Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 52 (2008) 957–970
Computer Networks
www.elsevier.com/locate/comnet

A router-based technique to mitigate reduction of quality (RoQ) attacks

Amey Shevtekar, Nirwan Ansari *

Advanced Networking Laboratory, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, United States

Received 21 February 2007; received in revised form 3 November 2007; accepted 22 November 2007
Available online 4 December 2007

Abstract

We propose a router-based technique to mitigate the stealthy reduction of quality (RoQ) attacks at the routers in the Internet. The RoQ attacks have been shown to impair the QoS sensitive VoIP and the TCP traffic in the Internet. It is difficult to detect these attacks because of their low average rates. We also show that our generalized approach can detect these attacks even if they employ the source IP address spoofing, the destination IP address spoofing, and undefined periodicity to evade several router-based detection systems. The detection system operates in two phases: in phase 1, the presence of the RoQ attack is detected from the readily available per flow information at the routers, and in phase 2, the attack filtering algorithm drops the RoQ attack packets. Assuming that the attacker uses the source IP address and the destination IP address spoofing, we propose to detect the sudden increase in the traffic load of all the expired flows within a short period. In a network without RoQ attacks, we show that the traffic load of all the expired flows is less than certain thresholds, which are derived from real Internet traffic analysis. We further propose a simple filtering solution to drop the attack packets. The filtering scheme treats the long-lived flows in the Internet preferentially, and drops the attack traffic by monitoring the queue length if the queue length exceeds a threshold percent of the queue limit. Our results show that we can successfully detect and mitigate RoQ attacks even with the source and destination IP addresses spoofed. The detection system is implemented in the ns2 simulator. In the simulations, we use the flowid field available in ns2 to implement per-flow logic, which is a combination of the source IP address, the destination IP address, the source port, and the destination port. We also discuss the real implementation of the proposed detection system.

Keywords: Denial of service (DoS); IP address spoofing; Network security; RoQ attack

1. Introduction

The CSI/FBI 2006 [1] survey showed that the denial of service (DoS) is still an issue leading to a significant revenue loss for many organizations. The low rate DoS attack poses a new threat to the Internet [2–8]. The low rate DoS attack was first publicly known from the work of [2], referred to as the Shrew attack. The RoQ attack [3] does not try to shut down the legitimate flows, but tries to reduce the quality of service experienced by them. Thus, it is even harder to defend against RoQ

* Corresponding author. Tel./fax: +1 973 596 3670.
E-mail addresses: abs6@njit.edu (A. Shevtekar), nirwan.ansari@njit.edu (N. Ansari).

1389-1286/$ - see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.comnet.2007.11.015

attacks. All these low rate types of DoS attacks can be defined by a general periodic waveform as shown in Fig. 1. A low rate type of DoS attack is characterized by three parameters, the attack period (T), the burst period or the burst length (t), and the burst rate (R). The Shrew attack [2] exploits the minimum RTO property and the exponential backoff algorithm of the TCP protocol. It works by sending a burst of attack packets with a rate greater than the target capacity for a short period like 20–200 ms at the target router every 1 s. This leads to packet drops of legitimate TCP connections, which subsequently enter timeout; as these connections attempt to retransmit lost packets after the minimum RTO of 1 s, they are again hampered by the attack traffic at the router. The connections now use the exponential back off algorithm to retransmit lost packets, attempting to retransmit packets at 2^n seconds, where n is an integer. These times are multiples of 1 s during which the attack traffic is present at the router, thereby continuously denying service to the legitimate TCP connections. Typically, the Shrew attack was shown to be lethal to the long-lived TCP flows in the Internet.

On the contrary, the RoQ attack targets to dampen QoS experienced by the TCP traffic by keeping the time period high. It tries to occupy the share of the legitimate network traffic by sending high rate bursts on longer timescales. For instance, by sending the periodic bursts of attack packets to a router, the attacker does not allow the queue to stabilize such that the QoS sensitive Internet traffic experiences degradation of quality [6]. In particular, the periodicity is not well defined in an RoQ attack, thereby allowing the attacker to keep the average rate of the attack traffic significantly low to evade the adaptive queue management techniques like RED and RED-PD [9]. The RoQ attack exploits the AIMD algorithm of the TCP protocol; it achieves this by intermittently sending the attack traffic during which the legitimate packets are dropped, and TCP reduces the current congestion window by half and enters the slow start phase. Since the attack time period is high, the legitimate TCP traffic regains some of the lost congestion window size slowly following the AIMD mechanism, and thus only suffering from reduction of quality. We consider the RoQ attack and the low rate DoS attack model in our study. To distinguish the two attacks, we classify an attack with time period greater than or equal to 5 s as a RoQ attack, and an attack with time period less than 5 s as a low rate DoS attack. It was suggested in [3], though not explicitly specified, to set a higher time period, but the time period value of 5 s is empirically found to perform well for our proposed attack filtering algorithm.

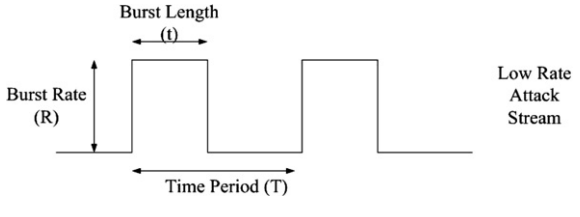

Fig. 1. The characteristics of the low rate DoS attacks.

In our earlier work [10], the detection system can detect the stealthy low rate DoS and RoQ attack by using a simple time difference method. The time difference technique uses a per-flow approach to store arrival times of the packets belonging to each flow, and computes 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. In this paper, we consider an attacker that will use the IP address spoofing to fool the per flow detection system. The approach presented in this paper works in conjunction with our previous approach, which can easily detect low rate DoS attack that do not use IP address spoofing, to provide a complete solution against these attacks. Traditional approaches to mitigate the IP address spoofing such as IP traceback are useful when an end-host is attacked [11,12]. On the contrary, the RoQ attack targets network elements, and so packets may not even reach the end host.

Our objective is to address the following question: can an individual router detect spoofed packets used in the low rate DoS and RoQ attack, and alleviate/mitigate the RoQ attack? Our solution provides both detection and mitigation against the RoQ attacks. We propose a scalable technique that passively detects the low rate DoS and RoQ attacks. After having confirmed the onset of an RoQ attack, we enable our filtering algorithm to separate long-lived legitimate flows at the router, and subsequently to drop the RoQ attack packets. The filtering algorithm is best suited for RoQ attacks; furthermore, we briefly discuss the problem of filtering low rate DoS attack packets using our approach.

The rest of the paper is organized as follows. Section 2 describes the problem in detail. Section 3 presents the detection system. Section 4 discusses the hardware implementation issue. Section 5 presents the ns2 simulation results and relevant discussion. Related works and comparisons with the proposed detection system are described in Section 6, followed by concluding remarks in Section 7.

2. Problem description

The MIT Spoofer project [13] has reemphasized the detrimental effect of the IP address spoofing. The subnet IP address spoofing [14] is easily orchestrated, as the ingress IP address filtering cannot contain the spoofing. To illustrate the subnet IP address spoofing, consider an attacker in the subnet, 12.28.34.0–12.28.34.100; 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. We assume 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 the paper. The open knowledge of the RoQ attack and the ON–OFF periodic blasting attack [9] shows that periodicity can be random for the low rate DoS and RoQ attack. As described in [9], 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 and RoQ 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 [15] 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 [16]. The primary scope of this paper 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. We shall next describe the architecture of the proposed detection system.

3. Detection system

3.1. Detection system architecture and logic

The low rate DoS and 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 loss observed in our experiments [6] was 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 [17,18]. In our case, we use the congestion signal from the adaptive virtual queue (AVQ) algorithm [18] to invoke the detection system. Thus, we can eliminate the overhead of performing memory intensive analysis under no attack. A network administrator can tune this parameter.

The detection system architecture and the attack detection procedure are depicted in Figs. 2 and 3, respectively. The detection system relies on readily available per-flow information in today's routers, for example, Cisco Netflow [19], as shown in Fig. 2. There has been significant advances and continued research effort in incorporating per flow processing in router design [20] as per flow information is of extreme importance for network management, accounting, billing, and security purposes to ISPs. Our detection system leverages on this readily available per flow information, thus requiring no major

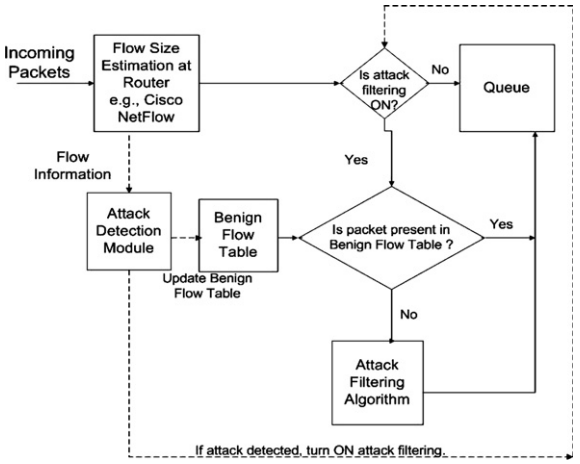

Fig. 2. The detection system architecture.

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.

Fig. 3. The attack detection procedure.

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 *lastaccessed time*; these parameters are the same as those facilitated in Cisco Netflow. The *lastaccessed 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 2 s [21]. 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 Section 3.2. To improve the scalability of the scheme, we rely on the Internet traffic observations made by various studies [22–24]. 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 property of the Internet traffic has been used before for designing AQM schemes such as RED-PD [25]. We use this knowledge about the long-lived flows 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 we discuss the filtering approach. 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, we have explained Steps 1–3 described in Fig. 3 except the functioning of the low rate DoS attack detection algorithm. Step 4 will be detailed in Section 3.2 when we consider some special cases of how an adversary can fool our detection system. We shall next explain the mechanism of the low rate DoS and RoQ attack detection algorithm in details of which the psuedocode is shown in Fig. 4.

The basic idea of the detection logic can be explained by using Fig. 5. Assuming the entire

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}

Fig. 4. Pseudocode of the attack detection algorithm.

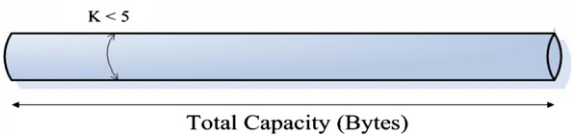

Fig. 5. The basic concept of the attack detection algorithm.

capacity of the link as a big pipe, we want to estimate how much of the pipe is filled by the flows containing *atleast k* packets in a second. For the time being, we assume the ON period of the attack to be less than or equal to 1 s. We want to see if the following holds true:

$$\left(\sum_{\text{All Flows } X<k} \text{TotalBytes per flow}\right) \gg \text{Threshold}.$$

The diagram in Fig. 5 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. We want to observe a sudden increase in the traffic load from a group of flows in a short time, and so, we consider the flows formed and expired within a second as seen in Step 1. The *createdtime* and *lastaccesed time* 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. We repeat Step 1 every 1 s 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, 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. We propose three thresholds to be used in Step 2 of the detection algorithm

1. $THRESHOLD \geqslant C + B$
2. $C/2 + B \leqslant THRESHOLD < C + B$
3. $C/4 + B \leqslant THRESHOLD < C/2 + B$

C is the capacity of the link, and B is the buffer size of the router. 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 Section 3.3. 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.

We shall next discuss how the detection system can detect an attacker using different IP address spoofing strategies. 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, we check the value of the *sum* variable 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. We check if the *sum* variable is higher than any of the three proposed thresholds in a short period of 1 s, and whether it repeats periodically. In the next subsection, to confirm the proposed heuristics, we keep the range of k as $0 < k < C/P$ and 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 $k < 2$, and checking if the value of the *sum* variable is greater than any of the proposed three thresholds. Once the attack is detected, we activate the proposed filtering mechanism 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 our simulation results. RED-PD [9] cannot detect such an attack.

3.2. 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 s, and then uses the same flowid to launch a low rate DoS or RoQ attack. The proposed detection system will incorporate our previous approach reported in [10], which can easily detect low rate DoS and RoQ attacks that do not use IP address spoofing. The time difference approach [10] relies on computing the time difference between the consecutive packets of a flow. It was shown in [10] 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 Fig. 2. 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 s 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, we apply our detection algorithm shown in Fig. 4 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 1 s. As in detection of the low rate DoS and RoQ attack described before, our 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 in Fig. 4 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, we propose to apply the attack detection algorithm 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 we use the low rate DoS and RoQ attack detection algorithm to detect the attack and not the time difference technique, we will rely on the attack filtering technique described in Section 3.4 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. Our approach can thus detect the low rate DoS and RoQ attack that uses any combination of the number of flows participating in the attack, any time period, any burst period, and any burst rate.

3.3. Trace evaluation

To confirm that the thresholds proposed in the previous subsection will work for the Internet traffic, we evaluate our strategy by analyzing the OC48 (2.5 Gbps) traces provided by CAIDA [27]. Using the Coralreef [28] software, we first obtained the expired flow statistics <Source IP address, Destination IP address, Source IP port, Destination IP port, Packetcnt, Bytecnt, Createdtime, and Lastaccessed time>, which are similar to the ones that we have proposed to collect for all the flows. We run the attack detection algorithm 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. Fig. 6 shows the sum variable statistics for a 5 min April 2003 trace. We have made similar analysis for longer periods, but they are similar to the ones presented here. The OC48 speed is 2.5 Gbps, i.e., 2.5E09, and so C/2 is 1.25E09, and C/4 is 62.5E07. We can see from Fig. 6 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 1 s as reported in the literature about low rate DoS attacks. We observe in Fig. 7 the values of the sum variable for every 2 s, in which

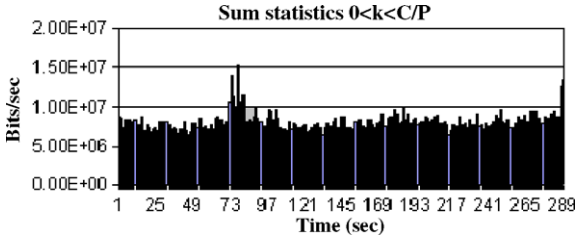

Fig. 6. Sum statistics for every second.

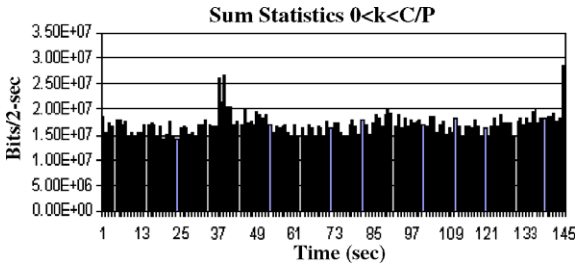

Fig. 7. Sum statistics for every 2 s.

case we assume an ON period of 2 s. The attack with ON period greater than 2 s can certainly be detected by our approach, but they can be easily detected by RED-PD and many other existing AQM schemes too.

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 Fig. 7 does not exceed the proposed thresholds. Similar observations were found in other traces. We also propose to select the lower thresholds carefully. 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 Fig. 2 to understand the unique properties of traffic distributions on each link, and to adjust the attack detection thresholds accordingly. We want to highlight that in a short period of 1–2 s the sum variable does not exceed the proposed thresholds in the absence of the low rate DoS and RoQ attack, as observed in Figs. 6 and 7. Thus, only during the low rate DoS and RoQ attack, the sum will exceed the proposed thresholds.

3.4. Filtering logic

We propose a filtering scheme to mitigate the RoQ attacks. The low rate DoS and RoQ attack detection algorithm in Fig. 4 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, we propose a nondeterministic approach 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. We have developed a simple method to address this problem. As mentioned before, we separate the long-lived flows in the benign flow table, 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 2 s, 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 3.2 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 enqueued packet is dropped immediately; otherwise, the enqueued packet is treated normally. Our 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, we know that most likely, these packets are RoQ attack packets, and hence we drop these packets. The filtering is also activated at the approximate time instants for 1 s when attack packets start arriving at the queue. We filter for 1 s because the attack detection algorithm runs every 1 s. 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 the simulation results 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 $(100 - \alpha)\%$ of the buffer space is denied to them till 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 enqueueing less frequently, e.g., every 5 s, 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 2 s [21], 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, our filtering solution is simple and effective. Our simulation results demonstrate the effectiveness of our 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 in Fig. 4. 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 enqueuing up more frequently. The queue length exceeds $\alpha\%$ 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. 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 [29,30], at relatively low speeds show that our proposed approach can be realizable. That 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, are the two critical considerations in provisioning high speed monitoring. In this section, we briefly discuss some of the recent techniques proposed in the literature that can facilitate realization of the proposed detection system architecture.

To estimate the size of the benign flow table, we consider the commonly used Internet traces for analysis as conducted in [31–33]. It is mentioned in [31] that this ISP trace for OC48 speed 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 [22–24], 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 [34], 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 2 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 [33], 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 1.8 MB using the same parameters as before. Thus, the benign flow table has a modest memory requirement of about 2 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 [19], 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 [19], The benign flow table represents the additional requirement if the proposed system is deployed at a router.

5. Simulation results and discussion

We used the ns2 simulator [35] to demonstrate the performance of the proposed detection scheme. The topology used in our experiment is shown in Fig. 8. We used the PackMime [36] HTTP traffic generator, which was developed by using the real traces of the Internet traffic. It is the most recent work about the modeling of the HTTP traffic that improves over the previous HTTP traffic models. We have adopted the PackMime traffic model in our 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 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 100 Mbps links to the delaybox, and two routers with a buffer size of 500 packets each and a bottleneck link of 10 Mbps between them. The queue size is fixed and is equal to twice the delay bandwidth product, which was found to perform optimally in routers in the Internet [37]. The link between the delaybox and the router is 100 Mbps. The delaybox is used to provide per flow dropping probability, round-trip times, and bottleneck link speeds. In our setting, the dropping probability is zero and the available server bandwidth is uniformly distributed from 1 to 20 Mbps. We have modified the file test-PackMime-delaybox.tcl script [38] for our simulations. All the access links have

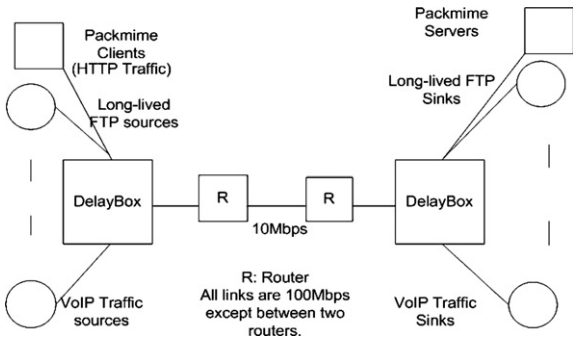

Fig. 8. The simulation topology.

random delays obtained by using a uniform distribution from 50 to 250 ms. The access links connecting to the sink agents and the bottleneck link has link delay of 10 ms. We also have 10 long-lived flows using FTP in the network. The details of the SACK TCP used in the simulations are: window size 50 packets, segment size 1460, minimum RTO 1 s 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. We have five VoIP flows that are modeled as G711 FEC 96 Kbps traffic using the exponential ON–OFF traffic model in ns2. The attacker uses UDP constant bit rate traffic (CBR). The proposed detection system code is embedded in the AVQ algorithm [18] 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. We ran the simulation for 650 s with a warm up time of 50 s; the attack was introduced 50 s 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 our simulation setup. It also conforms with 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 [22–24]. About 14,000 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, we demonstrate the ability of the attack detection algorithm in detecting the low rate DoS and the RoQ attack as explained in Fig. 4. We set the attack threshold as explained in Fig. 4 to be 312.5 KB close to the value of C/4, and test it with two attack scenarios: 1. $T = 1$ s, $t = 0.3$ s, and $R =$ 10 Mbps (low rate DoS attack) and 2. $T = 5$ s, $t = 0.3$ s, and $R = 10$ Mbps (RoQ attack). We now present results of the SUM variable described in Fig. 4. One representative SUM value in absence of the attack is 88.85 KB, and it never exceeds the threshold of 312.5 KB throughout the simulation. In attack scenario 1, the SUM values are

391.3 KB at 50.8 s, 425.7 KB at 51.8 s, 390.6 KB at 52.8 s, 393.9 KB at 53.8 s, and so on. In attack scenario 2, the SUM values are 391.3 KB at 50.8 s, 375.6 KB at 55.8 s, 371.7 KB at 60.8 s, 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 s, the burst period of 0.3 s, and the burst rate of 10 Mbps. Our detection system detects the RoQ attack as explained in the attack detection algorithm. As shown in Fig. 9, the throughput for the long-lived flows is restored to almost that of the case with no attack. Apparently, flow four was not affected by the attack; we conjecture 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 four achieved higher throughput because other flows were suffering from reduction in throughput.

In Fig. 10, we can see 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 2 s. A study by a VoIP carrier network vendor Nextone has reported the average VoIP call length to be around 9 min in one of their February 2005 case study [39]. For the RoQ attack with the time-period of 5 s, we kept the value of α to be 80 because above that leads to loss of throughput to the legitimate long-lived flows in the benign flow table (see Fig. 9). 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 our simulations. In Figs. 11–13, we show the statistics of each HTTP flow generated during the simulation obtained from the PackMime program. 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 1.8 s, the detection system case is 2.3 s, and the attack case is 1.2 s. The standard deviation of the response time in the no attack case is 1.3 s, the detection system case is 2.2 s, and the attack case is 1.3 s. In the attack case, the HTTP performance

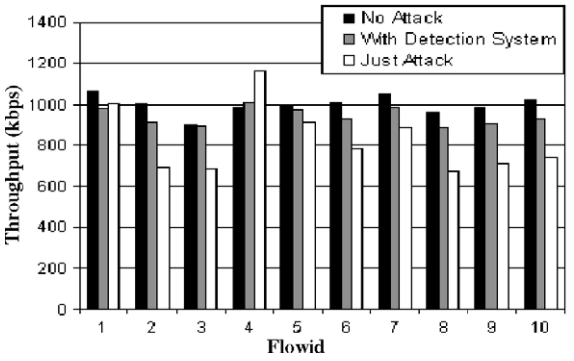

Fig. 9. The throughput comparison under an RoQ attack.

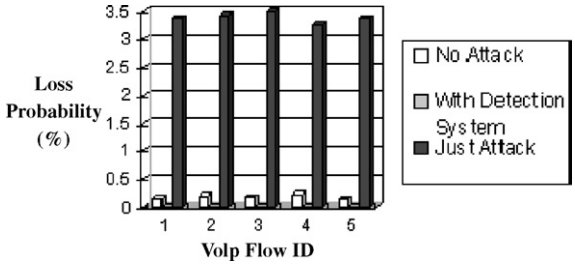

Fig. 10. VoIP packet loss comparison under an RoQ attack.

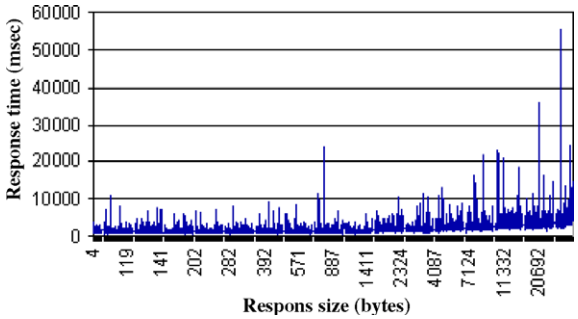

Fig. 11. HTTP performance under no attack.

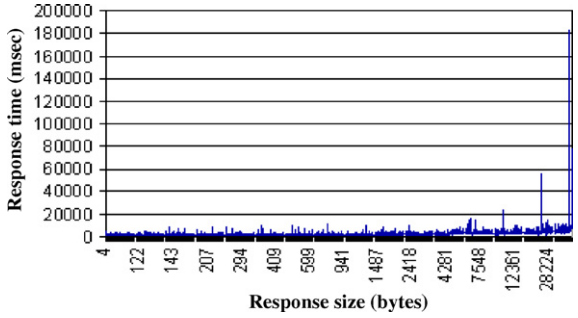

Fig. 12. Restoration of HTTP performance under an RoQ attack using the proposed detection system.

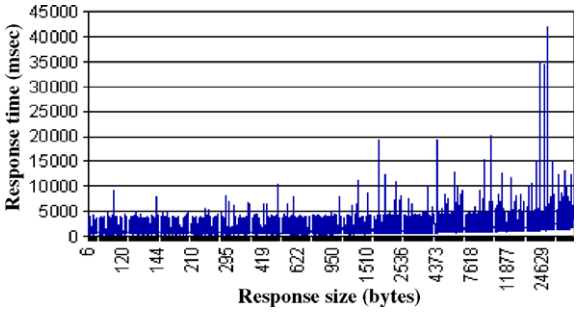

Fig. 13. HTTP performance under an RoQ attack.

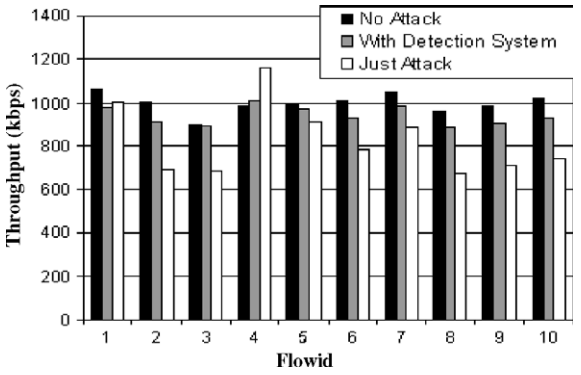

Fig. 14. The throughput comparison under an RoQ attack with continuous cycle IP address spoofing.

is better because the long-lived flows are suffering from losses and their bandwidth is claimed by HTTP flows. We compare the no attack case and detection system case, and observe that the average response time increases by 0.5 s and the standard deviation of response time increases by 0.9 s with the detection system. Thus, the legitimate clients are not denied service when our detection system is employed; however, there is a slight increase in response time.

We also conducted experiments when an attacker uses continuous cycle IP address spoofing in which a new IP address is used for every ON period. In our simulation, an attacker should send approximately 500, 5, 10, 15, and 20 K packets, each of size 210 bytes, to fill up the bottleneck link during the ON period of 10, 110, 210, 310, and 410 ms, respectively. The attacker in this scenario sends 4 K packets with a new IP address every ON period. We kept the time period of 5 s, the burst period of 0.3 s, and the burst rate of 10 Mbps for the RoQ attack as before 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 Fig. 14. We can observe that the throughput of the legitimate long-lived flows is restored, and a significant number of attack packets are dropped using the proposed detection system. The proposed architecture facilitates identification and filtering of the RoQ attack traffic in which IP address spoofing is instigated. Our approach thus addresses most of the issues where RED-PD and several other approaches fail to defend against RoQ attacks.

6. Related works

We briefly review some of the countermeasures proposed in the literature to mitigate the low rate DoS and RoQ attacks in the Internet although none of them has made a comprehensive attempt to address such attacks that can use IP address spoofing or botnets. In [26], an autocorrelation and dynamic time warping algorithm is proposed to detect the low rate DoS attacks. The paper proposes deficit round robin algorithm to filter the attack flows, which will fail 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. On the contrary, our proposed scheme can detect and filter RoQ attacks, which use IP address spoofing or botnets. The randomization of RTO [40] proposed to mitigate the low rate TCP DoS attack cannot defend against the RoQ attack, which targets the network element rather then the end host.

The collaborative detection and filtering scheme proposed in [41] involves cooperation among routers to throttle and push the attack traffic towards the source. The scheme relies on the autocorrelation property to distinguish the periodic behavior of the attack traffic from the legitimate traffic. It uses per-flow analysis to detect low rate DoS attacks, which will fail under IP address spoofing. It maintains a malicious flow table, and a suspicious flow table, which can be overwhelmed under the presence of IP address spoofing. The novel part is the cumulative traffic spectrum that can distinguish traffic with the attack 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. In contrast, our proposed approach can work without cooperation among the routers. A similar frequency domain approach is used in [42] to distinguish RoQ attack traffic but it fails to tackle the RoQ attack packet filtering problem. It uses a simple table to store hash of five-tuple flowid, which can be easily overwhelmed if large botnets are used to launch an RoQ attack at the router.

The wavelet based [5] approach identifies abnormal change in the incoming traffic rate and the outgoing acknowledgments to detect presence of low rate TCP DoS attacks. This approach cannot identify the RoQ attack. The wavelet approach does not filter the attack traffic. In [43], the authors have proposed to regulate the buffer size to expose the attack flows to the RED-PD filtering. 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. A modified AQM scheme referred to as HAWK [44] works by identifying bursty flows on short timescales, but fails 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. In a similar work [45], the authors estimate the bursty flows on shorter and longer time scales. The main idea in [45] 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; it is thus very easy to penalize a normal flow as a bursty flow. They do not consider the random IP address spoofing, where every packet will have a new flow id. They use a very complex filtering technique. On the contrary, the idea behind the approach proposed in this paper is to identify a group of flows with fewer packets showing high rates on shorter timescales. 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. Our technique of filtering and scalably isolating the legitimate flows thus helps mitigate most of the problems caused by the IP address spoofing.

7. Concluding remarks

We have proposed a router-based approach 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. At the time of writing this paper to best of our knowledge, there is no effective solution to defend RoQ attacks that employ IP address spoofing or botnets. Our solution fills up this gap in the network security area.

Acknowledgement

This work has been supported in part by the National Science Foundation under Grants 0435250 and 0726549.

References

[1] CSI/FBI Computer Crime and Security Survey. <http://www.gocsi.com/>, 2006, online.
[2] A. Kuzmanovic, E. Knightly, Low-rate TCP-targeted denial of service attacks (The Shrew vs. the Mice and Elephants), in: ACM SIGCOMM 2003, 2003, pp. 75–86.
[3] M. Guirguis, A. Bestavros, I. Matta, Exploiting the transients of adaptation for RoQ attacks on Internet resources, in: IEEE ICNP 2004, 2004, pp. 184–195.
[4] S. Ebrahimi-Taghizadeh, A. Helmy, S. Gupta, TCP vs. TCP: a systematic study of adverse impact of short-lived TCP flows on long-lived TCP flows, in: IEEE INFOCOM 2005, 2005, pp. 926–937.
[5] X. Luo, R.K.C. Chang, On a new class of pulsing denial-of-service attacks and the defense, in: NDSS 2005, 2005.
[6] A. Shevtekar, N. Ansari, Do low rate dos attacks affect QoS sensitive VoIP traffic? in: IEEE ICC 2006, 2006, pp. 2153–2158.
[7] R. Chertov, S. Fahmy, N. Shroff, Emulation versus simulation: a case study of TCP-targeted denial of service attacks, in: TridentCom 2006, 2006, pp. 316–325.
[8] M. Delio, New Breed of Attack Zombies Lurk. <http://technews.acm.org/articles/2001-3/0514m.html#item2>, 2001.
[9] Y. Xu, R. Guerin, On the robustness of router-based denial-of-service (DoS) defense systems, ACM Computer Communications Review 35 (3) (2005) 47–60.
[10] A. Shevtekar, K. Anantharam, N. Ansari, Low rate TCP denial-of-service attack detection at edge routers, IEEE Communication Letters 9 (4) (2005) 363–365.
[11] Z. Gao, N. Ansari, Tracing Cyber attacks from the practical perspective, IEEE Communications Magazine 43 (5) (2005) 123–131.
[12] A. Belenky, N. Ansari, On IP traceback, IEEE Communications Magazine 41 (7) (2003) 142–153.
[13] R. Beverly, S. Bauer, The Spoofer project: inferring the extent of source address filtering on the Internet, in: USENIX SRUTI'05, 2005, pp. 53–59.
[14] J. Mirkovic, P. Reiher, A taxonomy of DDoS attack and DDoS defense mechanisms, ACM SIGCOMM Computer Communications Review 34 (2) (2004) 39–54.

[15] D. Dagon, C. Zhou, W. Lee, Modelling botnet propagation using time zones, in: NDSS 2006, 2006.
[16] M. Guirguis, A. Bestavros, I. Matta, Bandwidth stealing via link targeted RoQ attacks, in: CCN'04, 2004.
[17] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking 1 (4) (1993) 397–413.
[18] S. Kunniyur, R. Srikant, An adaptive virtual queue (AVQ) algorithm for active queue management, IEEE/ACM Transactions on Networking 12 (2) (2004) 286–299.
[19] Cisco Netflow. <http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html>, 2007, online.
[20] R. Kompella, C. Estan, The power of slicing in Internet flow measurement, in: IMC 2005, 2005.
[21] N. Brownlee, K. Claffy, Understanding Internet traffic streams: dragonflies and tortoises, IEEE Communications Magazine 40 (10) (2002) 110–117.
[22] K.C. Claffy, H.-W. Braun, G.C. Polyzos, A parameterizable methodology for Internet traffic flow profiling, IEEE Journal on Selected Areas in Communication 13 (8) (1995) 1481–1494.
[23] M. Fomenkov, K. Keys, D. Moore, K. Claffy, Longitudinal study of Internet traffic from 1998 to 2003, in: Winter International Symposium on Information and Communication Technologies, 2004, pp. 1–6.
[24] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, C. Diot, Packet-level traffic measurements from the sprint IP backbone, IEEE Network Magazine 17 (6) (2003) 6–16.
[25] R. Mahajan, S. Floyd, D. Wetherall, Controlling high-bandwidth flows at the congested router, in: IEEE ICNP 2001, 2001, pp. 192–201.
[26] H. Sun, J.C.S. Lui, D.K.Y. Yau, Defending against low-rate TCP attack: dynamic detection and protection, in: IEEE ICNP 2004, 2004, pp. 196–205.
[27] The OC48 Data. <http://www.caida.org/data/passive/index.xml#oc48>, online.
[28] The CoralReef Software. <http://www.caida.org/tools/measurement/coralreef/>, online.
[29] H.J. Chao, R. Karri, W.C. Lau, CYSEP – a Cyber security processor for 10 Gbps networks and beyond, in: IEEE MILCOM 2004, 2004, pp. 1114–1122.
[30] A. Kumar, J. Xu, Sketch guided sampling – using on-line estimates of flow size for adaptive data collection, in: IEEE Infocom 2006, 2006.
[31] A. Kumar, M. Sung, J. Xu, J. Wang, Data streaming algorithms for efficient and accurate estimation of flow distribution, in: ACM SIGMETRICS 2004, 2004, pp. 177–188.
[32] C. Cranor, T. Johnson, O. Spatschek, Gigascope: a stream database for network applications, in: SIGMOD 2003, 2003, pp. 647–651.
[33] B. Grot, W. Mangione-Smith, Good memories: enhancing memory performance for precise flow tracking, in: ANCHOR 05, 2005.
[34] P. Manolios, Bloom Filter Calculator. <http://www-static.cc.gatech.edu/~manolios/bloom-filters/calculator.html>, online.
[35] UCB/LBNL/VINT Network Simulator. <http://www.isi.edu/nsman/ns/>, online.
[36] J. Cao W.S. Cleveland, Y. Gao, K. Jeffay, F.D. Smith, M.C. Weigle, Stochastic models for generating synthetic HTTP source traffic, in: IEEE Infocom 2004, 2004, pp. 1546–1557.
[37] M. Christiansen, K. Jeffay, D. Ott, F.D. Smith, Tuning RED for web traffic, IEEE/ACM Transaction on Networking 9 (3) (2001) 249–264.
[38] PackMime Traffic Generator. <http://dirt.cs.unc.edu/packmime>, online.
[39] Latinode Case Study. <http://www.nextone.com/files/LatiNode%20Case%20Study.pdf>, 2005, online.
[40] G. Yang, M. Gerla, M.Y. Sanadidi, Randomization: defense against low-rate TCP-targeted denial-of-service attacks, in: IEEE Symposium on Computers and Communications, 2004, pp. 345–350.
[41] Y. Chen, K. Hwang, Collaborative detection and filtering of Shrew DDoS attacks using spectral analysis, Journal of Parallel and Distributed Computing, Special Issue on Security in Grids and Distributed Systems 66 (9) (2006).
[42] Y. Chen, K. Hwang, Spectral analysis of TCP flows for defense against reduction-of-quality attacks, in: IEEE ICC 2007, 2007.
[43] S. Sarat, A. Terzis, On the effect of router buffer sizes on low-rate denial of service attacks, in: IEEE ICCCN 05, 2005, pp. 281–286.
[44] Y. Kwok, R. Tripathi, Y. Chen, K. Hwang, HAWK: Halting anomaly with weighted choking to rescue well-behaved TCP sessions from Shrew DoS attacks, in: ICCNMC 2005, 2005.
[45] Y. Xu, R. Guerin, A double horizon defense for robust regulation of malicious traffic, in: SecureComm, 2006.

Amey Shevtekar received the B.S. degree in Electronics and Telecommunications Engineering from University of Mumbai, India, and the M.S. degree from the New Jersey Institute of Technology (NJIT) in Telecommunications. He is pursuing his doctorate in Computer Engineering at New Jersey Institute of Technology (NJIT), focusing on network security, Low Rate DoS Attacks, DDoS Attacks, and computer networks.

Nirwan Ansari received the B.S.E.E. (*summa cum laude*) from the New Jersey Institute of Technology (NJIT), Newark, in 1982, the M.S.E.E. degree from University of Michigan, Ann Arbor, in 1983, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 1988.
He joined NJIT's Department of Electrical and Computer Engineering as an Assistant Professor in 1988, and has been a Full Professor since 1997. He has also assumed various administrative. He authored *Computational Intelligence for Optimization* (Springer, 1997, translated into Chinese in 2000) with E.S.H. Hou, and edited *Neural Networks in Telecommunications* (Springer, 1994) with B. Yuhas. His current research focuses on various aspects of broadband networks and multimedia communications. He has also contributed over 300 technical publications, of which over one third in refereed journals and magazines.

He is a Senior Technical Editor of the *IEEE Communications Magazine*, and also serves on the editorial board of *Computer Communications*, the *ETRI Journal*, and the *Journal of Computing and Information Technology*. He was the founding general chair of the First IEEE International Conference on Information Technology: Research and Education (ITRE2003), was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award and a 2003 Chapter Achievement Award, served as Chair of the IEEE North Jersey Section and in the IEEE Region 1 Board of Governors during 2001–2002, and has been serving in various IEEE committees such as Chair of IEEE COMSOC Technical Committee on Ad Hoc and Sensor Networks, and (TPC) Chair/Vice-chair of several conferences/symposia. His awards and recognitions include the NJIT Excellence Teaching Award in Graduate Instruction (1998), IEEE Region 1 Award (1999), an IEEE Leadership Award (2007), and designation as an IEEE Communications Society Distinguished Lecturer.