Monitoring the Macroscopic Effect of DDoS Flooding Attacks

Jian Yuan and Kevin Mills, *Senior Member*, *IEEE*

Abstract—Creating defenses against flooding-based, distributed denial-of-service (DDoS) attacks requires real-time monitoring of network-wide traffic to obtain timely and significant information. Unfortunately, continuously monitoring network-wide traffic for suspicious activities presents difficult challenges because attacks may arise anywhere at any time and because attackers constantly modify attack dynamics to evade detection. In this paper, we propose a method for early attack detection. Using only a few observation points, our proposed method can monitor the macroscopic effect of DDoS flooding attacks. We show that such macroscopic-level monitoring might be used to capture shifts in spatial-temporal traffic patterns caused by various DDoS attacks and then to inform more detailed detection systems about where and when a DDoS attack possibly arises in transit or source networks. We also show that such monitoring enables DDoS attack detection without any traffic observation in the victim network.

Index Terms—DDoS attack, monitoring, network traffic, attack dynamics, spatial-temporal pattern.

1 Introduction

The success of the Internet derives in large part from the end-to-end principle [1], which enabled deploying a simple network infrastructure (of packet-forwarding nodes supported by a few routing protocols), allowing network applications to evolve independent of the core network. In particular, the end-to-end congestion-control mechanisms of the TCP (Transmission-Control Protocol) played a key role in achieving a robust and stable Internet. At the same time, the existing end-to-end mechanisms have proven ineffective at protecting the Internet from distributed denial-of-service (DDoS) attacks, an increasingly frequent, global disturbance [2].

A DDoS attack is a simultaneous network attack on a victim (e.g., a Web server or a router) from a large number of compromised hosts, which may be distributed widely among different, independent networks. By exploiting asymmetry between network-wide resources and local capacities of a victim, a DDoS attack can build up an intended congestion very quickly at an attacked target. The Internet routing infrastructure, which is stateless and based mainly on destination addresses, appears extremely vulnerable to such coordinated attacks.

DDoS attacks cannot be detected and stopped easily because forged source addresses and other techniques are used to conceal attack sources. DDoS attacks can take a victim network off the Internet even without exploiting particular vulnerabilities in network protocols or weaknesses in system design, implementation, or configuration. While applying security patches may avert attacks against protocol or system vulnerabilities, congestion-inducing DDoS attacks exploit an inherent weakness in the Internet design and, thus, present a serious threat to Internet stability. This paper focuses on such congestion-inducing (so-called flooding) DDoS attacks.

Packet filtering is the main response to confirmed DDoS attacks; however, attacks must first be detected. Flooding attacks can be most easily detected at the victim, where all attack packets can be observed [3]. Unfortunately, an attack victim cannot defeat a flooding attack simply through detection. Instead, attack packets must be filtered in transit networks, preferably close to attack sources, before they converge on the victim. Attempts in transit networks to detect such attacks often lead to a high false-alarm rate. Similarly, networks hosting attack sources may observe only a normal outgoing pattern of Internet traffic [4], which shows high variability [5]. Stealthy attacks (such as increasing rate, fluctuating rate, and natural-network-congestion-like attacks [3], [6]) increase detection difficulty.

In this paper, we argue that DDoS flooding attacks alter internal characteristics of network-wide traffic so that an appropriate monitoring method can detect the attacks without observing traffic in the victim network. To avoid congestion in the Internet, all TCP flows adapt themselves in a self-organized manner. Adaptive behaviors of flows in different directions play a crucial role to keep the Internet stable and to form macroscopic (or aggregate) traffic patterns. DDoS attack packets do not observe TCP congestion-control algorithms; rather, they overwhelm the intended victim, causing well-behaved flows to back off and then ultimately to starve. DDoS attacks also impair other traffic flows that happen to share a portion of the congested network. Such network-wide phenomena might show themselves in shifting spatial-temporal patterns, which can be captured by using a novel technique we developed to analyze macroscopic behavior [7]. Our technique can infer the congestion state of specific network areas without directly measuring them. This proves advantageous in

- *J. Yuan is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084 China. E-mail: jyuan@tsinghua.edu.cn.*
- *K. Mills is with the Information Technology Laboratory of the National Institute of Standards and Technology, NN 445, Gaithersburg, Maryland 20899. E-mail: kmills@nist.gov.*

Manuscript received 5 Feb. 2005; revised 2 June 2005; accepted 15 Aug. 2005; published online 3 Nov. 2005.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-0026-0204.

1545-5971/05/$20.00 © 2005 IEEE Published by the IEEE Computer Society

attack detection because congested routers near the victim may fail to collect and transfer measurement data.

In this paper, we show that macroscopic (space-time) behavior of network traffic could provide significant information to detect DDoS attacks, which exhibit apparent effects on network congestion even while attackers constantly modify their techniques to avoid detection. When coupled with a dynamic monitoring capability deployed in transit or source networks, our analysis method could provide an alert function to warn detection systems about where and when an attack probably arises. Since attacks are becoming more sophisticated, we experiment in this paper with different attack modes: constant rate, increasing rate, natural-network-congestion-like, subgroup, pulsing, and TCP-targeted attacks. We use simulation results to show how our technique monitors spatial-temporal patterns under diverse DDoS flooding attacks. We show that these attacks, which have an apparent effect on network congestion, reveal themselves as shifts in spatial-temporal traffic patterns without any observations from the suffering victim. The dynamic nature of some stealthy attacks may even become an advantage because our technique benefits from increased correlation arising under shifting patterns in network traffic.

The rest of this paper is structured as seven sections. In Section 2, we place our ideas within the context of prior and on-going research related to DDoS defense. In Section 3, we explain our technique to analyze macroscopic behavior. In Section 4, we describe our simulation model. In Section 5, we report our results. In Section 6, we discuss strengths and limitations of our proposed technique. We outline future work in Section 7, before concluding in Section 8.

2 Related Work

Our literature survey found researchers attempting to detect DDoS attacks from three different perspectives: 1) near the victim [8], [9], [10], [11], [12], [13], 2) near attack sources [14], [15], [16], and 3) within transit networks [17], [18], [19], [20], [21]. From all these perspectives, researchers are investigating various approaches to analyze packet data (e.g., headers, aggregate flows, and correlations) in an effort to distinguish normal traffic from attack packets. The problem is difficult because DDoS attackers can release high volumes of normal-looking packets without being easily traceable or filtered. To date, no known set of characteristics clearly and accurately distinguishes between normal and attack traffic. Further, Staniford et al. [22] argue that, by controlling only one million hosts on the Internet, attackers can initiate DDoS attacks sufficiently diffuse to avoid detection using current techniques. Chang [23] discusses the difficulties in more detail.

Techniques that envision placing detection mechanisms near the attack victim share similar advantages and disadvantages. First, positioning near the victim provides a clear point to observe all incoming traffic, which should ease detection because attack traffic may be a significant portion of incoming traffic. Unfortunately, attack traffic could overload local resources and degrade the effectiveness of response mechanisms. Further, identifying attack traffic does not necessarily identify sources (because addresses may be forged), and does nothing to stop the attack. Selected researchers investigate attack suppression methods; however, such methods require deployment of cooperative, distributed mechanisms within elements throughout a network. Attack suppression can be carried out more effectively near sources; however, detecting attacks near sources proves more difficult because attack traffic might be a small fraction of all traffic generated near a source. Detecting attack traffic in transit networks (as we propose) provides a unique perspective because traffic monitors placed throughout a network might be able to collect information that can be subjected to correlation analysis. Such analysis could possibly detect some classes of stealthy attacks difficult to detect near sources or victims and could serve as an early-warning system to alert and activate more detailed monitoring near suspicious sources and suspected victims. Below, we survey representative research from each perspective: 1) near the victim, 2) near sources, and 3) in transit networks.

Most DDoS-related research has focused on detection mechanisms deployed near vulnerable servers, where incoming attack traffic could deny access to legitimate users. Many mechanisms attempt to detect attacks by analyzing specific features, e.g., header information, connection counts, correlations, and congestion. For example, Noh et al. [8] attempt to detect attacks by computing the ratio of TCP flags to TCP packets received at a Web server. (TCP flags denote how to interpret other fields within the header.) Noh et al.'s algorithms also consider the relative proportion of arriving packets devoted to TCP, UDP (User-Datagram Protocol), and ICMP (Internet Control Message Protocol) traffic. The premise of Noh's work is that characteristics of DDoS traffic differ from normal traffic. Similarly, Basu et al. [9] proposed techniques to extract features from connection attempts and then to classify attempts as suspicious or not. The rate of suspicious attempts over a day helped to expose stealthy denial-of-service attacks, which attempt to maintain effectiveness while avoiding detection. Li and Chi [10] used autocorrelation of arrival traffic as a feature to distinguish between normal and attack traffic. However, assuming absence of correlation between legitimate and malicious traffic seems implausible because both types of traffic must transit some shared portion of the network.

Other researchers have investigated techniques to discard attack traffic. Ioannidis and Bellovin [11] proposed an aggregate-based congestion control (ACC) algorithm to be deployed in routers (intermediate nodes that forward packets between source and destination). ACC combines detection with rate limiting. Here, each router periodically classifies discarded packets (from the router's drop history) into "aggregates," based on a selected set of characteristics. To block an aggregate upstream, a cooperative mechanism called "pushback" enables routers to transfer their findings hop-by-hop toward attack sources, which can then be subjected to rate limiting. Some other researchers have also devised response mechanisms [12], [13] triggered by DDoS-induced congestion, although they do not specify ways to determine the congestion.

Some researchers have focused on techniques that can be deployed near traffic sources. In general, detecting DDoS traffic near attack sources is ineffective because individual sources do not generate sufficient packets to induce local congestion. Instead, congestion arises closer to the victim, where attack packets arrive in aggregation. However, DDoS traffic may still show up to some extent at sources since attack packets do not observe TCP congestion-control algorithms. Mirkovic et al. [14] devised D-WARD, a DDoS defense system installed in edge routers (that connect clients and servers to a transit network) to monitor the asymmetry of two-way packet rates and, thus, to identify attacks. Similarly, Gil and Poleto [15] proposed a heuristic data-structure (MULTOPS), which exploits correlation of inbound and outbound packet rates for subnet prefixes (i.e., higher-order bits in a network address) at different aggregation levels to detect ongoing congestion-based attacks in a source network. Kim et al. [16] proposed a discrete-wavelet transform technique to statistically analyze correlation of destination addresses in outgoing traffic at an edge router.

In the future, Internet Service Providers (ISPs) may offer subscribers DDoS defenses in the form of enhanced security services, such as virtual-private networks (VPNs), which ensure traffic flows only among a designated set of trusted computers, and managed firewalls [17]. More sophisticated schemes have been proposed [18], but remain too immature to deploy. Talpade et al. [19] designed NOMAD, a network-traffic monitor deployed in a single transit router to detect network anomalies by analyzing packet-header information, such as time-to-live (TTL) and source and destination addresses. Akella et al. [20] proposed a detection mechanism to identify anomalies by comparing current traffic profiles with profiles of normal traffic, as observed at edge routers, which exchange information with other edge routers to increase confidence. In work most closely related to our own, Lakhina et al. [21] suggested a subspace method for characterizing network-wide anomalies by examining the multivariate time series of all origin-destination flows among routers in a transit network. Using Principal Component Analysis [24], origin-destination flows are decomposed into constituent eigenvectors (explained below in Section 3), where the top few eigenvectors depict normal traffic and remaining eigenvectors expose anomalies. Our own approach reveals congestion by examining only the eigenvector corresponding to the largest eigenvalue, which we associate with a strong correlation over the whole network [7], [25]. In addition, by taking advantage of increased correlations arising in a large network, our approach might require only a few observation points.

3 Analysis Technique

Networked computers exchange packets across transport connections that adapt their rate of flow based on apparent congestion (measured by round-trip delay and packet loss). Microscopic behavior (packet flows on individual connections) can look quite different from macroscopic behavior (aggregate flows across all connections in space and time). In large networks, interactions among adaptive connections and variations in user demands lead to emergence of spatial-temporal correlations among various characteristics, such as congestion [26]. Capturing macroscopic patterns of such correlation over time could help us to understand shifting traffic patterns, to identify operating conditions, and to reveal traffic anomalies. Motivated by these insights, we developed a novel method to analyze spatial-temporal traffic at large scale [7], [25]. Our analysis method can infer a shift in traffic pattern for large areas of interest outside those few areas where measurements are made. In this paper, we apply our method to watch network-wide patterns based on measurements from only a few observation points. In this section, we introduce our method.

3.1 Spatial-Temporal Correlation

The TCP congestion-control algorithm exhibits a self-organizing property: When a large number of connections share the Internet, underlying interactions among the connections avoid router congestion simultaneously over varying spatial extent. A number of empirical studies have convincingly shown that the temporal dynamics of Internet traffic exhibits long-range dependence (LRD) [27], [28], [29], [30], [31], which implies existence of nontrivial correlation structure at large timescales. A recent study of correlations among data flows in a French scientific network, Renater [32], detected the signature of such spatial-temporal correlation.

The Renater study uses random matrix theory (RMT) to analyze cross-correlations among network flows. RMT compares a random correlation matrix—constructed from mutually uncorrelated time series—against a correlation matrix for data under investigation. Deviations between properties of the two matrices convey information about "genuine" correlations. In the Renater study, the largest eigenvalue is approximately 100 times larger than predicted for uncorrelated time series, and the eigenvector component distribution of the largest eigenvalue deviates significantly from the Gaussian distribution predicted by RMT. Further, the Renater study reveals that all components of the eigenvector corresponding to the largest eigenvalue are positive, which implies their collective contribution to the strong correlation in congestion over the whole network. Since all network flows contribute to the eigenvector, the eigenvector can be viewed as an indicator of spatial-temporal correlation in network congestion.

In this view, congestion emerges from underlying interactions among flows crossing a network in various directions. Our hypothesis: congested destinations, caused by flash crowds (i.e., legitimate users simultaneously accessing the same Web site) or by flooding DDoS attacks, should be exposed through their correlation, as revealed by components of the eigenvector of the largest eigenvalue. Therefore, we can define a weight vector by grouping eigenvector components corresponding to a destination together to build up information about the influence of the destination over the whole network. Contrasting weights of the weight vector against each other, we can summarize a network-wide view of spatial-temporal correlation, locate congested destinations, and observe how traffic patterns change. In particular, using relatively few observation points, we could infer a shift in the spatial-temporal correlation of large areas of interest outside those few areas

where measurements are made [7], [25]. This approach could significantly reduce requirements for data, perhaps to the point where analysis could occur in real time. In what follows, we describe the mathematical details of our analysis method.

3.2 Representing Network Flow Data

We assume a network organized in four levels:

1. backbone routers,
2. subnet routers,
3. leaf routers, and
4. host computers.

Backbone routers connect to each other with high capacity links to forward packets among geographically distributed sites, which we call subnets. Each subnet router connects one geographic area (an aggregation of leaf routers) to a backbone router, allowing packets to be forwarded between the local subnet and remote subnets. Each subnet router also connects to a set of leaf routers, each of which forwards packets between a specified set of host computers and the connected subnet router. In this conception, a source host computer in one subnet may send a packet to a receiving host computer in a remote subnet by forwarding the packet to a designated leaf router. The leaf router then forwards the packet to a designated subnet router, which then forwards the packet to a connected backbone router. The backbone router forwards the packet toward the backbone router connected to the (destination) subnet containing the receiving host computer. The destination subnet router forwards the packet to the leaf router to which the receiving host computer connects. The leaf router then forwards the packet to the receiving host. Modern computer networks share a similar topological organization.

Assume N subnets, interconnecting through backbone routers to form a network, with L subnet routers selected as observation points to log outbound traffic. First, we need to represent packets flowing between source-destination pairs (of subnets). Let $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)^T$ denote the flow vector of corresponding packet counts, observed in L subnets during a given time interval. Each element of this flow vector is itself a vector defining the number of packets flowing into the corresponding subnet from each observation subnet. To obtain the flow variables in this vector, we first enumerate the destination subnets and then the observation posts by 1 to L, and group indices by subnet: subnets sending to the first subnet in the first block, $\boldsymbol{x}_1$, and those sending to the second subnet in the second block, $\boldsymbol{x}_2$, and so forth. Thus, we form $\boldsymbol{x}$ with subvectors in the order

$$\begin{aligned}\boldsymbol{x}_1 &= (x_{11}, x_{21}, \ldots, x_{L1})^T,\\ \boldsymbol{x}_2 &= (x_{12}, x_{22}, \ldots, x_{L2})^T, \ldots,\\ \boldsymbol{x}_N &= (x_{1N}, x_{2N}, \ldots, x_{LN})^T,\end{aligned}$$

where x_{ij} represents packet flow from the ith observation point ($i = 1, 2, \ldots, L$) to the jth subnet ($j = 1, 2, \ldots, N$). Each flow variable x_{ij} is normalized as f_{ij} by its mean m_{ij} and standard deviation σ_{ij},

$$f_{ij} = (x_{ij} - m_{ij})/\sigma_{ij}. \tag{1}$$

The normalized flow vector $\boldsymbol{f}$ corresponding to $\boldsymbol{x}$, comprises N normalized subvectors $\boldsymbol{f_k}(k = 1, 2, \ldots, N)$, where each subvector is formed from normalized flow variables f_{ik} ($i \le L$ and $k \le N$).

3.3 Cross-Correlation Analysis

Cross-correlation analysis is a tool commonly used to analyze multiple time series. We can compute the equal-time cross-correlation matrix $\boldsymbol{C}$ with elements

$$C_{(ij)(kl)} = \langle f_{ij} f_{kl} \rangle_t, \tag{2}$$

which measures correlation between f_{ij} and f_{kl}, where $\langle \cdots \rangle_t$ denotes an average over a selected time interval t. We are free to select any useful value for t. The cross-correlation matrix is real and symmetric, with each element falling between -1 and 1.

We can further analyze the correlation matrix $\boldsymbol{C}$ through eigenanalysis [33]. The equation

$$\boldsymbol{C}\boldsymbol{w} = \lambda \boldsymbol{w} \tag{3}$$

defines eigenvalues and eigenvectors, where λ is a scalar, called the eigenvalue. If $\boldsymbol{C}$ is a square K-by-K matrix, then $\boldsymbol{w}$ is the eigenvector, a nonzero K by 1 column vector. (In our application, we set $K = L(N-1)$, which is the number of sampling points multiplied by the number of subnets, omitting flows within the same subnet.) Eigenvalues and eigenvectors always come in corresponding pairs. An eigenvector is a special kind of vector for its associated matrix, because the action of the underlying operator represented by the matrix takes a particularly simple form: rescaling by a real number multiple. The eigenvector $\boldsymbol{w^1}$ corresponding to the largest eigenvalue λ_1 often has special significance. The eigenvalues and eigenvectors can be computed in various ways [33]. We exploit the MATLAB **eig** command, which uses the QR algorithm [34].

3.4 Defining the Weight Vector

The cross-correlation matrix contains information about underlying interactions among various flows among subnets. The components of the eigenvector $\boldsymbol{w^1}$ of the largest eigenvalue λ_1 represent the corresponding flows' influences on macroscopic behavior, abstracted from the matrix $\boldsymbol{C}$ into the pair (λ_1, $\boldsymbol{w^1}$). The eigenvector $\boldsymbol{w^1}$ comprises N subvectors, i.e., $\boldsymbol{w^1} = (\boldsymbol{w_1^1}, \boldsymbol{w_2^1}, \ldots, \boldsymbol{w_N^1})^T$. The kth subvector $\boldsymbol{w_k^1}$, corresponding to the kth subnet, is formed from components w_{ik}^1 ($i \le L$ and $k \le N$) representing the ith obsevation point's contribution to the kth subnet. We consider the square of each component, $(w_{ik}^1)^2$, instead of w_{ik}^1 itself because $\sum_{i,k}(w_{ik}^1)^2 = 1$ [35]. We define the weight S_k ($k = 1, 2, \ldots, N$) to be the sum of all $(w_{ik}^1)^2$ in the kth subvector $\boldsymbol{w_k^1}$, i.e.,

$$S_k = \sum_{i}^{L} (w_{ik}^1)^2. \tag{4}$$

S_k represents the relative strength of flows from L observation points toward the kth routing domain. Thus, the knowledge of weight vector $\boldsymbol{S} = (S_1, S_2, \ldots, S_N)$ across varying k constitutes a summary view of network-wide traffic. The evolving pattern of spatial-temporal correlation

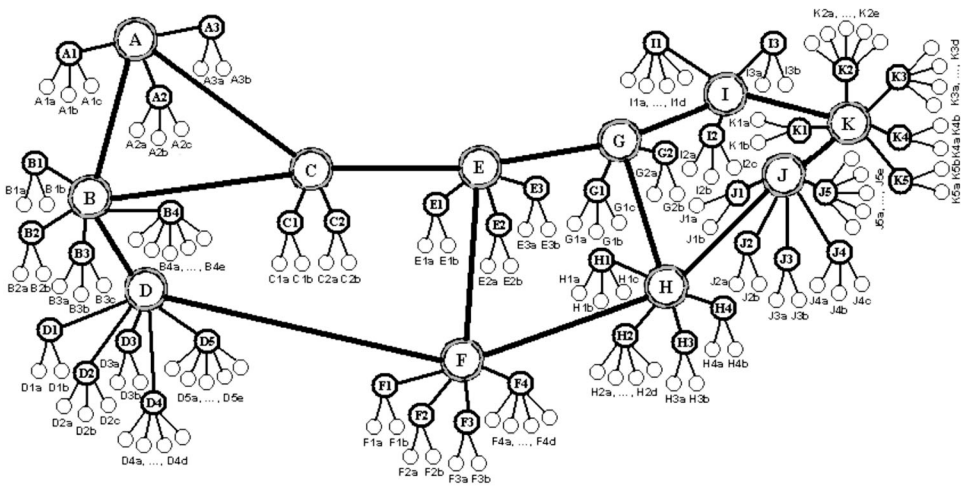

Fig. 1. The simulation model with 11 backbone routers, 40 subnet routers, and 110 leaf routers.

might allow us to infer where and when network congestion emerges.

4 Simulation Model

Simulation plays a key role in understanding network behavior. Choosing a proper level of abstraction for a model depends very much on the objective. Studying collective phenomena seems to require simulating networks with large spatial extent, but also including essential details representing protocol mechanisms across several layers of functionality (e.g., application, transport, network, and link). Yet, computational requirements must be restricted in space and time. Most available network simulators represent detail not germane to our study. Such a simulator would require that we configure numerous low-level parameters and consume computation simulating irrelevant details. For that reason, we chose to base our study on an abstract model that we developed and validated against current understanding of Internet dynamics.

Previously, we adopted a two-tier modeling approach that maintains the individual identity of packets [26]. While our two-tier model has been applied successfully to represent some traffic characteristics in large networks, doubts exist about the realism of the regular network structure of our model. In this paper, we retain the individual identity of packets and the Reno TCP congestion-control algorithm, but replace the regular network structure of our previous two-tier model with an irregular topology based on a real network.

4.1 Topology

We transform our two-tier model into an irregular four-tier topology, as shown in Fig. 1. (The host-computer tier is not shown.) While the network becomes heterogeneous and hierarchical, (tier-four) host-computer behavior remains homogeneous. The (tier-one) backbone topology, including 11 (backbone) routers $(A, B, \ldots K)$, resembles the original Abilene network [36]. We model links between backbone routers to have varying delays. The longest link (between routers D and F) has a 20 ms propagation delay; the shortest link (between routers J and K) has a 3 ms delay. Forty (tier-two) subnets connect to the backbone through subnet routers, represented by designators such as A1 and B2. Each subnet contains a variable number of (tier-three) leaf routers, such as A1a and B2b. Each leaf router supports an equal number (200) of (tier-four) source hosts, and a variable number (≤ 800) of (tier-four) receivers, activated on demand. Link capacities gradually increase from host to backbone, with backbone links being hundreds of times faster than links to hosts.

4.2 Traffic Sources

Our model includes 22,000 sources (200 per leaf router times 110 leaf routers), which operate at the packet level. Each source generates traffic as an ON/OFF process, alternating between wake and sleep periods with average durations λ_{on} and λ_{off}, respectively. At the beginning of each ON period, a destination receiver is chosen randomly from among leaf routers other than the local routers, i.e., all flows transit through at least one backbone link. Though we choose destinations with equal probability, our irregular network topology (Fig. 1) leads to higher probability of selecting leaf routers in some subnets over others.

Modern TCP implementations contain four intertwined algorithms: slow start, congestion avoidance, fast retransmit, and fast recovery. We simulate Reno TCP, except that our model reduces the congestion window to half the current window size after receiving one, instead of three, duplicate ACKs. While packets can be lost in our model, all packets on a connection take the same route, so no reordering occurs. We previously verified [7], [25], [26] that our TCP model generates behavior consistent with the dynamics of Internet traffic at various timescales.

When awake, a source may send, subject to any restrictions imposed by our TCP model, one packet at each time-step to the source's attached leaf router. The packets are placed at the end of the router's queue. When sleeping, the source does not generate new packets at each time-step. ON/OFF sources provide a convenient model of user behavior.

Empirical measurements on the Internet observe a heavy-tailed distribution of file sizes [5], [27], [28], [29], [30], [31]. Here, we use the Pareto distribution to model heavy-tailed characteristics. The Pareto distribution function has the following form: $P[X \leq x] = 1 - (k/x)^{\alpha}$, $k \leq x$, where $0 < \alpha < 2$ is the shape parameter. Using empirical measurements of Internet traffic, Crovella and Bestavros [28] showed that ON (data transfer) periods are heavy-tailed with $\alpha = 1.0 - 1.3$, and OFF (thinking) periods are heavy-tailed with $\alpha = 1.5$. For Web traffic, they found a strong preference for small file sizes. For our experiments, we selected the same shape parameter, $\alpha = 1.5$, for both ON and OFF processes; however, we selected different means. Here, $\lambda_{on} = 50$ (packets) is selected to represent the preference for small files, as is typically the case with Web page downloads. Empirical observations of OFF periods change dramatically between observations made at night or in the day. We selected $\lambda_{off} = 5,000$ (milliseconds) to represent the average thinking time before a user requests another file. For our purposes, we need to simulate background traffic that is neither too sparse nor too congested. A network driven by sparse traffic cannot produce the network-wide coherence required for a few observation points to sense shifts in traffic patterns.

4.3 Routers

Packets, the basic transmission unit on the Internet, contain destination addresses (used by routers to correctly forward) and source addresses (used by receivers to identity the destination for acknowledgments). Our four-tier topology requires us to make some assumptions about the relative packet-forwarding rate between each tier. Our model allows sources to generate one packet every millisecond, or 1,000 packets per second (pps). We assume that leaf routers forward at 5,000 pps, subnet routers forward at 20,000 pps, and backbone routers forward at 160,000 pps. Selecting these relative rates mirrors the way in which capacity increases with aggregation in a hierarchical network. Of course, real networks exhibit larger link capacities than our model, but also carry more connections. In particular, real backbone links can sustain much higher rates than we model; however, we find that further increasing capacity has no influence on our results because backbone links are already lightly loaded.

To store and forward packets (which in our model travel a constant, shortest path between a source-destination pair for each flow), all routers maintain a queue of limited length, where arriving packets are stored until they can be processed: first in, first out. The maximum queue length in routers may influence both network performance and simulation results: small queue lengths lead to many losses during TCP slow start, while large queues produce excessive delays. To achieve a reasonable balance, TCP simulations often [37], [38] set router queue lengths in a range of 10 to 200 packets. We assume that setting maximum queue length (160 packets, here) within this range would not influence our qualitative findings.

We select several subnet routers (B4, D5, F4, I1, and J5) as observation points, which record all outbound flows to destination leaf routers. We assume a central collector reliably receives a continuous stream of measured data from the observation points in time to perform analysis.

4.4 Model Summary

In summary, we model a hierarchical network with increasing forwarding capacity moving upward from sources to backbone routers. While forwarding capacities in our model are below those of real networks, we simulate fewer (up to 22,000) transport connections than would be carried in real networks. We size forwarding capacities to yield moderate levels of congestion given the number of sources we simulate. We select a maximum queue length for routers (within the range used often in TCP simulations) so that excessive packet discards or queuing delays do not influence our results. Our model ensures at least 10,000 connections are simultaneously active (i.e., in an ON period). Each active connection generates 2,000 pps (when including acknowledgments from the destination). In aggregate, then, our model generates at least 20 million pps of background traffic to create the haystack in which attack traffic may hide.

5 Simulation Experiments

DDoS attacks directed against the network infrastructure can lead to more widespread damage than those directed against individual Web servers. Here, a subnet router (I1) will be the attack target. (Elsewhere in [25], we report results where the attack target is a leaf router.) Since routers under attack may fail to collect and transfer measurement data, we assume in our experiments that the attack disables the observation point deployed at the subnet-router I1; thus, we perform our analysis using data from only four observation points (B4, D5, F4, and J5), which encompass 10 percent ($L = 4$) of the 40 subnets. Since stealth of attack strategies may vary, we experiment with a range of attacks: constant rate, increasing rate, natural-network-congestion-like, pulsing, TCP-targeted, and subgroup attacks. Fig. 2 illustrates four attack classes.

We uniformly distribute 50 DDoS attack sources throughout the simulated network. In our least stealthy (constant-rate) attack, each attack source generates 200 pps, for an aggregate rate of 10,000 attack pps (one for every 2,000 normal packets). In our most stealthy attack, three groups of (~ 17) attack sources alternate, yielding an aggregate rate of 3,400 attack pps (less than one for every 5,900 normal packets). We describe each attack further below, as we apply our analysis technique to detect the attacks.

5.1 Constant Rate Attack

Constant rate, the simplest attack technique, is typical of known DDoS attacks [3]. We arrange for all the 50 attack sources simultaneously to launch constant-rate attacks. We do not have attack sources generate packets with full force [14], so they cannot be easily identified through attack intensity at the source or in transit networks. We assume a variable H represents the intensity of an attack source. Since sources can only create one packet every millisecond, the maximum attack rate is one packet per millisecond; thus, $H \leq 1$ packet/ms. Fig. 2a shows the attack dynamics with a constant rate, starting from time t_0. We experiment with a constant-rate DDoS attack where $H = 1/5$, that is, each attack source creates one attack

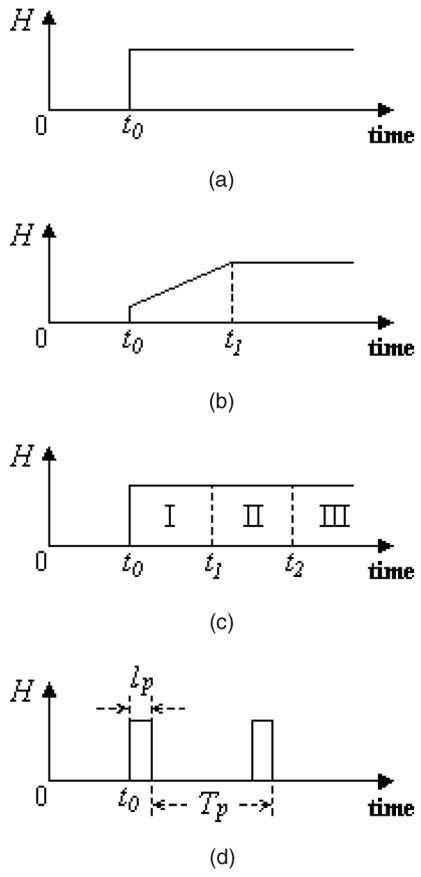

Fig. 2. Attack dynamics for: (a) constant rate, (b) increasing rate, (c) subgroup, and (d) pulsing.

packet for every five milliseconds (200 pps per source) beginning from t_0 (= 500 s).

To show spatial changes in traffic under this constant-rate attack ($H = 1/5$), we calculate the weight vector S using M data points within a moving time window MT from one period to the next. Fig. 3 shows S evolving with $T = 2\,\mathrm{s}$ and with the time window MT ($= 200 \times 2\,\mathrm{s} = 400\,\mathrm{s}$) sliding ahead every 20 s. The time axis indicates the end of the moving time window. The location axis represents 11 backbone-router zones, each of which denotes the subset of subnet routers therein. We can see that the attack leads to a network-wide shift in spatial-temporal correlation, and congestion at the victim (I1) reveals itself. Since we can observe this phenomenon and get the time and location of the attack (and without any help from the suffering victim), this type of monitoring should be meaningful to trigger detailed detection and filtering. On the other hand, Fig. 3 also contrasts the distinct effect of the transient period (during onset of the attack) with indistinctness after the new pattern becomes steady (say around $t = 900\,\mathrm{s}$). With fewer observation points, the effect of transient periods is helpful for monitoring the network-wide pattern shifting over time [7], [25].

Attack intensity H may influence spatial-temporal dynamics. Fig. 4 shows the spatial-temporal pattern of a constant-rate attack with $H = 1/10$ (100 pps per source), and the weaker visibility of I1 (compared against Fig. 3 when $H = 1/5$). We can easily imagine an attack weak enough not to cause an apparent shift of spatial-temporal correlation. In such cases, our method would prove ineffective.

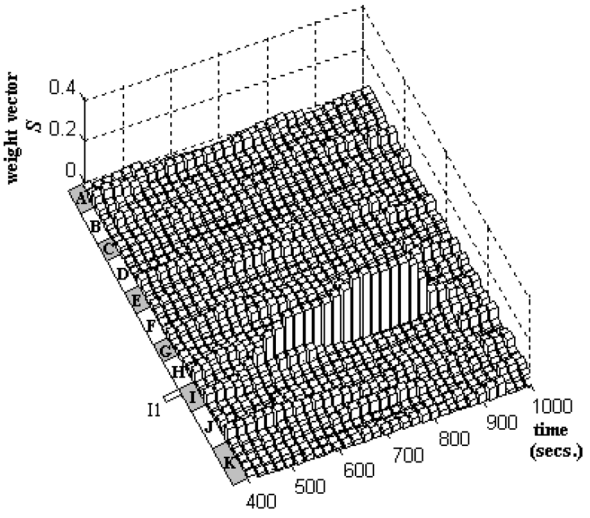

Fig. 3. Spatial-temporal pattern of constant-rate attack with $H = 1/5$.

While current detection methods seem designed mainly to counter constant-rate DDoS attacks, attackers may choose more sophisticated attack dynamics. As defense mechanisms are deployed to counter simple attacks, we are likely to see more complex attack patterns that make countermeasures against attacks more difficult.

5.2 Increasing Rate Attack

Usually, an abrupt change in traffic volume is an important signal to initiate anomaly detection. Attacks that exhibit a gradually increasing rate, as shown in Fig. 2b, may lead to slow exhaustion of a victim's resources. The state change in the victim network could be so gradual that services degrade slowly over a long period, delaying detection of

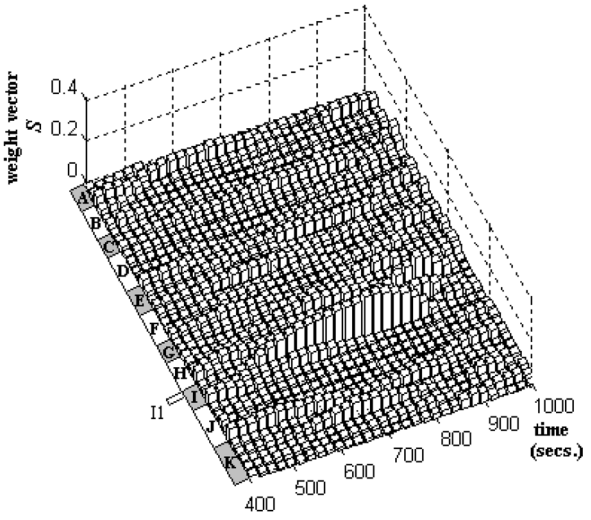

Fig. 4. Spatial-temporal pattern of constant-rate attack with $H = 1/10$.

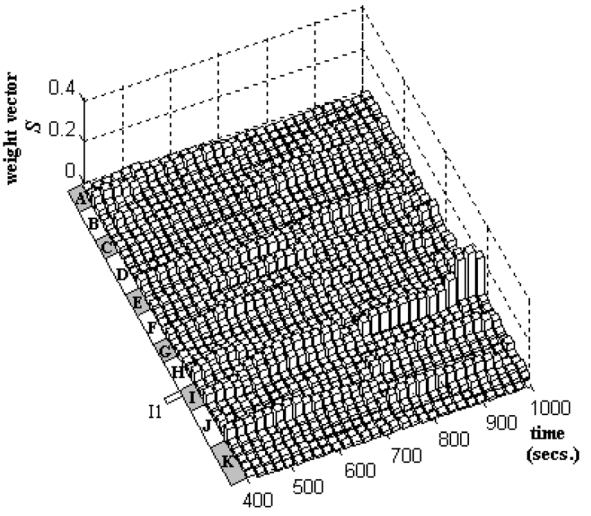

Fig. 5. Spatial-temporal pattern of increasing-rate attack with $H = 1/5$ increased from $H = 1/35$ over 300 seconds from $t_0 = 500$ s to $t_1 = 800$ s.

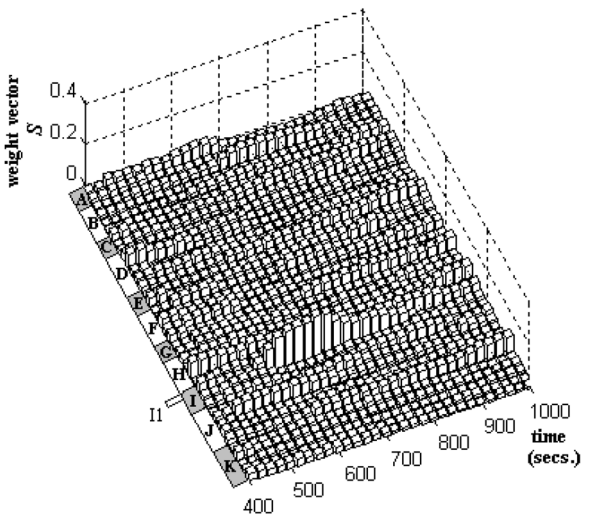

Fig. 6. Spatial-temporal pattern of natural-network-congestion-like attack with $\lambda'_{on} = 5$ ms, $\lambda'_{off} = 50$ ms, and $N_s = 50$.

the attack; thus, some other form of anomalous pattern must be identified.

In this experiment, we assume that the steady attack rate ($H = 1/5$) of each attack source is achieved gradually over 300 seconds (from $t_0 = 500$ s to $t_1 = 800$ s) starting from a weak rate ($H = 1/35$). Here, each attack source begins by generating about 29 pps, which increases gradually to a full rate of 200 pps. Fig. 5 shows the spatial-temporal pattern of this increasing-rate attack. The attack gradually builds up congestion in router I1 as attack intensity increases and, thus, reveals itself in the spatial-temporal evolution. An increasing-rate attack might deliberately accelerate slowly taking a very long time to reach a steady intensity; however, we believe that the congestion caused by this class of attack will still be discovered eventually [25].

5.3 Natural-Network-Congestion-Like Attack

Flash crowds on the Internet can trigger false alarms among DDoS attack-detection mechanisms. Further, DDoS attacks may mimic natural network congestion in order to avoid detection [6]. Usually, legitimate traffic occurs in waves, while DDoS attacks, such as ICMP ping flooding, direct malicious traffic toward victims in a fairly persistent form. An experienced attacker may design each attack source to behave like a normal user, exhibiting bursts of traffic followed by silent periods, so that the flooding attack would appear similar to natural network congestion. In particular, if attack packets use the same forged source addresses during each burst period, then the attacks cannot be detected simply by recognizing sudden change in the number of connections [39].

Following the expression formulated by Huang et al. [6], we can calculate the minimum number of attack sources required to deplete the link capacity (C_v packets/ms) of a victim,

$$N_s \geq C_v(\lambda'_{on} + \lambda'_{off})/\lambda'_{on}, \tag{5}$$

where λ'_{on} and λ'_{off} denote the mean ON and OFF periods for each source. In our experiments, the capacity of the victim subnet is 20 packets/ms, i.e., $C_v = 20$. If we set $\lambda'_{on} = 5$ ms and $\lambda'_{off} = 50$ ms, then $N_s \geq 220$. Here, we use only 50 such attack sources to degrade victim performance, and we observe if this low-grade attack leads to a corresponding shift in spatial-temporal pattern. Fig. 6 shows the spatial-temporal pattern formed by this low-grade attack, which mimics natural network congestion. The spatial-temporal evolution reveals the induced congestion.

5.4 Pulsing Attack

Pulsing attacks, exhibiting a fluctuating rate oscillating between H and zero, periodically reduce attack traffic in order to avoid detection. The dynamics of a pulsing attack appear as an ON/OFF pattern with period T_p and burst duration l_p, as shown in Fig. 2d. During a pulsing attack, attack sources periodically abort the attack only to resume it later.

In our simulated pulsing attacks, we set $T_p = 300$ s, $l_p = 60$ s, and $H = 1/5$. Fig. 7 shows the resulting spatial-

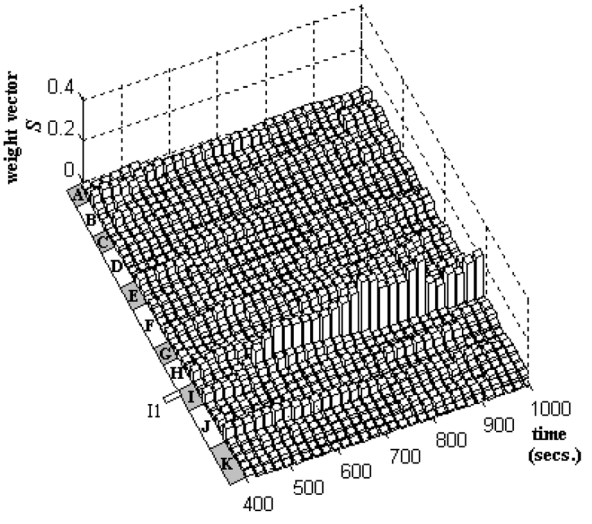

Fig. 7. Spatial-temporal pattern of pulsing attack with $T_p = 300$ s, $l_p = 60$ s, and $H = 1/5$.

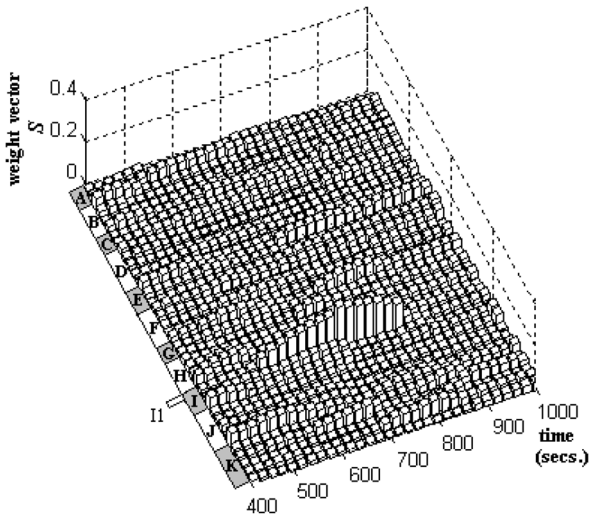

Fig. 8. Spatial-temporal pattern of TCP-targeted attack with $T_p = 1.1$ s, $l_p = 100$ ms, and $H = 1$.

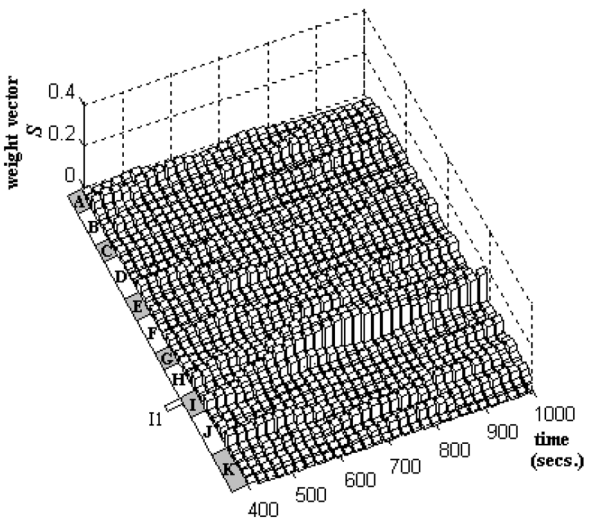

Fig. 9. The spatial-temporal pattern of the subgroup attack with three subgroups and $H = 1/5$.

temporal pattern; the attack is clearly revealed. The dynamic nature of the pulsing attack leads to frequent shifts in the traffic pattern, which strengthens correlation and causes the greater weight of victim I1 to persist.

A sophisticated attacker may attempt to launch a special variant of the pulsing attack: a low-rate TCP-targeted DDoS attack [40], which exploits the TCP retransmission time-out mechanism to throttle TCP flows, while eluding detection. Under such an attack, TCP flows to the victim may continually incur loss as they try to exit their timeout states. The TCP-targeted DDoS attack transmits short-duration high-rate bursts periodically. We simulate such a TCP-targeted attack with the same parameters used by Kuzmanovic and Knightly [40], i.e., $T_p = 1.1$ s, $l_p = 100$ ms, and $H = 1$. Fig. 8, which shows the spatial-temporal pattern of this TCP-targeted DDoS attack, still reveals the induced congestion at the victim I1.

Note that the weight of the victim in Fig. 8 does not benefit from the dynamics of this TCP-targeted attack. Fig. 8 is similar to Fig. 4 and Fig. 6, which exhibit natural patterns of congestion. In fact, the actual intensity

$$\left(\frac{l_p \times H}{T_p} = \frac{0.1 \times 1}{1.1} = \frac{1}{11}\right)$$

of our simulated TCP-targeted attack is the same as the intensity

$$\left(\frac{\lambda'_{on}}{\lambda'_{on} + \lambda'_{off}} = \frac{5}{5 + 50} = \frac{1}{11}\right)$$

of our attack that mimics natural network congestion, which is similar to the intensity ($H = 1/10$) of our simulated constant-rate attack (Fig. 4). The TCP-targeted and the natural-congestion attacks both exhibit fluctuating rates but over small timescales (near or under 1 s); so, the associated spatial-temporal traffic patterns tend to become steady more quickly, causing loss of the increased correlation associated with changing traffic patterns. However, the dynamics of the pulsing attack (Fig. 7) varies over a larger time scale (about 300 s); so, the increased correlation is present not only at onset of the attack, but continues to appear during the entire attack.

5.5 Subgroup Attack

If compromised sources are divided into several subgroups that coordinate so that one subgroup is always active, then successive attacks by the subgroups can still induce continuous denial of service at an attack victim. To simulate a subgroup attack, we divide 50 attack sources into three subgroups. As shown in Fig. 2c, each of the three subgroups (I: 17 sources, II: 17 sources, and III: 16 sources) is active in turn: the first subgroup attacking from t_0 to t_1, the second subgroup attacking from t_1 to t_2, and the third subgroup starting from t_2. Here, we set $t_0 = 500$ s, $t_1 = 680$ s, $t_2 = 860$ s, and $H = 1/5$. This results in a situation where the aggregate attack rate does not exceed 3,400 pps (or about one in 5,900 normal packets). We also arrange the three subgroups spatially in the left, the middle, and the right of the network so that the attack changes direction dynamically. Such attack dynamics make it difficult for *traceback* approaches [41], [42] to identify the attack sources and for aggregate congestion-control (ACC) mechanisms [43] to capture the congestion signature.

Fig. 9 shows the spatial-temporal pattern of our simulated subgroup attack, which reveals itself through the signature of congestion at victim I1. Comparing against the constant rate attack (Fig. 4), we find the dynamic nature of the subgroup attack seems advantageous for detection by our analysis method because the increased correlation induced by shifts in attack traffic keeps the weight of the victim I1 salient over a longer time range.

Fig. 10 shows the spatial-temporal pattern of only five backbone-router zones (G to K), where the weight vector S evolves with $T = 2$ s and with the time window MT ($= 200 \times 2$ s $= 400$ s) sliding ahead every 20 s, revealing the congestion arising in the subnet I1. The effects of the subgroup attack remain evident, while the observation of only a portion $[L(N-1) = 72]$ of the network requires less

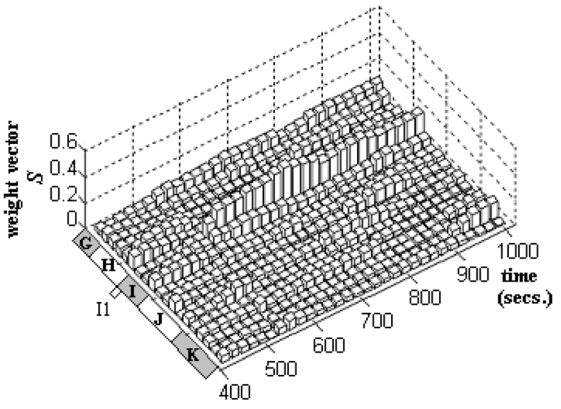

Fig. 10. Spatial-temporal pattern of subgroup attack, observed in five zones of backbone routers from G to K.

computing time and memory than for the case of Fig. 9, where $L(N-1) = 156$. This hints at future investigation into trade-offs between detection ability and space-time granularity in order to understand how much computing time might be required to achieve specific levels of detection.

6 Discussion

The approach we propose aims at monitoring network traffic at a macroscopic level in order to reveal dynamic shifts in congestion patterns, which might signal onset of a DDoS attack. Our method reveals possible attacks without observations near the victim. On the other hand, our technique cannot readily distinguish the cause of observed congestion, which might result from flash crowds or partial network outages, as well as from DDoS attacks. For this reason, our method can only serve as an alert function to trigger more detailed monitoring mechanisms, focused on particular points where congestion appears. Such a two-pronged strategy could improve the efficiency and effectiveness of monitoring functions because detailed monitoring processes consume substantial resources. Incorporating our approach could permit such processes to be activated only where and when needed.

We illustrated that our approach could succeed when observing at only a subset of possible measurement points; thus, reducing associated computation and memory cost. On the other hand, such reduction can be effective only when the network operates with sufficient space-time correlation; thus, lightly loaded networks might prove unsuitable for monitoring with only a few observation points. Elsewhere in [7], we found our method effective when monitoring at only 10 percent of available points. Effectiveness diminished, however, when we monitored at only 5 percent of available points.

Overall, our method provides significant information only if attacks cause apparent effects on network congestion. On the other hand, we showed our method could detect stealthy DDoS attacks more readily than constant-rate attacks, which inject higher volumes of attack traffic. For example, with our analysis method, pulsing and subgroup attacks exhibit increased congestion over a longer period than constant-rate attacks. This occurs because our method benefits from increased correlations arising during periods of changing traffic patterns. Stealthy attacks can extend periods of change. This demonstrates that our approach is not wholly dependent on the intensity of attack traffic. Of course, we can imagine a DDoS attack weak enough not to cause any apparent shift in spatial-temporal correlation.

Regarding efficiency, the technique we propose should require fewer resources than spatial approaches [21], [36] to macroscopic-level monitoring. This derives from the fact that we consider only the eigenvector corresponding to the largest eigenvalue, which we associate with a strong correlation over the whole network. Even so, our method requires computing a $K \times K$ cross-correlation matrix and then finding all K eigenvectors. Recall that $K = L(N-1)$, where L is the number of observation points and N is the number of areas to be observed. This suggests limits on monitoring area and frequency and on the number of observation points.

In previous work [7], we investigated the computational and memory requirements of our technique. Storing our matrix requires at least $b \cdot K^2$ bytes, where b is the number of bytes allocated to each cell. Computing the correlation matrix and all eigenvalues and eigenvectors with MATLAB on a 1 GHz personal computer (PC) required: 1) 0.06 s for $K = 108$, 2) 1.1 s for $K = 320$, and 3) 9.98 s for $K = 640$. Extrapolating from these observations, we estimate computing a $K = 800$ matrix within the 20 s monitoring cycle used in this paper. Storing such a matrix would require about five Mbytes (we used eight bytes per cell). Similarly, we estimate computing a $K = 1,000$ matrix within 35 s at a memory cost of eight Mbytes. Storing a network-wide (e.g., $K = 10,000$ matrix) would require 800 Mbytes of memory and take about seven hours to compute; thus, our approach cannot be adopted directly to monitor the Internet. Despite such scaling limitations, our approach could be applied by cooperating Internet service providers (ISPs) to offer DDoS alert services to paying subscribers.

Assume that $k1$ ISPs form a peer-to-peer (P2P) network, consisting of $k2$ nodes ($k \geq k1$), to share data and offer a DDoS alert service about M monitored targets of interest to a subscriber. Further, assume that $k3$ observation points ($k3 \geq k2 \geq k1$) are distributed within the ISPs. Each of the $k2$ P2P nodes is charged with calculating a weight vector for a subset N ($\leq M$) of the monitored targets. From L observation points ($L \leq k3$), each P2P node would gather flow data (x_{ij}) about the M monitored targets, where each x_{ij} represents traffic from the ith $(1..L)$ observation point to the jth $(1..M)$ monitored target. Each P2P node then collects flow data (but only for its N monitored targets) from the other $(k2-1)$ P2P nodes and aggregates this with the relevant flow data measured within its own domain. Suppose this process of gathering the flow data needs time $t1$. After aggregating flow data from the other nodes, each P2P node uses our approach to calculate the weight vector for just its N monitored targets and, subsequently, forwards the weight vectors to an alert process operating on behalf of

the subscriber. Suppose this process of calculating the weight vectors needs time $t2$. To achieve real-time monitoring, each P2P node, assuming a given number of observation points ($k3$), may not take on too many monitored targets (N), and the granularity of the monitoring time (T) must be larger than $t1 + t2$.

For example, assume $L = 10$ observation points assigned to each of $k2 = 4$ P2P nodes; thus, $k3 = 40$ ($L \times k2$) observations points. Supposing that a subscriber wishes to monitor $M = 100$ targets, so that each P2P node is responsible for monitoring $N = 25$ targets, then each P2P node would need to collect an $L \times M$ [i.e., 10*100] element flow data set. Assuming each element takes 4 bytes, then transmitting this 4,000 bytes would take only $t1 = 320$ us (assuming 100 Mbps transmission speed). Further, assuming use of MATLAB on a 1 GHz PC, computing a weight vector for a

$$K = (N - 1) \times k3 \approx N \times k3 = 25 \times 40 = 1,000$$

matrix would require about $t2 = 35$ s. Under these assumptions, our method could provide a DDoS alert service to a subscriber interested in monitoring $M = 100$ targets within a $T = 90$ s lag time.

From this discussion, the trade-offs become clear between time lag, the number of observation points, and the number of monitored targets. One may increase the number of observation points by lowering the number of targets monitored or by adding more cooperating nodes to keep the number of targets within a suitable range to achieve the desired time lag. The time lag may also be increased as an alternative to increasing the number of cooperating nodes. Of course, the cooperating ISPs may deploy additional sets of cooperating nodes to serve the requirements of more subscribers, each of whom presumably is interested in monitoring a different set of targets. One could imagine cooperating ISPs offering such DDoS alert services to paying subscribers, particularly businesses that might wish to monitor on the order of a few hundred sites deployed around the globe.

7 Future Work

We plan to investigate the ability of our method to detect DDoS attacks under more realistic conditions. First, we may consider alternate patterns of background traffic in two respects: effects of correlation and effects of higher variability. What is the minimum level of correlation in network traffic required to sense shifts in spatial-temporal traffic patterns with only a few observation points? More generally, can we establish trade-offs among level of correlation and needed number of observation points? We suspect that higher variability in background traffic might benefit our analysis method by increasing correlation in network traffic. To explore effects of higher variability, we may select destinations from nonuniform distributions. Second, we may explore relationships between attack characteristics and the detection ability of our method. The results reported here suggest our technique is more sensitive to attacks with changing patterns than to attacks with higher, sustained intensity. This indicates potential for our method to reveal several classes of stealthy attack. We seek a clearer understanding of relationships between attack characteristics and detection sensitivity.

8 Conclusions

Creating defenses for DDoS attacks requires monitoring dynamic network activities in order to obtain timely and significant information. While much current effort focuses on detecting constant-rate attacks, attack patterns appear likely to become more sophisticated. In this paper, we proposed a means for early detection of DDoS flooding attacks by monitoring macroscopic (network-wide) effects. We experimented with different attack modes: constant rate, increasing rate, natural-network-congestion-like, pulsing, TCP-targeted, and subgroup attacks. We found that these attacks, which have the apparent effect of inducing network congestion, reveal themselves through shifts in spatial-temporal patterns that exhibit the same signature: congestion at the victim network. Our simulation results show that macroscopic-level monitoring could capture shifting traffic patterns during transient periods with relatively few observation points. Our analysis method reveals the time and location of an attack without observations from the suffering victim. We also find the dynamic nature of selected attack types (e.g., subgroup and pulsing attacks) may be advantageous for our analysis method because increased correlation induced by changes in attack traffic keeps the weight of the attack victim salient for longer periods. Our results suggest that macroscopic-level monitoring might be both practical and helpful for triggering more focused detection and filtering in transit or source networks. Further work is needed to investigate engineering trade-offs among the space and time granularity of monitoring, the number of observation points, and the ability to detect attacks under diminishing levels of intensity.

References

[1] J. Saltzer, D. Reed, and D. Clark, "End-to-End Arguments in System Design," *ACM Trans. Computer System,* vol. 2, no. 4, pp. 277-288, Nov. 1984.

[2] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial of Service Activity," *Proc. USENIX Security Symp.,* Aug. 2001.

[3] J. Mirkovic, J. Martin, and P. Reiher, "A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms," Technical Report CSD-TR-020018, Computer Science Dept., Univ. of California, Los Angeles, 2001.

[4] J. Mirkovic, G. Prier, and P. Reiher, "Challenges of Source-End DDoS Defense," *Proc. Int'l Symp. Network Computing and Applications,* 2003.

[5] A. Feldmann, A.C. Gilbert, W. Willinger, and T.G. Kurtz, "The Changing Nature of Network Traffic: Scaling Phenomena," *ACM SIGCOMM Computer Comm. Rev.,* vol. 28, no. 2, pp. 5-29, Apr. 1998.

[6] Q. Huang, H. Kobayashi, and B. Liu, "Analysis of a New Form of Distributed Denial of Service Attack," *Proc. 37th Ann. Conf. Information Science and Systems (CISS'03),* Mar. 2003.

[7] J. Yuan and K. Mills, "A Cross-Correlation Based Method for Spatial-Temporal Traffic Analysis," *J. Performance Evaluation,* vol. 61, nos. 2-3, pp. 163-180, 2005.

[8] S. Noh, C. Lee, K. Choi, and G. Jung, "Detecting Distributed Denial of Service (DDoS) Attacks through Inductive Learning," *Lecture Notes in Computer Science,* vol. 2690, pp. 286-295, 2003.

[9] R. Basu, K.R. Cunningham, S.E. Webster, and P.R. Lippmann, "Detecting Low-Profile Probes and Novel Denial of Service Attacks," *Proc. 2001 IEEE Workshop Information Assurance,* 2001.

[10] M. Li and C. Chi, Decision Analysis of Statistically Detecting Distributed DoS Flooding Attacks," *Int'l J. Information Technology and Decision Making,* vol. 2, no. 3, pp. 397-405, 2003.
[11] J. Ioannidis and S.M. Bellovin, "Implementing Pushback: Router Defense against DDoS Attacks," *Proc. Network and Distributed Systems Security Symp.,* Feb. 2002.
[12] Y. Huang and J.M. Pullen, "Countering Denial of Service Attacks Using Congestion Triggered Packet Sampling and Filtering," *Proc. 10th Int'l Conf. Computer Comm. and Networks,* 2001.
[13] Y. Xiong, S. Liu, and P. Sun, "On the Defense of the Distributed Denial of Service Attacks: An On-Off Feedback Control Approach," *IEEE Trans. Systems, Man, and Cybernetics—PART A: Systems and Humans,* vol. 31, no. 4, pp. 282-293, 2001.
[14] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the Source," *Proc. Int'l Conf. Network Protocols 2002,* pp. 312-321, 2002.
[15] T.M. Gil and M. Poleto, "MULTOPS: A Data-Structure for Bandwidth Attack Detection," *Proc. 10th Usenix Security Symp.,* pp. 23-38, Aug. 2001.
[16] S.S. Kim, A.L. N. Reddy, and M. Vannucci, "Detecting Traffic Anomalies through Aggregate Analysis of Packet Header Data," *Proc. Networking 2004,* pp. 1047-1059, May 2004.
[17] D. Pappalardo, "ISPs Take on DDoS Attacks," *Computerworld,* Nov. 2003.
[18] L.-C. Chen, T.A. Longstaff, and K.M. Carley, "Characterization of Defense Mechanisms against Distributed Denial of Service Attacks," *Computers and Security,* vol. 23, no. 8, pp. 665-678, 2004.
[19] R.R. Talpade, G. Kim, S. Khurana, "NOMAD: Traffic-Based Network Monitoring Framework For Anomaly Detection," *Proc. Fourth IEEE Symp. Computers and Comm.,* 1998.
[20] A. Akella, A. Bharambe, M. Reiter, and S. Seshan, "Detecting DDoS Attacks on ISP Networks," *Proc. ACM SIGMOD/PODS Workshop Management and Processing of Data Streams (MPDS) FCRC 2003,* 2003.
[21] A. Lakhina, M. Crovella, and C. Diot, "Characterization of Network-Wide Anomalies in Traffic Flows," *Proc. ACM/SIGCOMM Internet Measurement Conf.,* 2004.
[22] S. Staniford, V. Paxson, and N. Weaver, "How to Own the Internet in Your Spare Time," *Proc. USENIX Security Symp.,* pp. 149-167, Aug. 2002.
[23] R.K. C. Chang, "Defending against Flooding-Based Distributed Denial of Service Attacks: A Tutorial," *IEEE Comm. Magazine,* vol. 40, no. 10, pp. 42-51, 2002.
[24] J.O. Ramsay and B.W. Silverman, *Functional Data Analysis, Springer.* New York, 1997.
[25] J. Yuan and K. Mills, "Macroscopic Dynamics of Large-Scale Data Networks," *Complex Dynamics in Comm. Networks,* pp. 191-212, 2005.
[26] J. Yuan and K. Mills, "Exploring Collective Dynamics in Communication Networks," *J. Research of the Nat'l Inst. of Standards and Technology,* vol. 107, no. 2, pp. 179-191, 2002.
[27] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *Proc. ACM SIGCOMM '94 Conf.,* pp. 257-268, 1994.
[28] M. Crovella and A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," *Proc. 1996 ACM SIGMETRICS Int'l Conf. Measurement and Modeling of Computer Systems,* May 1996.
[29] A. Feldmann, A.C. Gilbert, P. Huang, and W. Willinger, "Dynamics of IP Traffic: A Study of the Role of Variability and the Impact of Control," *Proc. ACM SIGCOMM '99 Conf.,* pp. 301-313, 1999.
[30] W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson, "Self-Similarity through High-Variability: Statistical Analysis of Ethernet Lan Traffic at the Source Level," *Proc. ACM SIGCOMM '95 Conf.,* pp. 100-113, 1995.
[31] T. Karagiannis, M. Molle, and M. Faloutsos, "Long-Range Dependence: Ten Years of Internet Traffic Modeling," *IEEE Internet Computing,* pp. 57-64, Sept.-Oct. 2004.
[32] M. Barthelemy, B. Gondran, and E. Guichard, "Large-Scale Cross-Correlations in Internet Traffic," *Physical Rev. E 66,* 2002.
[33] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide.* Philadelphia, Penn.: Soc. for Industrial and Applied Math., 2000.
[34] *MATLAB User's Guide,* The MathWorks, Inc., Natick, Mass., 1998.
[35] K.I. Goh, B. Kahng, and D. Kim, "Spectra and Eigenvectors of Scale-Free Networks," *Physical Rev. E 64,* 2001.
[36] M. Crovella and E. Kolaczyk, "Graph Wavelets for Spatial Traffic Analysis," *Proc. IEEE Infocom 2003,* Apr. 2003.
[37] M. Claypool, R. Kinicki, M. Li, J. Nichols, and H. Wu, "Inferring Queue Sizes in Access Networks by Active Measurement," *Proc. PAM (Passive and Active Measurement) Workshop,* Apr. 2004.
[38] M. Weigle, K. Jeffay, and F.D. Smith, "Quantifying the Effects of Recent Protocol Improvements to Standards-Track TCP," *Proc. 11th IEEE/ACM Int'l Symp. Modeling, Analysis, and Simulation of Computer and Telecomm.,* 2003.
[39] P. Barford and D. Plonka, "Characteristics of Network Traffic Flow Anomalies," *Proc. ACM SIGCOMM Internet Measurement Workshop,* Nov. 2001.
[40] A. Kuzmanovic and E.W. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks," *Proc. ACM SIGCOMM 2003 Conf.,* Aug. 2003.
[41] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical Network Support for IP Traceback," *Proc. ACM SIGCOMM'2000 Conf.,* 2000.
[42] A.C. Shoeten, C. Partridge, L.A. Sanchez, C.e.E. Jones, F. Tchakountio, S.T. Kent, and W.T. Strayer, "Hash-Based IP Traceback," *Proc. 2001 ACM SIGCOMM Conf.,* Aug. 2001.
[43] R. Mahajan, S.M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High Bandwidth Aggregates in the Network," technical report, AT&T Center for Internet Research at ICSI (ACIRI) and AT&T Labs Research, Feb. 2001.

Jian Yuan received the PhD degree in electrical engineering from the University of Electronic Science and Technology of China, in 1998, and both the BS and MS degrees in electrical engineering from the South-East University, China, in 1986 and 1989, respectively. From 2000 to 2004, he was a guest researcher at the Information Technology Laboratory of the National Institute of Standards and Technology. He is currently an associate professor in the Department of Electronic Engineering at Tsinghua University, Beijing, China. His research interests include large-scale network analysis, anomaly detection, and complex networks.

Kevin Mills received the PhD degree (1996) in information technology from the George Mason University in Fairfax, Virginia. He is a senior research scientist with the Information Technology Laboratory of the National Institute of Standards and Technology (USA), where he has served in various research and management positions since 1982, except for a term (1996 through 1999) as program manager with the US Defense Advanced Research Projects Agency (DARPA). He is also on the adjunct faculty (1996 through the present) of the School for Information and Technology Engineering at George Mason University. His research interests include complex distributed systems. He is a senior member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.