International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-3, Issue-4)

Research Article | **April 2014**

Efficient Packet Flow Clasification in Network by Using Bayesian Prediction

T.V. Austin Thomas[1]
[1]*PG Scholar,*
Department of CSE,
Sri Lakshmi Ammal Engineering College,
Tamil Nadu, India

R.Deepak[2#]**,**
[2#] *Assistant Professor,*
Department of CSE,
Sri Lakshmi Ammal Engineering College,
Tamil Nadu, India

Abstract:

Distributed Denial of Service (DDoS) attacks are a critical threat to the Internet. The defense scheme that supports automated online attack characterizations and accurate attack packet discarding based on statistical processing is proposed here. The key idea is to prioritize a packet based on a score which estimates its legitimacy given the attribute values it carries. Once the score of a packet is computed, this scheme performs score-based selective packet discarding where the dropping threshold is dynamically adjusted based on the score distribution of recent incoming packets and the current level of system overhead. Special considerations are made to ensure that the scheme is amenable to high-speed hardware implementation through scorebook generation and pipeline processing. A simulation study indicates that Packet Score is very effective in blocking several different attack types under many different conditions.

Keywords- Network level security and protection, performance evaluation, traffic analysis, network monitoring, security,simulation.

I. INTRODUCTION

One of the major threats to cyber security is Distributed Denial-of-Service (DDoS) attacks in which victim networks are bombarded with a high volume of attack packets originating from a large number of machines. The aim of such attacks is to overload the victim with a flood of packets and render it incapable of performing normal services for legitimate users. In a typical three-tier DDoS attack, the attacker first compromises relay hosts called agents, which in turn compromise attack
machines called zombies that transmit attack packets to the victim. Packets sent from zombie machines may have spoofed source IP addresses to make tracing difficult [25]. DDoS attacks can be launched by unsophisticated casual attackers using widely available DDoS attack tools such as trinoo, TFN2K, Stachedraht, etc. Since an attack in February 2000 [9] that targeted several high profile Web sites, including Yahoo, CNN, eBay, etc., the frequency and magnitude of DDoS attacks has been increasing rapidly, making it a growing concern in our Internet-dependent society. According to a 2003 CSI/FBI Computer Crime and Security Survey [6], 42 percent of respondents of the survey had suffered from DDoS attacks, 28 percent reported financial losses due to DDoS attacks, and the average losses due to DDoS attacks had increased 4.8 times since the year 2002. Recently, the FBI listed a suspect in the Most Wanted list for the charge of launching a DDoS attack against a competitor's Web site [7]. The DDoS problem has attracted much attention from the research community recently. In our observation, there are three major branches of research in DDoS, namely, 1) attack detection, e.g., by monitoring protocol behavior [30], 2) attack traceback, e.g., by packet marking [29], and 3) attack traffic filtering as described below more in detail. The Packet Score scheme discussed in this paper belongs to group 3) attack traffic filtering. Research in attack traffic filtering can be roughly categorized into three areas based on the point of protection:

Source-initiated: Source sites are responsible for guaranteeing that outgoing packets are attack-free. Examples include network ingress filters [8], disabling ICMP, or removing unused services to prevent computers from becoming attack agents, or filtering unusual traffic from the source [24]. However, the viability of these approaches hinges on voluntary cooperation among a majority of ingress network administrators Internet-wide, making these approaches rather impractical given the scale and uncontrollability of the Internet.

Path-based: In this approach, only the packets following the correct paths are allowed [15]. Any packet with a wrong source IP for a particular router port is considered a spoofed packet and dropped, which eliminates up to 88 percent of the spoofed packets [4], [28]. In another approach [11], if the number of traveled hops is wrong for a source IP, the
packet is dropped, thereby eliminating up to 90 percent of the spoofed packets. These approaches are considered practical, but they have a somewhat high probability of false negatives, i.e., falsely accepting attack packets. Apparently, when packets use unspoofed addresses, which is an emerging trend, none of these approaches works.

Victim-initiated: The victim can initiate countermeasures to reduce incoming traffic. For example, in the pushback scheme [10], the victim starts reducing excessive incoming traffic and requests the upstream routers to perform rate reduction as well. There are other methods based on an overlay network [14], packet marking [18], [31], TCP flow

filtering [17], [33] and statistical processing [19], [20], etc. Although victim-initiated protections are more desirable, some methods require changes in Internet protocols or are too expensive to implement.

The industry is adopting more practical approaches, 1) overprovisioning and proven to be vulnerable in recent attacks. More commonly, ISPs rely on manual detection and blocking of DDoS attacks. Once an attack is reported, an offline finegrain traffic analysis is performed by a subject-matter expert to identify and characterize the attack packets. New filtering rules on access control lists are then constructed and installed manually on the routers. But, the need for human intervention results in poor response time and fails to protect the victim before severe damages are realized. Furthermore, the expressiveness of existing rule-based filtering is too limited, as it requires an explicit specification of all types of packets to be discarded.

II. CONDITIONAL LEGITIMATE PROBABILITY

The most challenging issue in blocking DDoS attacks is to distinguish attack packets from legitimate ones. To resolve this problem, we utilize the concept of Conditional Legitimate Probability (CLP) for identifying attack packets probabilistically. CLP is produced by comparing traffic characteristics during the attack with previously measured, legitimate traffic characteristics. The viability of this approach is based on the premise that there are some traffic characteristics that are inherently stable during normal network operations of a target network. We named this scheme PacketScore because CLP can be viewed as a score which estimates the legitimacy of a suspicious packet. We will use the terms CLP and score interchangeably. By taking a score-based filtering approach, the prioritization of different types of suspicious packets is possible and we can avoid the problems of conventional binary rule-based filtering discussed in Section 1. The ability to prioritize becomes even more important when a full characterization of attack packets is not feasible. By dynamically adjusting the cutoff score according to the available traffic capacity of the victim, our approach allows the victim system to accept more potentially legitimate traffic. In contrast, once a rule-based filtering scheme is configured to discard specific types of packets, it does so regardless of the victim network's available capacity. To formalize the concept of CLP, we consider all the packets destined for a DDoS

attack target. Each packet would carry a set of discrete-value attributes A; B;C; . . . . For example, A might be the

protocol type, B might be the packet size, C might be the TTL values, etc. We defined {a2; a3; . . .g} as the possible values for attribute A, { b2; b3; . . .g} as the possible values for attribute B, and so on. During an attack, there are Nn legitimate packets and Na attack packets arriving in T seconds, totaling Nm.

$$CLP(packet_p)=P(packet_p \text{ is legitimate} \mid p\text{'s attribute } A= a_p, \text{attribute } B=b_p)$$

$$CLP(p)=p((p=legitimate)n(A=a_p,B=b_p))$$

$$p(A=a_p,B=b_p)$$

created from the training set using a Gaussian Let's say we have equiprobable classes so get the same answer.

ESTIMATING LEGITIMATE TRAFFIC DISTRIBUTION

That we can calculate the probability of a packet's legitimacy by observing the probabilities of the attribute values in legitimate traffic ðPnÞ and in total traffic ðPmÞ. However, it is practically impossible to know how many packets are legitimate during the attack period, let alone the number of legitimate packets bearing a particular attribute value. For that reason, we utilize an estimate P0nin place of true Pn. The estimate P0n is called a nominal profile and is collected in advance during normal operations. A nominal traffic profile consists of single and joint distributions of various packet attributes that are considered unique for a site. Candidate packet attributes from

IP headers are:

1. packet size,
2. Time-to-Live (TTL) values,
3. protocol-type values, and
4. source IP prefixes.

Those from TCP headers are:

5. TCP flag patterns and
6. server port numbers, i.e., the smaller of the source port number and the destination port number. Server port number is more stable than sort/destination port numbers because most of the well-known port numbers are small numbers (e.g., below 1,024) and a large portion of Internet traffic uses the well-known port numbers. To increase the number of attributes, we can

employ joint distributions of the fraction of packets having various combinations, such as:

7. <packet-size and protocol-type>,
8. <server port number and protocol-type>, and
9. <source IP prefix, TCP flags and packet size>, etc. Joint distributions often better represent the uniqueness of the traffic distribution for a site, and are harder to guess for the attackers. As many different combinations of single attributes as needed may be used while the storage space permits During the nominal profiling period, the number of packets with each attribute value is counted and the corresponding ratio is calculated. However, if the profile is created only once during the profiling period, temporally localized traffic characteristics may be misrepresented. To avoid it, the profiling period is broken into subperiods, then the ratios are measured for each subperiod, and one value representing all the

subperiods is selected. The principle of PacketScore is to punish the traffic whose attribute value ratio is higher than in profile. Therefore, to accommodate an occasional surge of particular attribute values in legitimate traffic, the highest ratio among the periodic ratios is selected. This strategy has little impact on blocking attack traffic while giving the legitimate traffic a safety margin. Table 1 illustrates this process with an example of TTL values. The boldface values are the highest ratios observed among the periodic values, which are then stored in the profile. If the variance among the periodic ratios is too great to be reliable, it is possible to include only those attribute

NOMINAL PROFILE PORT NUMBER AND ITS DISTIBUTION						
PORT NO	21	23	25	80	110	161
RELATIVE FREQUENCY	10%	5%	15%	45%	15%	5%
NOMINAL PROFILE PROTOCOL TYPE DIDTRIBUTION						
PROTOCOL TYPE	TCP		UDP		ICMP	
RELATIVE FREQUENCY	86%		10%		4%	
NOMINAL PROFILE PACKET SIZE DISTRIBUTION						
PACKET SIZE IN BYTES	0-40	41-100	101-500	501-1000	1001-1500	1500---
RELATIVE FREQUENCY	35%	10%	5%	15%	30%	5%

Due to the number of attributes to be incorporated in the profile and the large number of possible attribute values of each attribute, especially for the joint attributes, an efficient data structure is required to implement the profile. For example, the attribute values for TTL are 0; 1; 2; . . . 255, thus there are 256 possible attribute values. Each attribute value has a ratio in the profile as illustrated in Table 1 (e.g., 1.1 percent for TTL value 1). To reduce the storage space, we use iceberg-style profiles [2], in which only the most frequently occurring attribute values are stored along with their ratio. Two approaches are possible for selecting the icebergs, i.e., by static threshold and by adaptive threshold. In the static threshold approach, the profile only includes those attribute values which appear more frequently than a preset threshold ratio, say x percent. For the attribute values which are absent from the iceberg-style profiles, we use the upper bound (x percent) as their ratios. For example, with Table 1, if the preset threshold is 1 percent, TTL value 2 is removed from the profile, and TTL value 2 is considered to have percent share in the traffic during the scoring process later. In the adaptive threshold approach, the most frequently appearing attribute values that constitute a preset coverage of the traffic, e.g., 95 percent, are selected. The corresponding cutoff threshold y percent for the given values with low variance to have a more stable profile. coverage serves as the adaptive threshold, which is also used as the default ratio for the absent items. With such iceberg-style profiles, the nominal profile can be kept to a manageable size. Joint attributes experience an additional problem of combinatorial explosion, so buckets of preset ranges are used instead of the tuples of raw attribute values. Typical storage requirements for storing six single attributes and two joint attributes are shown in Table 2 for different threshold methods. For the static threshold, we used 0.01, 0.001, and 0.0001, respectively, for single attribute, two-dimensional and three-dimensional joint attributes. Although the static threshold method produces the most space-efficient profiles, adaptive thresholds are considered more robust against wide variations in traffic trace. The iceberg-based profile is similar to [5] where its unidimensional cluster and multidimensional clusters are comparable to our single During the nominal profiling period, the number of packets with each attribute value is counted and the corresponding ratio is calculated. However, if the profile is created only once during the profiling period, temporally localized traffic characteristics may be misrepresented. To avoid it, the profiling period is broken into subperiods, then the ratios are measured for each subperiod, and one value representing all the subperiods is selected. The principle of PacketScore is to punish the traffic whose attribute value ratio is higher than in profile. Therefore, to accommodate an occasional surge of particular attribute values in legitimate traffic, the highest ratio among the periodic ratios is selected. This strategy has little impact on blocking attack traffic while giving the legitimate traffic a safety margin. Table 1 illustrates this process with an example of TTL values. The boldface values are the highest ratios observed among the periodic values, which are then stored in the profile. If the variance among the periodic ratios is too great to be reliable, it is possible to include only those attribute

attributes and joint attributes, respectively. However, the approaches and purposes are different. While [5] provides valuable information for the network administrator by identifying the dominant traffic type in any combination of attributes within the current traffic, our goal is to take a comprehensive snapshot of the traffic for future reference. The

iceberg approach is merely used to reduce the storage requirement rather than traffic aggregation. Since the profiling in PacketScore is based on packet counting and does not require aggregation, it can be done very rapidly, i.e., in a matter of seconds as opposed to minutes [5] for similar size trace data.

Traffic Profile Stability

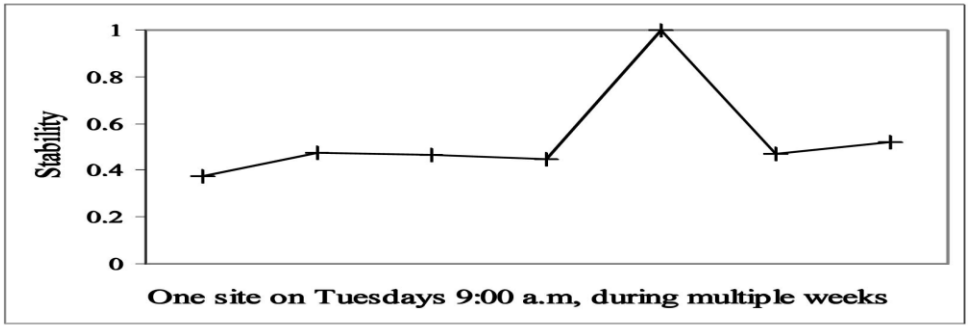
PacketScore depends on the stability of the traffic profile for estimating Pn. It has been known that for a given subnet, there is a distinct traffic pattern in terms of packet attribute value distribution for a given time and/or given day [12], [21], [23]. In general, the nominal traffic profile is believed to be a function of time which exhibits periodic, time-ofday, and day-of-the-week variations as well as long-term trend changes.

To further verify traffic profile stability, we conducted an analysis with the packet trace data available from NLANR packet trace archives [26]. All trace data were collected for 90 seconds from 17 sites within the US with the link speed ranging from OC-3 to OC-48. We randomly selected the four sites in Table 3 and total of 49 trace files were downloaded for analysis. shows general statistics for 10 selected traces.

A quick examination of Table 4 revealed that each site had a distinct traffic composition. Especially, we observed that the traffic in AIX is mostly GRE rather than TCP or UDP. For each trace, a profile was created using a 99 percent adaptive coverage method over a series of 10-second windows. We employed a joint attribute composed of three attributes (Protocol type, server port, and packet size) because joint attributes are believed more unique per site than single attributes. For an objective comparison of two profiles, we defined the stability metric S as follows:

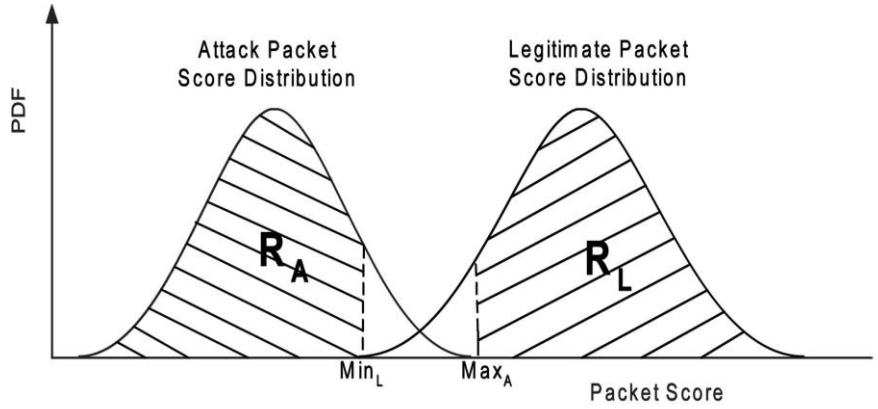
S=c *d

C=(number of common items in both profiles =n)/
(number of total items in both profiles)



(c)

D=different time of day (approximately 8:00 p.m.). Fig. 1c compared the profiles for seven Tuesdays at the same time of day (approximately 9:00 a.m.) from 23 August 2005 to 11 October 2005. Although it spans seven weeks, it still shows a similar correlation to the short-term profiles of 9:00 a.m. as in Fig. 1a. These seven Tuesday morning profiles are slightly closer than the evening profiles in Fig. 1b. However, in Fig. 1d, when compared with other sites, the SL is much lower, showing a much weaker correlation.



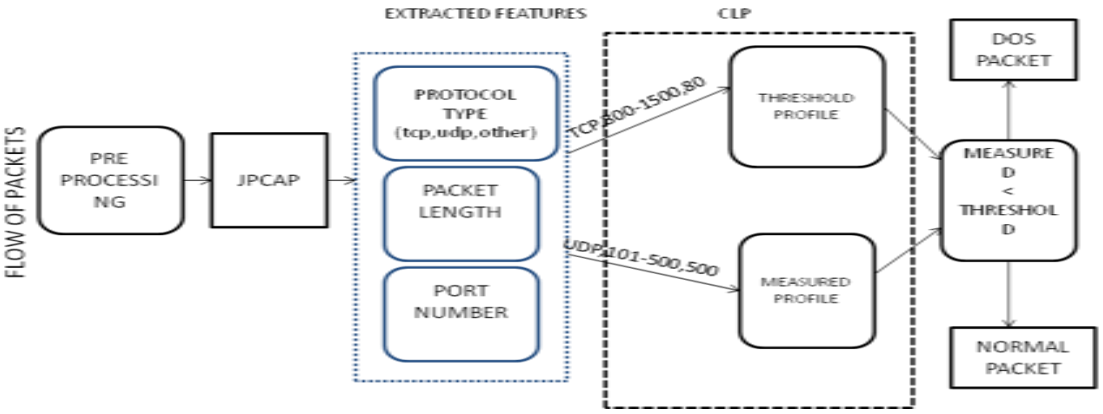
ATTACK PROFILE PORT NUMBER AND ITS DISTIBUTION						
PORT NO	21	23	25	80	110	161
RELATIVE FREQUENCY	3%	2%	5%	25%	5%	40%
ATTACK PROFILE PROTOCOL TYPE DIDTRIBUTION						

PROTOCOL TYPE	TCP	UDP	ICMP
RELATIVE FREQUENCY	45%	50%	5%
ATTACK PROFILE PACKET SIZE DISTRIBUTION			

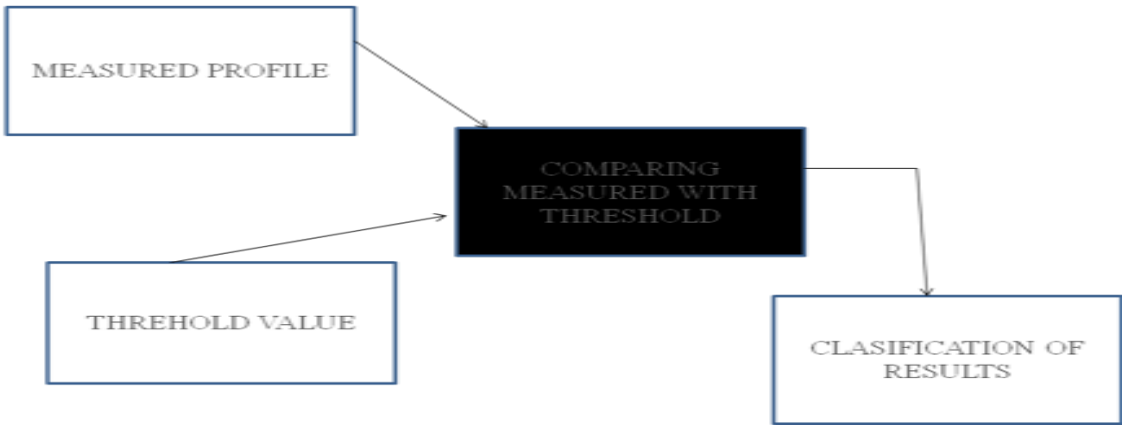
PACKET SIZE IN BYTES	0-40	41-100	101-500	501-1000	1001-1500	1500---
RELATIVE FREQUENCY	25	5%	40%	7%	23%	3%

III. THE INTEGRATED PROCESS

The integrated operation between CLP computation and the determination of a dynamic discarding threshold for CLP. A load-shedding algorithm, such as the one described in [13], is used to determine the amount (_) of suspicious traffic arriving that needs to be discarded in order to keep the utilization of the victim below a target value. Typical inputs to a load-shedding algorithm include current utilization of the victim, maximum (target) utilization allowed for the victim, and the current aggregated arrival rate of suspicious traffic. Once the required packetdiscarding percentage (_) is determined, the corresponding CLP discarding threshold, Thd, is determined from a recent snapshot of the CDF of the CLP values. The snapshot is updated periodically or upon significant changes in the packet score distribution. The adjustment of the CLP discarding threshold is done on a time-scale which is considerably longer than the packet arrival time-scale. The entire PacketScore process can be best performed in a pipelined approach as discussed in Section 4.2 in which time is divided into fixed intervals, and each operation is performed based on the snapshot of the previous period.



Specifically, the following three operations are performed in pipeline when a packet arrives:
1.Incoming packet profiling: At the end of the period, P0n =Pm is calculated and scorebooks are generated.
2. Scoring: The packets are scored according to the most recent scorebooks. .At the end of the period, CDF is generated and the cut-off threshold is calculated.
3. Discarding:
The packets are scored according to the most recent scorebooks. .The packet is discarded if its score is below the cut-off threshold score. It is also important to reemphasize that, while CLPcomputation is always performed for each incoming packet, selective packet discarding is only performed when the system is operating beyond its safe (target) utilization level. Otherwise, it will set _ to zero.



IV. CONCLUSIONS

We have outlined the process the PacketScore scheme uses to defend against DDoS attacks. The key concept in PacketScore is the Conditional Legitimate Probability (CLP) produced by comparison of legitimate traffic and attack traffic characteristics, which indicates the likelihood of legitimacy of a packet. As a result, packets following a legitimate traffic profile have higher scores, while attack packets have lower scores. This scheme can tackle never-before-seen DDoS attack types by providing a statistics-based adaptive differentiation between attack and legitimate packets to drive selective packet discarding and overload control at highspeed. Thus, PacketScore is capable of blocking virtually all kinds of attacks as long as the attackers do not precisely
mimic the sites' traffic characteristics. We have studied the performance and design tradeoffs of the proposed packet scoring scheme in the context of a stand-alone implementation. The newer simulation results in this paper are consistent with our previous research [19]. By exploiting the measurement/scorebook generation process, an attacker may try to mislead PacketScore by changing the attack types and/or intensities. We can easily overcome such an attempt by using a smaller measurement period to track the attack traffic pattern more closely. We are currently investigating the generalized implementation of PacketScore for core networks. PacketScore is suitable for the operation at the core network at high speed, and we are working on an enhanced scheme for core network operation in a distributed manner. In particular, we plan to investigate the effects of update and feedback delays in a distributed implementation, and implement the scheme in hardware using network processors. Second, PacketScore is designed to work best for a large volume attack and it does not work well with low-volume attacks. We intend to explore and improve PacketScore performance in the presence of such attack types, e.g., bandwidth

REFERENCES

[1] Akamai Technologies, Inc., http://www.akamai.com, 2006.
[2] B. Babcock et al., "Models and Issues in DataStream Systems," ACM Symp. Principles of Database Systems, June 2002.
[3] M.C. Chuah, W. Lau, Y. Kim, and H.J. Chao, "Transient Performance of PacketScore for Blocking DDoS Attack," Proc. IEEE Int'l Conf. Comm., 2004.
[4] Cisco IOS Security Configuration Guide, Release 12.2, "Configuring Unicast Reverse Path Forwarding," pp. SC-431-SC-446, http:// www.cisco.com/univercd/cc/td/doc/product/software/ ios122/122cgcr/fsecur_c/fothersf/scfrpf.pdf, 2006.
[5] C. Estan, S. Savage, and G. Varghese, "Automatically Inferring Patterns of Resource Consumption in Network Traffic," Proc. 2003 ACM SIGCOMM, pp 137-148, 2003. [6] CSI/FBI Survey, http://www.gocsi.com/forms/fbi/csi_fbi_ survey.jhtml, 2006.
[7] FBI Fugitive, http://www.fbi.gov/wanted/fugitives/cyber/ echouafni_s.htm, 2006. [8] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing," RFC 2827, 2000.
[9] L. Garber, "Denial-of-Service Attacks Rip the Internet," Computer, pp. 12-17, Apr. 2000. [10] J. Ioannidis and S.M. Bellovin, "Implementing Pushback: Router- Based Defense against DDoS Attacks," Proc. Network and Distributed System Security Symp., Feb. 2002.
[11] C. Jin, H. Wang, and K.G. Shin, "Hop-Count Filtering: An Effective Defense against Spoofed Traffic," Proc. ACM Conf. Computer and Comm. Security (CCS '03), Oct. 2003.
[12] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites," Proc. Int'l World Wide Web Conf., May 2002.
[13] S. Kasera et al., "Fast and Robust Signaling Overload Control," Proc. Int'l Conf. Network Protocols, Nov. 2001.
[14] A.D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An Architecture for Mitigating DDoS Attacks," IEEE J. Selected Areas in Comm., vol. 22, no. 1, pp. 176-188, Jan. 2004.

[15] A. Kuzmanovic and E.W. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants)," Proc. ACM SIGCOMM 2003, Aug. 2003.
[16] H. Kim and I. Kang, "On the Effectiveness of Martian Address Filtering and Its Extensions," Proc. IEEE GLOBECOM, Dec. 2003.
[17] Y. Kim, J.Y. Jo, H.J. Chao, and F. Merat, "High-Speed Router Filter for Blocking TCP Flooding under Distributed Denial-of-Service Attack," Proc. IEEE Int'l Performance, Computing, and Comm. Conf., Apr. 2003.