Transient Performance of PacketScore for blocking DDoS attacks

Mooi Choo Chuah[1]
CSE Department
Lehigh Univ.
Bethlehem, PA
(chuah@cse.lehigh,edu)

Wing Cheong Lau
Bell Laboratories
Lucent Technologies
Holmdel, NJ
(lau@bell-labs.com)

Yoohwan Kim
EECS Department
Case Western Reserve Univ.
Cleveland, OH
(yoohwan@ieee.org)

H. Jonathan Chao
ECE Department
Polytechnic Univ.
Brooklyn, NY
(chao@poly.edu)

Abstract--**Distributed Denial of Service (DDoS) attack is a critical threat to the Internet. Recently we have proposed the PacketScore scheme, a DDoS defense architecture that supports automated attack detection, on-line attack characterization and attack blocking. Its key idea is to use a statistics-based packet scoring mechanism to distinguish between legitimate and non-legitimate packets and discard packets based on the packet scores. In order for such an approach to work, we need to perform on-line traffic characterizations, and compare such characterizations with nominal profiles (generated from past history or off-line analysis). The threshold used for the score-based selective packet discard decision is dynamically adjusted based on the score distribution of recent incoming packets. In our previous paper [Kim04], we discuss how our proposed system performs in different attack scenarios.**

In this paper, we first give a brief review of the PacketScore approach and further elaborate on the transient performance under varying attack types and intensities, which may be exploited in more sophisticated attacks. We then show that PacketScore is well capable of blocking such sophisticated attacks by simply adjusting the measurement window time scale to closely track the attack profile.

I. INTRODUCTION

One of the major threats to cyber security is Distributed Denial-of-Service (DDoS) attack in which the victim network element(s) are bombarded with high volume of fictitious, attacking packets originated from a large number of machines. The aim of DDoS attack is to overload the victim and render it incapable of performing normal transactions. Recently, the DDoS problem has attracted much attention from the research community. So far, the focus has been on the design of traffic marking and traceback protocols [Be01, Pa01, Sa01, Sn01] which enable downstream routers to determine and notify the upstream routers of the attacking packets. Once the upstream sources of the attack have been identified, proposed pushback mechanisms [Io02, Ya02] are used to contain the damage of the attack. However, the effectiveness of such an approach is contingent upon the ability to extract a precise characterization of the attacking packets. While there has been recent work by the data-mining research community to recognize intrusion patterns using offline machine-learning approaches [Le98, Ma99], these schemes are mostly offline-oriented. An exception to this trend is the D-WARD approach [Mi02], which does perform limited statistical traffic profiling at the edge of the networks to detect new types of DDoS attacks online. D-WARD aims at stopping DDoS attacks near their sources, i.e., the ingress routers. While such "source-side" tackling approach is attractive in terms of having less demanding operating-speed and scalability requirements, its viability hinges on the voluntary cooperation of majority of ingress network administrators internet-wide. There are also a small set of commercial products [Mazu, Rive] which advertise limited support of statistics-based adaptive filtering techniques. However, most of these solutions do not fully automate packet differentiation or filter enforcement. Instead, they only recommend a set of binary filter rules to the network administrator to be installed in their routers or firewalls.

Recently we have proposed a statistics-based scheme, called *PacketScore*, for automated on-line attack characterization and packet discarding [Kim04], where we provided a detailed description of our PacketScore approach and simulation results under different attack scenarios. However, we did not provide any sensitivity analysis on how the score distribution changes with time when different attack types occur. In this paper, we concentrate more on the transient performance of the PacketScore system. In particular, we discuss how the performance changes with different attack types, different attack intensities and different measurement window times.

The rest of this paper is organized as follows: in Section II, we provide an overview of the PacketScore DDoS defense architecture. In Section III, we discuss how PacketScore performs in different attack scenarios and show its transient behavior. We examine the performance under changing attack types composed of four primary attack types (generic, TCP-SYN flooding, SQL Slammer worm, and nominal attack), and changing attack intensities. We conclude in Section IV with some future work that we intend to explore.

II. PACKETSCORE OVERVIEW

In PacketScore, we perform online traffic profiling of the incoming traffic and compare it with the nominal traffic profile for abnormality detection. The key concept in profiling is the notion of "Conditional Legitimate Probability" (CLP) which indicates the likelihood of a packet being a legitimate one given the attribute values it possesses. Packets are selectively discarded by comparing the CLP of each packet with a dynamic threshold. 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, in the absence of DDOS attacks. This "invariant" premise was supported by recent traffic measurements and analysis reported in [Fra03,Liu02,Kim04]. The PacketScore system is composed of two parts, the nominal profile generation part (off-line) and the scoring/discarding part (on-line). We will review them in the following subsections.

[1] This work was done while Professor Chuah was with Bell Labs.

A. Conditional Legitimate Probability

Let N_n be the number of packets arriving during a time period T during normal operation (n for *n*ormal). Let {*A, B, C*, …} be the packet attributes, $\{a_1, a_2, a_3, \ldots\}$ be the possible values for attribute A, $\{b_1, b_2, b_3, \ldots\}$ be the possible values for attribute B, and so on. We denote $P_n(A=a_i)$ as the ratio of the packets having value a_i for attribute A, which is equivalent to the number of packets having the value of a_i for attribute A divided by N_n. ($\sum_i P_n(A=a_i)=1$)

Let us assume that during the same period T, N_a attack packets are added. The total number of packets during T is then $N_n + N_a$, which we denote by N_m (a for *a*ttack, m for *m*easured). Similarly to $P_n(A=a_i)$, we define $P_a(A=a_i)$ and $P_m(A=a_i)$ as following:

- $P_a(A=a_i)$: the ratio of packets having value a_i for attribute A among N_a attack packets
- $P_m(A=a_i)$: the ratio of packets having value a_i for attribute A among N_m total packets

We define the conditional *legitimate* probability (*CLP*) as the conditional probability of a packet being legitimate given the set of attribute values it carries, that is,

- *CLP* (packet p) = P (packet p is legitimate | attribute $A = a_p$, attribute $B = b_p$,…), where a_p, b_p, … are the attribute values of a packet p

It can be rewritten as follows:

$$CLP(p) = \frac{P((p = legitimate) \cap (A = a_p, B = b_p, \ldots))}{P(A = a_p, B = b_p, \ldots)} \quad \text{... Eq. (1)}$$

$$= \frac{N_n \times P_n(A = a_p, B = b_p, \ldots)/N_m}{N_m \times P_m(A = a_p, B = b_p, \ldots)/N_m} = \frac{N_n \times P_n(A = a_p, B = b_p, \ldots)}{N_m \times P_m(A = a_p, B = b_p, \ldots)}$$

If the attributes are independent, then $P(A=a_p, B=b_p, \ldots) = P(A=a_p) * P(B=b_p) * \ldots$, and the CLP can be further rewritten as,

$$CLP(p) = \frac{N_n \times P_n(A = a_p) \times P_n(B = b_p) \times \ldots}{N_m \times P_m(A = a_p) \times P_m(B = b_p) \times \ldots} \quad \text{..... Eq. (2)}$$

Consider one of the terms $P_n(A = a_p)/P_m(A = a_p)$ in Eq.(2) during a DDoS attack. If the packet p is one of the many attack packets with attribute value a_p, then P_a ($A = a_p$) is high during attack and P_m ($A = a_p$) becomes high. Since P_n ($A = a_p$) is stable, the resulting CLP becomes low. On the other hand, for other attributes values that few attack packets have, a_p', the ratio P_a ($A = a_p'$) is low even if the actual number of such attack packets are greater than the number of normal packets. This makes P_m ($A = a_p'$) low and subsequently makes the CLP low.

CLP is considered a score for indicating the legitimacy of a packet and the effectiveness is amplified as we employ more packet attributes, including joint attributes of multiple attributes, and as the attack volume increases. The potential packet attributes are any field in the IP packet header that are expected to change substantially under DDoS attack, such as source IP prefix, protocol type, packet size, server port number (since the server bound traffic is more stable), TTL, TCP flag patterns, and combination of multiple attributes such as TTL + Source IP prefix.

B. Off-line Nominal Profile Generation

Since the actual legitimate traffic distribution during attack is unknown, we need to establish a reference profile, called nominal profile, from the past traffic. This profiling information is stored in the form of normalized histograms of one or higher dimensions. Due to the large number of attributes and attribute values to be incorporated in a profile, an efficient data structure is required to implement such histograms. Towards this end, we use *iceberg-style* histograms [Ba02], where the histogram only includes those entries in the population which appear more frequently than a preset percentage threshold, say *x%*. For entries which are absent from the iceberg-style histogram, we will use the upper bound, i.e., *x%* as their relative frequency. Tradeoffs between iceberg-threshold, histogram storage requirement and packet differentiation performance are discussed in [Kim04].

C. On-line PacketScore Operation

While attack packets arrive continuously and the true profile changes as new packets arrive, continuous profile generation and scoring is very difficult. Instead we take pipeline approach where the time is divided into fixed intervals, and each operation is performed based on the snapshot of the previous period. In PacketScore, the following 3 stages are performed in pipeline, namely, incoming packet profiling, scoring, and discarding. First, packet profile is measured at period T_1, and a scorebook is generated at the end of the period. Using the scorebook, the subsequent packets are scored at period T_2, and the score distribution graph is generated along with the cutoff threshold score at the end of period T_2. At period T_3, the incoming packets are either accepted or discarded based on the threshold score.

Incoming Packet Profiling

Similar to the off-line packet profile, an on-line profile is generated as packets arrive. Rather than performing expensive calculation of the CLP on the fly, we employ a scorebook approach. A frozen set of recent histograms in period T_1 is used along with the stored off-line nominal profile to generate a set of "scorebooks" which maps a specific combination of attribute values to its corresponding "score". To accelerate the computation, a logarithmic version of Eq. (2) is used.

Scoring and Selective Discarding

The objective of PacketScore is to prioritize the packets based on their CLP values and discarding the most likely attack packets. Since an exact prioritization would require offline, multiple-pass operations, e.g., sorting, we take the following alternative approach to realize an online, one-pass operation. First, all the incoming packets during T_2 are scored. Second, we construct a cumulative distribution function (CDF) with the CLP scores of the packets. Then a load-shedding algorithm [Ka01] is used to determine the fraction (Φ) of arriving packets to be discarded in order to control the utilization of the victim to be below a target value. Once the required packet-discarding percentage, Φ, is determined, the corresponding CLP discarding threshold, Thd, is looked up from the CDF of the CLP scores. We then discard a suspicious packet if its CLP score is below the threshold, Thd. While

CLP-computation is always performed for each incoming packet, selective packet discarding only happens when the system is operating beyond its safe (target) utilization level ρ_{target}. Otherwise, the overload control scheme will set Φ to zero.

D. Sample score distribution

Figure 1 shows a typical score distribution under a constant DDoS attack. The attack packets (red) are concentrated in the lower scored region while legitimate packets (blue) have higher scores. The black bar represents the cutoff threshold scores for discard decision, which removes the majority of the attack traffic.

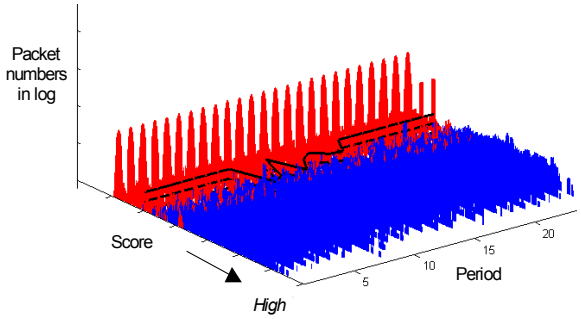

Figure 1. Periodic packet score distribution

III. Performance Analysis

A. Performance Metrics

We quantify the performance of PacketScore in the following three aspects:

First, we examine the differences in the score distribution for attack and legitimate packets. Such differences are quantified using 2 metrics, namely, R_A and R_L. Let Min_L (Max_A) be the lowest (highest) score observed for the incoming legitimate (attacking) packets. Define R_A (R_L) to be the fraction of attacking (legitimate) packets which have a score below Min_L (above Max_A). The closer of the values of R_A and R_L to 100%, the better the score-differentiation power. To avoid the masking effect of isolated outlier packets having extreme scores, we take Min_L (Max_A) to be the 1^{st} (99^{th}) percentile of the score distribution of legitimate (attacking) packets.

Second, we measure the false positive (i.e., fraction of legitimate packets got falsely discarded), and false negative (i.e., fraction of attacking packets got falsely admitted) ratios of PacketScore. While R_A and R_L can quantify the score differentiation power, the final outcome of selective discarding also depends on the dynamics of the threshold update mechanisms.

Third, we observe whether the amount of passing traffic after discarding is within an acceptable range for the destination. To measure the effectiveness of the overload control, we compare the actual output utilization ρ_{out} against the target maximum utilization ρ_{target} set by the scheme (ρ_{out}/ρ_{target}). In our simulations ρ_{target} is read from the nominal traffic profile.

We review the performance of PacketScore under constant attack, and then observe the performance under varying attacks. We study three types of changing attacks, i.e., attack type change, attack intensity change, and both attack type and intensity change. The default attack intensity is 10 times of nominal traffic, and the measurement window time is 10 seconds. For all attack scenarios, the simulation is performed for 300 seconds. For the traffic data, the packet trace data from WIDE project is used [WIDE].

B. Changing Attack Types

Table 1 shows the performance of PacketScore under constant attack with four primary attack types and mixed attack as defined as following. (All attack packets have randomized source IP address and source port number)

- *Generic attack*: All attribute values of the attack packets are uniformly randomized over their corresponding allowable ranges.
- *TCP-SYN flooding attack:* All attack packets are TCP SYN packets
- *SQL Slammer Worm attack*: All attack packets are UDP packets to port 1434 with the size between 371 to 400 bytes
- *Nominal attack*: All attacking packets resemble the most dominant type of legitimate packets observed in Internet, i.e. 1500-byte TCP packets with server-port 80 and TCP-flag set to ACK.
- *Mixed attack*: above 4 attacks are equally mixed

PacketScore shows remarkable effectiveness in discarding attack packets under constant attack, including mixed attack. After PacketScore processing, 1100% of nominal traffic volume is reduced down to about 100% of nominal traffic.

Attack Type	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Generic	1.73	0.961	99.45	99.61	0.97
TCP-SYN flooding	0.95	0.47	100	99.99	0.96
SQL Slammer Worm	0.94	0.84	99.99	100	1.00
Nominal	1.04	0.56	100	100	0.97
Mixed	2.29	0.97	98.51	99.00	1.00

Table 1: PacketScore performance under consistent attack type (Measurement window = 10 seconds)

We now observe the effect of a changing attack where an attack type randomly selected from the four primary attack types continues for an exponentially distributed period before another attack type is picked. Table 2 tabulates the simulation results for changing attacks. Changing attacks are more challenging to block due to their complex/ time-varying attacking packet characteristics. When a change occurs, it takes two measurement periods for PacketScore to establish new profiles and scorebooks due to the nature of pipeline processing. During this adjustment periods, PacketScore scheme can be misled to defend against some no-longer-existing attack packets. This situation is described in Figure 2, showing the sudden divergence upon change occurrences. The effect becomes worse if the attack type changes rapidly compared to the measurement period time scale.

Attack Type	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Changing (Ave. one attack duration = 10 sec)	11.98	21.80	79.14	90.57	3.02
Changing (Ave. one attack duration = 30 sec)	8.49	17.05	89.55	96.63	2.57
Changing (Ave. one attack duration = 60 sec)	4.15	11.11	92.53	98.28	2.01

Table 2: PacketScore performance under changing attack type (Measurement window = 10 sec., without fine-grain overload control)

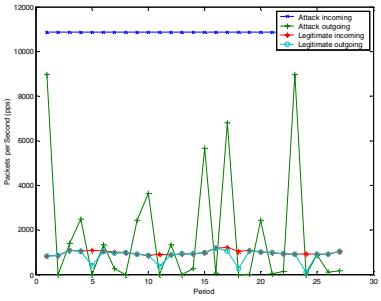

Figure 2: Incoming vs. outgoing traffic under changing attack type (Ave. attack duration = 10 sec., Measurement window = 10 sec.)

One way to improve the effectiveness of the PacketScore scheme is to apply fine-grain overload control within a measurement window as in our earlier study [Kim04] where 0.1-second fine-grain control is used in a 60-sec measurement window case. Figure 3 shows that the fine-grain overload control achieves some improvement. However, its effectiveness may be limited because the scorebook and CDF remains the same throughout the measurement window period even after the attack traffic profile changes.

Attack Type	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Changing (fine-grain OC window = 1 sec)	10.88	14.19	92.64	77.78	2.26
Changing (fine-grain OC window = 0.1 sec)	6.27	4.06	93.83	87.67	1.28

Table 3: PacketScore performance under changing attack type (Measurement window = 10 sec., ave. attack duration = 10 sec. with fine-grain overload control)

Another way is to shorten the measurement window time to speed up scorebooks/ CDF updates without fine-grain overload control. When the measurement window time is reduced to 1 second (Table 4), PacketScore can track the attack change very closely and show very good performance. The overloading is reduced from 202% to 111% of the target utilization rate where the measurement window is the same as the average attack-changing rate. (10 seconds for both) From Tables 2 and 3, we observe that shortening the measurement period works better than fine-grain overload control. Figure 3 shows the resulting incoming/outgoing traffic per period using 1-second measurement window. Some attack packets are still being admitted, but their volume is much smaller compared to the 10-second window case. As a result, the outgoing rate is more accurately controlled.

Attack Type	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Changing (Ave. one attack duration = 10 sec)	4.76	2.32	98.63	98.86	1.11
Changing (Ave. one attack duration = 30 sec)	3.91	1.24	98.41	99.63	1.01
Changing (Ave. one attack duration = 60 sec)	3.51	1.19	98.84	99.80	1.01

Table 4: PacketScore performance under changing attack type (Measurement window = 1 seconds)

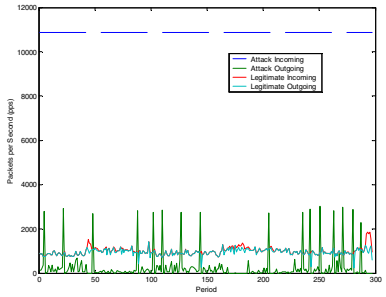

Figure 3: Incoming vs. outgoing traffic under changing attack type (Ave. attack duration = 10 sec., Measurement window = 1 sec)

C. *Changing Attack Intensities*

The attack intensity expresses the volume of attack packets in terms of multiples of the nominal packet volume. We measure the performance with varying attack intensity using generic attack. As a baseline, Table 4 shows the performance using constant attack intensity. By design, the differentiation power of PacketScore improves as the DDoS attack intensifies. This is because as attack traffic volume increases, the difference between the current traffic profile and the nominal one also increases.

Attack intensity	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
X 1	3.57	11.69	74.30	94.69	1.01
X 2	2.28	5.93	90.89	98.23	1.02
X 4	1.92	2.59	98.73	99.36	1.01
X 8	1.71	1.33	99.57	99.66	1.02
X 16	1.84	0.68	99.64	99.66	1.02
X 32	2.15	0.41	99.65	99.68	1.04

Table 5: PacketScore performance under changing attack intensities (Measurement window = 10 seconds)

In varying attack, the attack intensity changes randomly among 1,2,4,8, 16 and 32 times of nominal traffic volume. Table 6 shows the results where the attack intensity change interval is 10 seconds and 30 seconds.

Attack intensity	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Changing Randomly every 10 seconds	2.87	8.98	89.61	97.94	2.01
Changing randomly every 30 seconds	2.05	6.87	93.49	98.76	1.61

Table 6: PacketScore performance under changing attack intensities (Measurement window = 10 second)

Similar to the changing attack type case, when the attack intensity change interval is the same as the measurement window period, the performance of PacketScore degrades. However, as we reduce the measurement window time to 1 second (Table 7), the performance of PacketScore stabilizes and the outgoing rate is controlled to 125% of the target rate.

Attack intensity	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
			% R_A	% R_L	
Changing randomly every 10 seconds	7.13	7.81	63.24	93.15	1.25
Changing randomly every 30 seconds	6.31	1.46	68.07	95.56	1.05

Table 7: PacketScore performance by changing attack intensities (Measurement window = 1 second)

D. Changing Attack Types and Intensities

A clever attacker may try to vary both attack types and intensity. We combine the varying attack type and varying attack intensity, and examine the results. Using a 10 second measurement window, PacketScore only manages to throttle attack traffic such that outgoing rate is 1.5 to 3.5 times of target rate. But by reducing the measurement window time, PacketScore detects the changes in traffic pattern quickly and adapts to it. Table 8 shows the results under combined attack using 1-second window. The outgoing rate is controlled within 119% of the target rate.

Attack Type	Attack intensity	% False + ve	% False - ve	Average PDF Separation		$\frac{\rho_{out}}{\rho_{target}}$
				% R_A	% R_L	
Changing (Ave. one attack duration = 10 sec)	Changing randomly every 10 sec.	4.96	2.7	97.85	92.66	1.10
Changing (Ave. one attack duration = 10 sec)	Changing randomly every 30 sec.	4.68	2.08	98.04	92.69	1.19
Changing (Ave. one attack duration = 30 sec)	Changing randomly every 10 sec.	3.80	1.86	98.26	93.10	1.10
Changing (Ave. one attack duration = 30 sec)	Changing randomly every 30 sec.	4.41	2.27	82.11	97.90	1.05

Table 8: PacketScore performance under changing attack types and intensities (Measurement window = 1 second)

Although the false positive, negative ratios and the outgoing rates are slightly higher than in the constant attack cases, the results are well within acceptable range for practical use. Any combination of changing attacks can be effectively blocked by PacketScore scheme if the measurement window scale is shorter than the attack change granularity, e.g., 1:10 in our case. PacketScore scheme is effective with very short measurement windows e.g. 0.l-10 seconds as long as the number of packets within a window is sufficient for statistical processing. A few hundred packets or more within a measurement window is sufficient for profiling. It is difficult for an attacker to launch a precisely time-coordinated attack below the resolution of seconds due to the nature of Internet. This allows PacketScore scheme to block practically all types of changing attacks.

IV. CONCLUSIONS

We have reviewed the architecture of the PacketScore scheme that defends against DDoS attacks and studied its transient performance under changing attacks. PacketScore can tackle never-seen-before DDoS attack types by providing a statistics-based adaptive differentiation between attacking and legitimate packets. It is capable of blocking virtually all kinds of attacks as long as the attackers can't precisely mimic the sites' traffic characteristics. The packets following the nominal traffic profile have higher score while others have lower score. The more different the attack profile is from the nominal profile, the wider the score separation is.

By exploiting the profile/scorebook generation process, a clever attacker may try to mislead PacketScore by changing the attack types and/or intensities. PacketScore can easily overcome such an attack by using shorter measurement window to track the attack traffic pattern more closely.

There are other interesting issues on how different types of nominal profile affect the performance and how PacketScore performs under different iceberg coverage. They are discussed in [Kim04]. We plan to investigate the performance with packet number-based window rather than time intervals and other overloading control algorithms.

REFERENCES

[Ba02] B. Babcock et al, "Models and Issues in DataStream Systems," *ACM Symp. on Principles of Database Sys.,* Jun 2002.

[Be01] S. Bellovin, M. Leech, T. Taylor, "ICMP Traceback Messages," draft-ietf-itrace-01.txt, *Internet draft,* Oct 2001.

[Fra03] C. Fraleigh et al, "Packet –Level Traffic Measurements from the Sprint IP backbone", *IEEE Network*, November, 2003

[Io02] J. Ioannidis, S.M. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks", *Network and Distributed System Security Symp.*, Feb. 2002.

[Ka01] S. Kasera et al, "Fast and Robust Signaling Overload Control," *ICNP,* Nov. 2001.

[Kim04] Y. Kim, W.C. Lau, M. Chuah, J. Chao, "PacketScore: Statistics-based Overload Control against Distributed Denial-of-Service attack," *IEEE Infocom,* March 2004

[Lau03] W. Lau, M. Chuah, J. Chao, Y. Kim, "PacketScore - A proactive defense scheme against distributed denial-of-service attacks", *NSF Proposal under submission*

[Le98] W. Lee, S.J. Stolfo, "Data Mining Approaches for Intrusion Detection," *the 7th USENIX Security Symp.*, Jan 1998.

[Liu02] D. Liu, F. Huebner, "Application Profiling of IP Trafic," *LCN* 2002

[Ma99] D. Marchette, "A Statistical Method for Profiling Network Traffic," *the 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, Apr 1999.

[Mazu] Mazu Networks Inc. http://www.mazunetworks.com

[Mi02] J. Mirkovic, G. Prier, P. Reiher, "Attacking DDoS at the Source," *ICNP*, Nov. 2002.

[Pa01] K. Park and H. Lee, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack," *Infocom,* 2001.

[Rive] Riverhead Networks Inc. http://www.riverheadnetworks.com

[Sa01] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network Support for IP Traceback," *IEEE/ACM TON,* Vol. 9, no. 3, June 2001.

[Sn01] A. Snoeren et al, "Hash-based IP Traceback," *SIGCOMM,* Aug. 2001.

[WIDE] MAWI Traffic Archive, http://tracer.csl.sony.co.jp/mawi/

[Ya02] D.K.Y. Yau, J.C.S. Lui, F. Liang, "Defending Against Distributed Denial-of-Service Attacks with Max-min Fair Server-centric Router Throttles," *IWQoS*, 2002.