Use of Spectral Analysis in Defense Against DoS Attacks

Chen-Mou Cheng, H.T. Kung, Koan-Sin Tan
Division of Engineering and Applied Science
Harvard University

***Abstract* -- We propose using spectral analysis to identify normal TCP traffic so that it will not be dropped or rate-limited in defense against denial of service (DoS) attacks. The approach can reduce false positives of attacker identification schemes and thus decrease the associated unnecessary slowdown or stoppage of legitimate traffic. For the spectral analysis, we use the number of packet arrivals of a flow in fixed-length time intervals as the signal. We then estimate the power spectral density of the signal, in which information of periodicity, or lack thereof, in the signal reveals itself. A normal TCP flow should exhibit strong periodicity around its round-trip time in both flow directions, whereas an attack flow usually does not. We validate the effectiveness of the approach with simulation and trace analysis. We argue that the approach complements existing DoS defense mechanisms that focus on identifying attack traffic.**

I. Introduction

In recent years DoS attacks have become one of the most serious security threats to the Internet. This is because they may result in massive service disruptions and also because they have proven to be difficult to defend against [6]. These attacks are attempts by attackers to deny access of legitimate users to network services. Though it is possible to exploit software vulnerabilities such as buffer overflow, DoS attacks are usually achieved by continually and excessively consuming finite resources that are necessary to provide the services, such as link bandwidth, server CPU processing power, or memory storage [5].

A general networking approach to mitigate DoS attacks is to identify and rate-limit attack traffic, preferably at points as close to sources as possible, in order to reduce the collateral damage. However, the problem of identifying attack traffic is generally difficult because attackers can manipulate their traffic and packets to defeat detection.

In this paper we describe a novel use of spectral analysis in identifying normal TCP traffic and show how this new method can complement existing DoS defense mechanisms that focus on identifying attack traffic. More specifically, the method exploits the fact that normal TCP flows must exhibit periodicity in packet transport associated with round-trip times. Based on this fact it is possible for the method to identify normal TCP traffic reliably. After other DoS defense methods such as those in [8] have identified certain traffic aggregates as candidates for attack traffic, our new spectral analysis based method can rule out those candidates which are deemed to be normal TCP traffic. Thus our method can help reduce the impact of false positives of these other methods.

The rest of this paper is organized as follows. In Section II, we empirically establish the relationships between various packet processes and the corresponding power spectral densities. We go on to validate our concept of spectral analysis based identification of TCP traffic, using simulation and traces in Section III and IV. We describe a possible way to integrate our method into existing DoS defense mechanisms in Section V. In Section VI, we discuss limitations and potential issues of our method, based on our findings and the experience obtained from simulations and trace analysis. We compare our work with those of other researchers in Section VII and conclude this work in Section VIII.

II. Power Spectral Density of Packet Process

We first review basic properties of TCP and Power Spectral Density (PSD) and then give several examples illustrating how PSD can be used to discover periodicity in a packet process.

A. Background

TCP [12], the underlying protocol used by the majority of the traffic on today's Internet, is a sliding-window based, acknowledgement (ACK) driven transport protocol. The window size of a TCP flow limits the number of in-flight packets it can have in the network. The window size is determined by the advertised window size of the receiver and the estimated congestion level of the network. Packet transmission of TCP can be characterized by the packet conservation principle [9]. According to this principle, every arriving data packet at the receiver allows the departure of an ACK packet, and every arriving ACK packet at the sender enables the injection of a new data packet into the network. Consecutive packets within a window are sent out in a bursty manner, constrained only by the transmission time of the bottleneck link.

A consequence of the packet conservation principle is that TCP flows exhibit periodicity. By periodicity, we mean that, if we see a TCP packet at any point in the network, then chances are that after one round-trip time (RTT), we will see another packet belonging to the same TCP flow passing through the same point.

We consider a random process $\{X(t), t = n\Delta, n \in N\}$, where Δ is a constant time interval, *N* is the set of positive integers, and for each *t*, *X(t)* is a random variable. Here *X(t)* represents the number of packet arrivals for a TCP flow in (t - Δ, *t*]. We refer to this random process as the *packet process* in the rest of this paper.

To study the periodicity embedded in the packet process $\{X(t), t = n\Delta, n \in N\}$, we use its autocorrelation function:

$$R_{xx}(\tau, t) = E[X(t)X(t+\tau)]$$

where $R_{xx}(\tau, t)$ captures the correlation of the packet process and itself at lag τ. We assume that the packet process is (wide-sense) stationary. Thus we can drop the dependence of R_{xx} on *t*.

To compute the periodicity embedded in the packet process amounts is equal to finding the maxima of R_{xx}. In practice, it is often computationally more efficient to find periodicity in the frequency domain using the Power Spectral Density (PSD). The PSD function of the packet process is the Discrete Fourier Transform of its autocorrelation function:

$$S_x(f) = \sum_{k=-\infty}^{\infty} R_{xx}(k) e^{-i2\pi fk}$$

In the absence of a complete mathematical description of the random process, one often resorts to PSD estimators, instead of the true PSD, to discover periodicity from realizations (measured signals) of the process. Periodogram is a commonly used PSD estimate technique [1], which captures the "power" that a signal contains at a particular frequency. Here power means the strength of periodicity at the corresponding period.

Traffic signals are noisy due to randomness introduced by factors such as queueing delay, user think time, etc. Raw periodogram in itself is an inconsistent estimator of PSD: if we compute periodogram of several realizations of a same random process, we will often end up with very different spectrum estimates. To reduce the estimate fluctuation, we use Welch's averaged, modified periodogram method to compute PSD estimates [13].

B. Illustrative Examples

We present some simple examples to illustrate the information one can obtain from PSD. We first consider Poisson processes and processes with heavy tail distributions. Poisson processes are commonly used to model packet arrival time in traditional queueing theory, whereas heavy-tailed processes are believed to be a more accurate model for Internet traffic in recent literature. Then we introduce periodicity into packet processes and show how periodicity reveals itself in the resulting PSD estimates. Furthermore, the corresponding frequency domain feature is salient in the face of limited superposition of similar processes.

We first look at a Poisson packet process formed by counting the number of arrivals in each of the 10 ms bins, with the inter-arrival times independently drawn from an exponential distribution, and with mean arrival rate equal to 200 arrivals per second. By applying Welch's method to obtain a PSD estimate, we can show both a realization of the packet process and its PSD estimate in Fig. 1. The resulting PSD estimate has a rather flat power distribution over all frequencies, which corresponds to that of a white noise process.

In the second example, we use a similar packet process, but this time the inter-arrival times are drawn from a Pareto distribution. Recall that the probability density function of a Pareto distribution is:

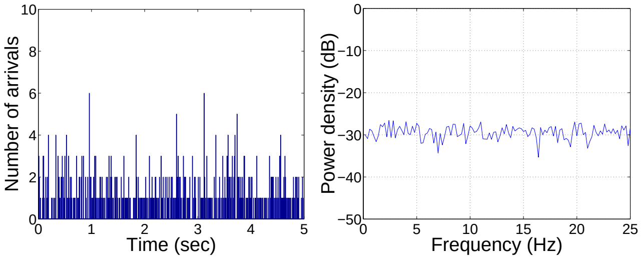

Fig. 1. A realization (left) of a Poisson packet process with exponential inter-arrival times and its associated PSD estimate (right), in which power is evenly spread across all frequencies due to lack of periodicity

$$P(x) = \frac{\alpha k^{\alpha}}{x^{\alpha+1}}, \qquad x \geq k$$

where α controls the tail behavior of the distribution. For $1 < \alpha < 2$ the distribution has a finite mean but infinite variance, resulting in a slowly decaying autocorrelation function and thus long-range dependence [10]. In the example we use $\alpha = 1.3$, $k = 0.001$. Fig. 2 shows that the resulting PSD estimate has more power at low frequencies than in Fig. 1.

We introduce periodicity into the packet process in the following examples by generating deterministic arrivals interleaved with probabilistic arrivals. In the third example, probabilistic arrivals have exponentially distributed inter-arrival times and each of them further triggers a deterministic arrival after 130 ms. In Fig. 3 we see how the resulting PSD estimate reveals this periodicity: it has peaks at the integral multiples of a fundamental frequency approximately at 7.7 Hz, which converts to 130 ms and agrees with what we have set forth. The harmonics do not exhibit a decaying envelope as seen in most band-limited signals, because the probabilistic arrivals form a Gaussian-like process and thus have a flat PSD.

In the last example, we show periodicity is preserved under small degrees of spatial superposition and small perturbations in temporal displacement. More specifically, instead of triggering a new arrival exactly 130 ms later, now the periods are drawn from a uniform distribution in $130 \pm 10\%$ ms. Fig. 4 shows that the resulting PSD estimate of superposition of 16 processes still exhibits periodicity, though the peaks are lower compared with that of one single process in Fig. 3, due to small

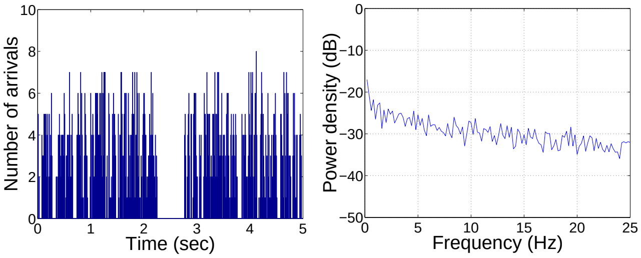

Fig. 2. A realization (left) of a heavy-tailed packet process with Pareto inter-arrival times and its associated PSD estimate (right): the PSD contains more power at low frequencies, compared with Fig. 1

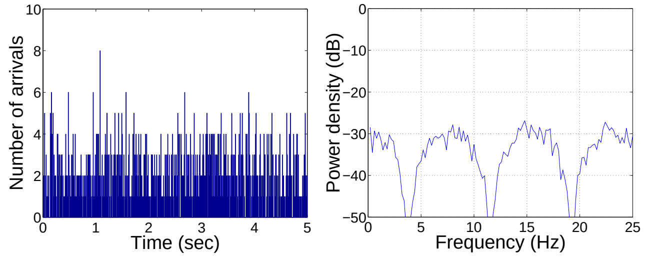

Fig. 3. A realization (left) of a packet process with exponential inter-arrival times mixed with deterministic arrivals and its associated PSD estimate (right): the first peak is located at the frequency corressponding to the time lag of probabilistic arrivals and the triggered deterministic arrivals

variations in periodicity.

III. Network Simulations of the Spectral Analysis Based Approach

In this section, we use simulation to validate our idea of using spectral analysis to identify TCP traffic. We use a network whose topology is a binary tree. The binary tree topology is deliberately chosen to help gain insights into the frequency domain behavior of aggregates of TCP flows, as well as to find limitations of this spectral analysis based technique.

As depicted in Figure 5, the simulated network is structured as a binary tree of depth d, which makes it easy to investigate how different levels of aggregation affect periodicity. For illustrative purposes, we set d = 10 in the following description. Node S_0 is the traffic sink, which sits behind a 100 Mbps link. All other links are 1 Gbps. All internal links L_1 through L_{511} have a propagation delay of 10 ms, whereas leaf links L_{512} through L_{1023} have propagation delays uniformly distributed in the range between 10 ms and 20 ms. This makes the RTTs between the sink node and leaf nodes uniformly distributed in the range between 200 ms and 220 ms. At each link there is a 750-packet Random Early Detection (RED) queue [3]. RED parameters are set as follows: minimum threshold set to 125 packets, maximum threshold set to 375 packets, gentle_ bit set.

The attack traffic, from node S_{513} to node S_0, is modeled using a constant bit rate UDP packet process with randomized inter-packet times and an average bit rate of 10 Mbps. There are long FTP sessions between all other leaf nodes, S_{513} through S_{1023}, and the sink node. All packets contain 1000 bytes. The configuration allows the pipes, along with the queue in front of the bottleneck link, to hold approximately five to six packets for each TCP flow. In the simulation, we aim to validate that large-volume TCP flows exhibit periodicity around RTT. Thus, we deliberately make TCP flows operating in congestion avoidance phase without experiencing many retransmission timeouts (RTO). In real traffic, however, some of the TCP flows could experience quite a number of RTOs from time to time, but such flows are unlikely to pose serious threats in terms of bandwidth usage. We further discuss the issues on timeouts in Section VI.

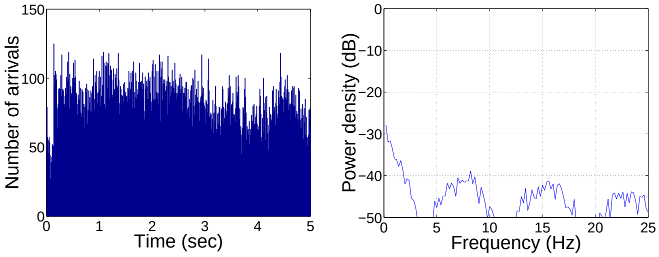

Fig. 4. The superposition of 16 realizations (left) of a packet process with Pareto inter-arrival times mixed with slightly perturbed deterministic arrivals and its associated PSD estimate (right): note that the PSD has lower peaks due to superposition and RTT variation, as well as a decaying envelope, compared with that in Fig. 3, due to long-range dependence

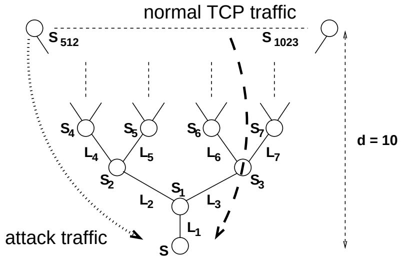

Fig. 5. The topology of the simulation, in which the attack flow is from the leftmost leaf node, whereas legitimate traffic comes from the rest of the leaf nodes

Fig. 6 shows two typical PSD estimates of TCP packet arrival processes. TCP flows show strong periodicity, for the same reasons as we have seen in the examples in Section II. The PSD estimate has peaks at the integral multiples of a fundamental frequency around 4.7 Hz, which corresponds to a period of 210 ms in time domain, since the time resolution is also 10 ms. Also note that periodicity is preserved after aggregating 256 similar flows, though the power gets spread out as the degree of statistical multiplexing increases.

Fig. 7 shows the power at the first peak in the PSD estimates of TCP aggregates measured along the S_2, S_4, S_8,...,S_{256} path. As we have pointed out, the height of the first peak decreases as level of statistical multiplexing in an aggregate of TCP flows increases. Another factor that affects the height of the first peak is whether the aggregate has been contaminated by attack flows. At places near the source of the attack, there is little power under the fundamental frequency, because the aperiodic attack flows have stronger signal strength than legiti-

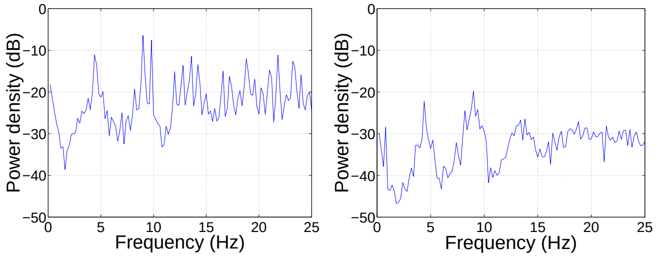

Fig. 6. The PSD estimates for aggregates consisting of a single TCP flow at S_{513} (left) and 128 TCP flows at S_5 (right): the height of the peak at fundamental frequency decrease as the level of aggregation increases

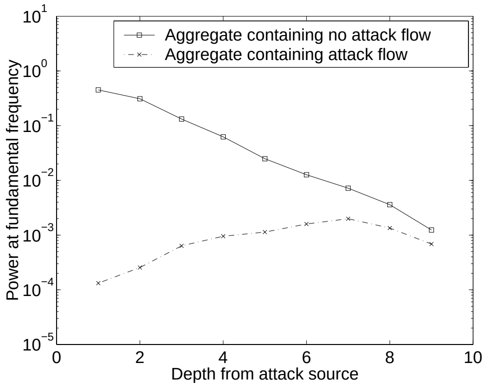

Fig. 7. The relative power under fundamental frequencies of PSD estimates for two groups of TCP aggregates; the one with UDP attack traffic has lower power at fundamental frequencies

mate TCP flows. As we move closer to the victim, the signal strength of TCP flows gets stronger, and hence the increasing trend. At the node where there are 127 TCP flows, the effect of statistical multiplexing starts to dominate, so we see a decreasing trend after we have more than 127 flows. This imposes limitations on the level of aggregation, which is an important implementation consideration because we need to keep per-aggregate states, as will be addressed in Section VI.

IV. Validation using Traces

We further validated our method with traffic traces. The trace files were obtained on May 6 and 7, 1999, at a 100Mbps link that connected the Harvard Faculty of Arts and Sciences to the rest of the campus.

We extract connection information for TCP flows from trace files using *tcptrace* [11]. The traces show that more than 66% of the packets are carried by flows that last longer than 40 seconds. This length of time appears to be sufficient for spectral analysis to identify TCP traffic. Furthermore, the variation in RTTs from one trip to another is quite small: 88% of the flows have relative standard deviation of less than 50%, as shown in the cumulative probability function of relative standard deviation in Fig. 9. Note that the *tcptrace* uses the time difference between a TCP data segment and its corresponding ACK to calculate RTT, so that the RTTs reported include delay at end-hosts. Thus, the measured variation in RTTs is often an overestimate.

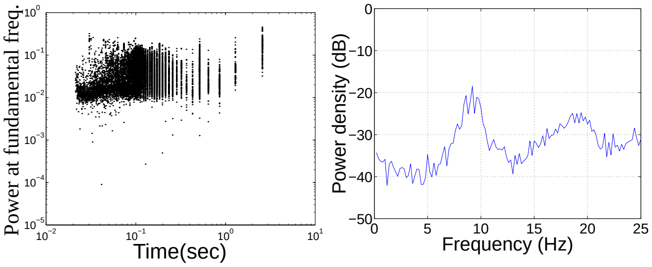

Fig. 8. RTT estimates (left) and a typical PSD estimate (right) calculated from the 1999 Harvard trace; it confirms that most of the long TCP flows exhibit periodicity, which can be seen in PSD

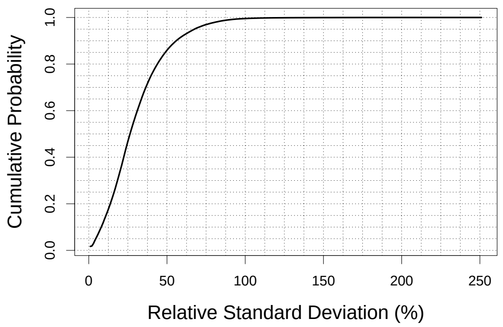

Fig. 9. The cumulative probability distribution of relative standard deviation of round-trip times, indicating that over 88% of the flows have small RTT variations

We use simple heuristics to find the power at fundamental frequency: out of the highest five peaks from the PSD estimate, we select the one with the lowest frequency and use it to estimate RTT. The result and a typical PSD estimate are plotted in Fig. 8, in which we can see that for most of the TCP flows, the relative power at the fundamental frequency is above 5×10^{-3}. We use such heuristics to determine whether a flow is TCP. The results on the Harvard trace are summarized in Table 1.

Though the ratio of false positives and false negatives is small, we did discover a few flows, often between sites that are connected with high-speed short-RTT links, to have white noise like PSD. Furthermore, though 96% of the flows have RTT estimates smaller than 500 ms, there are cases in which we observe stronger periodicity at time scales larger than one second. These are usually contributed by on-off type of TCP flows, such as web sessions with persistent TCP. Fig. 10 shows an example of this type of flow.

V. Usage Examples

Our method complements the defense mechanisms proposed in [2] and [8]. We first consider the case where attack packets reveal true source IP addresses. This may result from wide deployment of ingress filters, which mitigates IP spoofing. In this case, for example, the spectral analysis techniques of this paper can complement the Aggregate Congestion Control (ACC) and Pushback mechanism at routers [8] in the following way. The ACC mechanism will mark any aggregate with high traffic volume as suspects for causing the congestion and rate-limit them accordingly. With the aid of spectral analysis

TABLE 1

	TCP flows	Non-TCP flows
Identified as periodic traffic	81.8%	15.7%
Identified as non-periodic traffic	18.2%	84.3%

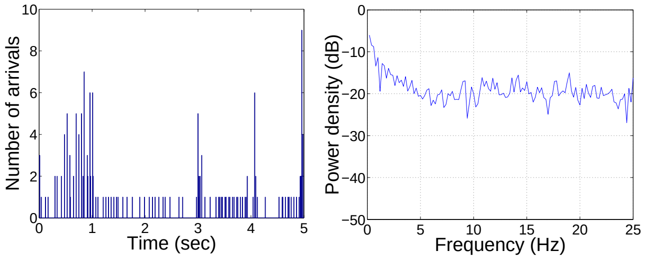

Fig. 10. The arrival packet count (left) and the corresponding PSD estimate (right) of an on-off type of TCP flow, in which stronger periodicity is found at time scales larger than seconds

techniques, we can lower the volume threshold to flag flows as suspicious more aggressively and then examine them more closely. Once a high-volume traffic aggregate has been identified as a legitimate TCP traffic, it no longer needs to be rate-limited, thereby reducing the impact of the false positives caused by a more aggressive threshold.

A similar approach can work even when attack packets may use spoofed source IP addresses. In this case, the pushback requests will back-propagate to the points where the routers can classify normal and attack traffic in terms of other types of traffic sources, such as the network interfaces they come from. As illustrated in Fig. 11, if the attacker were to randomize source IP addresses, legitimate long traffic flows would be contaminated only slightly, so spectral analysis techniques would still be able to pick up these flows. The routers can classify flows according to which network interfaces they come in and then label interfaces that do not contain enough legitimate traffic as suspicious. Pushback requests can back-propagate to the routers that are able to accurately differentiate attack flows with minimum collateral damage, where preferential drops are performed to mitigate the attacks.

VI. Discussions

One might argue that attackers can have attack flows to mimic the periodicity of normal TCP flows by sending out packets periodically. A countermeasure is to consider return paths along with forward paths. Since TCP flows are two-way traffic, the reverse flows must also exhibit similar periodicity. It would be difficult for attackers to trigger flows in the reverse direction to have the desired periodicity, unless they actually use closed-loop protocols to launch attacks. In this case the difficulty of launching DoS attacks has been significantly raised, that is, attackers have to consume an amount of resources comparable to that of a normal TCP sender.

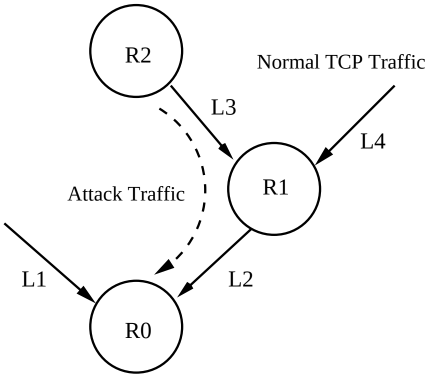

Fig. 11. The pushback mechanism when source IP addresses of attack packets are randomized: router R0 marks L2 as suspicious since it contains less legitimate TCP flows than L1 (router R1 does so recursively), and therefore the pushback request is back-propagated along the L2-L3 path to R2

Our spectral analysis method best deals with long TCP flows. For short TCP flows, the effect of their statistical multiplexing may outweigh their intrinsic periodicity, and as a result, our method will not be able to identify them as normal TCP flows. Fortunately, as we have seen earlier in our traces, short TCP flows usually represent a small percentage of the total TCP load to a network in terms of packet counts. This means that when there are a high percentage of packets belonging to short TCP flows, these packets most likely belong to attack traffic.

For our method, short flows may also be created by TCP timeouts which segment flows into shorter ones. Again, these short flows resulting from TCP timeouts are not expected to constitute a large percentage of the total TCP load.

The RTT of a TCP flow may vary slightly from trip to trip, due to queueing delay variations. The sampling period has to be large enough to tolerate RTT fluctuation, while small enough to make the periodicity to be observed distinguishable. Thus, a limitation of our method is that it can not identify TCP flows with very small RTTs. These flows generally do not pose severe security threats because they are mostly local traffic, or traffic between two administratively close networks. Nevertheless, one possible remedy on this shortcoming of our method is to set up a list of neighboring sites and treat the traffic related to these sites separately. Another possible remedy is to add artificial delay at the router where we take measurements, so that the range within which RTTs vary is relatively small.

When estimating PSD, it is possible to aggregate flows of the same RTT, as demonstrated in Section II and III. Because TCP flows between two networks should have the same RTTs, it is natural to aggregate flows according to subnet prefixes, using a similar mechanism as MULTOPS [4].

VII. Related Work

It was proposed in [7] the use of spectral analysis techniques to study network performance, in particular, the use of wavelets to infer and detect the qualitative aspects of various network performance problems. However the wavelet technique is not suitable for identifying normal or attack traffic in defense against DoS attacks because it does not have enough resolution at the time scale of interest. Rather, the scale-localization ability of wavelets is more useful for finding the qualitative properties of packet processes.

It was proposed in [14] a simple statistics-based mechanism to detect TCP SYN flood attacks. The idea is to detect deviation from an expected balanced SYN/FIN packet ratio using a non-parametric, cumulative sum method. However, such simple heuristics are unlikely to sustain, since the attackers can try to fool the system by mixing their SYN and FIN packets.

VIII. Concluding Remarks

We propose using spectral analysis techniques to identify normal TCP flows, in order to avoid dropping packets from legitimate TCP traffic in defense against DoS attacks. We investigate the feasibility of the proposed method and find that the periodicity of normal TCP traffic is preserved under reasonable levels of aggregation. Combined with other volume-based attack identification mechanisms, the result of this paper offers an approach that can mitigate DoS attacks without hurting normal TCP traffic.

Acknowledgment

This research was supported in part by grant 2000-DT-CX-K001 from the Office of Justice Programs, National Institute of Justice, U.S. Department of Justice, in part by DARPA through AFRL/SNZW under contract F33615-01-C-1983, and in parts by research grants from Microsoft Research. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the research sponsors.

Reference

[1] Bloomfield, P, “Fourier Analysis of Time Series: An Introduction,” 2nd Edition, John Wiley & Sons, 2000.

[2] Ferguson, P. and Senie, D., “Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing,” RFC 2827, May 2000.

[3] Floyd, S. and Jacobson, V., “On traffic phase effects in packet-switched gateways,” *Internetworking: Research and Experience*, vol. 3, no.3, pp. 115–156, September 1992.

[4] Gil, T. and Poletto, M., “MULTOPS: a Data Structure for Bandwidth Attack Detection,” in Proceedings of the 10th Usenix Security Symposium, August 2001.

[5] Houle, K. and Weaver, G., “Trends in Denial of Service Attack Technology,” October 2001, http://www.cert.org/archive/pdf/DoS_trends.pdf.

[6] Householder, A., Manion A., Pesante, L., Weaver G., and Thomas, R., “Managing the Threat of Denial-of-Service Attacks,” October 2001, http://www.cert.org/archive/pdf/Managing_DoS.pdf.

[7] Huang, P., Feldmann, A., and Willinger, W., “A Non-Intrusive, Wavelet-Based Approach to Detecting Network Performance Problems,” in Proceedings of ACM SIGCOMM Internet Measurement Workshop 2001, November 2001.

[8] Ioannidis, J. and Bellovin, S., “Implementing Pushback: Router-Based Defense against DDoS Attacks,” in Proceedings of NDSS’02, February 2002.

[9] Jacobson, V., “Congestion avoidance and control,” *ACM Computer Communication Revie*w, vol. 18, no. 4, pp. 314–329, August 1988.

[10] Leland, W., Taqq, M., Willinger, W., and Wilson, D., “On the Self-Similar Nature of Ethernet Traffic,” in Proceedings of ACM SIGCOMM’93, August 1993.

[11] Ostermann, S., Tcptrace, http://jarok.cs.ohiou.edu/software/tcptrace/index.html

[12] Postel, J., “Transmission Control Protocol – DARPA Internet Program Protocol Specification,” RFC 793, September 1981.

[13] Stoica, P. and Moses, R., “Introduction to Spectral Analysis,” pp. 52–54, Prentice Hall, 1997.

[14] Wang, H., Zhang, D., and Shin, K., “Detecting SYN Flooding Attacks,” in Proceedings of Infocom 2002, June, 2002