognition for their support.

Finally, a special thanks to my mother, my uncle, and all my family members who gave me constant support without which this dissertation would not have been possible.

TABLE OF CONTENTS

Chapter **Page**

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

LIST OF FIGURES
(Continued)

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of this dissertation is to achieve two goals: 1) to evaluate the threat of the low rate Denial of Service (DoS) attack and develop effective mitigation mechanisms to counter the low rate DoS attacks, and 2) to model new possibilities of the low rate DoS attacks.

1.2 Background Information

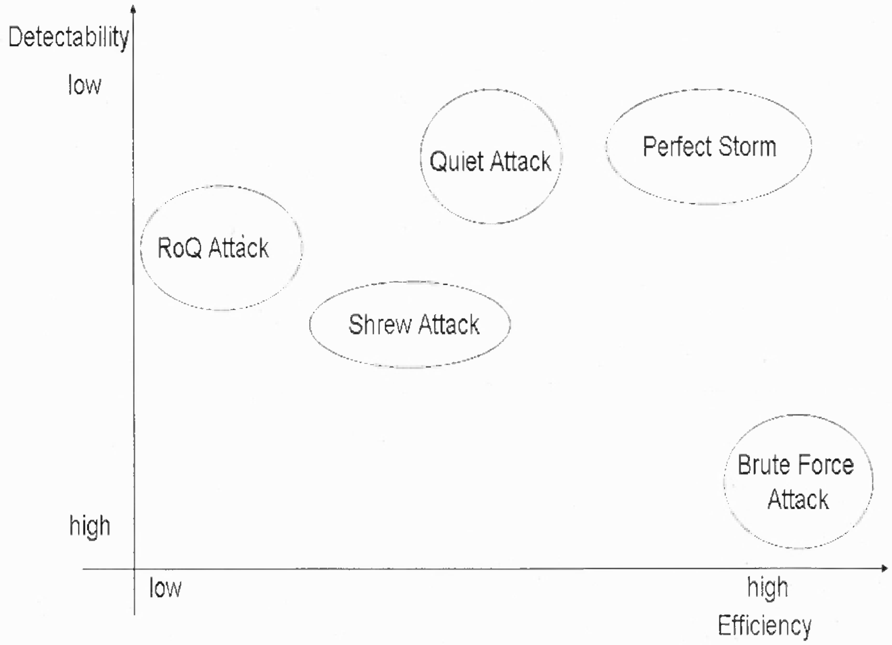


Figure 1.1 The trend of DDoS attacks in the Internet.

Internet has been plagued by a variety of security threats over the past several years. The DDoS attacks received hype after 2000 when yahoo.com was attacked. After that event, DDoS attacks have been rampaging in the Internet frequently. A famous website [1] has been keeping track of DDoS related news since long time. A tremendous amount of work has been done in the industry and academia to mitigate DDoS attacks, but none have been able to eradicate them successfully. The motivations for launching attacks have also been shifted significantly. Initially, they were for publicity but now they are for economic or political incentives. Thus, DDoS attacks are a complex problem to mitigate.

Figure 1.1 depicts the trend of DDoS attacks in the Internet. The x-axis indicates the efficiency of the attack, indicating how much damage it can cause to the good traffic. The y-axis indicates the detectability of the attack, indicating the exposure of the attack to defense systems. The early brute force attacks relied on sending high rate attack traffic continuously to a website. They are now easily detected because there have been many defense systems that can distinguish such anomalous attack traffic [2]. The shrew attack and RoQ attack are the new low rate DoS attacks that are difficult to detect as compared to the brute force attack, but they primarily affect only long-lived TCP traffic. One major contribution of this dissertation is to forewarn and model the emerging sophisticated attacks, namely, the quiet attack and the perfect storm. Such attacks have higher impact on good traffic and yet they are evasive. This Chapter will briefly discuss old brute force attacks, and then introduce recent low rate DoS attacks.

1.3 Old Brute Force Attacks

A DoS attack is defined as an attack that causes a network or a computer incapable of providing service to the users [2]. It typically targets the bandwidth of the victim. A DDoS attack is defined as an attack that uses multiple unwilling computers to send the attack traffic to the victim. A DDoS attack is more lethal since it has a large capacity to send attack traffic as compared to a DoS attack. These attacks are also refereed to as brute force attacks as they send attack traffic at high rates and lack characteristics required to be stealthy. There are several types of brute force DDoS attacks that have been reported in the literature, and a few of commonly used attacks are described below. The DDoS attacks can be characterized as shown in Figure 1.2 [2].

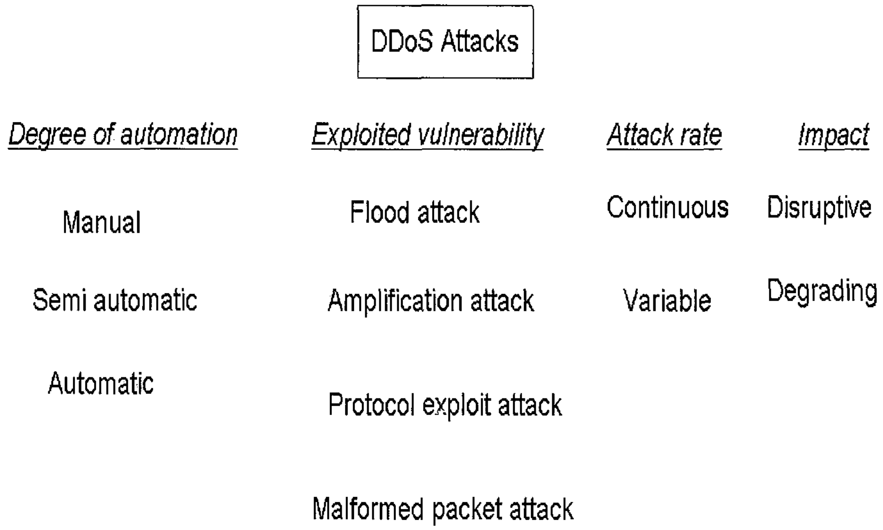


Figure 1.2 Classification of DDoS attacks.

Classification of DDoS attacks can be classified by the degree of automation, i.e., the level of sophistication of the attack mechanism. Earlier attacks were manual, and were improved gradually. In a manual attack [2], the victims are scanned for a particular vulnerability which is exploited to gain access into the victim's system by the attacker. An attacker would then use commands to control the victim during the attack. In a semi-automatic attack, some steps of the attack procedure, which were originally manually performed become automated, for example, some of the victims are compromised to act as attack agents who coordinate the attack by issuing commands to conceal the identity of the attacker even if the attack is detected [2]. The attack agents are pre-programmed with the necessary required commands during the attack by an attacker. All the recent attacks are highly automatic requiring minimal communication between the attacker and the compromised machines once the attack is launched. All the attack steps are pre-programmed and delivered as a payload to infect clients, also referred to as zombies or bots. Some new attack payloads, which fail to detect a specific vulnerability in a victim machine, will automatically scan for another vulnerability in the same machine. Recent botnet attacks on Estonia's websites employed fully automated mechanisms [1]. Botnet is a network of bots or zombies controlled by a botmaster [3].

Classification of DDoS attacks based on an exploited vulnerability takes into account property of the network or the protocol used in the attack. The category to which the flood attack belongs is the simplest of all categories in which an attacker relies on denying the network bandwidth to the legitimate users. The common example of this category is the UDP flood attack [4]. In a UDP flood, attacker sends UDP packets at a high rate to the victim so that the network bandwidth is exhausted. UDP is a connectionless protocol so it is easy to send UDP packets at any rate in the network. Another attack in this category is the ICMP echo flood; it involves sending many ICMP echo request packets to a host. The host replies with an ICMP echo reply to each of the two ICMP echo request packet, and many such requests and reply packets fill up the network bandwidth.

In the category of amplification attack, an attacker exploits a protocol property such that few packets will lead to amplified attack traffic. The SMURF attack exploits the ICMP protocol [5] and it falls in this category. It involves forging a source IP address of the ICMP echo request packet with the address of the victim. The destination address of the ICMP echo request is the broadcast address of the LAN or so-called directed broadcast addresses. On receiving such a packet, each active host on a LAN responds an ICMP echo reply packet to the victim. Typically, a LAN has many active hosts, and so a tremendous amount of attack traffic is generated to cripple the victim. To avoid such an attack, most system administrators are advised to disable the directed broadcast addresses.

In the category of protocol exploit, again a property of the protocol is exploited. The SYN attack [6] exploits the TCP protocol's three way handshake mechanism. Web servers use port 80 to accept incoming HTTP traffic that runs on top of the TCP protocol. When users wants to access a webpage, it sends a SYN packet to the web server's open port 80. The web server does not know the user's IP address before the arrival of the SYN packet. The server, upon receipt of a SYN packet, sends a SYN/ACK packet and thus puts the connection in the LISTEN state. A legitimate user's machine replies to the web server's SYN/ACK packet with an ACK packet and establishes the connection. However, if the SYN packet has been sent from an attack machine which does not respond to the server with an ACK packet, the web server never gets an ACK packet and the connection remains incomplete. Every web server has a finite amount of memory resources to handle such incomplete connections. The main goal of the SYN attack is to exhaust the finite amount of memory resources of a web server by sending a large number of SYN packets. Such an attack causes the web server to crash. Other similar protocol exploit attack example is PUSH + ACK attack [7] is another example falling in this category.

In the category of malformed packet attack, the packet header fields are modified to instigate a crash of the operating system of the receiver. An IP packet having the same source and destination IP address is a malformed packet [2]. In another kind of malformed packet attack, the IP options fields of the IP header are randomized, and the type of service bit is set to one. Ping of death attack involves sending a ping packet larger than the maximum IP packet size of 65535 bytes [8]. Historically, a ping packet has a size of 56 bytes, and most of the systems cannot handle ping packets of larger size. Operating systems take more time to process such unusual packets, and has a large quantity of such packets can crash the systems.

DDoS attacks can also be classified by their attack rates, namely, continuous vs. variable. Likewise, they can also be classified by their impacts: disruptive vs. degrading. Disruptive attacks aims for denial of service while degrading attacks aim for reduction of quality. Examples of the class of degrading attacks are discussed in more details in the next Section. This dissertation includes the evaluation of the effects of degrading attacks as well as schemes proposed to detect them.

1.4 Low Rate DoS Attacks

As mentioned before, the security in the Internet is increasingly challenging as more professionals are getting into this lucrative business. An article in the New York Times [9] describes one such business of selling the software exploits. Attacks are also getting more sophisticated, as the attackers are not merely interested in achieving publicity. The shrew attack is one such intelligent attack, which was first reported in [10], followed by a series of variants [11]-[15]. This study considers these attacks as low rate DoS attacks. It is typically illustrated by a periodic waveform shown in Figure 1.2., where *T* is the time period, *t* is the burst period, and *R* is the burst rate.

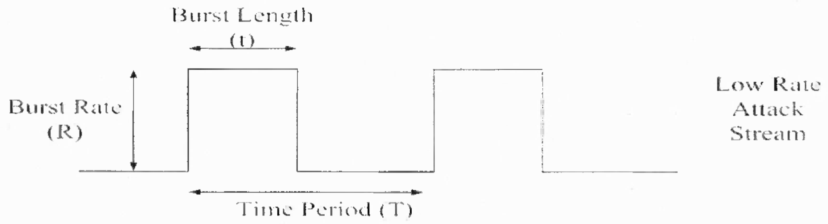


Figure 1.3 An example of a generic low rate DoS attack pattern.

A shrew attack exploits widely implemented minimum RTO [16] property of the TCP protocol. The following characterize the low rate TCP DoS attack:

- It sends periodic bursts of packets at one-second interval.
- The burst rate is equal to or greater than the bottleneck capacity.
- The burst period is tuned to be equal to the round-trip times of the TCP connections; this parameter determines whether the attack will cause DoS to the TCP connections with small or long round trip times.
- The exponential back off algorithm of the TCP's retransmission mechanism is eventually exploited.

In a Reduction of Quality of Service (RoQ) attack [11], the attacker sends high rate short bursts of the attack traffic at random time periods, thereby forcing the adaptive TCP traffic to backoff due to the temporary congestion caused by the attack bursts. In particular, the periodicity is not well defined in a RoQ attack, thus allowing the attacker to keep the average rate of the attack traffic low in order to evade the regulation of adaptive queue management like RED and RED-PD [17]-[19]. By sending the attack traffic, the RoQ attack introduces transients and restricts the router queue from reaching the steady state. The open knowledge of these stealthy attacks presses for early fixes. For simplicity, from now on, the term "low rate DoS attack" refers to both the shrew and RoQ attack, unless otherwise stated as shown in Figure 1.4.

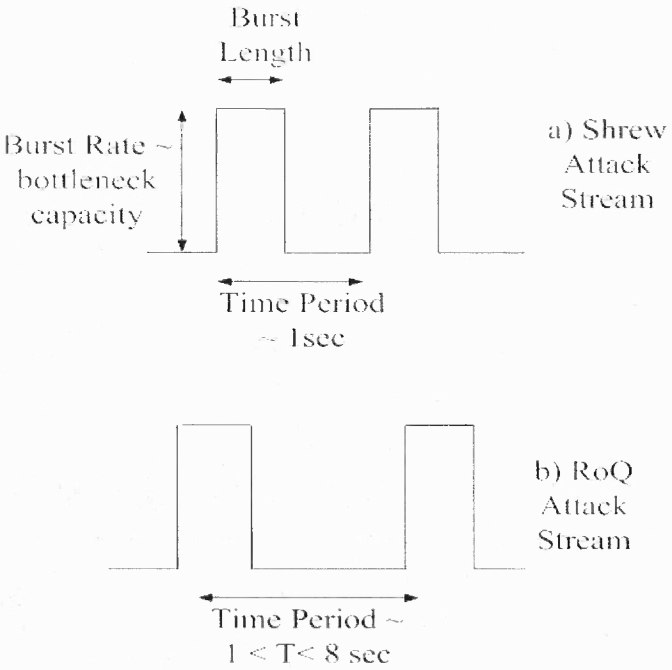


Figure 1.4 Low rate DoS attacks: a) shrew attack, and b) RoQ attack.

The attacker can also use different types of IP address spoofing to evade several other detection systems. Owing to the open nature of the Internet, IP address spoofing can still evade ingress and egress filtering techniques at many sites [20]. A low rate DoS attack can use IP address spoofing in a variety of ways like random IP address spoofing and continuous IP address spoofing [18]. The use of IP address spoofing most importantly divides the high rate of a single flow during the burst period of the attack among multiple flows with spoofed identities. This way, an attacker can evade detection systems that concentrate on finding anomalous traffic rate. The detection systems that rely on identifying periodicity of the low rate DoS attack in the frequency domain can detect the periodicity, but they fail to filter the attack traffic as it is difficult to know the IP addresses that an attacker will use in the future.

This problem is further exacerbated by the use of botnets; a botnet is a network of compromised real hosts across the Internet controlled by a master [3]. As an attacker using botnets has control over thousands of hosts, it can easily use these hosts to launch a low rate DoS attack; this is analogous to a low rate DoS attack that uses random or continuous IP address spoofing. Now, with the use of botnets, the IP addresses of bots are not spoofed and so these packets cannot be filtered by spoofing prevention techniques. In fact, these attack packets are similar to the HTTP flows. This random and continuous IP address spoofing problem described above is unique to the low rate DoS attacks, and is different from other types of DDoS attacks. These attacks [11] can be launched at any routers in the Internet; the edge routers can be easy targets as their capacities are small, and hence attackers can easily incite denial of service to the VoIP users traversing those routers. Low rate DoS attacks fall in the DDoS attack category of variable attack rate and degrading impact.

1.5 Defenses for Brute Force Attacks

Mitigating DDoS attacks is a widely studied problem and some of the popular approaches are described below. Defense systems can be broadly classified based on their functions. There are four main categories of defense systems: intrusion prevention, intrusion detection, intrusion response, and intrusion mitigation [2].

Intrusion prevention systems prevent an attack from occurring. Ingress and egress filtering control IP address spoofing being used in the attack. Ingress filtering only allows packets destined for the source network to enter the network, thereby filtering all other packets. It is implemented at the edge routers of the network, and it limits the attack traffic entering the network. Egress filtering is an outbound filter that allows only packets with source IP addresses originated from the source network to exit the source network. Use of egress filtering controls attack traffic going to destination networks [21]. Disabling IP broadcasts prevents smurf attacks. Honeypots are network decoys [22]. They allow studying attack behavior before the onset of an attack. Honeypots act as early warning systems. Honeypots mimic all aspects of a real network like web server and, mail server to lure attackers. The primary goal of the honeypots is to determine/derive the exploit mechanism of the attack in order to build defense signatures against the exploit. Intrusion prevention systems cannot completely prevent an attack, but they contain the damage of the attack. They allow building better defense systems by studying the attack.

Intrusion detection systems detect an attack based on attack signatures or anomalous behaviors. Snort is a popular signature based network intrusion detection systems [23]. It performs protocol analysis and content matching to passively detect a variety of attacks like buffer overflows, port scans, and web application attacks. Snort uses Berkeley's libpcap library to sniff packets. It uses a chain structure to maintain rules. The header of each chain is a tuple of source IP address, destination IP address, source port, and destination port. Various rules are then attached to the header so that packet information is matched to a header and corresponding rules to detect an intrusion.

Anomaly detection systems rely on detecting shift in the normal traffic patterns of the network. A system based on SNMP MIB variables use statistical techniques to detect network anomalies [24]. A network management system is widely deployed in the

Internet and is effectively used for intrusion detection. Consider the ping flood attack in which many ICMP echo request packets are sent to the target. The SNMP ICMP MIB group has a variable icmpInEchos, which shows the sudden increase in its count during the ping flood attack. During the UDP flood attack, SNMP UDP MIB group's udpInDatagrams shows a similar increase in its count. To detect localized variations in important MIB variables, a time series is segmented in small sub-time series which are compared to the normal profiles. In a DDoS attack, variations are so intense that averaging the time series on properly chosen time intervals enables anomaly detection.

The above are network based intrusion detection systems, but intrusion detection can also be performed at a host. A data mining based approach is one such method [25]. Datasets consisting of normal and abnormal data points are gathered and fed to a classification algorithm to obtain classifiers. These classifiers once trained on the training datasets are then used to find abnormal data points. D-WARD [26] is an intrusion detection system to be installed at the network edges to detect attack sources. It monitors network traffic rates to determine asymmetry in the traffic rate. Typically in a DDoS attack like the SYN attack, there are more SYN packets leaving the network as compared to ACK packets entering the network. D-WARD emphasized on stopping the attacks close to the sources so that network congestion is reduced. Attack traffic in general even affects traffic not intended to the victim, and thus D-WARD also minimizes the collateral damage from the attack.

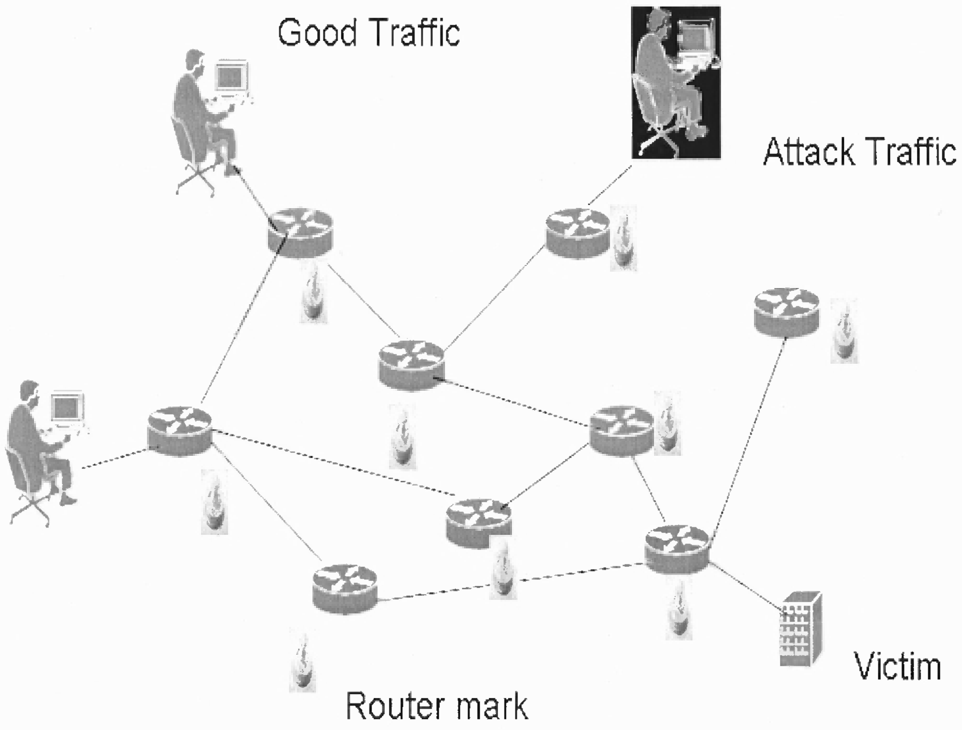


Figure 1.5 Conceptual diagram of the PPM scheme where each router marks packets probabilistically so that victim can reconstruct the entire path from the source router.

Intrusion response systems are required to find the source of the attack to stop the attack. Blocking the attack traffic is sometimes done manually by contacting network administrators who change the filtering policies to drop the attack traffic at routers. If an attacker is using source IP address spoofing, manual filtering is not useful and schemes like IP traceback are required. IP traceback traces the IP packets back to their sources and helps reveal attack sources [27][28]. In probabilistic packet marking (PPM) [29], routers mark their addresses on packets that traverse through them. Packets are selected randomly with some fixed probability of marking. A victim upon receiving many packets can construct the route back to the sources by reading router marks. Router vendors need to enable the marking scheme, and so ISP participation is required. This scheme does not require additional bandwidth overhead, which is an important advantage of this scheme.

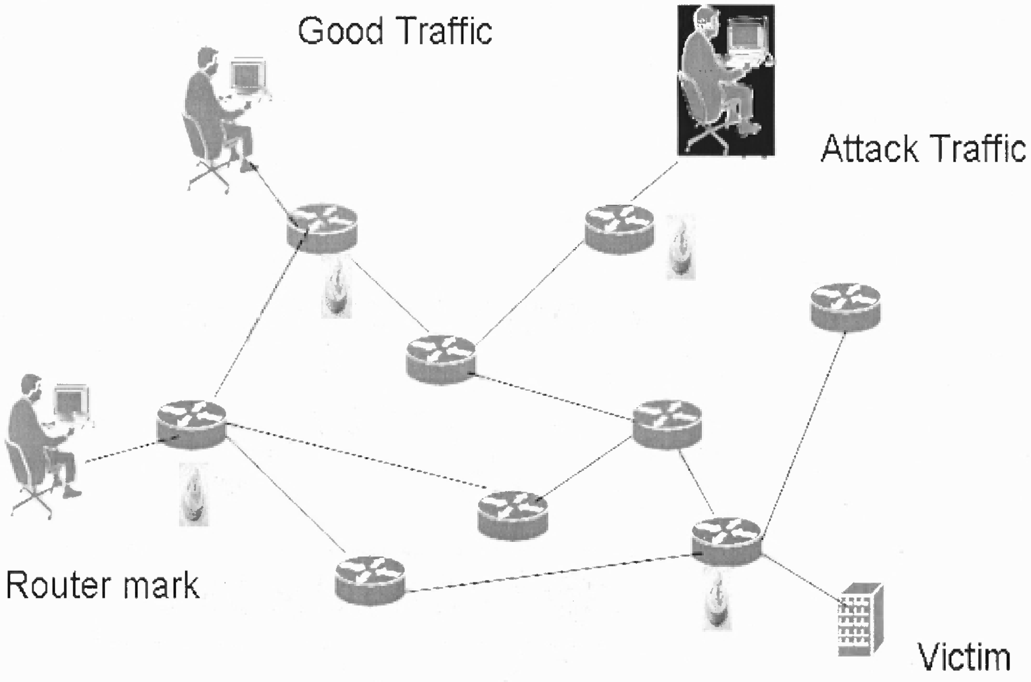


Figure 1.6 Conceptual diagram of the DPM scheme where only edge routers mark all packets so that the victim can determine the edge router from which the attack is initiated.

In a contrast, a scheme referred to as deterministic packet marking (DPM) [30], only marks packets passing through edge routers of the network. At the victim, a table is maintained for mapping between source addresses and router interface addresses. This facilitates the reconstruction and identification of source of the packets.

ICMP traceback, referred to as iTrace [31], requires a router sending an ICMP traceback message for 1 in every 20000 packets to a destination address which is the same as the destination address of the selected packet. The traceback message consists of the timestamp, the next hop, the previous hop, and the packet payload information to which the ICMP traceback message is sent. The TTL field of the packet payload field is set to 255. During the attack, the victim can construct routes that the packets are traversing based on the ICMP messages it has received. This method requires separate

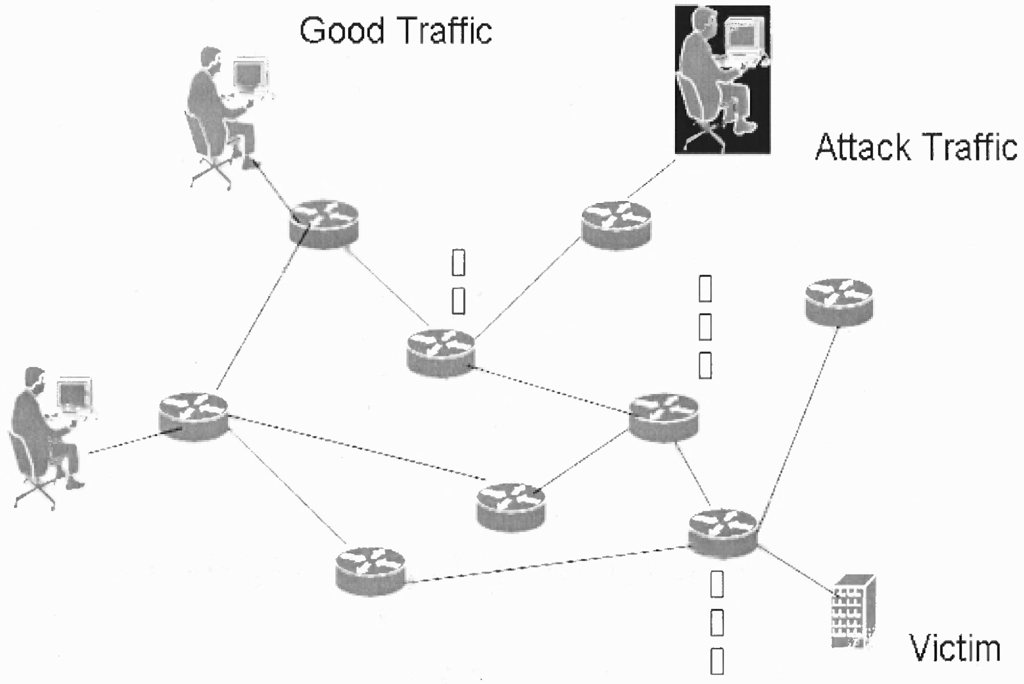
ICMP messages to be sent during the attack, and thus extra bandwidth is needed. If the DDoS attack is severe, the iTrace packets may be dropped. To prevent dropping of iTrace packets, routers should implement complex scheduling mechanisms that give higher priority to iTrace packets.

IP traceback schemes which do not use marking techniques are discussed next. Centertrack is a completely different IP traceback method in which all the edge routers route traffic via an overlay network to central tracking routers during the attack [32]. The source of the attack traffic is logically one hop away from the central routers by virtua of the overlay network. ISP involvement is high in this method as ISP has to implement the overlay system to divert traffic to central tracking routers and block malicious traffic. Hash based IP traceback, referred to as source path isolation engine (SPIE), saves partial packet information of packets passing through a router [33]. It uses several hash functions and bloom filters to optimize storage of packet information. As the bloom filter becomes 70% full, it is archived and another one is used. The routers participating in this scheme are called data generation agents, and the network is divided into several regions. In every region, the SPIE collection and reduction agents connect to all data generation agents to query for packet information during the attack. Querying collection and reduction agents reveals the source of the attack packets. Controlled flooding is another IP traceback technique; it is feasible only during a DoS attack. It relies on the fact that during the attack network links should be congested, and so flooding these links with more traffic would increase the packet drop rates on that link. The process of flooding is repeated to next hop links until the source of the attack is reached. This proposal requires the ability to flood a link when a host itself is under attack and it also contributes to the attack traffic making it a rather controversial idea.

Intrusion mitigation systems defend against the on-going attack. Adaptive Queue Management (AQM) schemes aim to detect and regulate misbehaving flows in the Internet, i.e., flows which do not follow congestion control, such as UDP flows. AQM schemes monitor the router buffer to detect misbehaving flows, and mark or drop packets that belong to these flows. The Adaptive Virtual Queue (AVQ) scheme [34] is described since AVQ is a relatively novel scheme that outperforms most of the other AQM schemes. AVQ works as follows: a router with capacity C maintains a virtual queue with capacity $C^* \leq C$ and queue size equal to the actual queue size. Upon arrival of a real packet, a fake packet is added to the virtual queue if there is a space in the queue. If the fake packet is dropped from the queue, then a real packet is either dropped or marked to notify the sender of the congestion. The virtual queue capacity is updated on each packet arrival according to the following equation:

$$C^* = \alpha(\gamma C - \lambda) \tag{1.1}$$

where λ denotes the packet arrival rate at the link, γ is the desired link utilization, and α is the smoothing parameter. Thus, in brief, the idea of AVQ is to mark or drop as a function of the link utilization: if the utilization is high, the marking probability is high. The pushback scheme [35] has also been proposed to mitigate the impact of DDoS attacks, but its approach is different from an AQM scheme. Pushback, also known as aggregate congestion control scheme (ACC), drops DDoS attack traffic by detecting the attack and sends signals to drop the attack traffic closer to the source. The rationale behind the pushback scheme is that the attack traffic has a unique signature in an attack, where the signature consists of identifiers such as port numbers, IP addresses, and IP prefixes. By detecting a signature in the aggregate attack traffic, upstream routers can be instructed to rate-limit flows that match the signature. A router in a pushback scheme has two components: a local ACC mechanism and a pushback mechanism.



▯ : Attack Packets are dropped by the pushback routers

Figure 1.7 Conceptual diagram of the pushback scheme where a router drops attack traffic based on the attack aggregate signature; it also requests upstream routers to perform attack filtering.

The local ACC mechanism is invoked if the packet loss percentage exceeds a threshold of 10%. It then tries to determine the aggregate congestion signature and correspondingly tries to rate limit the aggregate traffic. If the rate limiting does not reduce the arrival rate of the attack traffic below a pre-defined target rate, ACC invokes the pushback mechanism which sends pushback messages to the upstream routers to filter the attack traffic. By repeating this scheme upstream, pushback aims at rate-limiting the attack traffic at the source network. Throttling is another approach to defend web servers from a DDoS attack. It uses max-min fairness algorithm to compute rate to drop excess traffic. Upstream routers also participate in the scheme so as to drop the attack traffic near the source. There have been several ways to mitigate specific DDoS attacks like SYN attacks. A technique, referred to as a SYN cookie, avoids giving server resources to the SYN packet until a SYN/ACK is received [36].

1.6 Defenses For Low Rate DoS Attacks

Some countermeasures to mitigate the low rate DoS attacks in the Internet have been reported although none of them has made a comprehensive attempt to address such attacks with IP address spoofing.

The autocorrelation and dynamic time warping algorithm [37] relies on the periodic property of the attack traffic to detect the low rate DoS attacks. It proposes a deficit round robin algorithm to filter the attack flows; however it fails to drop attack packets when the attacker uses the continuous cycle and randomized IP address spoofing as each attack flow is a combination of multiple flows and each will be treated as a new flow. Thus, the attacker can easily evade the filtering mechanism. The randomization of RTO [38] proposed to mitigate the low rate TCP DoS attack cannot defend against the RoQ attack, which targets the network element rather than the end host. The main idea reported in [38] is to randomize the minimum RTO instead of setting it to be one second should be randomized. However, it ignores the advantages of having the minimum RTO of one second, which was chosen as a balance between an aggressive value and a conservative value.

The Collaborative Detection and Filtering scheme proposed in [39] involves cooperation among routers to throttle and push the attack traffic towards the source. They rely on the autocorrelation property to distinguish the periodic behavior of the attack traffic from the legitimate traffic. Thus, it needs extra DSP hardware for implementation, and extra memory to store the flow information of the attack packets to be dropped. The scheme maintains a malicious flow table and a suspicious flow table, which can be overwhelmed under the presence of the IP address spoofing. The novel part is the cumulative traffic spectrum that can distinguish traffic with and without the attack. In the traffic spectrum with the attack, the energy is found more localized in lower frequencies. The attacker can randomize the attack parameters in the RoQ attack. This work does not provide clear guidelines to activate attack packets filtering.

The wavelet based approach identifies the abnormal change in the incoming traffic rate and the outgoing acknowledgments to detect the presence of low rate TCP DoS attacks [13]. This approach cannot identify the RoQ attack. The wavelet approach does not filter the attack traffic. The buffer sizing method [40] proposes to regulate the buffer size so that the attack flows can be detected as high rate flows by the RED-PD filter, and subsequently dropped. This work does not consider the RoQ attack in their analysis; it is difficult for this approach to detect the RoQ attack because the average rate of an RoQ attack is very low. The buffer sizing scheme fails if an attacker uses IP address spoofing as the high rate attack flow is a combination of multiple low rate individual flows. A modified AQM scheme referred to as HAWK [41] works by identifying bursty flows on short timescales, but lacks good filtering mechanisms to block the attack flows that can use the IP address spoofing. This approach can penalize the legitimate short bursty flows, thereby reducing their throughput. A filtering scheme similar to HAWK is proposed to estimate the bursty flows on shorter and longer time scales [42]. The main idea is to use per-TCP flow rate as the normal rate, and anything above that rate is considered abnormal. The identification of flow rates is done online. On a shorter time scale, it is very easy to penalize a normal flow as a bursty flow. The proposal did not consider the random IP address spoofing, where every packet may have a new flow id. They used a very complex filtering technique. With the use of the IP address spoofing, it is difficult to come up with the notion of a flow as the number of packets per flow can be randomized in any fashion during every ON period.

The rest of the dissertation is organized as follows. Chapter 2 covers the discussion of the effects of low rate DoS attacks on VoIP traffic. Chapter 3 presents a simple time difference technique to detect the low rate DoS attacks. Chapter 4 describes the proposed mitigation technique against the RoQ attacks, which use IP address spoofing. Chapter 5 presents the proposed quiet attack, which uses several novel techniques to maintain stealthiness, and Chapter 6 discusses a variant of the quiet attack, referred to as the perfect attack. Concluding remarks along with a discussion on future works are finally given in Chapter 7.

CHAPTER 2

LOW RATE DOS ATTACKS AND VOIP TRAFFIC

2.1 Introduction

Voice over IP (VoIP) technology has emerged as one of the killer application on the Internet, as web traffic did in early 90's. VoIP is a QoS sensitive application, and the best effort service model of the Internet is not designed to carry such traffic.

To address the challenges of sending voice over the Internet, a few intelligent protocols have been devised for carrying these data. At the transport layer, UDP, which is a connectionless protocol, has been the choice to transport voice. UDP serves better than TCP for voice, as TCP's "acknowledge/rate adjustment" concept is applicable to the reliable and network-aware data delivery model, but is unfit for voice traffic, which needs data at the "right-time". VoIP applications use RTP-UDP protocols at the transport layer, where RTP [43] is a real time transport protocol that provides some basic transport layer functions. RTP uses RTCP [43] conjunctionally to provide feedback about statistics of the received data back to the sender. Four major protocols ITU-T H.323, IETF SIP, MGCP, and MEGACO/H.248 [44] have been proposed for call setup. Naturally, one should be concerned about security as voice packets are sent over a network, like the Internet, which is not controlled by any single entity. Several challenges for VoIP security are highlighted in [45]. VoIP imposes some tight budget requirements like delay to be less than 150msec and the packet loss to be less than 1%. DoS attacks have been rampaging the Internet and they are bound to cause an impact on VoIP traffic too. It is also obvious that high rate UDP will affect VoIP traffic, but the literature lacks a systematic study of the effects of the low rate DoS attacks on VoIP traffic, as they are typically known to cause damage to TCP traffic. In addition, the low rate UDP attack does not continuously send UDP traffic, and so it is interesting to study the behavior of such periodic high rate UDP flows on legitimate UDP VoIP flows. This Chapter thus investigates the effects of the new low rate DoS attacks on this delay sensitive VoIP traffic, as the Internet traffic is a mix of UDP and TCP traffic [46], [47]. Forward error correction (FEC), which is used to recover the lost VoIP packets, is also considered in the analysis presented in this Chapter. This Chapter studies the performance of VoIP calls by using ITU-T Emodel rather than providing abstract numbers of packet loss, and provides comprehensive analysis.

2.2 Problem Description

In this Section, evaluation methodology to assess the impact of low rate DoS attacks on VoIP traffic is presented. A low rate DoS attack model is considered in this study. Figure 1.4 shows the variation of the periodicity of the low rate attack models. To make an easy distinction between the two attacks, an attack with time period less than or equal to one second is classified as a low rate DoS attack while an attack with a time period greater than one second is classified as an RoQ attack. The RoQ attack is an attack whose objective is to reduce the quality of service received by an application. It may not cause denial of service, which is not its goal, but will lead to reduction in quality of service. It is important to note that the reduction of quality will be different for different applications. For VoIP, standard models are used to determine quality rating and depict reduction of quality. As mentioned before, Internet traffic is constituted by traffic generated by a variety of applications running on top of TCP or UDP [46], [47]. Any attack on the Internet will have some impact on both the UDP and the TCP traffic. Since UDP does not have any congestion control or accountability mechanism, it pumps data at the set rate. VoIP traffic performance is characterized by three parameters: the delay, the jitter, and the packet loss. All of these parameters are closely tied to the time varying network state. We have decided to evaluate the impacts of the periodic attacks on VoIP traffic for the following reasons.

Low average rate attacks rely on sending high rate, bursts of packets with small duration to fill up the targeted router queue, and thereby deny access to the competing flows. These sudden transients are bound to cause abrupt changes in the jitter/packet loss/delay of VoIP traffic. The length of the burst can be tuned to maximize its impact on the jitter/packet loss/delay. The period of the attack can be kept further high to attain a low average rate. We conjecture that the low rate attack UDP traffic will affect the low rate UDP (VoIP) traffic owing to the fundamental property of UDP of being non-adaptive.

In this Chapter, the goal is to characterize these changes due to a low rate DoS and RoQ attack, and decide if these attacks possess sufficient potential to force a VoIP call, which is in different ranges [48], [49] of quality, into unsatisfactory regions of operation. Tremendous research efforts have focused on minimizing the end-to-end delay, the packet loss, and the packetization delay [43] of a VoIP flow, and so it is important to analyze effects of these transients due to the attack. On the links where calls are experiencing poor quality, due to the network conditions, these attacks can be detrimental [50]. The VoIP delay can consist of different components; this study focuses only on the propagation, transmission and queuing delay in the network, and the deviation of this segment of the delay due to the attack. Two commonly used models are used to analyze VoIP packet loss to understand the behavior of packet loss due to the attack.

Random Loss Model: In this model, a packet is lost according to the simple probability p, which is referred to as the average loss probability in this study. If X_n is a boolean random variable where 1 means packet lost and 0 means no loss, then $p = E[X_n]$.

Gilbert Bursty Loss Model: Many studies about the packet loss nature have found that the Internet packet loss is bursty [51], [52]. This proposed model is widely used to describe bursty losses. The Gilbert loss model is characterized by two probabilities called conditional loss probability $P_c = P[X_{n+1} = 1 \mid X_n = 1]$ and unconditional loss probability $P_u = E[X_n]$. The conditional loss probability quantifies the burstiness of the packet loss. A state model representation with the transition probabilities p and q is shown in Figure 2.1.

$$q = 1 - P_c$$

$$p = {P_u q}\big/{1 - P_u}$$

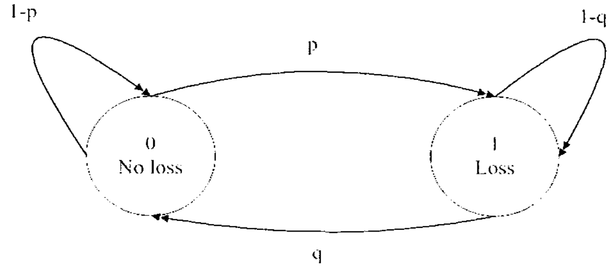


Figure 2.1 The Gilbert bursty loss model.

2.3 Error Correction and E-model

To mitigate packet loss in the Internet, Forward Error Correction (FEC) [53] is used. A media independent FEC recovers lost packets in a bit-exact form, and the commonly used code is called Reed-Solomon, $RS(n,k)$, where n and k are the number of all and non-FEC data units, respectively. The basic idea to recover the lost packets is to receive at least k packets from a block of n packets. The $RS(n,k)$ code works by piggybacking the forward packet information in the piggybacked packet, and so even if one packet is lost, its contents can be obtained from the following piggybacked packet. There is another media dependent FEC known as Low Bit Redundancy (LBR) , which works by encoding a lower bit rate version of the same audio in subsequent packets. So, even if the higher bit rate audio packets are lost, the lower bit rate audio packets can be used to reconstruct the voice. It was shown in [54] that $RS(n,k)$ FEC performs better than LBR, and so $RS(3,2)$ FEC was used in this analysis. Two VoIP codecs commonly used in the Internet were chosen, with following specifications: G711 $RS(3,2)$ FEC with a media bit rate of 96Kbps and G729a $RS(3,2)$ FEC with a media bit rate of 12Kbps. For $RS(n,k)$, the probability to recover packets given the average loss probability p is [55]:

$$P_{FEC} = p(1-\sum_{i=k}^{n-1}\binom{n-1}{i}(1-p)^{i}p^{n-i-1}) \tag{2.1}$$

In this work, $n=3$ and $k=2$. It is well-known that performance of VoIP decreases significantly at the same average loss rate when the loss is bursty [54]. We consider two other metrics apart from the conditional loss probability of the Gilbert model to analyze the loss burstiness, as suggested by the IP performance metrics (IPPM) working group: consecutive packet loss run length and interloss distance (ILD) [56]. A consecutive loss run length is defined as the length of consecutive packet losses. The ILD determines the distance between two consecutive loss run lengths. Consider a binary sequence 0110001111000 of 1s and 0s, where 1 indicates packet lost, and 0 indicates packet received. The consecutive loss runs in this example are two and four. In addition, ILD is three in the same example. The ILD metric is important because small ILD values affect the final voice quality. Small ILD values also have a considerable negative impact on the FEC performance [56], which is comprehensible, considering the characteristics of FEC, which needs at least k packets in a group of n packets to recover lost packets. A small ILD value means more consecutive losses spaced close to each other.

Mean Opinion Score (MOS) and R-factor are used to evaluate the quality of voice. MOS is known as the arithmetic average of opinion; the R-factor is computed from formulas supplied by the E-model. MOS is related to the R-factor by following formula:

$$MOS = 1 + 0.035R + 7 \times 10^{-6} R(R-60)(100-R) \tag{2.2}$$

Cole and Rosenbluth [57] provided a simplified formula to estimate the R-factor:

$$R = 94.2 - I_d - I_e \tag{2.3}$$

where I_d and I_e are delay and loss impairment. The delay and loss impairment can be easily obtained from the graphs provided in [57] by curve fitting. A similar approach to evaluate MOS and R-factor is found in several studies [54]. The two previously described codec's are chosen because there are established results that map the loss probability to the corresponding MOS as shown in Figure 2.2 where T is the packet size such as 10ms/packet for G711. The R-factor and MOS calculation procedures are described in later sections when results are evaluated. R-factor in the range of 90 to 100 is considered best while that between 60 and 50 is considered poor or the result of a DoS. Thus, anything between 90 and 60 is considered the result of a RoQ.

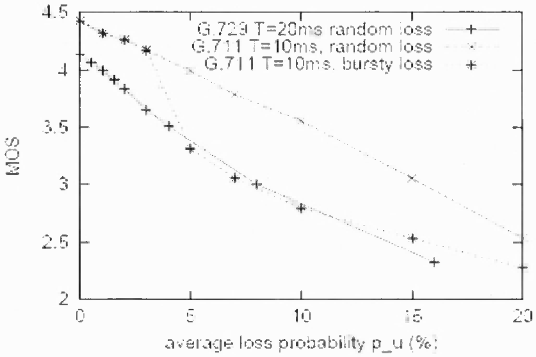


Figure 2.2 MOS behaviors from the E-model.

2.4 Simulation Results

The network model used to test effects of the low rate DoS and RoQ attacks on VoIP traffic is described in the beginning. This section describes the experimental setup by using ns2 for this study. To conduct any network study based on simulations, it is important to have correct simulation settings [58] or else the results will not be insightful. The network model uses a typical dumbbell topology as shown in Figure 2.3. The network topology in Figure 2.4 shows that the traffic passes through a bottleneck link of 10Mbps from the access links of 100Mbps. All the access links have random delays uniformly distributed in the 50-100ms range [58]. The access links connecting the sink agents to the bottleneck link has a link delay of 10ms. The bottleneck link uses a simple droptail queuing policy with a queue length of 500 packets.

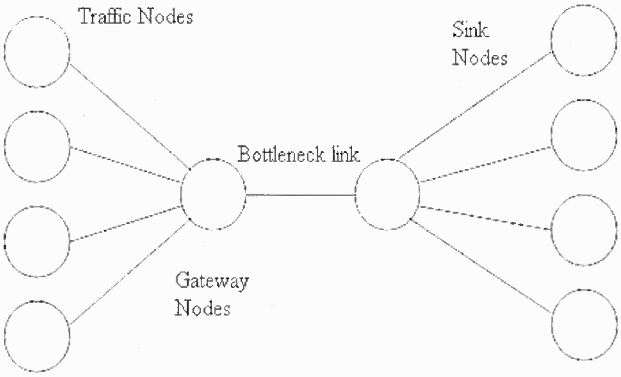


Figure 2.3 The network topology.

The queue size is kept twice the bandwidth delay product (BDP) as it was suggested in [59] to keep the queue size between 1.25 BDP and 2 BDP for reasonable response times for droptail queues. The droptail policy is used because it is widely used in most of the routers. By using a droptail policy, packets will only be dropped when the queue is full. The network traffic is composed of five FTP flows, many HTTP flows, five VoIP flows, and a single attack flow. The FTP flows use the TCP selective acknowledgment algorithm, which is widely deployed in the Internet. Important ns2 full-TCP parameters adopted are a maximum segment size of 1460 bytes, window size of 50 packets, minimum RTO of 1sec, and the rest of the parameters are defaults set by ns-2. We have used the packmime HTTP traffic generator [60] in the network model. There are two packmime client and server instances in the diagram. The packmime traffic generator data is based on the real world traces of the Internet traffic. The packmime client and server agents initiate HTTP connections by using the user-specified average rate parameter. We keep the average rate as ten connections per second. The delaybox is used to provide per flow dropping probability, round trip times, and bottleneck link speeds. The per flow dropping probability is kept zero and the server bandwidth is uniformly distributed from 1 to 20 Mbps. The test-packmime-delaybox.tcl script [61] was modified for simulations presented in this Chapter.

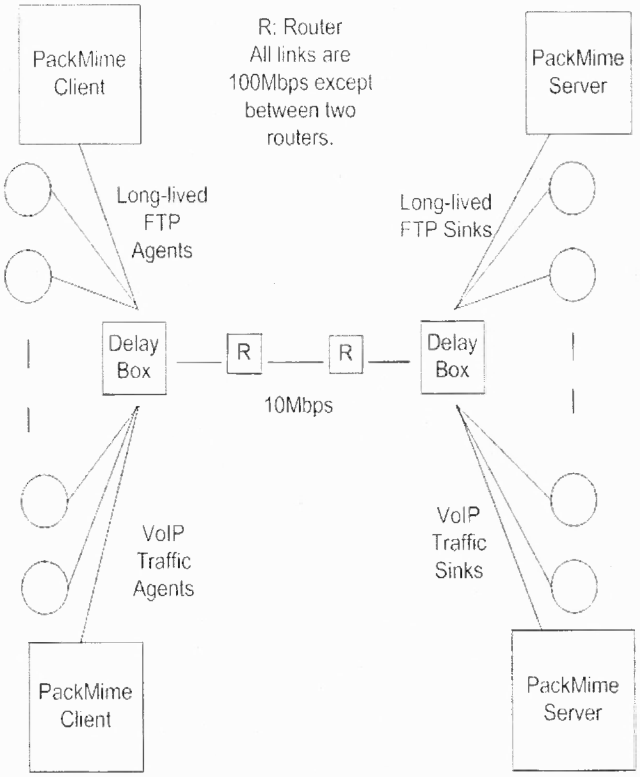


Figure 2.4 Simulation topology.

The VoIP traffic is modeled by using G711 and G729a codecs with RS (3,2) FEC. The rate of a G711 flow is 96Kbps with packet size equal to 10ms and that of a G729a is 12Kbps with packet size equal to 20ms. The simulation runs for 600 seconds with VoIP flows being active for 180 seconds. The attack traffic is introduced 10 seconds later in the same network that passes through the bottleneck link. Figure 2.5 shows different attack scenarios that are considered, where T is the time period, t the burst period, and R the burst rate of the attack. The first two attack scenarios are shrew attacks with the attack rates equal to and double that of the bottleneck capacity in the first and second case, respectively. The next four are RoQ attack scenarios with various attack rates and attack burst lengths. An initial simulation is done without attack and referred to as the baseline scenario. In the baseline scenario, packet loss is zero for all VoIP flows.

Scenario	T	t	R
	Sec	Sec	Mbps
1	1	0.3	10
2	1	0.3	20
3	5	0.3	10
4	5	0.3	20
5	5	1.2	10
6	5	1.2	20

Figure 2.5 The attack scenarios.

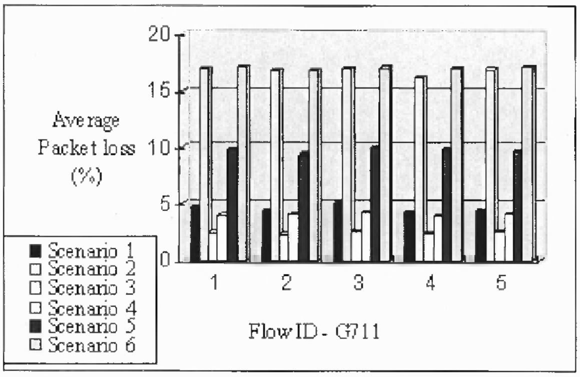


Figure 2.6 The packet loss for the G711 codec.

The one-way delays for VoIP calls in simulations and corresponding delay impairment values are obtained by curve fitting as 103, 98, 80, 77, and 99 msec, and 2.5, 2.4, 2.2, 2.2, and 2.4, respectively [57]. The delay impairment values are such that their impact on the *R*-factor is minuscule, and so effects of the attack on the loss impairment factor can be scrutinized, as low rate DoS and RoQ attacks increase the packet loss. These results will provide the lower bound on the effects of low rate DoS and RoQ attacks on the VoIP quality, and if the delay is high, these attacks will cause a higher degree of impact. The loss impairment can be obtained from the standard loss mapping between the average loss probability and MOS for G.711 and G.729 codecs provided by ITU-T's E-model by curve fitting, a technique used in several studies [57].

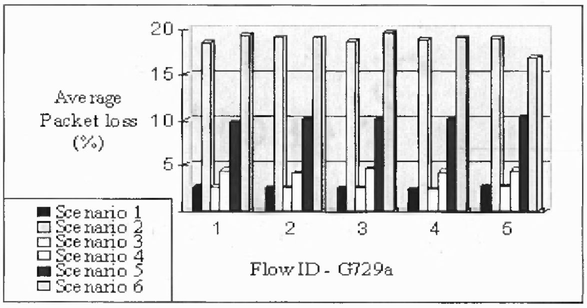


Figure 2.7 The packet loss for the G729a codec.

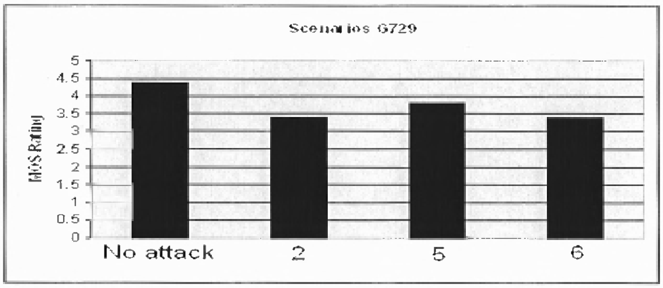


Figure 2.8 The average MOS for each scenario using G729a codec VoIP flows.

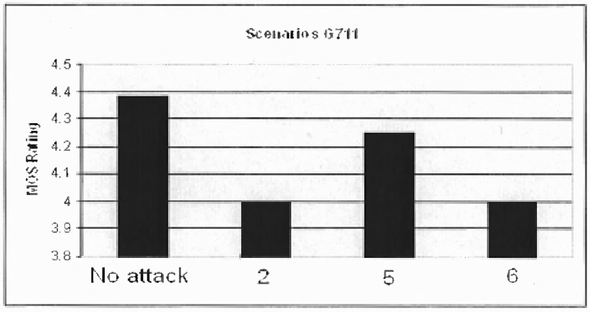
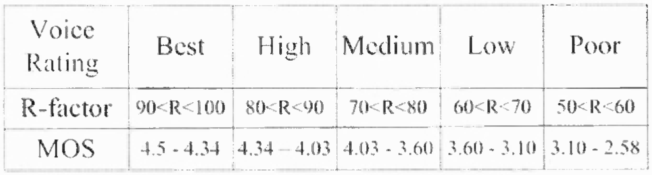


Figure 2.9 The average MOS for each scenario using G711 codec VoIP flows.

Figures 2.6 and 2.7 show packet loss results for the G711 codec and the G729a codec, respectively. The average packet loss is almost similar for both codecs in each case, but is high when the rate is double that of the bottleneck capacity. The most damaging attack scenarios are 2, 5, and 6. The last two attack scenarios demonstrate that RoQ attacks can be equally lethal as compared to the shrew attack when they have longer burst lengths. These RoQ attacks with longer burst lengths compromise attack potency by sending more attack traffic. Attack potency is a metric defined for RoQ attacks; it represents a tradeoff between the attack detectability and the attack effects on the legitimate traffic. With higher attack traffic, the attack effect on the legitimate traffic increases, but the attack potency decreases as the attack can be more easily detected and mitigated. The RoQ attack and shrew attack can always use botnets or IP address spoofing thereby distributing the attack traffic during each burst among different IP addresses. We now substitute the average loss probability (p) obtained from the simulations in (1) to calculate the probability of loss with FEC incorporated (PFEC). That is, PFEC takes into account of error correction. This improved loss probability due to FEC is then used to compute the loss impairment factor I_e. Finally, MOS can be obtained from Figure 2.2 which gives standard mapping for G711 and G729a codecs. We have plotted the average values of the MOS for scenarios 2, 5, and 6 using the G729a and G711 codecs in Figures 2.8 and 2.9, respectively. The different VoIP quality ratings as per E-model are shown in Figure 2.10. Scenarios 1, 3, and 4 are not shown in Figures 2.8 and 2.9 because the corresponding packet loss is not very high, and FEC can recover the losses thereby the voice quality can be classified as high or best. The voice quality is medium for all tested scenarios according to Figure 2.8 for G729a VoIP flows, indicating reduction of quality.

The voice quality is also medium for scenarios 2 and 6 of G711 VoIP flows. In other words, the voice quality is at an acceptable service level, and it can be concluded that the low rate DoS and RoQ attacks do cause a reduction of quality according to the E-model quality rating scale. In Section 2.3, it was decided that voice ratings of high, medium, and low correspond to reduction of quality. If FEC was not used, then the attack would have caused denial-of-service, i.e., voice rating of poor or below.



Voice Rating	Best	High	Medium	Low	Poor
R-factor	90<R<100	80<R<90	70<R<80	60<R<70	50<R<60
MOS	4.5 - 4.34	4.34 – 4.03	4.03 - 3.60	3.60 - 3.10	3.10 - 2.58

Figure 2.10 The E-model VoIP quality ratings.

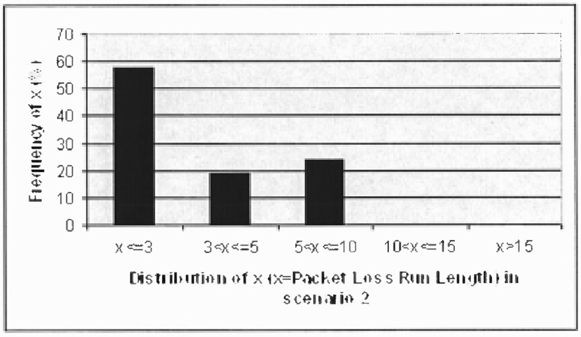


Figure 2.11 Packet loss run length for G711 codec for scenario 2.

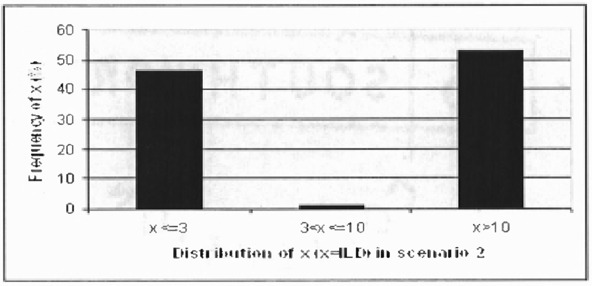


Figure 2.12 ILD for G711 codec for scenario 2.

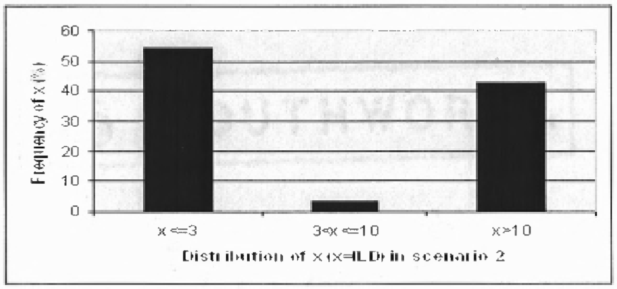


Figure 2.13 ILD for G729a codec for scenario 2.

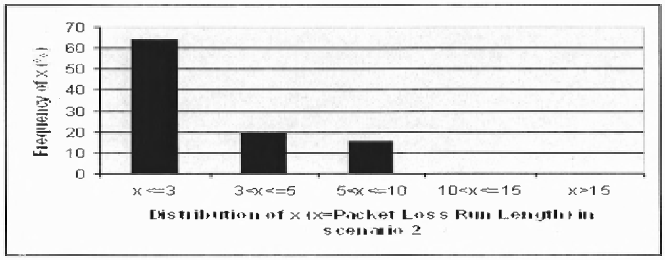


Figure 2.14 Packet loss run length for G729a codec for scenario 2.

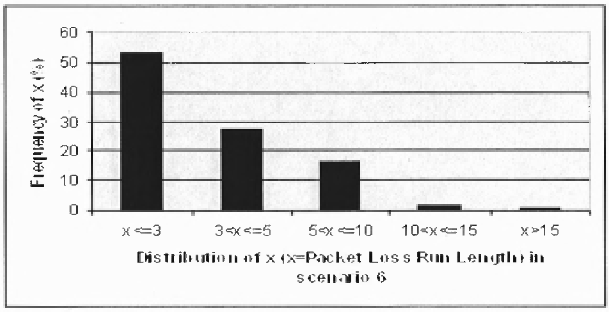


Figure 2.15 Packet loss run length for G711 codec for scenario 6.

We next investigate the effect of low rate DoS attacks on the burstiness of the VoIP packet loss by evaluating the conditional loss probability of the Gilbert loss model, consecutive packet loss run lengths, and ILD for scenario 6 as it has high burst length and burst rate, and for scenario 2 as it has short time period of 1 sec and high burst rate. The average conditional loss probabilities of each VoIP flow in scenarios 2 and 6 is 9.8% and

12.88%, respectively, for the G729 codec case. They are 7.3% and 9.1% for scenarios 2 and 6, respectively, for the G711 codec case. The conditional loss probability is much higher in scenario 6, thus implying that the burst length does play a major role; so a RoQ attack with higher burst lengths will increase the bursty losses of the VoIP flows. This observation is consistent with characteristics of the UDP protocol, which is not network adaptive, and thus high rate UDP bursts with long burst lengths will force packet drops spaced closed to each other. It is always possible to make burst length much longer than 1.2 sec but the objective of the RoQ attack is not to cause denial of service, though in reality, an attacker can do so. The effect of conditional loss probabilities on the final loss probabilities with FEC has been studied. For smaller conditional loss probabilities, the Gilbert model and random loss model are concluded to be similar [54]. The conditional loss probabilities were found to be much less than 30% as reported in [54]. So, further analysis using conditional loss probability is not necessary, but ILD and packet loss run lengths results are still presented which confirm that the burstiness of the VoIP packet loss will increase if an attacker chooses to use higher burst length.

ILD and packet loss run lengths are other indicators of burstiness, which are shown in Figures 2.11-2.18. Packet loss run lengths higher than 5 were not rare, and that of higher frequency of ILD less than 3 were not rare either. Both of these factors can severely affect VoIP quality of service [54]. The percentage of ILD fewer than three is high as compared to some experiments reported in [54]. This indicates that the losses are high enough to drop voice frames even with the use of FEC, thereby leading to dropped calls if a user becomes unsatisfied. The ILD and consecutive packet loss run lengths primarily depend on the attack burst length and the attack rate. We can see from scenario

2 in Figures 2.11-2.15 that the frequency of ILD less than 3 and packet loss run lengths greater than 5 is low, but it is high for scenario 6 as shown in Figures 2.16-2.19 for both G729a and G711 codec cases.

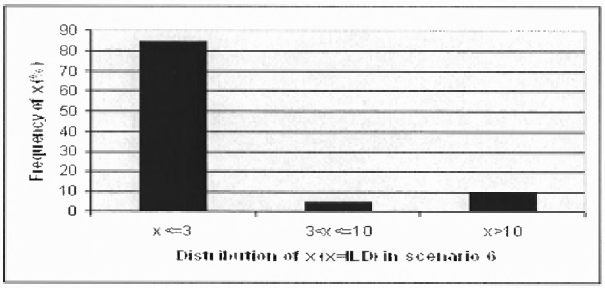


Figure 2.16 ILD for G711 codec for scenario 6.

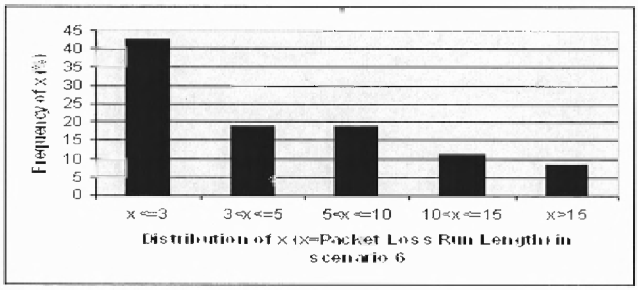


Figure 2.17 Packet loss run length for G729a codec for scenario 6.

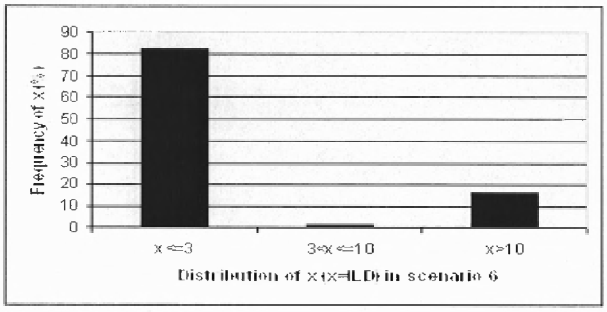


Figure 2.18 ILD for G729a codec for scenario 6.

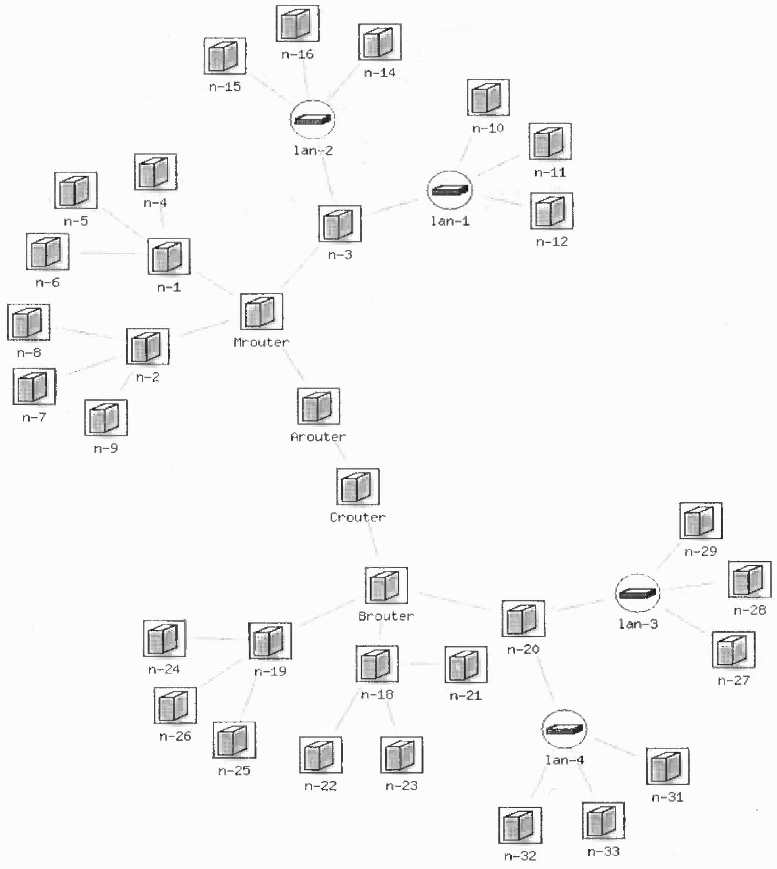


Figure 2.19 The deterlab topology snapshot.

Below are some critical observations about the packet loss from demonstrated results:

- The RoQ attacks with shorter burst lengths and burst rates fail to cause any damage in presence of FEC.
- The RoQ attacks with higher burst lengths and burst rates can lead to small ILD and high packet loss run lengths, thus leading to dropped calls, which can lead to denial of service.
- Low rate DoS attacks reduce VoIP quality in presence of FEC.
- The attack effect on the G729a codec is more severe as compared to G711 as it reduces the voice quality rating from Best to Low.
- Packet loss recovery schemes like FEC are required to recover from both low rate DoS and RoQ attacks.

It could have been possible to increase the attack parameters, particularly, burst length, but it would lead to exposure of the attacker, but effects of the low rate DoS and RoQ attack in the stealthy form were of more interest. Thus, demonstrated results will act as a lower bound on the effects of the low rate DoS and RoQ attacks on the VoIP traffic. Nevertheless, with representative attack scenario 6, it is demonstrated that these attacks can be more lethal by increasing the burst length.

2.5 Deterlab Experiments

The deterlab [62] is a testbed designed for network security research and is a collaborative effort of many universities. The important objective of the testbed is to allow experiments related to Internet attacks that are difficult to conduct in a normal testbed or over the Internet. This will help in developing robust systems and studying vulnerabilities of current protocols. We decided to evaluate the effects of the low rate DoS and RoQ attacks on the VoIP traffic by using the deterlab. Figure 2.19 shows the snapshot of a deterlab topology used for this experiment.

Each system is running a Red Hat Linux 9 OS. Conceptually, the deterlab topology is the same as that shown in Figure 2.3, which has a bottleneck link between two routers. Here, all links are 100Mbps with bottleneck link set to 10Mbps between Arouter and Crouter in Figure 2.19, and various delays obtained from a uniform distribution of 80-150ms along the access links. The Arouter and Crouter are running kernel level click modular router [63] with a drop tail queue size of 600 packets. The queue size is kept as twice the bandwidth delay product. All the routers use dual Pentium 4 Xeon processors with 2GB memory. The long-lived TCP flow is setup by using the ettcp program, and VoIP traffic is setup by using the D-ITG traffic generator [64] as a G729a codec VoIP flow. We have two VoIP and three long-lived TCP flows in the network. We also use the harpoon [65] traffic generator to create background network traffic similar to the real Internet traces. One node acts as a harpoon client sending on average 3 Mbps traffic throughout the experiment. We thus have 3 FTP and 2 VoIP flows sharing the remaining 7Mbps of the bottleneck link capacity in absence of the attack. The packet loss of VoIP traffic in absence of the attack is zero. We then introduce attack traffic based on six scenarios described earlier in Figure 2.5. Figure 2.20 shows the packet loss results, which indicate that the packet loss is slightly higher as compared to ns-2 simulation results for similar scenarios shown in Figure 2.7. The most damaging scenarios are 1, 2, 5, and 6. However, in the simulation results, the packet loss increased when the rate of the attack was doubled as shown in Figure 2.7, but increase in the rate had no prominent effect for testbed results. We conjecture that a high enough rate equal to that of the bottleneck was good enough to instigate a significant impact such that increasing the attack rate did not cause further damage. The corresponding average MOS value for each scenario is shown in Figure 2.21; it is obtained by performing similar analysis discussed in the previous section on simulation results. In scenarios 1 and 2, the MOS value is in the poor range, indicating a denial of service. In scenarios 5 and 6, the MOS value is in the low range, which indicates reduction of quality. These observations are consistent with the simulation results that the loss from some RoQ attacks can be recovered by using FEC, while some RoQ attacks with high burst rate and burst length can lead to reduction of quality. On the other hand, low rate DoS attacks can lead to poor VoIP quality according to the E-model quality rating scale. FEC is required to recover from losses; otherwise, VoIP users will be denied service because of the low rate DoS and RoQ attacks. Although high end PCs were used as routers in the deterlab tests, it can be assumed that the packet loss for VoIP flows might be little less on commercial routers with dedicated hardware and software used for efficient routing. Finally, it can be at least concluded that VoIP is indeed affected by low rate DoS and RoQ attacks as reconfirmed by results from the Deterlab testbed.

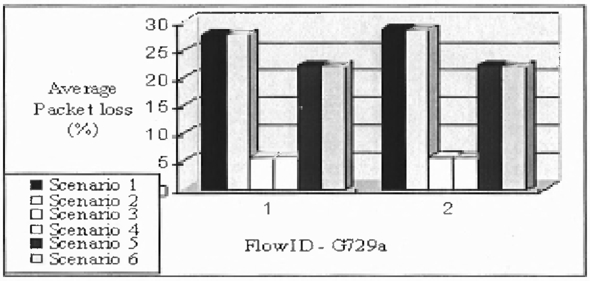


Figure 2.20 The packet loss on Deterlab for G729a codec VoIP flows.

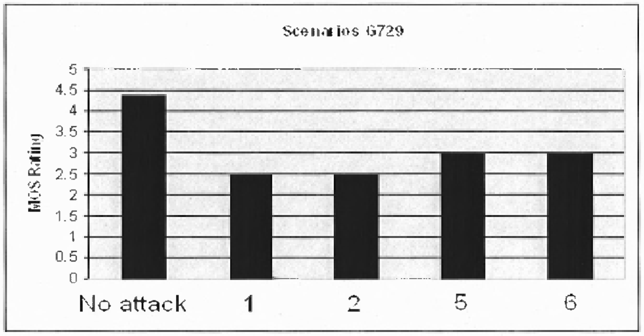


Figure 2.21 The average MOS for each scenario using G729a codec VoIP flows.

2.6 Summary

This chapter demonstrates that the low rate DoS and RoQ attacks not only pose a threat to the TCP traffic, but are also detrimental to the QoS sensitive UDP based applications like VoIP. The loss of packets is the main concern from these types of attacks. This study shows that these attacks pose serious threat to the Internet traffic and presses for immediate attention to deploy effective mitigation strategies against these attacks in the Internet.

CHAPTER 3

A SIMPLE DETECTION SCHEME

3.1 Approach and Methodology

A network scenario in Figure 3.1 is shown to explain where the proposed detection system is deployed. It shows edge routers connecting a local area network to the Internet with typical client server connections. Each edge router acts as an entry and exit point for traffic originating from that local area network; essentially all incoming and outgoing traffic will pass through this point. The proposed detection system can be deployed at the edge routers of a local area network in which the server is present. For illustrative purposes, it is assumed that all clients are outside the local area network in which the server is present, so that the detection system can monitor all flows connecting to the server.

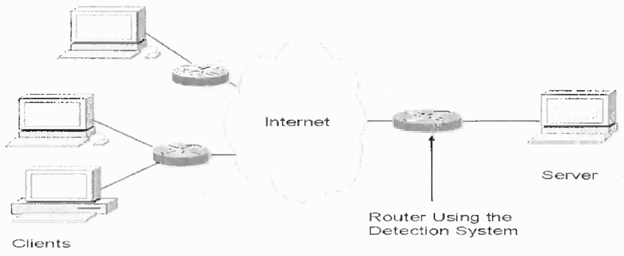


Figure 3.1 Proposed detection system deployed at an edge router.

Figure 3.2 shows the basic layout of the system. It has three basic blocks, namely, flow classifier, object module, and filter. Each block functions as follows:

- The flow classifier module classifies packets based on the flow ID by means of a combination of the IP source address, IP source port, IP destination address, and IP destination port. Flow information is obtained from these packets, and packets are forwarded as usual by the routing mechanism resulting in no additional delay apart from the lookup delay. The object module consists of various objects for each flow that is monitored. Detailed explanation is given in the next sub-section. A flow is monitored until it is considered normal.

- The filter is used to block flows that are identified as malicious by the object module.

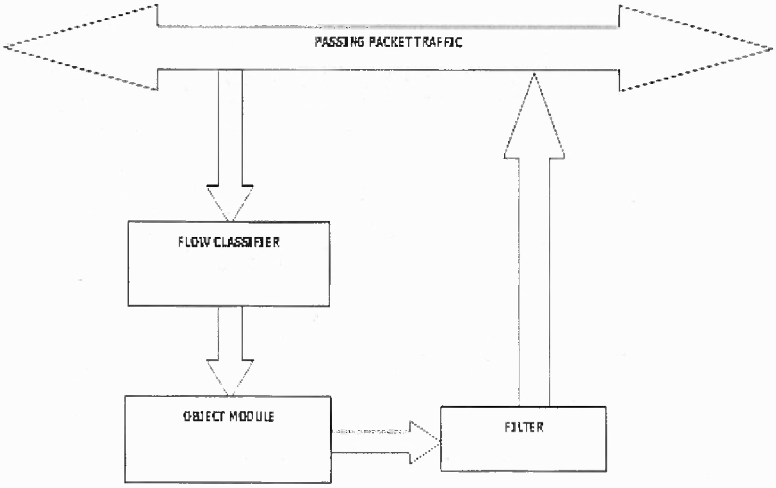


Figure 3.2 The detection system architecture.

Consider a DoS attack which tries to exploit protocol shortcomings. The object module maintains per flow information by creating objects per flow called flow objects. A thin data structure layer is designed to keep track of these flow objects. It maintains only those parameters which are exploited in the attack. This thin structure keeps separate track of information about flows classified as malicious. This information is then relayed to the filter module.

The flow objects maintain the arrival times of packets at the edge router in the pseudo transport layer. The malicious flow detection sub-module of the object module computes the time difference of consecutive packets of each flow. The sub-module computes the average high and average low of the time difference values. The average high value of the time difference repeats periodically for the attack flow; other flows do not exhibit this property. The malicious flow detection sub-module then estimates the burst length of a flow based on the packet arrival times. A flow exhibiting periodicity in the time difference graph is marked malicious as no legitimate flows will show such periodicity.

3.2 Results

NS2 is used to demonstrate the performance of the proposed detection scheme. The experimental topology is shown in Figure 3.3. As specified earlier, the time difference is computed for consecutive packets. Figures 3.4-3.6 show plots of this time difference for three different types of flows. Note that the attack traffic shows periodicity in the time difference while HTTP and FTP traffic lacks periodicity. This is the most important step in distinguishing the attack flow from other flows.

3.3 Major Drawbacks

The time difference technique uses a per-flow approach to store the arrival times of the packets belonging to each flow, and computes the inter-arrival times between the consecutive packets to detect periodicity. The attacker using IP address spoofing can easily deceive this simple per-flow approach as the time difference approach will not be able to detect periodicity in the attack flow, which is no longer a single flow.

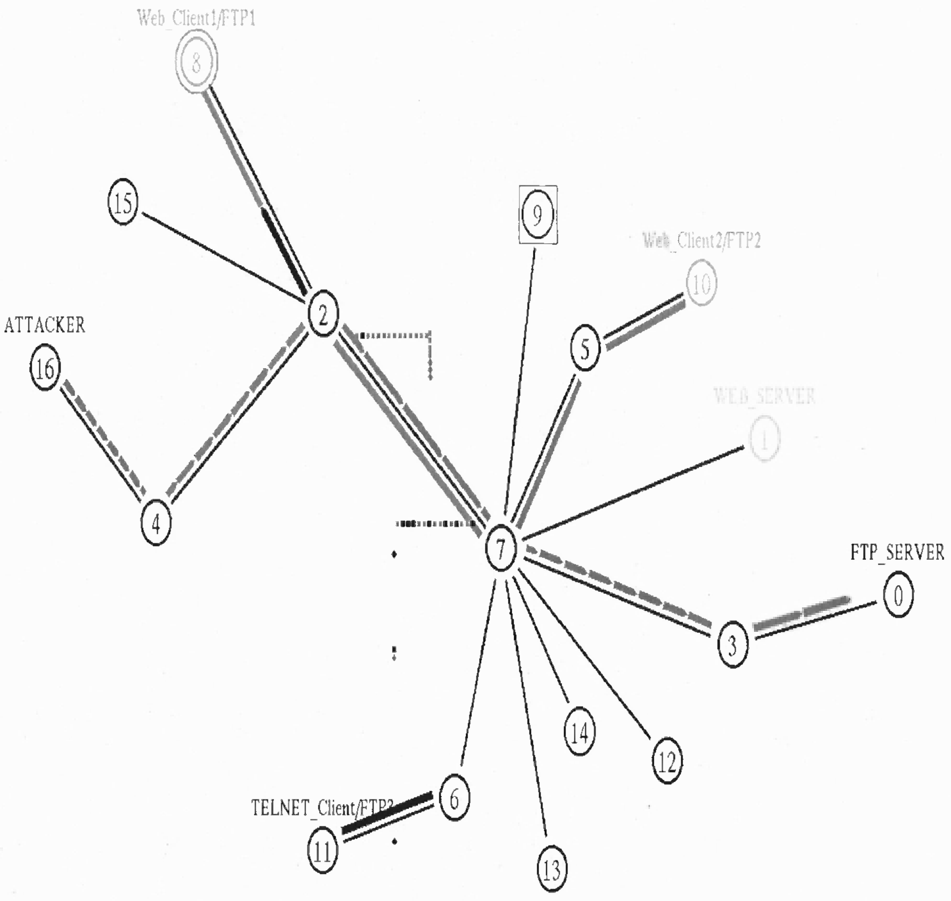


Figure 3.3 The simulation topology.

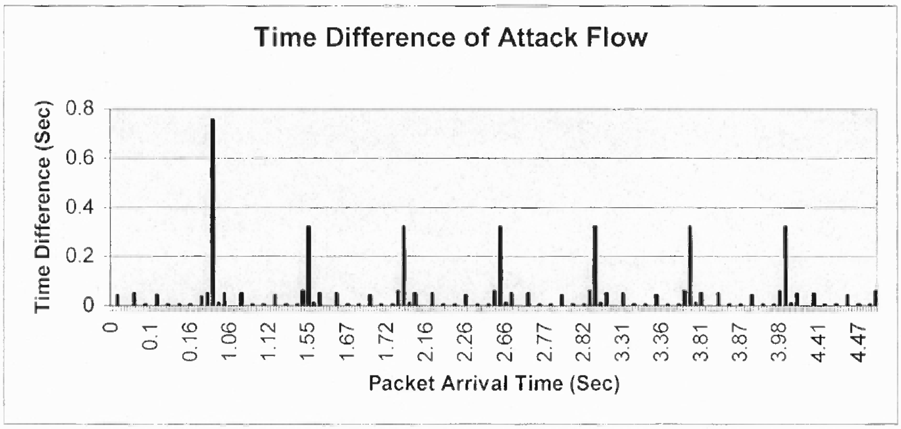


Figure 3.4 The time difference of the attack flow.

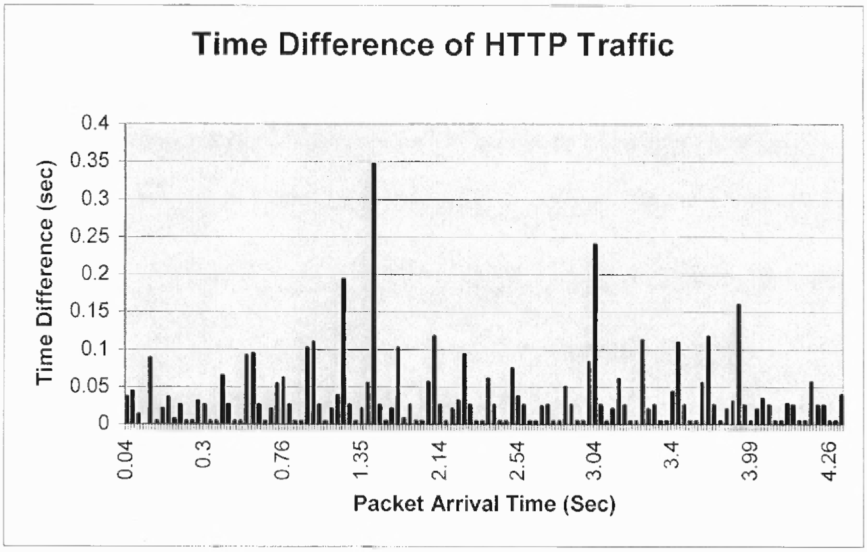


Figure 3.5 The time difference of the HTTP flow.

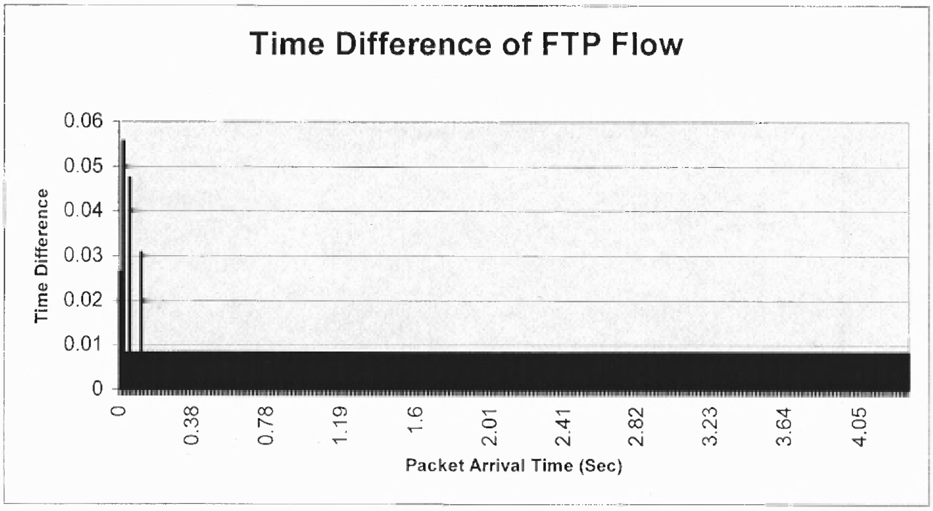


Figure 3.6 The time difference of the FTP flow.

3.4 Summary

This chapter describes the proposed time difference technique against the low rate TCP DoS attacks. The scheme is effective in detecting the low rate TCP DoS attack at the edge routers. The focus here is to present the framework of the proposed detection concept.

CHAPTER 4

AN IMPROVED DETECTION SCHEME

4.1 Problem Description

The MIT Spoofer project has reemphasized the detrimental effect of the IP address spoofing. The subnet IP address spoofing [66] is easily orchestrated, as the ingress IP address filtering cannot contain IP spoofing.

To illustrate the subnet IP address spoofing, consider an attacker in the subnet, 12.28.34.0 to 12.28.34.100; an attacker can easily use any address in this range for spoofing a source IP address inside this subnet. The IP address of every outgoing packet can be easily spoofed by randomly selecting an IP address from the pool of IP addresses available for spoofing; this is referred to as random IP address spoofing. It is assumed that the attacker has complete control of the source machine and can change the operating system stack as needed. The attacker can use either the UDP or the TCP protocol to send a packet with any possible value in the packet header. Let k be the number of packets in each flow. The flow-id of a flow is defined by the combination of a source IP address, a destination IP address, a source port, and a destination port. This definition will be adopted throughout this Chapter. The open knowledge of the RoQ attack and the ON-OFF periodic blasting attack [18] shows that periodicity can be random for the low rate DoS and RoQ attack. As described in [18], the attacker will have one IP address for every ON period; this is referred to as continuous cycle IP address spoofing. A variant of continuous IP address spoofing would be using a group of IP addresses in each ON period of the attack. The number of packets in the low rate DoS attack traffic using continuous or random IP address spoofing would be similar to the number of packets in a typical HTTP flow or a short-lived flow. The HTTP flows are hence referred to as mice as compared to long-lived flows known as elephants. Considering the widespread use of botnets by attackers, it is not difficult for an attacker to use the compromised machines with valid IP addresses to launch a low rate DoS attack. In addition, the master who controls botnets can sabotage machines in subnets scattered across the Internet, so that the attack traffic rate coming out of each subnet is not anomalous, but the aggregated traffic will lead to DoS when it reaches the targeted router. Use of botnets will also allow an attacker to use compromised machines to send attack traffic with random and continuous cycle IP address spoofing. In a low rate DoS attack, the required number of compromised machines can be low [67]. To make an easy distinction between the two attacks, an attack with time period less than or equal to one second is classified as a low rate DoS attack while an attack with a time period greater than one second is classified as a RoQ attack. The primary scope of this Chapter is to mitigate RoQ attacks with IP address spoofing in which an attacker can employ different types of IP address spoofing strategies while launching the RoQ attack.

4.2 Detection system

4.2.1 Detection System Architecture and Logic

The RoQ attacks cause fluctuations in the queue size and congestion levels at the router during the ON period of the attack. They can incur an increase in the instantaneous packet loss. The packet losses observed in experiments shown in Chapter 2 were greater than 2%. It is important that the detection system should be "off" when there is no attack.

Note that for the RoQ attack, the packet loss might not increase, and so the network administrator can also invoke the detection system by using the congestion signal from the active queue management (AQM) system. The proposed defense system in this Chapter uses congestion signal from the adaptive virtual queue (AVQ) algorithm to invoke the detection system. Thus, overhead of performing memory intensive analysis under no attack is eliminated. A network administrator can tune this parameter.

The detection system architecture and the attack detection procedure are depicted in Figures 4.1 and 4.2, respectively. The detection system relies on readily available per-flow information in today's routers, for example, Cisco Netflow, as shown in Figure 4.1 [68]. The proposed detection system leverages on this readily available per flow information, thus requiring no major architectural or system changes in the routers. The contents of the flow size estimation module are periodically flushed to the persistent storage memory for accurate attack identification. The following parameters are obtained for every new flow: the packet count (k), the packet size, the *createdtime* and the *lastaccesed* time; these parameters are the same as those facilitated in Cisco Netflow. The *lastaccesed* time field allows us to decide when a flow ends. The packet count keeps track of the number of packets of a particular flow, while the other parameters are self-explanatory. The benign flow table is used to separate the long-lived flows passing through the router. On the arrival of a new packet, if the attack filtering is ON, a query is made to the benign flow table to check whether a flow entry for that packet is present. If the entry exists, the packet is treated normally going through conventional steps of header decoding, route lookup, and forwarding. If the entry does not exist, the packet is passed to the module that implements the attack filtering algorithm which will be described later.

The entries in the benign flow table are updated once the processing performed in the persistent storage memory identifies the new legitimate long-lived flows. The long-lived flows are ones, which are still active after at least two seconds [69]. The benign flow table also sets an inactivity timer for each flow similar to the one used by Netflow, that helps eliminating entries of inactive flows in the benign flow table; a flow is considered inactive if no packet belonging to that flow has been observed for more than the inactive timeout constant. This also prevents an attacker from exploiting the benign flow table by sending normal traffic to be classified as benign and reusing the same flow id after a long time to send the attack traffic.

Further discussion of some related scenarios will be made in Subsection 4.2.2. To improve the scalability of the scheme, Internet traffic observations made by various studies [46] were incorporated. The key observation is that a small percentage of the flows contribute to the majority of the bytes and packets of the total traffic passing through a link. This knowledge about the long-lived flows is used to achieve considerable memory savings as well as to provide better throughput for the legitimate long-lived flows. The advantage of the benign flow table will become clearer, as the filtering approach is discussed later. This property of the Internet traffic is also the guiding principle of the low rate DoS and RoQ attack detection algorithm. As discussed earlier, the low rate DoS and RoQ attack flows that use IP address spoofing or botnets are similar to legitimate short lived flows in terms of the number of packets per flow, and the Internet traffic characteristic indicates that short lived flows occupy less percentage of the total traffic passing through a link. Thus, the Internet traffic property is violated when the low rate DoS and RoQ attack is instigated in the Internet. The key idea behind the detection approach is to detect violation of this property of the Internet traffic. So far, Steps 1 to 3 have been described in Figure 4.2 except the functioning of the low rate DoS attack detection algorithm. Step 4 will be detailed in Subsection 4.2.2 when some special cases of how an adversary can fool the proposed detection system are considered. The mechanism of the low rate DoS and RoQ attack detection algorithm is further explained in details along with the pseudo code shown in Figure 4.3.

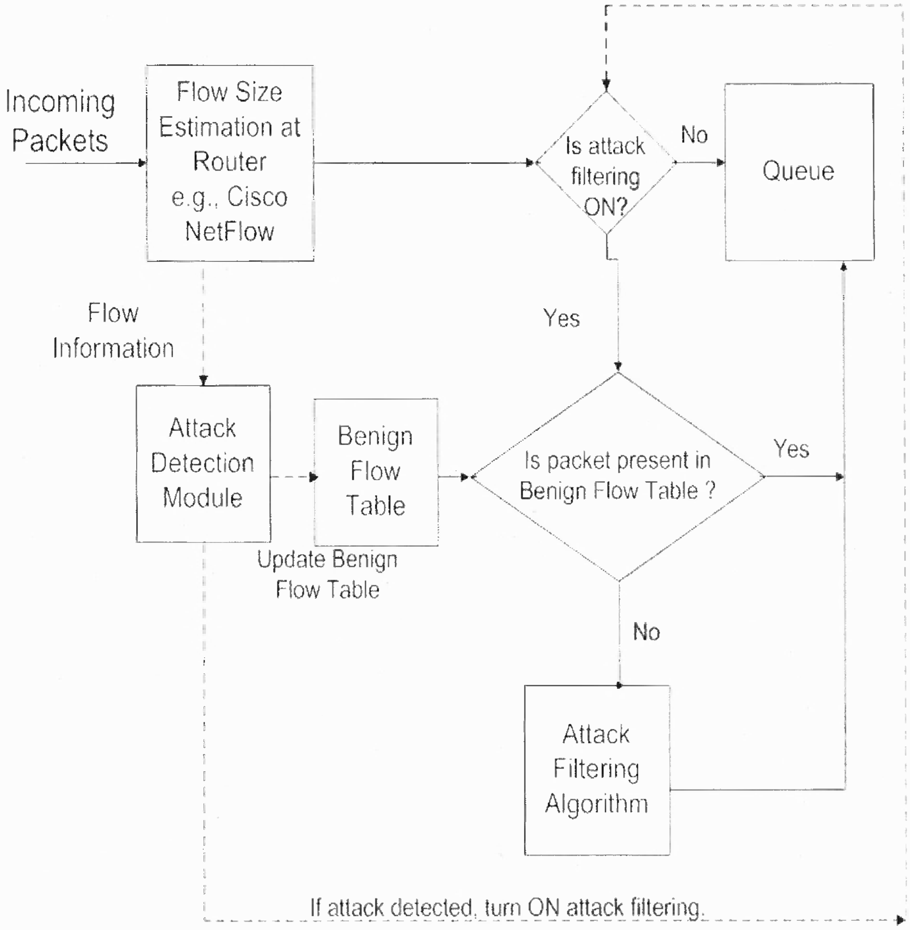

Incoming Packets
Flow Size Estimation at Router e.g., Cisco NetFlow
Is attack filtering ON?
No
Queue
Flow Information
Yes
Attack Detection Module
Benign Flow Table
Is packet present in Benign Flow Table ?
Yes
Update Benign Flow Table
No
Attack Filtering Algorithm
If attack detected, turn ON attack filtering.

Figure 4.1 The detection system architecture.

1) If the Virtual Queue > AVQ AQM threshold. Start the detection algorithm.
2) If attack filtering ON, check if the packet belongs to a flow in the benign flow table.
If yes, send the packet directly to the queue and continue normal operation.
If no, send the packet to the attack filtering module.
3) Periodically, run the low rate DoS and RoQ attack detection algorithm on the output of the flow size estimation module. Start the attack filtering algorithm if RoQ attack is detected.
4) Periodically, run the low rate DoS and RoQ attack detection algorithm and time difference algorithm on the benign flow information, while attack filtering is ON.

Figure 4.2 The attack detection procedure.

```
1) For all the expired flows with k packets in each flow
   in the persistent storage with C as link capacity
   and P as minimum packet size,

  IF {0< k < C/P, createdtime > current time – 1sec,
     lastaccessedtime < current time},

  SUM = SUM + Totbytecnt/flow.

2) If SUM > THRESHOLD {Low Rate DoS or ROQ
   Attack Detected}
```

Figure 4.3 Pseudocode of the attack detection algorithm

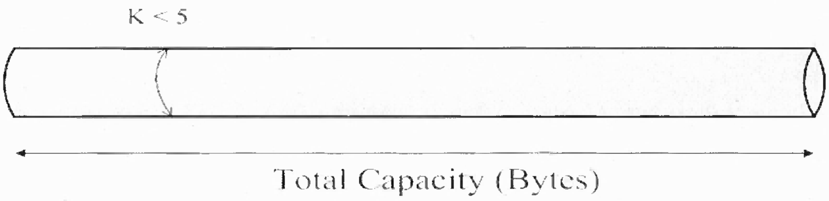


Figure 4.4 The basic concept of the attack detection algorithm.

The basic idea of the detection logic can be explained by using Figure 4.4. Assuming the entire capacity of the link as a big pipe, it is estimated how much of the pipe is filled by the flows containing atleast *k* packets in a second. For the time being, it is assumed that the *ON* period of the attack to be less than or equal to one second.

$$\left(\sum_{AllFlowsX<k} TotalBytes\ per\ flow\right) >> Threshold \tag{4.1}$$

The diagram in Figure 4.4 shows the entire capacity and the amount filled by the flows that have less than or equal to five packets in them every second. Step 1 of the algorithm computes the traffic load of a group of expired flows that satisfy the required criterion in the "if" statement. The expired flows are the ones that will not receive any more packets; it is simply a closed connection between the respective source and destination. Only the expired flows are considered in Step 1 of the detection algorithm. To observe a sudden increase in the traffic load from a group of flows in a short time, flows formed and expired within a second are considered as seen in Step 1. The *createdtime* and *lastaccesedtime* values of individual monitored flows will allow precise selection of the flows that were formed and expired within the observation time slice of a second. Step 1 is repeated every one second on all the expired flows stored in the persistent memory. The *totalbytecnt* of each flow is the *SUM* of the size of each packet in that flow that can be easily derived from the IP header. If the (*SUM*) in Step 2 of the algorithm exceeds a threshold in Equation (4.1), then the detection system invokes the attack filtering module. Thus, an attacker using IP address spoofing or botnets will try to send enough packets, each belonging to a different flow, that are sufficient to fill up the total capacity during the ON period of the attack. The proper selection of the threshold value is important to detect the low rate DoS attack. Three thresholds are proposed to be used in Step 2 of the detection algorithm:

$$\begin{aligned} &1) THRESHOLD \geq C+B \\ &2) C/2+B \leq THRESHOLD < C+B \\ &3) C/4+B \leq THRESHOLD < C/2+B \end{aligned} \tag{4.2}$$

C is the capacity of the link, and B is the buffer size of the router in Equation (4.2). A little modification might be required for lower values of thresholds for particular C and B, which can be obtained by the technique explained in Subsection 4.2.3. The threshold will be exceeded more often in a low rate DoS attack due to its low time period, and less often in a RoQ attack due to its high time period. The period of the attack can also be detected based on instants when the threshold is exceeded.

Various functions of the detection system to detect an attacker using different IP address spoofing strategies is discussed further. The attacker may use perfect random IP address spoofing; that is, a new IP address from the pool of available IP addresses is assigned to every packet. This implies one packet per flow. To detect an attack using continuous cycle IP address spoofing, or variations of random IP address spoofing with more than one packet per flow, the value of the *SUM* variable is checked by keeping k as $0 < k < C/P$, where k is the number of packets in each flow, C is the capacity of the link, and P is the packet size which can be assumed to be 64 bytes, the size of the smallest packet. The *SUM* variable is evaluated to be greater than any of the three proposed thresholds in a short period of one second, and whether it repeats periodically. In the next Subsection, to confirm the proposed heuristics, the range of k is kept as $0 < k < C/P$ to show that in the absence of the low rate DoS and RoQ attack, the value of the *SUM* variable does not exceed the proposed thresholds. The perfect random IP address spoofing case can be quickly detected by keeping *k* less than two, and checking if the value of the *SUM* variable is greater than any of the proposed three thresholds. Once the attack is detected, the proposed filtering mechanism is activated in which the attack packets are dropped as they start filling up the capacity. Thus, even if the attacker uses a different IP address in each ON period, the attack packets will be dropped, as demonstrated in the simulation results. RED-PD [18] cannot detect such an attack.

4.2.2 Intelligent Attacker

The only way an attacker can evade detection is if the attack flows are classified as the benign flows. This can be staged if an attack flow sends packets for more than 2 seconds, and then uses the same flowid to launch a low rate DoS or RoQ attack. The proposed detection system will incorporate an approach reported in [70], which can easily detect low rate DoS and RoQ attacks that do not use IP address spoofing. The time difference approach relies on computing the time difference between the consecutive packets of a flow. It was shown in [70] that only the low rate DoS and RoQ attack flows exhibit periodicity in the time difference, while other legitimate traffic flows lack this characteristic periodicity property of the low rate DoS and RoQ attack. The time difference technique can be easily integrated in the current detection system as it only requires the following per-flow information, the *createdtime*, the lastaccessed time, and each packet arrival time from the per-flow systems shown in Figure 4.1. The per-flow system will have to be configured to provide each packet arrival time as one of the flow fields, and just has to record the timestamp when it samples a packet belonging to a particular flow for this requirement. If a particular flow is found to be an attack flow by using the time difference technique, it can be easily blocked by filtering traffic coming from that IP address.

In another possibility, an attacker will use a group of machines (bots) to send flows for more than 2 seconds to forge classification as the benign flows, and then use these flowid's to launch a low rate DoS attack. To detect such an attack, the proposed detection algorithm shown in Figure 4.2 is applied on the benign flows. It requires adding the total byte count of each flow in the benign flow table for a short period of every one second. As in detection of the low rate DoS and RoQ attack described before, the proposed logic of the "*SUM*" value exceeding predefined thresholds will still hold as the attack flows are now part of flows classified as benign. To evade the above change in the detection system, an attacker can simply use some flows classified as benign flows and others as short-lived flows to launch a low rate DoS or RoQ attack. In this way, the *SUM* value of the detection algorithm shown in Figure 4.2 will not exceed the predefined thresholds when the detection algorithm is applied only on the benign flows. To detect this variant of a low rate DoS and RoQ attack, the proposed attack detection algorithm is applied on both the benign and short-lived flows simultaneously to see if the *SUM* variable exceeds the proposed thresholds. Now, the *SUM* variable should exceed one of the proposed thresholds. For the latter variations of the attack when the low rate DoS and RoQ attack detection algorithm is used to detect the attack and not the time difference technique, the attack filtering technique described in Subsection 4.2.4 is used to filter the attack traffic. This process of detecting an intelligent attacker can be activated once the attack-filtering algorithm is active. Attention should also be made in classifying flows as benign once the attack detection algorithm finds flows in the benign flow table causing the low rate DoS or RoQ attack. Typically, long-lived flows should be classified as benign after having been verified their lack of periodicity by using the time difference technique described earlier. The proposed approach can thus detect the low rate DoS or RoQ attack that uses any combination of the number of flows participating in the attack with any time period, any burst period, and any burst rate.

4.2.3 Trace Evaluation

To confirm that the thresholds proposed in the previous Subsection will work for the Internet traffic, the proposed strategy is evaluated by analyzing the OC48 (2.5Gbps) traces provided by CAIDA [71]. Using the Coralreef [72] software, the expired flow statistics <*Source IP address, destination IP address, dource IP port, destination IP port, packetcnt, Bytecnt, createdtime, and lastaccessedtime*> were obtained; they are similar to the ones that are proposed to collect for all the flows. The attack detection algorithm was executed using the flow information obtained by the coralreef software to observe the characteristics of the *SUM* variable in the absence of the low rate DoS and RoQ attack. The traces were recorded during August 2002 and April 2003. Figure 4.5 shows the *SUM* variable statistics for a 5-minute April 2003 trace. The OC48 speed is 2.5Gbps, i.e., 2.5E09, and so C/2 is 1.25E09, and C/4 is 62.5E07. It can be observed from Figure 4.5 that the *SUM* variable does not exceed any of the proposed thresholds. An ON period in a low rate DoS attack is typically less than one second as reported in the literature about low rate DoS attacks. It can be noted in Figure 4.6 the values of the *SUM* variable for every 2 seconds, in which case it was assumed an ON period of 2 seconds. The attack with ON period greater than 2 seconds can certainly be detected by the proposed approach, but they can be easily detected by RED-PD and many other existing AQM schemes too.

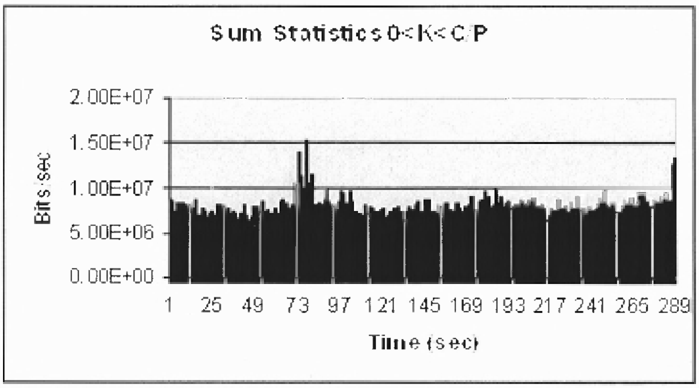


Figure 4.5 Sum statistics for every second.

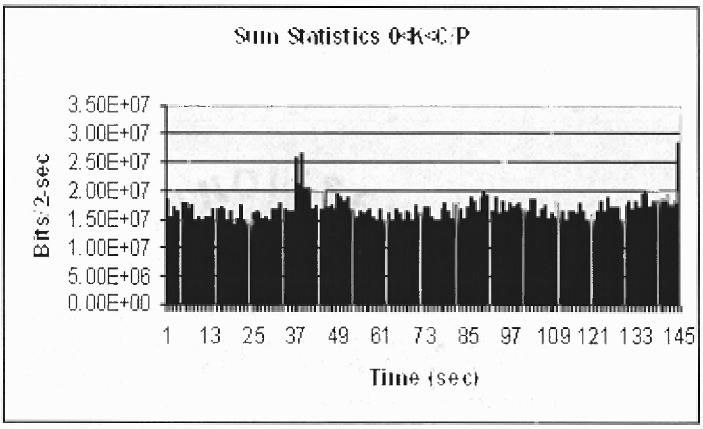


Figure 4.6 Sum statistics for every two seconds.

Interestingly, it can be argued that the attack cannot be a low rate DoS or RoQ attack, when the ON period is so long. The *SUM* variable in Figure 4.6 does not exceed the proposed thresholds. Similar observations were found in other traces. The tuning of lower thresholds below C/2 can be done by studying the *SUM* statistics in the absence of the attack during the normal router operation. The profile of flow sizes and the *SUM* distributions can be obtained in the absence of the attack scalably by using the per flow information from systems like Cisco Netflow in Figure 4.1 to understand the unique properties of traffic distributions on each link, and to adjust the attack detection thresholds accordingly. It is highlighted that in a short period of one to two seconds the *SUM* variable does not exceed the proposed thresholds in the absence of the low rate DoS and RoQ attack, as observed in Figures 4.5 and 4.6. Thus, only during the low rate DoS and RoQ attack, the *SUM* will exceed the proposed thresholds.

4.2.4 Filtering Logic

A filtering scheme to mitigate the RoQ attacks is proposed. The low rate DoS and RoQ attack detection algorithm in Figure 4.2 provides information on how frequently the attack bursts are instigated from which one may determine the attack type (i.e., RoQ attack). To filter the RoQ attack packets, which are using the spoofed IP addresses, a nondeterministic approach is proposed because it is difficult to know which IP address an attacker will use in future bursts. Thus, it is futile to store the attack IP addresses seen in the old bursts. A simple method to address this problem is developed. As mentioned before, the long-lived flows in the benign flow table are separated, and they are treated preferentially. On the arrival of packets belonging to these flows, unless the buffer is full, they are enqueued in the queue and are passed normally. Special attention is needed while identifying a new benign long-lived flow when the attack-filtering mode is ON by verifying that the difference between *createdtime* and lastaccessed time should be atleast two seconds, and the lastaccessed time should be close to the inspection time to classify the flow as a non-expired, legitimate, and long-lived flow. The detection algorithm also considers special attack scenarios discussed in Section 4.2.2 before classifying the flow as a non-expired, legitimate, and long-lived flow. Packets, which belong to the new flows and are not present in the benign flow table, are enqueued in the queue, and the current queue length is then computed. The current queue length is checked if it is greater than α% of the queue limit. If so, the enqeued packet is dropped immediately; otherwise, the enqeued packet is treated normally.

The proposed strategy is a preemptive strategy to prevent the attack packets from gaining access to the legitimate bandwidth. It can be empirically confirmed that the point after which the queue length exceeds α% of the queue limit will occur only during the attack epochs as the legitimate TCP flows will try to share bandwidth, and the attack packets will typically try to force the legitimate packets out of the queue once the occurrence of the attack is confirmed by the proposed attack detection algorithm. Nevertheless, under no attack during congestion, the legitimate flows can force other legitimate packets to be dropped, but in the RoQ attack case, they are most likely RoQ attack packets, and hence these packets are dropped. The filtering is also activated at the approximate time instants for one second when attack packets start arriving at the queue. Filtering is done for one sec because the attack detection algorithm runs every one second. The time period of the attack can be obtained from the attack detection algorithm. This idea is the essence of the proposed attack-filtering algorithm. A tradeoff exists in choosing the percentage of the queue limit for dropping the packets; this will decide how much attack traffic will be dropped as well as the penalty imposed on the legitimate short-lived and long-lived flows. The percentage value will be determined empirically; details will be provided in a later Section. One advantage of this approach is that too many legitimate short flows traversing the router need not be isolated as it is difficult to implement per-flow logic in hardware, and the difficulties involved will be discussed more in the next Section on hardware implementation strategies. The short-lived flows will get enough share of the total capacity as just (100 - α)% of the buffer space is denied untill the RoQ attack is filtered. However, some of the packets of the new short flows will be dropped, but they will be admitted after a few milliseconds when the attack burst has subsided. A normal TCP connection uses the exponential backoff algorithm to resend the dropped packets before giving up. In addition, the RoQ attack packets are enqueued less frequently, e.g., every 5 seconds, and so few legitimate short-lived TCP traffic, which lost packets, can easily enter slow start and increase their congestion windows, which were halved due to lost packets, to send data before the next attack burst occurs. In fact, most of the short-lived flows last less than two seconds, and so they will finish sending data during the slow start phase before the next attack burst occurs.

One more advantage from the implementation perspective as compared to the traditional filtering is that no memory is needed to store the list of the IP addresses to be dropped. The hardware implementation issue with per flow states will be discussed in the next section where it becomes clear why it is difficult to preserve per flow state at high speeds for all the flows. Thus, the proposed filtering solution is simple and effective. Simulation results demonstrate the effectiveness of the proposed filtering technique in dropping a significant number of attack packets while simultaneously provisioning the legitimate traffic enough bandwidth. The attack filtering can be stopped after having confirmed the no attack status by using the attack detection algorithm shown in Figure 4.2. The scheme has a drawback when used to filter the low rate DoS attack traffic, because the low rate DoS attack has a small-time period as compared to those of the RoQ attacks, and hence the attack packets are enqueued more frequently. The queue length exceeds α% of the queue limit more often, thus leading to higher drops of the legitimate packets. The legitimate short-lived traffic is penalized, and it suffers from reduction in throughput.

4.3 Implementation Discussion

Internet security is vital to facilitate e-commerce transactions, and there has been continued research effort to provision network traffic monitoring at high speeds. The hardware capabilities achieved by some other approaches like the deep packet inspection [73], at relatively low speeds show that the proposed approach can be realizable. The fast memory, i.e., SRAM, is exorbitantly costly, and the cheap memory, i.e., DRAM, is too slow to work at the high speed line rates, which are two critical considerations in provisioning high speed monitoring.

In this Section, some of the recent techniques proposed in the literature that can facilitate realization of the proposed detection system architecture are discussed. To estimate the size of the benign flow table, commonly used Internet traces are considered for analysis [74] - [76]. The ISP trace at OC48 speed used in [74] contains 11,341,289 flows. To maintain per-flow states for so many flows is difficult as the majority of the flows are short-lived leading to continuous updates and removal of flows from the memory. The high-speed memory SRAM which can support such operation is exorbitantly costly. Considering the previously described characteristics of the Internet traffic, approximately one-third (3,780,429) of the flows are the large flows. It is difficult to maintain per flow state for all the flows because of resource constraints. By using a bloom filter calculator [77], the amount of memory (SRAM) required for maintaining the entry of each flow in a bloom filter for approximately 3,780,429 flows with the probability of false positive of 0.001 and four hash functions is two MB. Note that these are not live flows at one instant, but the total number of flows found in the entire trace. In another work [78], the authors pointed out that the maximum number of live flows is 714,166 in another OC-48 trace for which the size of the bloom filter is 1.8MB using the same parameters as before. Thus, the benign flow table has a modest memory requirement of about two MB.

Apart from the memory requirement for the benign flow table, the proposed detection system also needs per flow size estimation module like Cisco Netflow and hence additional resources are needed for the per flow size estimation module. However, the additional resources for the per flow size estimation module are already included in the routers. The benign flow table represents the additional requirement if the proposed system is deployed at a router.

4.4 Simulation Results

The ns2 simulator is used to demonstrate the performance of the proposed detection scheme. The topology used in the experiment is shown in Figure 4.7. Packmime HTTP traffic generator was used; it was developed by using the real traces of the Internet traffic. This is the most recent work about the modeling of the HTTP traffic that improves over the previous HTTP traffic models. The packmime traffic model was adopted in this study because the HTTP traffic interaction with the detection system is important since an attack flow using the IP address spoofing or botnets would have a similar number of packets like in a typical HTTP flow or a short-lived flow. The topology consists of two

PackMime clients, two PackMime servers connected by 100Mbps links to the delaybox, and two routers with a buffer size of 500 packets each, and a bottleneck link of 10Mbps between them. The queue size is fixed and is equal to twice the delay bandwidth product, which was found to perform optimally for routers in the Internet. The link between the delaybox and the router is 100Mbps. The delaybox is used to provide per flow dropping probability, round-trip times, and bottleneck link speeds. In the experiment setting, the dropping probability is zero and the available server bandwidth is uniformly distributed from 1 to 20 Mbps.

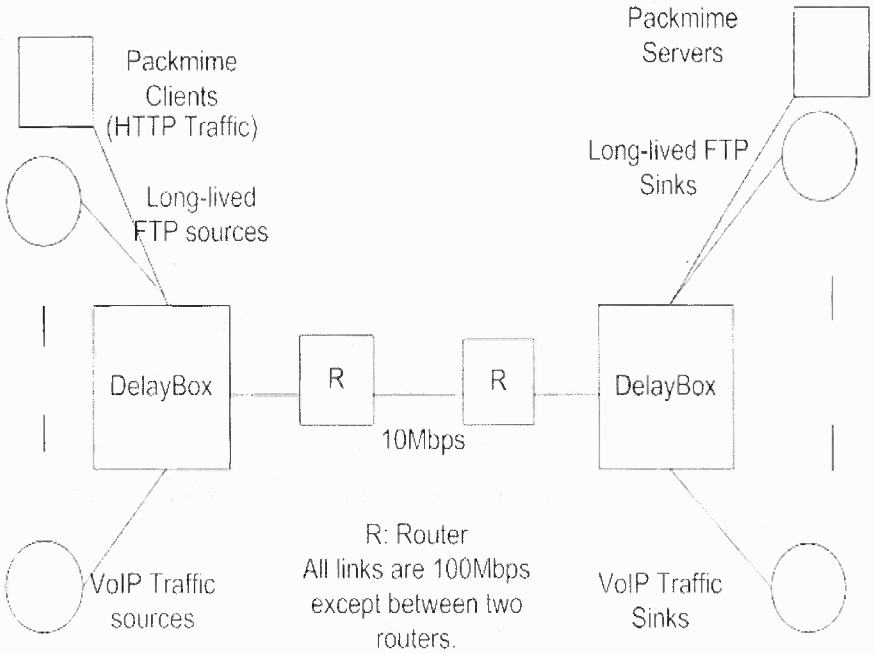


Figure 4.7 The simulation topology.

The test-packmime-delaybox.tcl script [42] was modified for the simulations. All the access links have random delays obtained by using a uniform distribution from 50-250ms. The access links connecting to the sink agents and the bottleneck link has link delay of 10ms. Ten long-lived flows simulating FTP are used in the network. The details of the SACK TCP used in the simulations are: window size 50 packets, segment size

1460, minimum RTO one second for the FTP flows, and the rest of the parameters are the default settings. The TCP of the packmime model also adopts the SACK TCP; other details are the same as that of the TCP used for FTP. Five VoIP flows modeled as G711 FEC 96Kbps traffic using the exponential on-off traffic model in ns2 are part of the network traffic. The attacker uses UDP constant bit rate traffic (CBR). The proposed detection system code is embedded in the AVQ algorithm of ns2, and the detection system is invoked when the virtual capacity exceeds the AVQ-defined threshold. It is invoked by the congestion signal from the AVQ algorithm. The simulation was executed for 650 seconds with a warm up time of 50 seconds; the attack was introduced 50 seconds later after the start of the simulation. The packmime connection rate was 40 connections per second, i.e., 40 new HTTP connections would start every second. The rate value of 40 is used because it generates the average throughput of 30% of the bottleneck link in absence of any other traffic on the bottleneck link in the simulation setup. It also conforms to the reported nature of the Internet traffic, in which many short-lived flows occupy around 30% of traffic in bytes, and the rest is occupied by the long-lived flows. About 14000 connections were generated during the lifetime of the simulation. The detection system code uses the flowid field available in ns2 to implement the per-flow logic. In practice, the flow-id will have to be replaced by the hash of the source IP address, the destination IP address, the source port, and the destination port.

Initially, the ability of the attack detection algorithm to detect the low rate DoS and the RoQ attack is demonstrated. The attack threshold is set as explained in Figure 4.2 to be 312.5KB close to the value of C/4, and was tested with two attack scenarios: 1) T=1s, t=0.3s, and R=10Mbps (low rate DoS attack), and 2) T=5s, t=0.3s, and R=10Mbps

(RoQ attack). One representative *SUM* value in absence of the attack is 88.85 KB, and it never exceeds the threshold of 312.5KB throughout the simulation. In attack scenario 1, the *SUM* values are 391.3KB at 50.8s, 425.7KB at 51.8s, 390.6KB at 52.8s, 393.9KB at 53.8s, and so on. In attack scenario 2, the *SUM* values are 391.3KB at 50.8s, 375.6KB at 55.8s, 371.7KB at 60.8s, and so on. Thus, clearly in both cases, the *SUM* value exceeds the threshold, thus indicating the presence of the low rate DoS and RoQ attack at the router, respectively.

In this simulation, the attacker uses random IP address spoofing with the time period of 5 seconds, the burst period of 0.3 second, and the burst rate of 10Mbps. The proposed detection system detects the RoQ attack as explained in the attack detection algorithm. As shown in Figure 4.8, the throughput for the long-lived flows is restored to almost that of the case with no attack. Apparently, flow 4 was not affected by the attack; it is conjectured that the attack packets could not force timeout during the burst period as the RTT was large, and hence the flow did not suffer from reduction in throughput. In addition, during the attack, flow 4 achieved higher throughput because other flows were suffering from reduction in throughput.

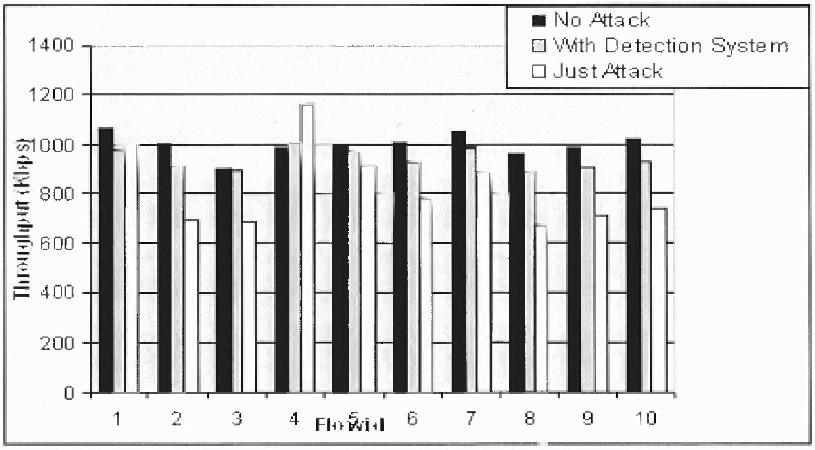


Figure 4.8 The throughput comparison under an RoQ attack.

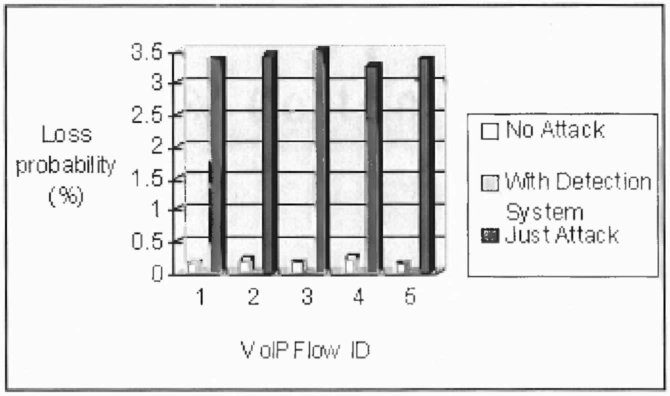


Figure 4.9 VoIP packet loss comparison under an RoQ attack.

In Figure 4.9, it can be seen that VoIP flows do not experience any packet loss with the use of the detection system. The VoIP flow is classified as a benign long-lived flow as it is active for more than two seconds. A study by a VoIP carrier network vendor Nextone has reported the average VoIP call length to be around 9 minutes in one of their February 2005 case study [79]. For the RoQ attack with the time-period of 5 seconds, the value of α is kept to be 80 because above that leads to loss of throughput to the legitimate long-lived flows in the benign flow table. The value of α can be higher than 50% for the RoQ attack filtering as the attack packets enter the network less frequently; this concurs with many of demonstrated simulations. In Figures 4.10-4.12, the statistics of each HTTP flow generated during the simulation obtained from the packmime program are shown. The x-axis is the response size in bytes of the replies from the server, and the y-axis is the response time between client sending HTTP request and client receiving complete HTTP response in seconds. The average value of the response time in the no attack case is 1.8s, that with the detection system is 2.3s, and that with the attack is 1.2s. The standard deviation of the response time in the no attack case is 1.3s, that with the detection system is 2.2s, and that with the attack is 1.3s. In the attack case, the HTTP performance is better because the long-lived flows are suffering from losses and their bandwidth is claimed by HTTP flows. In comparing the case with no attack and that with detection system, it is observed that the average response time increases by 0.5s and the standard deviation of the response time increases by 0.9s with the detection system. Thus, the legitimate clients are not denied service when the proposed detection system is employed; however, there is a slight increase in the response time.

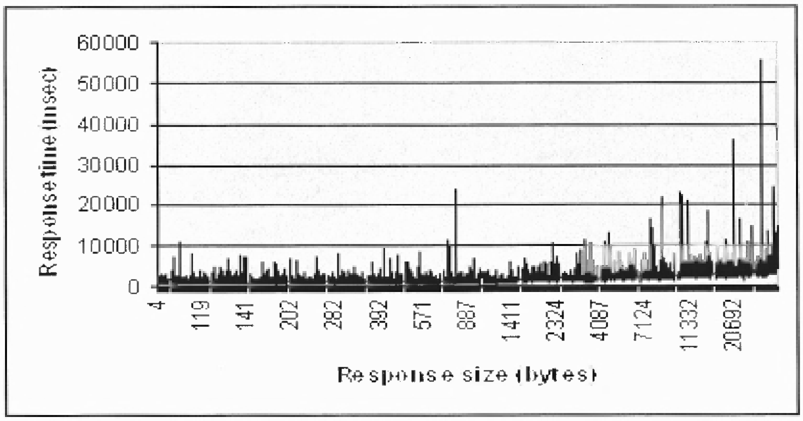


Figure 4.10 HTTP performance under no attack.

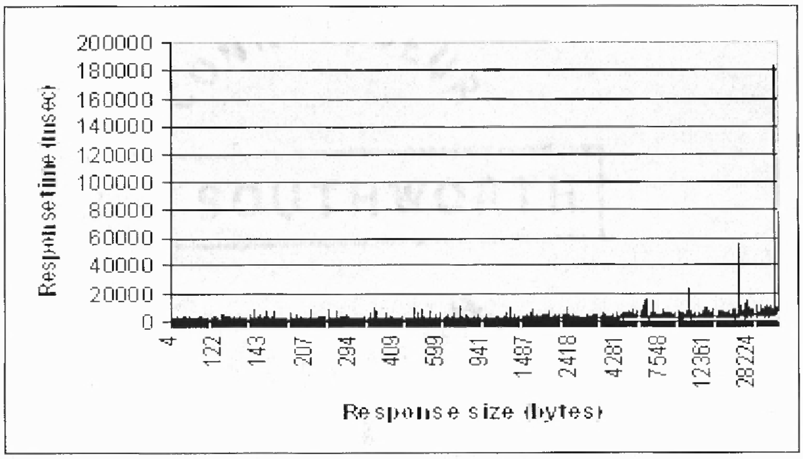


Figure 4.11 Restoration of HTTP performance.

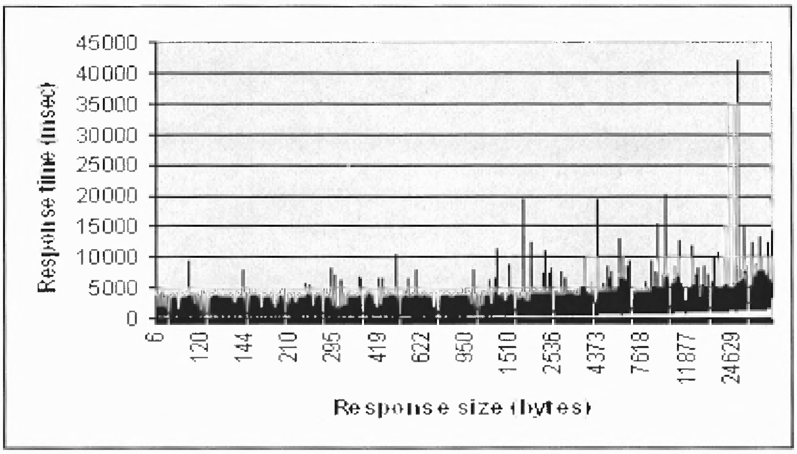


Figure 4.12 HTTP performance under an RoQ attack.

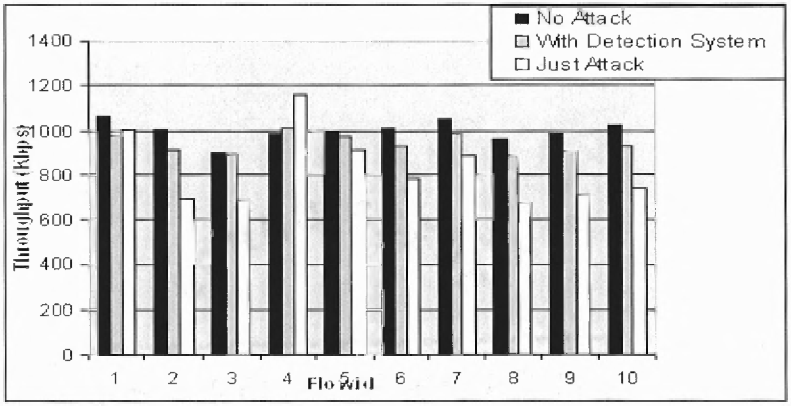


Figure 4.13 The throughput comparison under an RoQ attack with continuous cycle IP address spoofing.

Experiments were conducted on simulating an attacker by using continuous cycle IP address spoofing in which a new IP address is used for every ON period. In the simulation, an attacker sends approximately 500, 5K, 10K, 15K, and 20K packets, each of size 210bytes, to fill up the bottleneck link during the ON period of 10ms, 110ms, 210ms, 310ms, and 410ms, respectively. The attacker in this scenario sends 4K packets with a new IP address every ON period. The time period was kept 5 seconds, the burst period 0.3 second, and the burst rate 10Mbps for the RoQ attack, and all other simulation parameters were exactly the same as those in previous simulations but with a variant of continuous IP address spoofing instead of random IP address spoofing. The results are shown in Figure 4.13. It can be observed that the throughput of the legitimate long-lived flows is restored, and a significant number of attack packets are dropped by using the proposed detection system. The proposed architecture facilitates identification and filtering of the RoQ attack traffic in which IP address spoofing is instigated. The proposed approach thus addresses most of the issues where RED-PD and several other approaches fail to defend against RoQ attacks.

4.5 A Different Approach to Filter Attack Traffic

This section describes the proactive test based differentiation technique (PTDT) in detail for filtering the low rate DoS attack traffic. Figure 4.14 shows the block diagram of the system. The detection algorithm is the same as defined in Section 4.2.

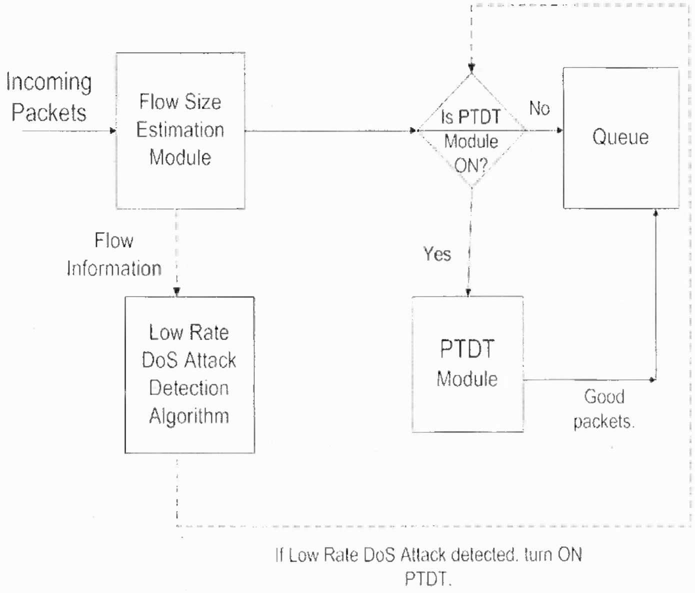

Incoming Packets
Flow Size Estimation Module
Is PTDT Module ON?
No
Queue
Flow Information
Yes
Low Rate DoS Attack Detection Algorithm
PTDT Module
Good packets.
If Low Rate DoS Attack detected, turn ON PTDT.

Figure 4.14 The new filtering system block diagram.

The detection and mitigation system will operate at routers primarily at the edges of a network, so that all the traffic entering the network can be examined by the system. The low rate DoS attack detection algorithm requires the per flow information of the traffic passing through an edge router. A flow or flow id is defined as a 4-tuple of the source IP address, the source port, the destination IP address, and the destination IP port. The flow size estimation module will provide the necessary per-flow information of the network traffic passing through the router to the low rate DoS attack detection algorithm. Incoming packets are enqueued in the outgoing queue of the router if the PTDT module is not ON. PTDT is activated if the low rate DoS attack detection algorithm detects a low rate DoS attack. Once activated, PTDT allows legitimate traffic to pass through the router, but blocks the attack traffic. The detailed functioning of the PTDT module is explained further. After the low rate DoS attack is detected, PTDT is activated. PTDT also leverages on the fact that TCP traffic is dominant in the Internet. Figure 4.15 shows the flowchart of the PTDT module. On arrival of a SYN packet, the PTDT module knows that it belongs to a new flow and sends a puzzle to the source of the packet. The technique of using visual puzzles or CAPTCHA's in the proposed system is similar to several proposals used to distinguish humans from the attack machines [80]. The user is admitted by the system after having given the correct answer of the puzzle. This way, PTDT can stop an attacker who can send SYN packets to launch a low rate DoS attack, and still allow the legitimate users to access the network. The technique of introducing a puzzle will also confirm that a human user, rather than an automated attack machine, is initiating the communications, and so the flow after having been admitted to the system will follow the standard protocol.

The remaining steps in the flowchart for an old flow will typically verify the behavior of the flow over a period of time before being classified as benign. Functions of the remaining blocks of the flowchart are explained hereafter, which are similar to the original proposal described in [81]. The two counters fail_num and pass_num achieve a balance between the conservative and aggressive approach against individual flows. A flow is allowed to prove its innocence by failing no more than a certain number of tests maintained by the fail_num counter; otherwise, it is dropped. Similarly, a good flow after having passed a certain number of tests maintained by the pass_num counter is trusted as innocent, and is not required to take further tests. Then, the PTDT module checks the current state of the flow; for a flow under test, its current rate should not exceed its previous rate. This can be achieved by dropping a single data packet, so that the legitimate TCP flow will reduce its sending rate. The attack flows will not reduce their rates, and thus can be distinguished. If a flow passes this test, its pass_num value is increased by one, and vice versa. The default values of various thresholds like certain numbers of tests that can be failed, though are applications specific, can be kept the same as in the original proposal [81]. Another advantage of using these proactive tests is to detect intelligent attackers who can overcome CAPTCHA's by using intelligent image recognition software's [82].

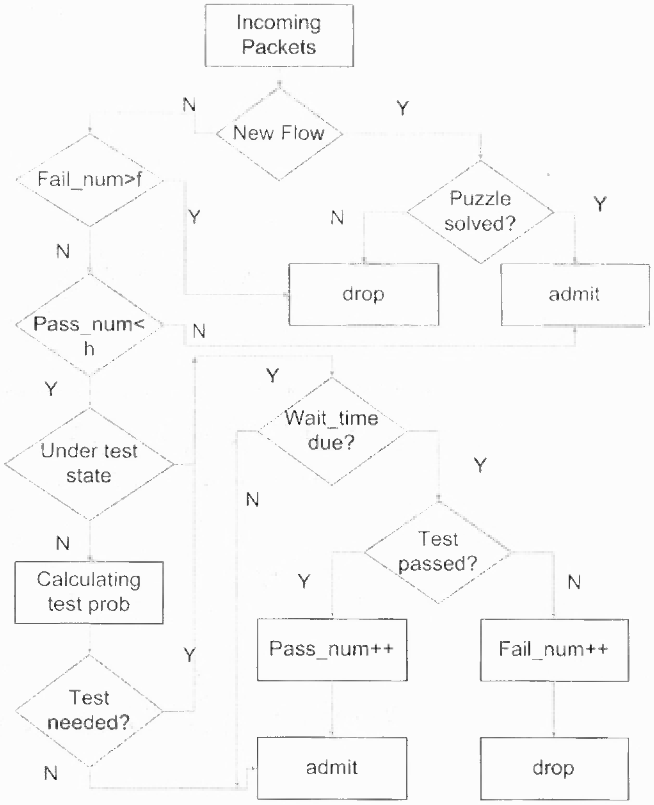


Figure 4.15 The PTDT module's flowchart.

Thus, by employing the PTDT module, the legitimate TCP traffic can still access the network in the presence of the low rate DoS attack. The PTDT module can also monitor the rates of some common UDP applications like VoIP and Video traffic, and if the rates conform to the standard application rates, then these flows can also be admitted into the network. Proposals such as CYSEP [73], which perform complex network security operations like encryption/decryption and intrusion detection at gigabit speeds, can be easily modified to implement the PTDT module. Currently, the entire detection system can work at network edges where such tasks can be performed by using a combination of different hardware implementations. It can be typically suitable for medium and small-sized networks, which host web servers.

4.6 Simulation Results of New Filtering System

Preliminary investigation to validate the proposed detection and mitigation system was done by using the ns2 simulator. The well-known dumbbell topology network was used with a single bottleneck link of 10Mbps. The access links were 100Mbps with random delays on each link to simulate varying roundtrip-time for each connection. The network traffic consisted of ten FTP flows and one attack flow. The FTP traffic used TCP sack with a window size of 43 packets, segment size 1460bytes, and the other parameters with default values. The attack flow was UDP constant bit rate traffic with the burst period of 0.4s, the burst rate of 15Mbps, and the time period of one second. The attack traffic was using random IP address spoofing. In the ns2 simulations, puzzles were not served before the connections were established, and so it was assumed that the FTP connections had solved the puzzles. It is decided to implement this functionality in the future on a Linux prototype of the detection system. Proactive tests were imposed on these connections to verify their legitimacy. Figure 4.16 shows the throughput of the legitimate FTP flows under different scenarios. Note that the throughput is restored back by using the detection and mitigation system. Figure 4.17 shows the sending rate of a legitimate flow with and without the proactive test, respectively. Typically, the legitimate flow that uses a stock TCP should reduce its rate after its packet is dropped. An attack flow will not obey the stock protocol which was shown in the original PTDT; this was reinforced by simulations. The contribution of this new filtering system has been augmenting the PTDT approach with the low rate DoS attack detection algorithm to mitigate the low rate DoS attacks orchestrated through botnets.

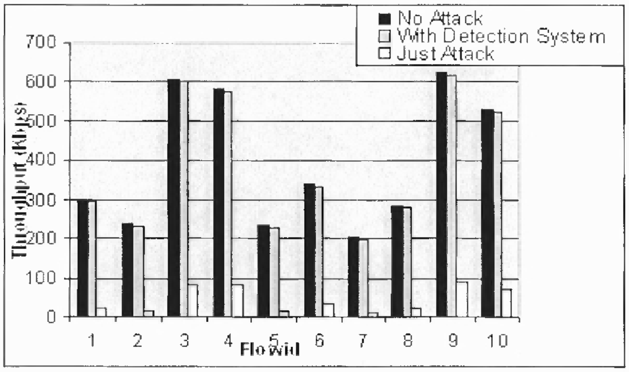


Figure 4.16 Throughput of the legitimate FTP flows.

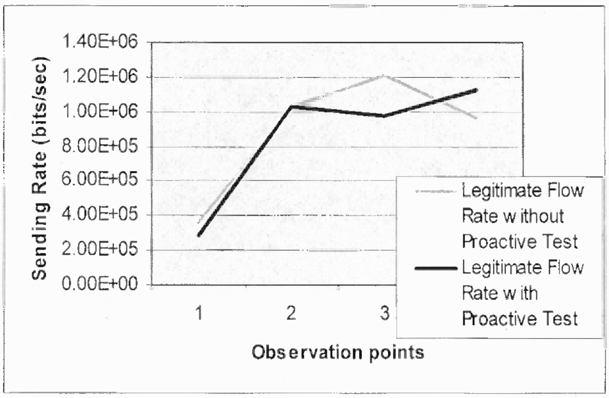


Figure 4.17 A proactive test demonstration on a legitimate FTP flow.

4.7 Summary

In this chapter, a router-based approach is proposed to detect the low rate DoS and RoQ attacks which use IP address spoofing or botnets. This work addresses the IP address spoofing and botnet problem in the context of the low rate DoS and RoQ attacks, and proposes an effective and realizable solution to defend against these attacks. The effectiveness of the proposed approach has been demonstrated via extensive experiments.

CHAPTER 5

A QUIET DDOS ATTACK

5.1 Quiet Attack Basics

In this chapter, a new low rate attack model is proposed. This work will hopefully help stimulate development to mitigate existing vulnerabilities before they may be exploited. The war against attackers will be won only if the white hat community stays ahead of the attackers. This work describes an attack that has the potential to disrupt the Internet which has become an integral part of economic, social, and political activities of any nation.

The proposed model uses the TCP protocol to launch the attack. TCP is considered to be a network adaptive protocol, and widely contributes to the current Internet traffic. The attack has the ability to disguise itself as a normal traffic, thereby making detection difficult. It relies on using readily available botnets to send the attack traffic. Botnets are formed at an alarming rate today because of increasing vulnerabilities in various software applications. The users of these applications are often common folks who are not security conscious, thus leaving their systems exposed to exploitations. Social engineering attacks are also used to increase the bot population. Thus, all these conditions lead to the breeding ground for rogues to develop new attacks. The basic attack model is explained along with effects of the attack on a single TCP flow. The attack exploits the effect of short-lived TCP flows on other TCP flows. A TCP flow strictly adheres to the congestion control algorithms like slow start and congestion avoidance. A long-lived TCP flow is typically known to enter and end in the congestion avoidance phase while a short-lived TCP flow to end in the slow start phase. In the quiet attack model, a set of short-lived TCP flows are started so that the aggregate attack traffic rate is approximately proportional to the link capacity under attack, and subsequently after every short time period *T*, a new set of short-lived TCP flows are injected to the network. A TCP flow rate is congestion dependent unlike a UDP flow, and so the exact rate cannot be imposed by using TCP as an attack flow. A botmaster will use capacity estimation techniques like capprobe [83] to approximate the link capacity. During this process, the available bandwidth is monitored by using tools like abget [84] or pathchirp [85], and is not allowed to increase. The time period *T* is kept random between 0s and 1s to remove any deterministic pattern. In fact, the time period is designed to look continuous. The available bandwidth remains negligible that leads to congestion at the router queue, and as the queue fills up, packets at the queue end are dropped by the droptail queue mechanism, i.e., any packet can be dropped as there is no preferential treatment given to any packet. As a legitimate long-lived TCP flow times out and enters the slow start phase, the congestion window is reduced, thus reducing the sending rate and throughput. The persistent congestion in the queue forces random packet drops of the legitimate long-lived TCP flows, thus affecting the throughput each time a packet is dropped. The attack concept is simple: keep the available bandwidth negligible by sending enough short-lived TCP flows.

The quiet attack exploits shortcomings of the end to end window flow control [86], which shows that if the number of connections increases, then delay or congestion will roughly increase proportionally to the number of connections or more precisely to the sum of window sizes of all connections. The basic trade off in end to end window flow control mechanism is selecting the right window size which is a well studied difficult problem. TCP initially does not know the network congestion state, and so it uses slow start in the beginning to increase the window size exponentially; this, however, gives unfair advantage to short-lived TCP flows in the quiet attack when other legitimate TCP flows are waiting for transmission in the stage of retransmission timeout. In this attack, short-lived flows are sent persistently that subsequently lead to considerable reduction in throughput.

The major threat of the attack model is to provide false impression of the attack as a "congestion scenario". Previous studies have considered effects of short-lived flows on other Internet traffic with the assumption that they occupy less volume of the total capacity although they are large in numbers. By using botnets in sending short-lived TCP flows during an attack, such Internet traffic assumptions can no longer be true.

5.2 Quiet Attack Execution in the Internet

5.2.1 Quiet Attack Reconnaissance Phase

The attack execution of the quiet attack model involves network reconnaissance and finally execution. The network reconnaissance involves deciding a router to be attacked, a botnet to be used, and gathering a list of web servers. It is assumed an attacker has access to a botnet which is formed by capitalizing on latest exploits. Active bots in a botnet are typically concentrated in the same time zone [3], and such group of bots will participate in the quiet attack. The next step is to determine which router to launch the attack that involves the botmaster issuing a command to the active bots to initiate a simple command like "tracert www.bu.edu" which will trace the path from the bot to the bu website. The tracert command even works for users with limited access or non-administrative accounts on windows XP and Vista. All bots then send the output of the tracert to the botmaster. An edge router whose IP address is observed in all the tracert outputs becomes a target for attack by the botmaster. It is assumed that each time zone will have few big ISPs; for example, there are Verizon, Comcast and AT&T in the eastern standard time zone. The big ISPs have a few transit points from which the traffic enters the Internet backbone. Such transit points of big ISPs will be the targets of the quiet attack. Espionage during a cyber war or stealing confidential information of ISPs can be another way in which an IP address of a router to be attacked can be found. To discover routers to be attacked, an attacker can also use a network mapping technique by using open source tools like cartoreso [87] or traceroute.

The last step is collecting web servers that could be attacked. The major component of the attack traffic is short-lived flows, and so an attacker needs web servers from which bots could request files via HTTP. It is assumed that bots can exploit a CAPTCHA or use websites that do not employ CAPTCHA [88]. There are simple scripts that can estimate the size of web pages [89]. A botmaster can set a limit of the web page size between 100KB and 1MB for a bot to request. A larger number of web servers will allow a sparser distribution of the attack traffic. This would make attack traffic realistic and evade detection from systems which target sudden increase in the traffic to a destination address. An attacker should avoid websites which use CDNs as it could lead to packets going to proxy web servers at the edges of ISPs. The largest CDN provider akamai claims to carry 20% of web traffic, i.e., 80% of web traffic does not pass through akamai. Most of the government websites, university websites, and small websites cannot afford CDNs.

5.2.2 Quiet Attack Execution Phase

Under normal circumstances, most of the DSL download rates are in the range of 1-2 Mbps for a typical home customer. Now, consider an OC192 (10Gbps) link under the attack. To fill up 10Gbps, 20,000 bots should be online, assuming each sending at the minimal rate of 500Kbps. In the Internet, there are easily around millions of websites, and so it can be assumed that a botmaster uses 20,000 websites in the quiet attack. The webpages (*p*) of websites (*W*) used in the attack are accessed by 20,000 bots as shown in the Figure 5.1.

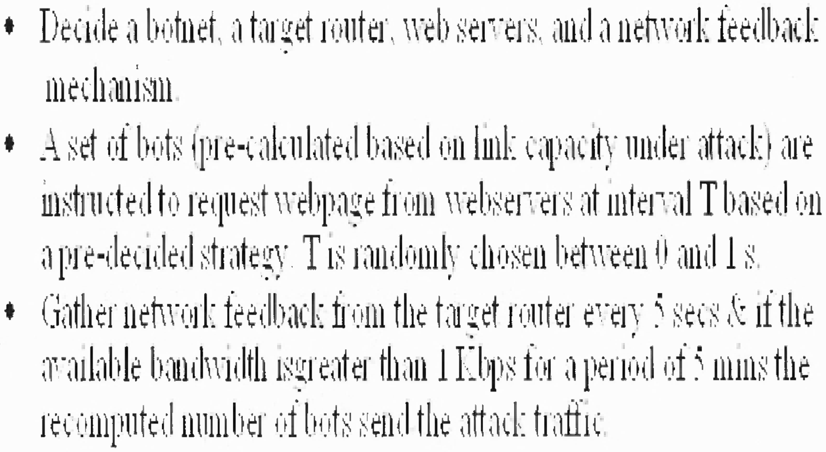
$$[W_1p_1W_2p_1W_3p_1\ldots W_{20000}p_1][W_1p_2W_2p_2W_3p_2\ldots W_{20000}p_2]\ldots$$

$$\langle ----1st\,sec----\rangle\langle ----2nd\,sec----\rangle\langle --$$

Figure 5.1 Webpage request strategy in a quiet attack.

Thus, every second each website only experiences one webpage worth of traffic which is clearly non-anomalous for any IDS at the website. For a signature based IDS at a router, there is nothing anomalous in packet contents or patterns. For an anomaly based IDS at a router, there is nothing anomalous if such a traffic pattern consisting of so many unique flows is occurring daily. Typically, there are millions of web flows like the attack short-lived TCP flows passing daily via a major router of any ISP. If an attacker has more than 20,000 websites, then the quiet attack can become much more difficult to be tracked and detected at the router.

While launching this attack, a botmaster will need network feedback on the amount of traffic at the target link. To get network feedback, a botmaster can use a tool like abget. The botmaster will initiate new bot requests at every interval T depending upon the network feedback. The network feedback is useful if ISPs do load sharing so that the botmaster can sense the condition of congestion, and add more bots to send additional attack traffic. Available bandwidth of more than 1000bps for more than 5mins is considered to be a threshold to add more attack short-lived TCP flows. A botmaster will re-use capacity estimation techniques like capprobe to approximate the size of the bottleneck link to adjust the number of bots. ISPs typically do not use dynamic load sharing techniques as it leads to some negative effects although they can advertise different CIDR blocks to peering ISPs for load sharing using BGP [9]. The attack model can be summarized as an algorithm, as shown in Fig. 5.2.



- Decide a botnet, a target router, web servers, and a network feedback mechanism.
- A set of bots (pre-calculated based on link capacity under attack) are instructed to request webpage from webservers at interval T based on a pre-decided strategy. T is randomly chosen between 0 and 1 s.
- Gather network feedback from the target router every 5 secs & if the available bandwidth isgreater than 1 Kbps for a period of 5 mins the recomputed number of bots send the attack traffic.

Figure 5.2 Quiet DDoS attack algorithm.

5.3 Simulation Results

Ns2 is used to demonstrate the feasibility of the proposed attack model. The dumbbell topology is used to focus on the bottleneck queue, and a provision is also facilitated to modify parameters such as the access link bandwidth, the delay, and the number of clients and servers. The bottleneck link has a capacity of C=10 Mbps and each of the access links has a capacity of 100 Mbps. The link delay of the client-side access links is uniformly distributed between 50 and 100ms to simulate the varying roundtrip-time for each connection; all other links have a delay of 10ms. The router has a simple droptail queue with size equal to 500 packets; following the specification that the buffer size should be twice that of the delay bandwidth product. The legitimate traffic consists of five FTP flows and HTTP traffic which is generated by PackMime in ns2. The HTTP traffic rate of 40 connections/sec is used to generate an average traffic of approximately 30% of the bottleneck link. Thus, the legitimate traffic contains the mix of short-lived and long-lived flows that is found in the Internet where measurements show that short-lived flows account for 30-40% of the traffic volume (in bytes), and the rest is occupied by long-lived flows. In the absence of the attack, the available bandwidth at the bottleneck queue is always negligible.

The attack traffic consists of a set of 20 ftp flows, each transferring a small file of size 750KB to simulate bots accessing webpages. These attack flows are introduced at every interval of T from different nodes, thus simulating bots of a botnet. The time period T is kept random between 0s and 1s to remove the deterministic pattern. To simulate a real botnet, nodes are simulated as zombies at 1, 2, 3, and 4 hops away from the target router. Each of the nodes has variable access link delays to the target router. Pathchirp is used to gather the available bandwidth at the bottleneck. The effect of attack traffic on ftp flows is shown in Figure 5.3. The throughput of ftp flows is reduced by almost 70% when there is attack. It was observed that the quiet attack does not affect the web traffic significantly due to lack of burstiness, but it is conjectured that increasing more short-lived flows can be damaging. This investigation is left as part of the future work.

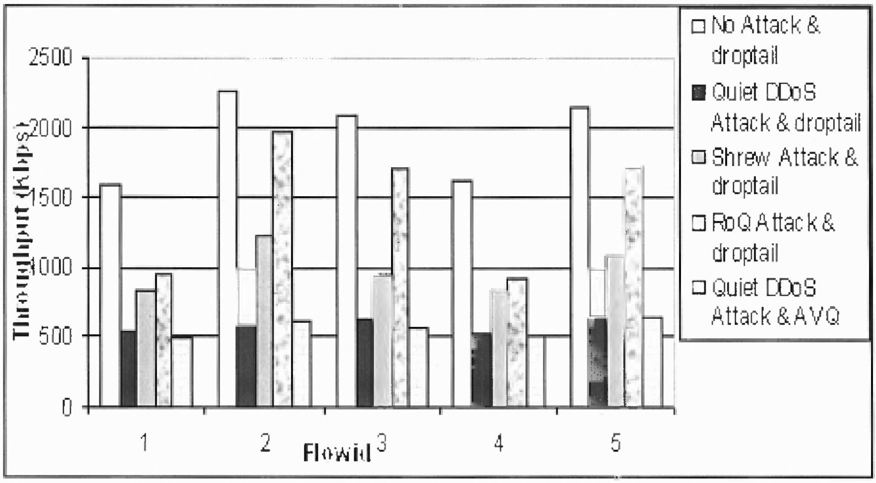


Figure 5.3 Effect of Quiet DDoS attack on throughput.

Experiments show that the quiet attack cannot be mitigated by adaptive queue management system like AVQ since the attack traffic adapts to the network congestion which is a property of the TCP protocol. The throughput of long-lived flows under AVQ is shown in Figure 5.3. The aggregate congestion control scheme pushback was also used to mitigate the quiet attack. The pushback scheme also failed because the attack traffic uses TCP and network feedback. For the pushback scheme, a slightly different topology with four routers between sources and sinks was used. The network traffic was composed of a legitimate TCP flow and the attack traffic. The throughput of the TCP flow was 277 Kbps with the attack and pushback scheme, and it was 954Kbps without the attack.

5.4 Existing Attack Models

The new breed of DoS attacks such as Shrew attacks and RoQ attacks rely on sending high rate UDP traffic periodically. Their deterministic periodic pattern and high rate make them detectable, and several detection systems have been proposed including contributions of this dissertation discussed in Chapters 3 and 4. TCP vs TCP attack [12] is similar to the shrew attack, but it uses short-lived TCP flows instead of UDP. It suffers from the same issues of sudden high rate and deterministic periodic attack pattern. Thus, in a Shrew and a RoQ attack, an attacker sends traffic at a rate equal to the target capacity to induce packet drops while the proposed quiet attack does not impose such requirement on the aggregate attack rate, thereby making the traffic less bursty. A typical DDoS attack sending high rate UDP, ICMP or TCP SYN packets continuously can affect web traffic, but these attacks can be detected by detection systems like pushback. DDoS attacks such as the SYN attack have distinguishable traffic characteristics like abnormal quantities of SYN packets as compared to normal traffic. ICMP and UDP attack traffic can be easily detected as they normally occupy a small percentage of the regular traffic as compared to the TCP traffic. The proposed sophisticated quiet attack model based on reconnaissance and execution is unique, and is adapted to be legitimate-like. Since no single server is under constant attack, the web servers are kept in the dark while ISPs who manage routers are made to believe that their routers are under real congestion. Finally, the simulation and comparison results demonstrate that the proposed attack model performs better than other attack models which have been reported in the literature so far in terms of evading detection.

5.5 Summary

In this chapter, a new DDoS attack model was proposed by using botnets that is evadable and can be easily mistaken as real congestion. The proposed quiet attack model incorporates several novel tactics to evade detection. The defense mechanisms for botnet mitigation, such as better CAPTCHAs, are an important part of the defense strategy against the quiet attack.

CHAPTER 6

THE PERFECT STORM

6.1 Attack Model Design Philosophy

This chapter describes the perfect attack model, which is a variant of quiet attack model described in the previous Chapter. The attack is first described by the scenario shown in Figure 6.1: a botnet with potentially millions of zombies sending data towards a target, in this case a router. One zombie acts as the botmaster m to coordinate the attack traffic of n zombies in the botnet. The bottleneck router is located at the server-side edge of the server that is assumed to be under a DDoS attack. The capacity of the outgoing link of the router is denoted as C, and the available bandwidth at a given time t is denoted as $B(t)$.

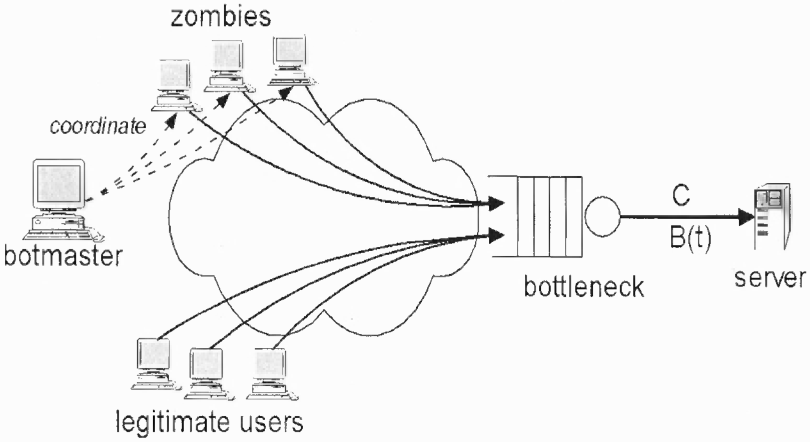


Figure 6.1 The network model.

Three innovative concepts towards designing the perfect attack are outlined, which distinguishes the perfect attack from existing sophisticated attacks:

- Use TCP along with a small amount of UDP traffic as the attack traffic.
- Incorporate network feedback mechanism in the attack process.
- Distribute the attack sources by using botnets to launch the attack.

The proposed attack model consists of two parts: an initial, short, deterministic high-rate pulse, and a feedback-driven sustained attack period with a network-adaptive attack rate, as depicted in Figure 6.2.

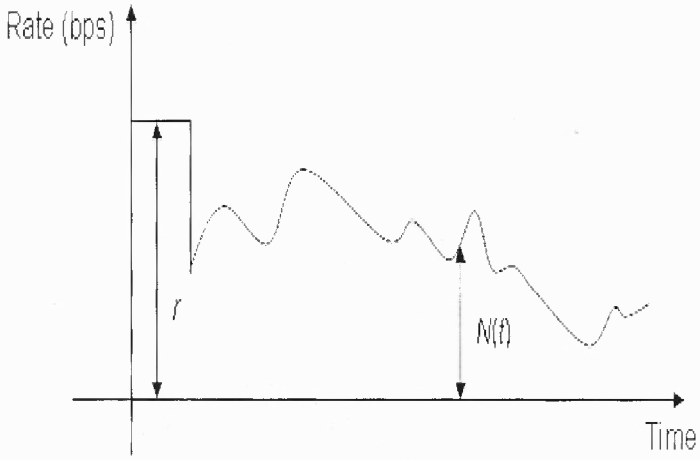


Figure 6.2 Attack traffic for a perfect DDoS attack.

To fulfill the above criteria, a perfect attack is envisioned to be executed as follows. The attack traffic is injected from a botnet towards a target with a bottleneck queue as described in Figure 6.2. Initially, a high-rate pulse is sent at a rate r for a duration of τ that overflows the buffer such that all packets are dropped. This pulse is similar to a shrew pulse, with the main difference that it only occurs at the beginning, once or twice, to drop all the packets in the queue. Thus, after this pulse, it is assumed that all long-lived TCP flows are in a timeout state, and thus do not send traffic. The only legitimate traffic that arrives immediately after the pulse is flows that are being established.

Thus, in the phase after the pulse, the attacker must fill the bottleneck queue with its own traffic as fast as possible and to sustain this level. Ideally, this filling ensures that only a small fraction of legitimate packets ever passes the bottleneck. Such a high drop rate per flow implies that (i) a large number of SYN packets are dropped, (ii) TCP flows experience packet loss already in slow start, and (iii) other non-TCP traffic incurs significant packet loss. Thus, in this second phase, the attack traffic is sent according to the following pattern. Denote $B(t)$ as the available bandwidth at the bottleneck and C as the link capacity of the attack target link. Then, the sustained attack traffic $N(t)$ is:

$$N(t) = C + B(t) \tag{6.1}$$

Under the assumption that all or the vast majority of the bottleneck bandwidth is consumed by the attack traffic, Equation (6.1) aims at maintaining a steady consumption. This attack traffic is the TCP traffic at a rate equivalent to the link capacity C. To fill the bottleneck and to compensate for drops in the TCP attack rate, UDP traffic at a rate of $B(t)$ is injected into the network. Note here that the UDP traffic is a function of the available bandwidth rather than the capacity as in a shrew attack or a RoQ attack. $N(t)$ is periodically updated and adjusted with the time period T. The update contains a rate adaptation but also the chance to exchange the zombies in the attack to create a diverse traffic pattern from different traffic sources. Thus, at the bottleneck, $N(t)$ creates a traffic pattern consisting of a superposition of many TCP flows, with a small fraction of UDP traffic, whose sources vary over time. After each period T, a new set of TCP flows are directed at the bottleneck link. These TCP flows begin in the slow start phase of TCP and end in the slow start phase as well. It is important to keep the attack TCP flows in slow start because they have been shown to affect long-lived TCP flows on shorter timescales, and also introducing a new TCP connection allows the congestion window to grow rapidly, otherwise attack TCP flows will enter congestion avoidance phase and will try to share the bandwidth with the legitimate TCP flows. The period T can be random so as to make detection difficult particularly for systems that try to find the deterministic attack pattern. Note that this interplay between TCP and UDP further complicates the detection. In contrast to the shrew attack where the repetitive pulses can be relatively detected, the perfect attack does not create such a repetitive pattern. Instead, the dynamics lead to an ever changing traffic pattern that cannot be observed and captured by the defense. In comparison between a normal TCP vs. a TCP attack [12], besides the non-determinism, the perfect attack monitors the bottleneck and injects traffic at a specific target rate to achieve a high impact. The TCP vs. TCP attack [12] cannot launch an attack on a bottleneck link as it relies on TCP which cannot force other packets to be dropped, but in the proposed attack model with the minimal use of UDP traffic, higher impact can be achieved at any link irrespective of its congestion state. In the perfect attack, more attack traffic is sent as compared to a shrew attack or an RoQ attack, but the shrew and RoQ attacks rely on UDP traffic to cause damage and the proposed model relies on network adaptive TCP traffic. In Section 6.3, it is shown that the TCP attack traffic is legitimate web requests, and is thus even more difficult to be detected.

In summary, the perfect DDoS attack can be visualized as a superposition of many attack flows, as seen in Figure 6.2. All attack parameters can be dynamically adjusted based on the feedback gathered during the attack. Therefore, the attack shows no deterministic pattern that can be captured by a defense mechanism. Moreover, the feedback and the primary use of TCP, with intermittent UDP fillings, ensure that the attack leads to a denial of service and not just a degradation of quality as in RoQ attacks.

6.2 Simulation Results

This section evaluates the impact of the perfect attack via simulations. In particular, the impact of the perfect attack on legitimate user traffic is evaluated. Simulations were conducted to compare the impact of the perfect attack with the situation where there is no attack as well as against the shrew and RoQ attacks. A small proof-of-concept experiment is demonstrated that validates the idea of using short-lived TCP flows against long-lived TCP flows as the attack flows in the perfect attack model described in the earlier section. A simple dumbbell topology is used in ns-2 with a legitimate long-lived TCP flow and the perfect attack flows. The bottleneck link has a capacity of $C = 1$ Mbps, and each of the access links has a bandwidth of 10 Mbps. Two initial bursts are sent at the rate of $r = 1$ Mbps for a period of 0.4 sec. Then, the attack traffic consists of the feedback-based attack cycle with an attack rate of $N(t)$ as defined in Equation (6). The $N(t)$ consists of a long-lived TCP flow in case 1, and a short-lived TCP flow in case 2 sent at random intervals. Case 3 does not contain the perfect attack. The throughput of the legitimate TCP flow is 558Kbps, 158Kbps, and 970 Kbps in case 1, 2, and 3, respectively. This proof-of-concept experiment shows that the use of short-lived TCP flows is an important condition to achieve the desired impact.

For the detailed analysis, the simulation settings for the dumbbell topology are as follows. This topology allows us to focus on the bottleneck queue, but it still provides the ability to modify parameters such as the access link bandwidth, the delay, and the number of clients, servers, and zombies. The bottleneck link has a capacity of $C = 10$ Mbps, and each of the access links has a bandwidth of 100 Mbps. The link delays of the client-side access links follow a uniform distribution in the range of 50 to 100ms to simulate the varying roundtrip-time for each connection; all other links have a delay of 10ms. The router has a simple droptail queue with size equal to 500 packets, following the specification that the buffer size should be twice that of the delay bandwidth product. The legitimate traffic consists of 5 FTP flows, 4 VoIP flows, and HTTP traffic generated by PackMime. The packmime HTTP traffic generator is incorporated in ns-2. During the 600 seconds of the simulation, PackMime generates around 14000 HTTP connections at the rate of 40 connections per second. The rate of 40 is used to generate an average traffic of approximately 30% of the bottleneck link. Thus, the legitimate traffic is consistent with measurements which show short-lived flows account for 30-40% of the traffic volume (in bytes), and the rest is occupied by long-lived flows. The 4 VoIP flows that are modeled as G711 codecs with FEC (Forward Error Correction) are sent at a rate of 96 Kbps by using the exponential on-off traffic model. From this host, two initial bursts are sent at the rate of $r = 15$ Mbps for a period of 0.4 sec. Then, the attack traffic consists of the feedback-based attack cycle with an attack rate of $N(t)$.

Given the fact that TCP is used as the primary attack traffic and that the sources are ideally distributed, the probability of identifying one of the flows as an attacking flow is very low. To monitor the status of the bottleneck queue and thereby to get an estimate of the available bandwidth, an active monitoring approach by using pathchirp is chosen. The available bandwidth measurement will be done by a botmaster in practice. The attack rate is updated at an interval of $T = 1$ sec.

6.2.1 Impact

The impact of the perfect attack on the legitimate traffic is assessed in this Subsection. Moreover, the perfect attack is compared against the impact of a shrew and RoQ attack. In particular, for the shrew attack, the burst length is set to 0.4 sec, the burst period is set to 1 sec, and the attack rate is set to 15 Mbps. For the RoQ attack, the burst length is set to 0.4 sec, the burst period is set to 7 sec, and the attack rate is set to 15 Mbps.

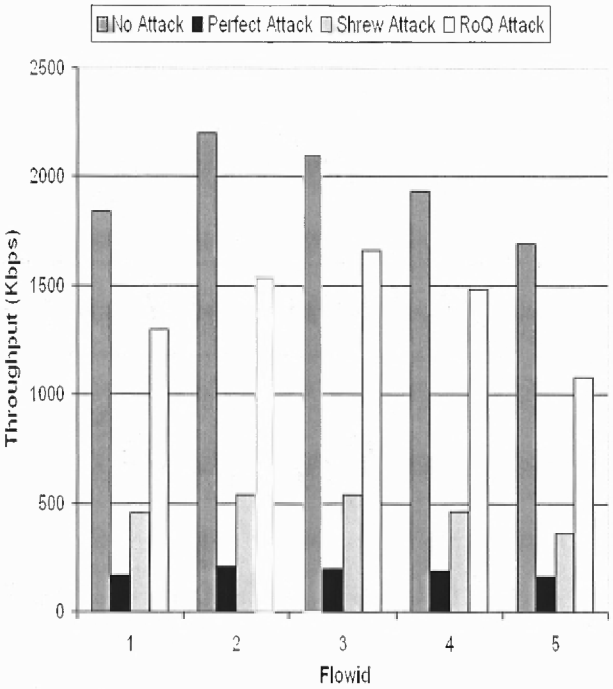


Figure 6.3 Attack effects on throughput of long lived TCP flows.

6.2.2 Impact on long-lived FTP flows

Impact on the 5 long-lived elephant FTP flows is analyzed in this Subsection. Figure 6.3 shows that the average throughput of each flow in kbps. Without an attack, the throughput varies between 1.6 and 2.2 Mbps as a function of the RTT between the client and the server. Using the perfect attack, the throughput of the FTP flows is reduced to a dismal percentage. The throughput is reduced to less than 200 kbps - a degradation by more than an order of magnitude as compared to the throughput when no attack is present. Next, in comparison of the perfect attack to a shrew attack, it is noted that the throughput is reduced by a factor of 2-3. Finally, the RoQ attack is clearly only reducing the quality of a long-lived flow, but is far from an efficient DoS attack because the average throughput is not even degraded by a factor of two as compared to the situation without an attack. This implies that the perfect attack also reduces the throughput as compared to an RoQ attack by one order of magnitude.

6.2.3 Impact on VoIP flows

Impact on the VoIP flows is analyzed in this Subsection. The average packet loss, delay, and jitter are used as metrics to quantify the impact. Figures 6.4-6.6 show the average packet loss, the delay and the jitter of the 4 VoIP flows as a function of the 4 attack types. First, without attack, the packet loss rate is 0. With the perfect attack, the packet loss increases to an average of 20%. In comparison, the shrew attack incurs a packet loss of 18%, and the RoQ attack incurs a packet loss of 3%. Besides the high average packet loss induced by the perfect attack, a high degree of burstiness was also noted in the packet loss pattern. Both the high average packet loss rate as well as the burstiness severely degrade the quality of the voice conversation. Similarly, the perfect attack also leads to extremely high delays around 450ms and high jitter values around 80ms for VoIP calls as shown in Figures 6.5 and 6.6. Delay of less than 150ms and jitter of less than 20ms are acceptable for VoIP calls.

6.2.4 Impact on short-lived HTTP flows

Impact on short-lived HTTP flows is evaluated in this Subsection. The packmime web traffic model provides statistics for the web traffic. It generates requests at an average rate of 40 connections per second, and the average reply size is 6KBytes.

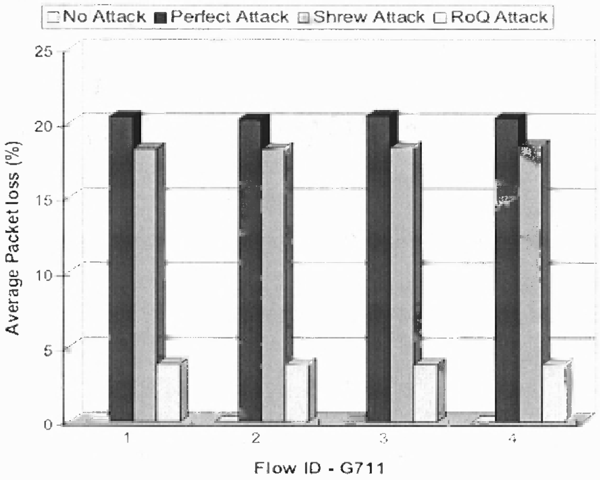


Figure 6.4 VoIP packet loss.

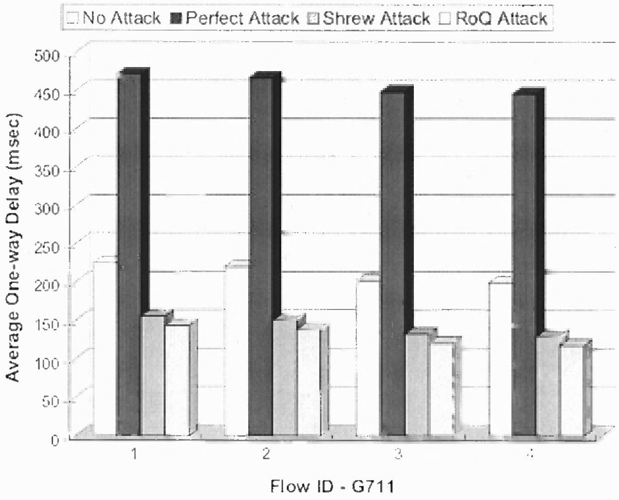


Figure 6.5 VoIP average one-way delay.

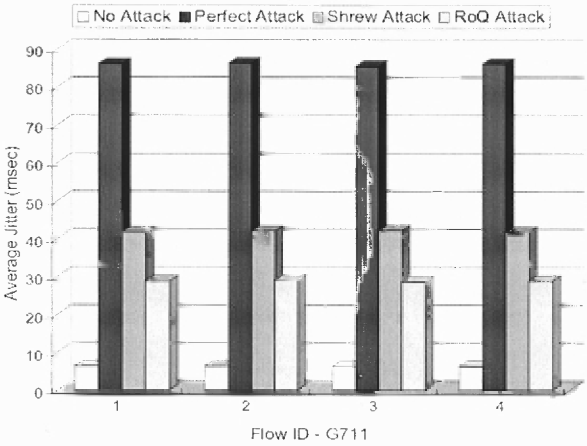


Figure 6.6 VoIP jitter.

Figure 6.7 quantifies the impact on the response time as a cumulative distributive function (CDF) plot on a log-scaled x-axis. The response time is measured from the sending of the request until the reply is successfully received by the client. Figure 6.7 clearly shows that the perfect attack outperforms the other attacks as well as the situation where no attack is present. As shown in Figure 6.7, the median response time without an attack is as low as 882 msec. With the perfect attack, it increases to 3148 msec, an almost 4-fold increase. As compared to the other attacks with medians of 1085 msec for the shrew attack and 750 msec for RoQ attacks, the perfect attack clearly outperforms the other attacks as well. In particular, the RoQ attacks have no impact on the median response time as the time is even lower than that without an attack. While looking at the bare numbers, an increase by a factor of 4 with the perfect attack as compared to no attack may limit the amount of excitement; the impact becomes obvious when considering user satisfaction. In particular, user satisfaction is high if Web page downloads are completed within 5 sec [91]. Now, without attack, 98% of the requests are completed, whereas only 70% complete with the perfect attack. Even worse, looking at the download time of 1 sec, 58% of the pages are successfully downloaded without an attack, whereas a dismal 10% are completed with the attack.

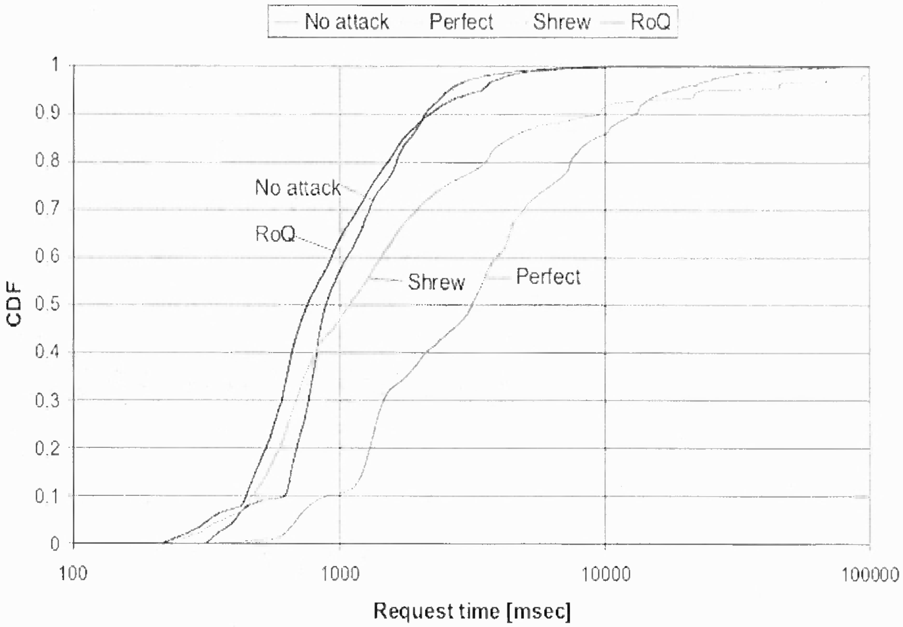


Figure 6.7 Impact on HTTP flows.

The above evaluation shows that the proposed attack scheme is significantly reducing the performance of long-lived TCP flows, short-lived TCP flows as well as VoIP flows. Since these categories occupy the vast majority of traffic in the Internet in terms of byte volume as well as the number of flows, it is shown that the proposed attack is highly efficient in denying service to legitimate users. The exact impact must be traded off against the probability that an attack flow is detected. At this point, it is argued that the chosen parameters for the perfect attack are far from aggressive. Therefore, a real attack can easily tune the attack parameter to increase the impact without increasing the probability of being detected.

6.2.5 Randomizing T

The effect of randomizing the period of update T of $N(T)$ defined in Section 5.1 is analyzed here. The parameter, T, determines when the new attack flows should enter the network. By randomizing T, the attack detection will become more difficult for detection systems that rely on finding a particular attack pattern. Impact on the long-lived TCP flows is shown in Figure 6.8. It shows that by choosing a random value between 1s and 6s leads to impact closer to that of T=1s. A little increase in T will naturally decrease the impact to some degree, but is no way closer to the no attack case in Figure 6.8.

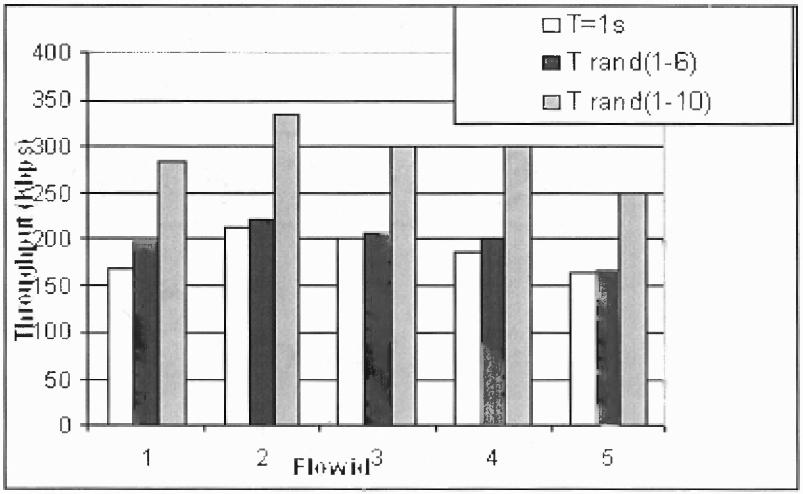


Figure 6.8 Attack effects by randomizing update interval.

6.2.6 Router Defense Mechanisms

The ability of router mechanisms such as Active Queue Management or pushback schemes to identify and mitigate the effects of the perfect attack is analyzed in this Subsection. Among the set of proposed AQM schemes, such as RED, RED-PD, and BLUE, the Adaptive Virtual Queue (AVQ) scheme is used in preliminary studies to investigate its ability to detect a perfect attack. AVQ is a relatively novel scheme that outperforms most of the other AQM schemes like RED, and hence it was chosen. To assess the impact of AVQ, the simulations above are repeated, but with AVQ enabled. The adopted parameters are λ=0.98 and α=0.15. Figure 6.9 compares the throughput without the attack traffic to that with the perfect attack with AVQ. The results show that AVQ is unable to mitigate the attack because the perfect attack sends a continuous stream of attack traffic (in contrast to the shrew bursts), and the traffic volume is comparable to the available bandwidth. The additional UDP traffic needed to compensate for TCP throughput loss is not sufficient to trigger a reaction by AVQ. The pushback scheme has also been proposed to mitigate the impact of DDoS attacks, but its approach is different from AQM. Pushback, also known as aggregate congestion control (ACC), pushes back DDoS attack traffic from the victim towards the source network of the attack.

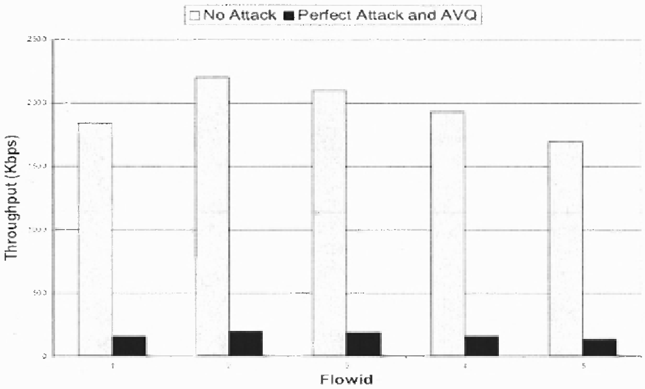


Figure 6.9 Impact with AVQ.

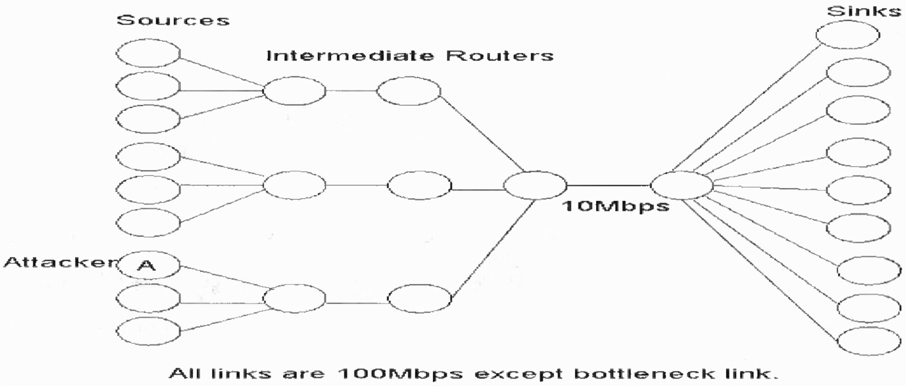


Figure 6.10 Topology for pushback simulation.

To assess the ability of a pushback scheme to defend against the perfect attack, the simulation topology is modified as depicted in Figure 6.10. In particular, intermediate routers are necessary to aggregate traffic and to push back potential attack traffic towards the source. Note, however, that the traffic is sent from a single attacker host only. Therefore, this scenario should be considered as an upper bound to the effectiveness of the pushback schemes. In contrast, the perfect attack will be launched from distributed zombies in a botnet. As for the legitimate traffic, only seven long-lived FTP flows were used, i.e., while VoIP and HTTP flows were omitted.

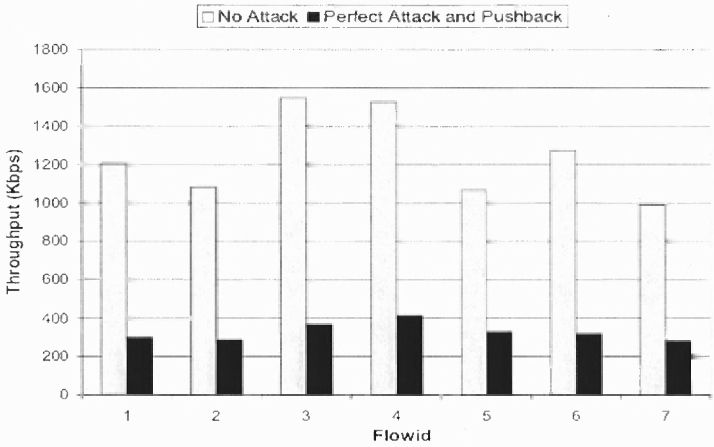


Figure 6.11 Impact with pushback.

Figure 6.11 compares the situation without attack to that with the perfect attack. It is noted that the throughput of the legitimate flows is again severely degraded due to the attack. Or, in other words, pushback fails to identify the attack traffic because it is stealthy, i.e., because it follows TCP fairness rules and because network feedback allows the attack traffic to remain at a low rate.

In summary, preliminary studies have demonstrated the potency of the perfect attack that can effectively evade existing detection mechanisms.

6.3 Perfect Attack in the Internet

The tools required to instigate the perfect attack are readily available. A simple traceroute from publicly available traceroute servers reveals IP addresses of routers along the path to a web server or any server to be attacked. Bots have been known to engage with web servers to mimic real web requests to occupy web server's resources like bandwidth and CPU [80]. Such requests are hard to distinguish, and as Kandula et al. [80] suggested that the only way to classify a request of a bot from a human is to use some form of a Turing test like CAPTCHA's. However, CAPTCHA's are not foolproof, and attackers regularly succeed in breaking new CAPTCHA designs [88]. Many websites, which do not implement CAPTCHA's like news websites, can be easily attacked.

The execution of the perfect attack will require the use of a botnet similar to the widely known clickbot [92] used against the Google's ad mechanism to generate false clicks on a website for a financial fraud. To execute the perfect attack, it is envisioned such a bot would issue web requests from a web server which is along the path from the bot to the web server. These web requests are issued by bots at every interval *T* defined before in Section 6.2. For every interval *T,* the botmaster also gets a chance to exchange the bots in the attack to create a diverse traffic pattern from different traffic sources. In the perfect attack, such web requests will be short-lived TCP flows acting as the attack TCP flows. Before an attack begins, an attacker can estimate different sizes of pages that may be requested, and so a certain amount of throughput is achieved by an individual TCP flow for a short period of time. Ultimately, the bulk of such requests would contribute the total amount of the attack TCP traffic. The abget or pathchirp tool performs the available bandwidth measurement which is done by the botmaster who can then instruct bots to just send attack traffic. Active bots in a botnet are shown to be present in the same time zones, thereby readily available for the attack. The initial UDP bursts and feedback based attack UDP traffic can be sent from any machine on which a botmaster has more control.

6.4 Summary

This chapter outlines future potential strategies towards the "perfect" DDoS attack. As compared to current attacks, the proposed model incorporates several techniques that allow the attack to be undetectable while achieving the denial of service at the target. The model includes the exploitation of feedback from the network, the use of a mixture of TCP and UDP, and the distribution and coordination of entire botnets. Simulation results have demonstrated the potential lethality of the perfect attack. By making this work public, this dissertation hopes to inspire others to work on thwarting such a grave threat that would ultimately benefit the society.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation, router-based approaches have been proposed to detect the stealthy low rate DoS and RoQ attacks which use IP address spoofing or botnets. This work addresses the IP address spoofing and botnet problem in the context of the low rate DoS and RoQ attacks, and proposes an effective and realizable solution to defend against RoQ attacks. The effectiveness of the proposed approach has been demonstrated via extensive experiments.

This dissertation have also proposed two new DDoS attack models by using botnets that are evadable and can be easily mistaken as real congestion. Botnet mitigation techniques, such as better CAPTCHAs, are an important part of the defense strategy against the quiet attack and perfect attack. Future work will be focused on building defenses against such grave threats.

REFERENCES

1. DDoS news. Retrieved Mar, 19, 2009 from the WWW: http://staff.washington.edu/dittrich/misc/ddos/

2. C. Douligeris and A. Mitrokotsa (2004). DDoS Attacks and Defense Mechanisms: classification and state-of-the-art. Computer Networks, Vol. 44 No. 5, 643-666.

3. D. Dagon, Z. Zhou, W. Lee (2006). Modeling Botnet Propagation Using Time Zones. NDSS.

4. UDP port Denial-of-Service Attack. Retrieved May, 1, 2007 from the WWW: http://www.cert.org/advisories/CA-1996-01.html

5. Smurf attack. Retrieved Jan, 1, 2008 from the WWW: http://www.nordu.net/articles/smurf.html

6. W. Eddy (2007). TCP SYN Flooding Attacks and Common Mitigations. IETF RFC 4987.

7. Push + Ack attack. Retrieved Jan, 1, 2005 from the WWW: http://www.csie.ncu.edu.tw/~cs102085/DDoS/protocolexploit/push%2Back/description.htm

8. Ping of Death attack. Retrieved Jan, 1, 2005 from the WWW: http://insecure.org/sploits/ping-o-death.html

9. A Lively Market, Legal and Not, for Software Bugs. Retrieved Jan, 30, 2007 from the WWW: http://www.nytimes.com/2007/01/30/technology/30bugs.html?ex=1327813200&en=99b346611df0a278&ei=5088&partner=rssnyt&emc=rss

10. A. Kuzmanovic and E. Knightly (2003). Low-Rate TCP -Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants). ACM SIGCOMM, 75-86.

11. M. Guirguis, A. Bestavros, and I. Matta (2004). Exploiting the Transients of Adaptation for RoQ Attacks on Internet Resources. IEEE ICNP, 184-195.

12. S. Ebrahimi-Taghizadeh, A. Helmy, and S. Gupta (2005). TCP vs. TCP: a Systematic Study of Adverse Impact of Short-lived TCP Flows on Long-lived TCP Flows. IEEE INFOCOM, 926-937.

13. X. Luo and R. K. C. Chang (2005). On a New Class of Pulsing Denial-of-Service Attacks and the Defense. NDSS.

14. A. Shevtekar and N. Ansari (2006). Do Low Rate DoS Attacks Affect QoS Sensitive VoIP Trafic? IEEE ICC, 2153-2158.

15. R. Chertov, S. Fahmy, and N. Shroff (2006). Emulation versus Simulation: A case study of TCP-Targeted Denial of Service Attacks. _Tridentcom_, 316-325.

16. V. Paxon and M. Allman (2000). Computing TCP's Retransmission Timer. _IETF RFC 2988_.

17. R. Mahajan, S. Floyd, and D. Wetherall (2001). Controlling High-Bandwidth Flows at the Congested Router. _IEEE ICNP_, 192-201.

18. Y. Xu and R. Guerin (2005). On the Robustness of Router-based Denial-of-Service (DoS) Defense Systems. _ACM Computer Communications Review, Vol. 2_, 47-60.

19. S. Floyd and V. Jacobson (1993). Random Early Detection Gateways for Congestion Avoidance. _IEEE/ACM Transactions on Networking, Vol.1, No. 4_, 397-413

20. R. Beverly and S. Bauer (2005). The Spoofer Project: Inferring the Extent of Source Address Filtering on the Internet. _USENIX SRUTI_, 53-59.

21. P. Ferguson and D. Senie (2001). Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Address Spoofing. _IETF RFC 2827_.

22. W. R. Cheswick (1992). An Evening with Berferd, in which a Cracker is lured, endured, and studied. _USENIX Winter Conference_ 163-174.

23. Snort IDS. Retrieved Mar, 1, 2004 from the WWW: http://www.snort.org

24. J. B. D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran, and R. K. Mehra (2001). Proactive Detection of Distributed Denial of service Attacks using MIB Variables – a Feasibility Study. _IFIP/IEEE Integrated Symposium of Network Management_, 1-14.

25. W. Lee and S. Stolfo (1998). Data Mining Approaches for Intrusion Detection. _USENIX Security Symposium_, 79-93.

26. J. Mirkovic, G. Prier, and P. Reiher (2002). Attacking DDoS at Source. _IEEE ICNP_, 312-321.

27. A. Belenky and N. Ansari (2003). On IP Traceback. _IEEE Communications Magazine Vol. 41, No. 7_, 142-153.

28. Z. Gao, and N. Ansari (2005). Tracing Cyber Attacks from Practical Perspective. _IEEE Communications Magazine Vol. 43, No. 5_, 123-131.

29. S. Savage, D. Wetherall, A. Karlin, and T. Anderson (2001). Network Support for IP Traceback. _IEEE ACM/Transactions on Networking Vol. 9, No. 3_, 226-237.

30. A. Belenky and N. Ansari (2003). IP Traceback with Deterministic Packet Marking. IEEE Communication Letters Vol. 7, No. 4, 162-164.

31. S. Bellovin (2000). ICMP Traceback Messages. IETF RFC Draft.

32. R. Stone (2000). CenterTrack: An IP Overlay Network for Tracking DoS Floods. USENIX Security Symposium,199-212.

33. A. C. Snoeren, C. Patridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer (2001). Hash-based IP Traceback. ACM SIGCOMM, 3-14.

34. S. Kunniyur and R. Srikant (2004). An Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. IEEE/ACM Transactions on Networking, Vol. 12, No. 2, 286-299.

35. R. Mahajan, S. Bellovin, S. Floyd, J. Ionnadis, V. Paxson, and S. Shenker (2002). Controlling high-bandwidth aggregates in the network. ACM SIGCOMM CCR, Vol. 32, No. 3, 62-73.

36. SYN cookies. Retrieved Mar, 19, 2007 from the WWW: http://cr.yp.to/syncookies.html

37. H. Sun, J. C. S. Lui, and D. K. Y. Yau (2004). Defending Against Low-rate TCP Attack: Dynamic Detection and Protection. IEEE ICNP, 196-205.

38. G. Yang, M. Gerla, and M. Y. Sanadidi (2004). Randomization: Defense against Low-Rate TCP-targeted Denial-of-Service Attacks. IEEE Symposium on Computers and Communications, 345-350.

39. Y. Chen, K. Hwang, and Y. Kwok (2006). Collaborative Detection and Filtering of Shrew DDoS Attacks using Spectral Analysis. Journal of Parallel and Distributed Computing. Retrieved Feb 01, 2009 from the WWW: http://gridsec.usc.edu/files/publications/JPDC-Chen-2006.pdf

40. S. Sarat and A. Terzis (2005). On the Effect of Router Buffer Sizes on Low-Rate Denial of Service Attacks. IEEE ICCCN, 281-286.

41. Y. Kwok, R. Tripathi, Y. Chen, and K. Hwang (2005). HAWK: Halting Anomaly with Weighted ChoKing to Rescue Well-Behaved TCP Sessions from Shrew DoS Attacks. ICCNMC, 2-4.

42. Y. Xu and R. Guerin (2006). A Double Horizon Defense for Robust Regulation of Malicious Traffic. SecureComm, 1-11.

43. H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson (2003). RTP: A Transport Protocol for Real-Time Applications. RFC3550.

44. B. Goode. (2002). Voice Over Internet Protocol (VoIP). IEEE Vol. 90, No. 9, 1495-1517.

45. T. J. Walsh and R. Kuhn (2005). Challenges in Securing Voice over IP. IEEE Security and Privacy Magazine, Vol. 3, No. 3, 44-49.

46. M. Fomenkov, K. Keys, D. Moore, and K. Claffy (2004). Longitudinal study of Internet traffic from 1998-2003. *Winter International Symposium on Information and Communication Technologies*, 1-6.

47. C. Fraleigh, and S. Moon, and B. Lyles, and C. Cotton, and M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot (2003). Packet-Level Traffic Measurements from the Sprint IP Backbone. IEEE Network Magazine, Vol. 17, No. 6, pp. 6-16.

48. P. Calyam, M. Sridharan, W. Mandrawa, and P. Schopis (2004). Performance Measurement and Analysis of H.323 Traffic. *Passive and Active Measurement Workshop (PAM)*, 137-146.

49. P. Calyam and C. G. Lee (2005). Characterizing voice and video traffic behavior over the Internet. *ISCIS, Vol. 1,* 83-92.

50. A. Markopoulou, F. Tobagi, and M. Karam (2002). Assessment of VoIP Quality over Internet Backbones. IEEE Infocom, 150-159.

51. M. Yajnik, S. Moon, J. Kurose, and D. Towsley (1999). Measurement and modeling of temporal dependence in packet loss. IEEE Infocom, 345-352.

52. W. Jiang and H. Schulzrinne (2000). Modeling of Packet loss and delay and their effect on real-time multimedia service quality. NOSSDAV.

53. C. Perkins and O. Hodson (1998). Options for repair of streaming media. RFC 2354.

54. W. Jiang and H. Schulzrinne (2002). Comparison and optimization of packet loss repair methods on VoIP perceived quality under bursty loss. NOSSDAV.

55. J. Rosenberg, L. Qiu, and H. Schulzrinne (2000). Integrating Packet FEC into Adaptive Voice Playout Buffer Algorithms on the Internet. IEEE Infocom, 1705-1714.

56. R. Koodli and R. Ravikanth (2002). One-way loss pattern sample metrics. RFC 3357.

57. R. G. Cole and J. Rosenbluth (2001). Voice over IP performance monitoring. *Computer Communication Review, Vol. 31, No. 2*, 9-24.

58. S. Floyd and E. Kohler (2002). Internet Research Needs Better Models. *Usenix Hotnets*, 29-34.

59. M. Christiansen, K. Jeffay, D. Ott, F. D. Smith (2001). Tuning RED for Web Traffic. *IEEE/ACM transaction on Networking Vol. 9, No. 3*, 249-264.

60. J. Cao W. S. Cleveland, Y. Gao, K. Jeffay, F. D. Smith, and M. C. Weigle (2004). Stochastic Models for Generating Synthetic HTTP Source Traffic. *IEEE Infocom*, 1546-1557.

61. PackMime Traffic Generator. Retrieved May 01, 2007 from the WWW: http://dirt.cs.unc.edu/packmime

62. Deterlab. Retrieved May 01, 2007 from the WWW: http://www.deterlab.net

63. E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Frans Kaashoek (2000). The Click modular router. *ACM Transactions on Computer Systems Vol.18, No. 3*, 263-297.

64. A. Botta, A. Dainotti, and A. Pescape (2007). Multiprotocol and Multiplatform traffic generation and Measurement. *IEEE Infocom Demo session.*

65. J. Sommers and P. Barford (2004). Self-Configuring network traffic generation. *ACM IMC 2004*, 68-81.

66. J. Mirkovic and P. Reiher (2004). A Taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communications Review, Vol. 34, No. 2*, 39-54.

67. M. Guirguis, A. Bestavros, and I. Matta (2004). Bandwidth Stealing via Link Targeted RoQ Attacks. *CCN*.

68. K. C. Claffy, H-W. Braun, and G. C. Polyzos (1995). A Parameterizable methodology for Internet traffic flow profiling. *IEEE Journal on Selected Areas in Communication, 13*, 1481-1494.

69. N. Brownlee and K. Claffy (2002). Understanding Internet Traffic Streams: Dragonflies and Tortoises. *IEEE Communications Magazine, Vol. 40, No. 10*, 110-117.

70. A. Shevtekar, K. Anantharam, and N. Ansari (2005). Low Rate TCP Denial-of-Service Attack Detection at Edge Routers. *IEEE Communication Letters, Vol.9*, 363-365.

71. The OC48 Data. Retrieved May 01, 2007 from the WWW: http://www.caida.org/data/passive/index.xml#oc48

72. The CoralReef Software. Retrieved May 01, 2007 from the WWW: http://www.caida.org/tools/measurement/coralreef/

73. H. J. Chao, R. Karri, and W. C. Lau (2004). CYSEP – a Cyber Security Processor for 10Gbps Networks and Beyond. IEEE MILCOM, 1114-1122.

74. A. Kumar, M. Sung, J. Xu, and J. Wang (2004). Data Streaming Algorithms for Efficient and Accurate Estimation of Flow Distribution. ACM SIGMETRICS, 177-188.

75. A. Cranor, T. Johnson, and O. Spatschek (2003). Gigascope: a stream database for network applications. SIGMOD, 647-651.

76. B. Grot and W. Mangione-Smith (2005). Good Memories: Enhancing Memory Performance for Precise Flow Tracking. ANCHOR.

77. P. Manolios (2007). Bloom Filter Calculator. Retrieved May 01, 2007 from the WWW: http://www-static.cc.gatech.edu/~manolios/bloom-filters/calculator.html

78. A. Kumar and J. Xu (2006). Sketch Guided Sampling - - Using On-Line Estimates of Flow Size for Adaptive Data Collection. IEEE Infocom, 1-11.

79. 2005 Latinode Customer Case Study. Retrieved May 01, 2007 from the WWW: http://www.nextone.com/files/LatiNode%20Case%20Study.pdf

80. S. Kandula, D. Katabi, M. Jacob, and M. Berger (2005). Botz-4-Sale: Surviving organized DDoS attacks that mimic flash crowds. USENIX NSDI 287-300.

81. Z. Gao and N. Ansari (2006). Differentiating malicious DDoS attack traffic from normal TCP flows by proactive tests. IEEE Communication Letters, Vol. 10, No. 11, 793-795.

82. G. Mori and J. Malik (2003). Recognizing Objects in Adversial Clutter – Breaking a Visual CAPTCHA. IEEE CVPR, 134-141

83. R. Kapoor, L. Chen, L. Lao, M. Gerla and M. Sanadid (2004). CapProbe: A simple and accurate capacity estimation technique. ACM SIGCOMM, 67-78.

84. D. Antoniades, M. Athanatos, A. Papadogiannakis, E. Markatos, and C. Dovrolis (2006). Available Bandwidth Measurement as Simple as Running Wget. PAM.

85. V. Rebeiro, R. Reidi, R. Baranuik, J. Navratil, and L. Cottrell (2003). pathChirp: Efficient Available Bandwidth Estimation for Network Paths. PAM.

86. D. Bertsekas and R. Gallager (2001). Data Networks. (2nd ed.). Prentice Hall,180-188.

87. Cartoreso. Retrieved Jan 01, 2009 from the WWW: http://cartoreso.campus.ecp.fr/

88. Is captcha moment passing? *Wired Magazine*. Retrieved May 01, 2008 from the WWW: http://blog.wired.com/monkeybites/2008/04/is-captchas-mom.html

89. Finding suitable files in a server. Retrieved Feb 01, 2009 from the WWW: http://www.ics.forth.gr/~papadog/abget/abget_tutorial.html#scripts

90. M. Ceasar and J. Rexford (2005). BGP routing policies in the Internet. *IEEE Network, Vol. 19, No. 6* 1-6.

91. N. Bhatti, A. Bouch, and A. Kuchinsky (2000). Integrating user perceived quality into web server design. *Computer Networks, Vol. 33, No. 1-6* 1-16.

92. N. Daswani, M. Stoppelman and the Google Click Quality and Security Teams (2007). The anatomy of Clickbot.A. *HotBots,* 11-11.