Identification of Repeated Denial of Service Attacks

Alefiya Hussain*†, John Heidemann* and Christos Papadopoulos*
* USC/Information Sciences Institute, 4676 Admirality Way, Marina del Rey, CA 90292, USA
Email: {johnh,christos}@isi.edu
† Sparta Inc, 2401 E. El Segundo Blvd #100, El Segundo, CA 90245, USA
Email: alefiya.hussain@sparta.com

***Abstract*— Denial of Service attacks have become a weapon for extortion and vandalism causing damages in the millions of dollars to commercial and government sites. Legal prosecution is a powerful deterrent, but requires attribution of attacks, currently a difficult task. In this paper we propose a method to *automatically fingerprint* and *identify* repeated attack scenarios—a combination of attacking hosts and attack tool. Such fingerprints not only aid in attribution for criminal and civil prosecution of attackers, but also help justify and focus response measures. Since packet contents can be easily manipulated, we base our fingerprints on the *spectral characteristics* of the attack stream which are hard to forge. We validate our methodology by applying it to real attacks captured at a regional ISP and comparing the outcome with header-based classification. Finally, we conduct controlled experiments to identify and isolate factors that affect the attack fingerprint.**

I. Introduction

Distributed denial of service (DDoS) attacks are a common phenomenon on the Internet [20]. A human attacker typically stages a DDoS attack using several compromised machines called *zombies*. The actual number of zombies on the Internet at any given time is not known, but it is estimated to be in the thousands [20]. To keep management simple, groups of zombies are typically organized in *attack troops*, that the attacker can then repeatedly use to flood a target. We define the combination of an attack troop and the attack tool as an *attack scenario*. An attack is identified as repeated when the same attack scenario is used to launch multiple DoS attacks over a period of time on a victim. Moore et al. have identified 35% of all Internet attacks are repeated attacks directed at the same victim using backscatter analysis [20]. The results indicate that repeated attacks are a very common and a serious security problem on the Internet.

Current approaches addressing DDoS are focused on attack prevention, detection, and response. Prevention of DDoS attacks encompasses techniques that preserve integrity of the hosts [31] and techniques to detect and rate-limit abnormal network activity [19]. Attack detection is typically based on techniques such as signature matching [23], [24] or anomaly detection [21]. Response to DDoS attacks typically involves filtering of attack packets, assuming a signature has been defined, and traceback techniques [26], [28], which attempt to identify the attack paths.

While approaches to detect and respond to DDoS are improving, responses such as traceback require new wide-area collaboration or deployment. We explore *attack attribution*, a complementary approach that allows identification and quantification of repeated attacks on a victim from the same attack troop.

Attribution of attacks is important for several reasons. The primary motivation is that attribution can assist in legal prosecution of attackers. Recently there have been numerous reports of extortion targeted against commercial web sites such as online banking and gambling [30], [29]. However, prosecuting attackers is still very challenging. Our system will provide the ability to identify repeated attacks which will help establish criminal intent and help meet monetary thresholds for prosecution. Additionally, attribution can also be useful in civil suits against attackers. Other motivations for deploying such a system include automating responses to repeated attacks to cut down reaction time, and it can also be used to quantify repeated and unique attacks to justify investment in defensive tools. Further, attack correlation over the global Internet can help track global attack trends. Such information is useful in designing the next generation of defense mechanisms and tools. We explore these motivations in more detail in Section III-A. Finally, our approach to attribution does not require global deployment, only deployment near the victim.

Packet header contents of an attack packet can be easily spoofed and provide very limited information about the attack scenario. Thus as ballistics studies of firearms can trace multiple uses of a weapon to the same gun, in this paper we develop a system for network traffic forensics to uncover structure in the attack stream that can be used to detect repeated attacks. Figure 1 illustrates the scenario we consider. Attackers have compromised two troops of machines in the Internet, labeled A and B; they use these machines to attack victims (labeled V) inside an edge network. A host at the edge network (labeled M) monitors a series of attacks, recording packet traces $t_1, t_2 \ldots t_i$. Our system then converts each attack t_i into a compact *fingerprint*, $f(t_i)$. We show that a fingerprint uniquely identifies an *attack scenario*. Thus if t_1 and t_3 are from troop A with the same tool while t_2 is from troop B, then $f(t_1) \sim f(t_3)$ while $f(t_1) \not\sim f(t_2)$, and some new attack t_i can be identified as similar to either t_1, t_2, or representing a new attack scenario.

This description raises several issues that must be explored. First, we must identify traffic features that indicate an attack scenario. Previous work on DoS attack classification has established the use spectral analysis to detect the presence of multiple attackers by extracting periodic behavior in the attack

1-4244-0222-0/06/$20.00 (c)2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the Proceedings IEEE Infocom.

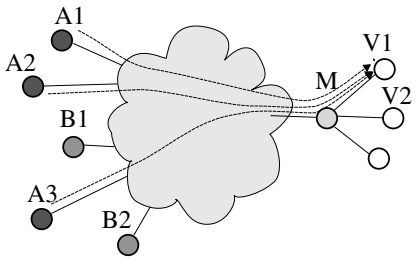


Fig. 1. Monitoring attacks in the Internet.

stream [12]. In this paper, we suggest that individual attack scenarios often generate unique *spectral fingerprints* that can be used to detect repeated attacks.

Second, to support this claim we must understand what systems factors affect fingerprints. We evaluate what aspects of the network cause regularity in the packet stream and affect the fingerprints in Section V. There we present a battery of experiments varying the attack tool, operating system, host CPU, network access speed, host load, and network cross-traffic. Our results indicate that even though there is sensitivity with respect to host load and cross traffic, fingerprints are consistent for a particular combination of attack troop and attack tool.

Third, DoS attacks are adversarial, so in Section VI we review active countermeasures an adversary can use to manipulate the fingerprint. There is a tension inherent in the desire to identify scenarios as distinct from each other, yet be tolerant to measurement noise. Our current method favors sensitivity, so while we show that modest changes to the attack scenario allow repeated attack detection, countermeasures such as including significant changes in number of attackers or attack tool result in different fingerprints. Clearly future work will be required to explore alternative trade offs, however our current approach does significantly raise the requirements in attack tool sophistication and attack group size.

We validate our fingerprinting system on 18 attacks collected at Los Nettos, a regional ISP in Los Angeles. We support our methodology by considering two approaches: (a) by comparing different attack sections of the same attack with each other to emulate an ideal repeated attack scenario, and (b) by comparing different attacks to each other. The results indicate that different sections of the same attack always provide a good match, supporting our attack scenario fingerprinting techniques. Further, comparing the different attacks indicated that seven attacks were probably from repeated attack scenarios. We describe these approaches in more detail in Section IV. We further investigate our methodology in Section V and Section VI by conducting controlled experiments on a testbed with real attack tools. The testbed experiments enabled testing the attack scenario fingerprints with changes in both environmental and adversarial conditions.

The contribution of this paper is to introduce attack attribution as a new component in addressing DoS attacks. We demonstrate a preliminary implementation that that can identify repeated attack scenarios and validate them through trace data, testbed experiments, and exploration of countermeasures. To our knowledge, there have been no previous attempts to identify or analyze attack scenarios for forensic purposes.

II. Related Work

Pattern recognition has been applied extensively in character, speech, image, and sensing applications [14]. Although, it has been well developed for applications in various problem domains, we have not seen wide-scale application of this technology in network research. Broido et al. suggest applying network spectroscopy for source recognition by creating a database of inter-arrival quanta and inter-packet delay distributions [2] and Katabi and Blake apply pattern clustering to detect shared bottlenecks [16]. Additionally there is a large body of work that analyzes timing information in network traffic to detect usage patterns [9]. Further, network tomography techniques such as those described by Duffield [7] correlate data from multiple edge measurements to draw interference about the core. In this paper, we make use of pattern classification techniques to identify repeated attack using spectral fingerprints and suggest that similar techniques can be applied in other areas of network research.

Signal processing techniques have been applied previously to analyze network traffic including to detect malicious behavior. Feldmann el al. were one of the first to do a systematic study on fingerprinting network path characteristics to detect and identify problems [8]. Cheng et al. apply spectral analysis to detect high volume DoS attack due to change in periodicities in the aggregate traffic [3] whereas Barford et al. make use of wavelet-based techniques on flow-level information to identify frequency characteristics of DoS attacks and other anomalous network traffic [1]. Hussain et al. make use of spectral density of the attack stream to characterize single and multi-source attacks [12]. In a broader context, researchers have used spectral analysis to extract information about protocol behavior in encrypted wireless traffic [22]. In this paper, we transform the attack stream into a spectral fingerprint to detect repeated attacks.

Intrusion detection refers to the ability of signaling the occurrence of an ongoing attack and is a very important aspect of network security. DoS attacks attempt to exhaust or disable access to resources at the victim. These resources are either network bandwidth, computing power, or operating system data structures. Attack detection identifies an ongoing attack using either anomaly-detection [21] or signature-scan techniques [23], [24]. While both types of IDS can provide hints regarding if a particular attack was seen before, they do not have techniques to identify if it originated from the same set of attackers.

III. Attack Scenario Fingerprinting

In this section we explore the applications and develop the algorithm used to identify similar attack scenarios. First, we discuss details about where and how a fingerprinting system should be deployed. We then provide an intuitive explanation

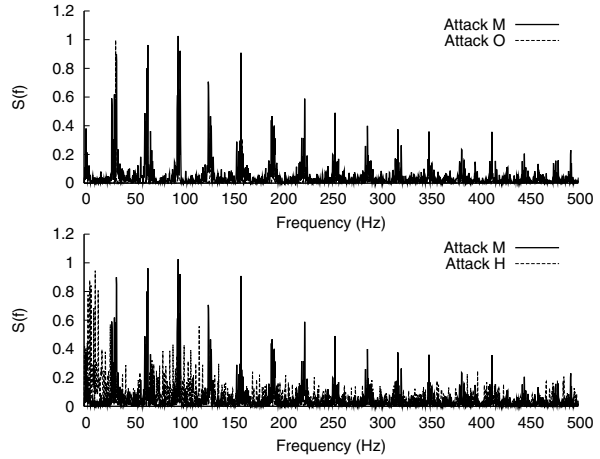


Fig. 2. Top plot: Frequency Spectra of two similar attacks overlap, Bottom plot: Frequency Spectra of two dissimilar attacks are distinct

of how the detection algorithm for repeated attacks works. Finally, we detail the algorithm with the help of an example.

A. Applications of Fingerprinting

Attack fingerprinting is motivated by the prominent threat of denial-of-service attacks today. Utilities and critical infrastructure, government, and military computers increasingly depend on the Internet for operation; information warfare is a reality for these applications. In the commercial world online businesses are under daily threats of extortion [29], [32], [30], attacks by competitors [17], and simple vandalism [11]. Similarly, many websites, including the RIAA, SCO, and political sites, are vulnerable to ideologically motivated attacks. Even universities and ISPs are subject to smaller-scale DoS attacks provoked by individuals following arguments on IRC channels. These small attacks can cause collateral damage on the network. Each of these cases motivate the need for better approaches to detecting, understanding, and responding to DDoS attacks. Finally, although not directly attacked, ISPs may wish to provide attack countermeasure services as a value-added service.

Our attack fingerprinting system helps identify repeated attack scenarios—attacks by the same group of machines and attack tool. This identification provides assistance in combating attacks. First, attack identification is important in criminal and civil prosecution of the attackers. The FBI requires demonstration of at least $5000 in damage before investigating a cyber-crime [5]. Quantifying the number of attacks is necessary to establish damages in a civil trial. Although our algorithms do not directly identify the individual behind the attack (which is a particularly hard problem), they can help associate a pattern of attacks with an individual identified by other means, allowing legal measures to complement technical defenses.

Further, the detection of repeated attacks can be used to automate technical responses. Responses developed for an attack can be invoked again automatically on detection of the same attack, cutting down on reaction time. In addition, estimating the number of attackers can help justify added investment in better defensive tools and personnel.

Finally, an attack fingerprinting system can evaluate the number of attack troops, number of unique and repeated attack scenarios and quantify the amount of malicious behavior on the Internet, providing more accurate "crime reports".

B. Our Approach in a Nutshell

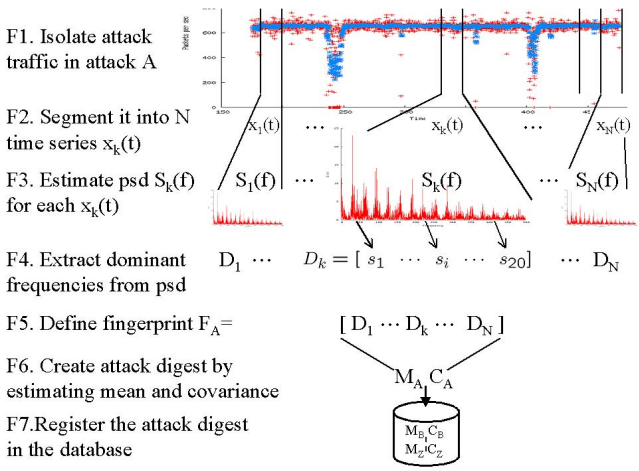
Every attack packet stream is inherently defined by the environment in which it is created, (that is, attack tool and host machine characteristics) and is influenced by cross traffic as it traverses through the network. These factors create regularities in the attack stream that can be extracted to create unique fingerprints to identify repeated attacks. In this section, we briefly outline the algorithms we use to detect and identify these regularities.

Given an attack, we wish to test if this scenario occurred previously. Figure 2 illustrates the intuition behind this concept. The figure shows three attacks in two groups, with attacks M and O on the top graph and attacks M and H on the bottom. We claim that the spectra of M and O are qualitatively similar, as shown by matching peaks at multiples of 30Hz in both spectra. By contrast M and H are different, since H shows distinct frequencies, particularly in 0–30Hz and 60-140Hz. We compare all 18 captured attacks in Section IV and Figure 4.

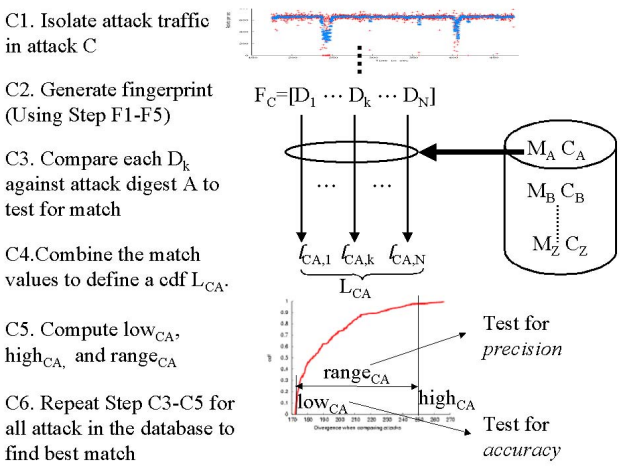
While Figure 2 provides intuition that spectra can identify repeated attacks, graphical comparisons of spectra are difficult to automate and quantify. We therefore define a procedure to reduce spectra to generate and then compare *fingerprints* that abstract the key features of the spectrum.

Figure 3(a) illustrates our process and Section III-C describes it in detail. Briefly, we first isolate the attack packet stream of attack A (step F1). Since attack traffic varies over time, we divide it into N segments (step F2) and compute the spectrum for each segment (step F3). We then extract dominant features in the spectra by by identifying twenty dominant frequencies of each segment (step F4) and merge these to form the fingerprint, an $20 \times N$ matrix, F_A (step F5). To facilitate matching, we create an *attack digest* from the mean (M_A) and covariance (C_A) of F_A (step F6). The digest values of each attack form the database of known attacks (step F7).

Given a new candidate attack C, Figure 3(b) summarizes the procedure for matching it against the database, Section III-D provides a more detailed explanation. We begin by isolating the attack and generating the fingerprint F_C using steps F1–F5 described above. We compare F_C against the mean and covariance of each attack in the database by breaking it into its component D_k vectors and comparing each segment D_k against a given attack, generating a match value (step C3). We then combine the match values for all segments to create an empirical distribution (step C4) and extract the low value as the 5% quantile and the range as difference between the 95% and 5% quantlies to estimate accuracy and precision of the match (step C5). Comparing these values for different matches can suggest which is best; comparing them against a fixed



(a) Extracting the attack fingerprint.



(b) Comparing candidate attack C with registered attacks in the database.

Fig. 3. Algorithm used to register and compare attack scenarios.

threshold can evaluate if that match is considered correct or not.

For our algorithm to be effective, it must be robust to noise and resistant to spoofing. Noise is introduced by changes in environmental conditions, such as change in host load or network cross traffic. We examine the underlying network influences on the spectrum and the impact of noise in Section V. We consider adversarial countermeasures, such as change in number of attackers or attack tool, in Section VI.

C. Creating the Attack Fingerprint

In order to generate the attack fingerprint, we first need to isolate the attack stream from the network traffic. This is done by filtering based on the attack signature, if identifiable. If a signature is not available, or is hard to determine, we filter based on the target's address. Since we consider only flooding attacks in our analysis, we assume that most other traffic is squeezed out (otherwise the attack is not very successful).

Next we extract feature data from the attack stream by converting the attack stream into a time series. We assume a given sampling bin of p seconds and define the arrival process $x(t)$ as the number of packets that arrive in the bin $[t, t+p)$. Thus, a T second long packet trace will have $M = \frac{T}{p}$ samples. The bin size p limits the maximum frequency that can be correctly represented to $\frac{1}{2p}$ Hz. We use a sampling bin of 1ms for the attack fingerprint.

Given attack A, we divide the attack stream into k, where $k = 1 \ldots N_A$, segments. For each segment we compute the power spectral density $S_k(f)$ where f varies between 0–500Hz using techniques discussed in [12].

The power spectrum $S_k(f)$ of the attack is obtained by the discrete-time Fourier transform of the ACF to obtain the frequency spectra for each attack segment, as shown in Figure 2. Formally:

$$S_k(f) = \sum_{\ell=0}^{2M-1} r(\ell) e^{-\imath \ell 2\pi f} \tag{1}$$

Next we define a technique to quantitatively compare each attack segment. We define a segment fingerprint D_k, a vector consisting of the twenty dominant frequencies in $S_k(f)$ to be the frequency representation for each segment k (where $k = 1 \ldots N_A$). Dominant frequencies are extracted by identifying frequencies that contain most power in $S_k(f)$. Ideally, when comparing two attacks, an exact match for the attack would consist of the complete frequency spectrum. However, handling the complete spectrum makes computation of the comparison more costly as well as requires significantly more attack segments. Therefore, formulating the signature as the dominant twenty frequencies helps reduce the number of samples to make robust comparisons, with minimal loss of information. To arrive at the optimal feature set we did a preliminary exploration by varying the number of frequencies used as features, and found that we get accurate match values as the size of feature set increases and the poor matches for smaller feature sets. We tested our algorithm with feature sets of 5 and 30 frequencies on the attacks and testbed experiments and obtained varying match results. The top 5 feature produced significantly lower quality matches while the top 30 frequencies did not improve the quality of our matches.

Thus, the dominant twenty frequencies provide a good estimate of the important periodic events that constitute the attack stream. In Section VII, we discuss additional factors we would like to explore to generate robust features.

Next for each attack A, we define F_A as the attack fingerprint consisting of all the segment fingerprints $D_k (k = 1 \ldots N_A)$. We can think of F_A as representing a sample

of the dominant frequencies of A. For easy comparison of candidate attacks against the database, we compute attack digests summarizing F_A. We do this by computing the mean and covariance of F_A defined as:

$$M_A = 1/N_A \sum_{k=1}^{N_A} D_k \quad (2)$$

$$C_A = 1/N_A \sum_{k=1}^{N_A} (D_k - M_A)(D_k - M_A)^T \quad (3)$$

A minimum ratio of 10 for the number of attack segments N_A to the size of the feature segment D_k is required to ensure robust estimates for the mean and covariance of F_A [6]. Since the feature segment consists of twenty dominant frequencies, we consider attacks that consist of at least 200 segments, ($N_A = 200$) each of two second duration, making the minimum attack duration 400 seconds. As a result, the attack fingerprint F_A is defined as a 20x200 matrix. The attack digest M_A is defined as a 20-element mean vector of the dominant frequencies and C_A is defined as a 20x20 element matrix of the covariances of the frequencies. Intuitively, these summarize the most common frequencies by representing them as distribution parameters of the attack sample.

We found the attack spectrum to be a good indicator of a unique attack scenario. In fact identifying repeated attacks was motivated by observing identical spectral behavior in different attacks when we were working on our previous paper [12]. We discuss alternate feature definitions in Section VII.

D. Comparing Two Attacks

Once we have a database of registered attack fingerprints, we can test if a new attack scenario, C, has been previously observed by applying the Bayes maximum-likelihood classifier [6]. The ML-classifier makes the following assumptions:

1) For a given attack scenario, the spectral profiles have a normal distribution with respect to each dominant frequency.
2) Every attack scenario is equally likely.
3) Every attack occurs independent of previous attacks.

To validate these assumptions, we verify that the every attack segment fingerprint F_A has an approximately normal distribution for each dominant frequency represented in each segment X_k, where $k = 1 \ldots N_A$. In each case, the χ^2 test at 90% significance level indicated all the dominant frequencies have normal distribution. The second and third assumption, regarding the attack likelihood and independence are more difficult to validate. Clearly attack occurrences are not completely independent since attack techniques and attackers change with time; for example, Smurf attacks are not as popular today as they were couple of years ago. But to quantify the comparisons we must make these assumptions. As future work, we will attempt to understand the impact of these assumptions and their impact on the fingerprint as discussed in Section VII.

We use the Bayes maximum-likelihood classifier to test if the current attack scenario C is similar to a registered attack fingerprint A. First, we need to create an attack fingerprint for attack C. We therefore segment the attack trace into N_C time series segments, $x_l(t)$, each of duration 2 seconds. We then compute the spectrum $S_l(f)$ for each attack segment, $l = 1 \ldots N_C$ and identify the dominant twenty frequencies to form the attack feature segment X_l collectively defined as the attack fingerprint F_C. The value of N_C depends solely on attack length and can be smaller than 200 seconds used for N_A. Because we are not estimating distribution parameters for making an attack comparison, there are no requirements on the minimum number of attack segments N_C.

Once the attack segment fingerprints are generated, we can compare the fingerprint F_C against the database of registered attack digests. We make comparisons using the maximum likelihood of each segment in F_C against all previously registered attacks A using:

$$l_{CA,l} = (X_l - M_A)^T C_A^{-1} (X_l - M_A) - log|C_A| \quad (4)$$

where X_l represents each attack feature segment in F_C, $l = 1 \ldots N_C$. Intuitively, Equation 4 quantifies the separation between the registered attack scenario A and the current scenario C and is also called the *divergence* of the attack scenario distributions. This procedure generates a set of N_C matches, L_{CA}, for each segment X_l of F_C against each attack digest. A match set is thus generated for all the attacks in the database.

E. Interpreting the Match Data

Once the match set L_{CA} for comparing current attack C with each attack digest in the database is generated, we must summarize this match data. For any comparison, some segments will match better than other segments. In this paper, we try to find good general comparisons by specifically answering the following two questions:

1) Are the comparisons accurate? i.e.: Does attack C match well with the attack digest A?
2) Are the comparisons precise? i.e.: Does attack C consistently have a small divergence with attack digest A?

To *test for accuracy* (TA) we compute low_{CA}, as the 5% quantile of L_{CA}. A small value for low_{CA} indicates at least 5% attack segments from attack C have a very accurate match with attack A. To *test for precision* (TP) we compute $high_{CA}$, as the 95% quantile of L_{CA} and the define the $range_{CA}$ as the difference between $high_{CA}$ and low_{CA}. A precise match will have a small range indicating a large percentage of the attack segments match with the attack digest.

To automate the matching procedure, we now need to identify what values of TA and TP indicate a good match and how they are related. We define the matching condition used for comparison of the attacks as Attack C matches attack A if and only if

$$range_{CA} < threshold_{range} \quad \text{AND}$$

$$low_{CA} < low_{CB} \ \ \forall B \neq A \quad \text{(Condition 1)}$$

We empirically derive the values of the $range$ threshold in Section IV-B by comparing separate sections of real-world

attack to itself. In addition to identifying the closest match for attack C in the database of attacks, we need to define a test for when attack C is a new attack we have not seen previously. We believe that the comparison of a new attack not present in the database will have a matching condition of the form

$$low_{CA} > threshold_{low} \quad \text{(Condition 2)}$$

. The identification of such a threshold if more difficult since we would need to observe completely new attacks in the wild. We believe such a threshold will emerge as the database increases in size.

IV. Evaluation of Basic Algorithm

We now evaluate our comparison technique on attacks captured at Los Nettos, a moderate size ISP located in Los Angeles [18]. Our approach was motivated by observing similar spectra during attack classification [12]. We observed that the spectral content of several attacks, even though they occurred at different times, was remarkably similar. We first describe the packet characteristics of the captured attacks. Since the trace data is from a live network, we cannot prove that independent attacks are from the same hosts. Instead, in Section IV-B we compare different sections of the same attack to show our approach can identify the repeated attack scenarios and use the results to define thresholds for a good match. We then present examples of different attacks that we hypothesize may be from the similar scenarios in Section IV-C.

A. Attack Description

Los Nettos is a moderate size ISP with diverse clientele including academic and commercial customers. We detected 18 long attacks during the months of July–November 2003. Although we detected many short duration attacks (more than 80) during the period too, we limited our analysis to attacks of at least 400s to generate fingerprint digests that capture steady-state behavior. This threshold is probably overly pessimistic; evaluating the appropriate attack duration needed for fingerprint generation is an area of future work.

Table I summarizes the the packet header content for each captured attacks at Los Nettos. The second column gives the packet type, the third column gives the TTL values and the last column summarizes the prefix-preserving, anonymized, source IP addresses seen in the attack packets. The TCP *no flags* refers to pure TCP data packets with no flags set, and the *mixed* refers to attacks that use a combination of protocols and packet types such as TCP, UDP, ICMP and IP proto-0. Few attacks subnet spoof the source addresses (for example: attack B), few attacks randomly spoof the source address (for example: attack A), whereas few attacks use constant IP addresses (for example: attack F). For the six echo reply reflector attacks the last column indicates the observed number of reflector IP addresses (along with the subnet address when possible). We believe the attacks that have very similar packet header content indicate the possibility that they are manifestations of the same attack scenarios.

TABLE I
Packet header content observed in the attacks captured at Los Nettos

Id	Packet Type	TTL	Source IP
A	TCP ACK+UDP	14, 48	random
B	TCP ACK	14, 18	6.13.8.0/24
C	TCP no flags	248	random
D	TCP SYN	61	12.9.192.0/24
E	Echo Reply		78 reflectors from 4.15.0.0/16
F	IP-255	123	31.5.19.166, 31.5.15.186, 31.5.23.11
G	IP-255	123	31.5.19.166, 31.5.15.186, 31.5.23.11, 31.5.15.8
H	Echo Reply		1262 reflectors
I	Mixed	27, 252	28.25.14.0/24
J	Mixed	27, 252	28.25.14.0/24
K	UDP	53	6.22.12.20
L	TCP SYN	4,7	random
M	Echo Reply		72 reflectors from 18.9.200.0/24
N	Echo Reply		72 reflectors from 18.9.200.0/24
O	Echo Reply		71 reflectors from 18.9.200.0/24
P	Echo Reply		73 reflectors from 18.9.200.0/24
Q	TCP no flags	248	random
R	IP-255	123	31.5.10.96, 31.5.89.96, 31.5.87.24, 31.5.87.13

B. Emulating the Same Attack Scenario

The attacks listed in Table I are "from the wild", therefore we have can only deduce limited information of the attack scenario. Comparisons among these attacks can only suggest, but not prove, reuse of the same attack hosts and tools. Hence, to establish the viability of our methodology in detecting similar attack scenarios, we emulate a repeated attack scenario by comparing different attack sections of a registered attack. We chose this approach on the assumption that an attack should best match itself and not match all other attacks, thus this comparison allows a controlled study of our technique. Additionally, this approach also helps establish what threshold values of TA and TP indicate a good match for the matching conditions described in Section III-E.

We divide each attack (A–R from Table I) in two parts, a head and a tail section. The head section is composed of the first 400s of the attack, that is used to define the attack fingerprint by applying the technique described in Section III-C. The tail section is made up of at least 20s of the remaining attack to ensure reasonable number of segments to allow statistical comparison against the fingerprint database.

For each attack, we compare the tail of the attack against all registered fingerprints (computed from the heads of each attack) using the technique outlined in Section III-D and Section III-E. Figure 4 represents the accuracy and precision of each attack compared against a database consisting of all attacks. For each attack, we consider it a trial attack (represented as a row) and compare it against the fingerprint of each other attack (each column). For each combination the graph shows the accuracy (TA_{AB}) and precision (TP_{AB}) of the result. Accuracy is presented by the a line, the length of which is linearly proportional to the inaccuracy of the match,

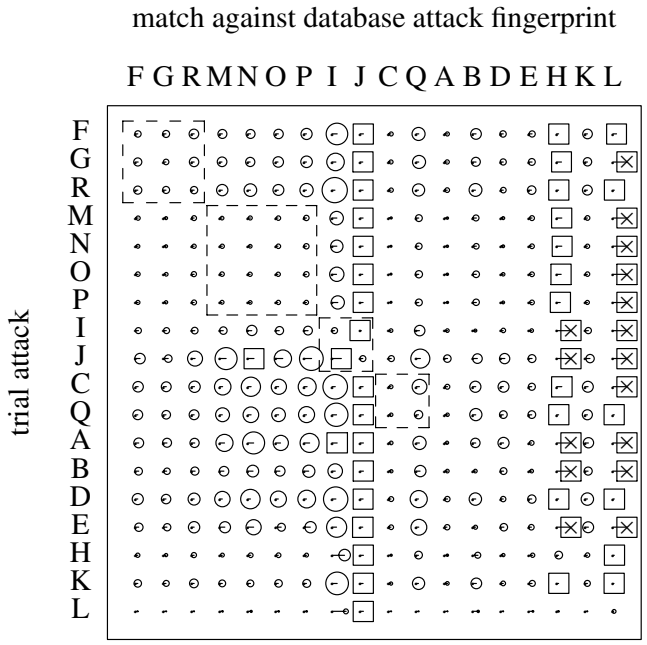


Fig. 4. Graphical representation of TA_{XY} and TP_{XY} statistics for 18 attacks captured at Los Nettos (values greater than 1000 is indicated as square and X). Each row represents a trial attack, while each column represents a database fingerprint; the intersection is a comparison of the attack against a particular database entry.

so short lines represent better accuracy. Accuracies greater than 1000 are considered "too inaccurate" and are instead plotted as an X. Precision is represented with a circle whose area is linearly proportional to the precision of the match, thus large circles represent imprecise results. Ranges greater than 1000 are considered "too imprecise" and and plotted as a large square. A numeric representation of this data can be found in our technical report [13].

We can observe several things from this table. First, the diagonal from A–A to R–R represents the comparison of attacks against their own fingerprints. We see that attacks almost always have the most accurate match against themselves, as we would expect. For example, we get $TA_{AA} = 201$ and $TP_{AA} = 15$ when comparing trial segments of attack A with the attack digest A. Surprisingly, this is not always the case, as in attack M and attack P where the $TA_{MP} = 171$ is more accurate then $TA_{MM} = 174$. We discuss this exception in more detail later in the Section. Additionally, we observe that in some cases as in attack H, $TP_{HH} = 80$ is fairly large. A high TP when an attack is compared against itself indicates that the attack has a large amount of internal variation. We consistently observe comparing the head and tail sections of the same attack provide the closest matches for nearly all attacks validating our comparison techniques.

We can also use self-comparisons to evaluate what values of TP are reasonable. Since self-comparisons indicate internal variations of up to 100 are common, we select this as a threshold for TP in match Condition 1 (Section III-E) to indicate a good match.

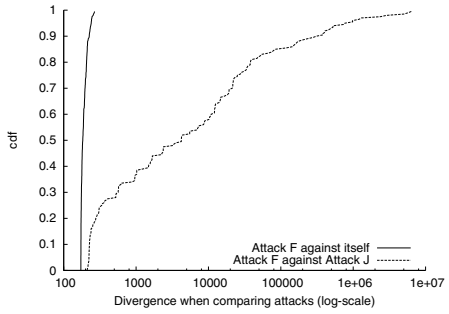


Fig. 5. The cumulative distribution of the maximum-likelihood values when comparing the same attack scenario and when comparing different attacks.

Second, Figure 4 compares 18 by 18 possible matches between attack scenarios. As an example of two of those matches, we take comparing the tail of attack F to the registered fingerprint of attack F ($TA_{FF} = 172$ and $TP_{FF} = 57$ represented by a circle and a short line) and the registered fingerprint of attack J ($TA_{FJ} = 223$ and $TP_{FJ} = 768333$ represented by a square and a line), and visually analyze the difference in the cumulative plots of the values. We plot the cumulative distribution of the set of matches L_{FF} in Figure 5 (shown by the solid line). Observe the small TP_{FF} indicated by a nearly vertical line in the graph. In contrast, the cumulative distribution of set of matches L_{FJ} is spread across a large TP of values (show by the dashed line). The difference in the cumulative plot arises since the ML-classifier consistently returns a small divergence value for the similar attacks and large divergence values when comparing dissimilar attacks.

Additionally, we observe that some trials match poorly against all attacks. Attacks H, J, and L are in this category. Although we might expect the graph to be symmetric, it is not (for example, compare L_{CA} and L_{AC}). Asymmetric occurs because matching considers the entire attack while fingerprint generation considers only a 200s period.

We also evaluate false negatives, that is attacks where self-matches are poorer than matches against other attacks. The $TA_{MM} = 174$, $TA_{NN} = 175$, and $TA_{OO} = 170$ diagonal elements are slightly less accurate than the non-diagonal elements $TA_{MP} = 171$, $TA_{NP} = 174$, and $TA_{OP} = 168$ indicating more accurate matches with attack P. While one might consider this a false negative, an alternative explanation is that these attacks are very similar and hence generate small differences in accuracy values. False positive conditions in the algorithm occur when an attack is identified as repeated when infact it is a new type of attack. In Section V we do a battery of experiments to evaluate when such conditions may occur, and in Section VII we describe how a larger attack database would aid in evaluating the false positives.

We have demonstrated that our approach can detect repeated attack scenarios by considering the ideal case of matching attacks with themselves. This "success" may not be surprising since we knew that each candidate attack had a match; however lack of mismatches in this case is promising. The

above process also provided thresholds for TP values that can be used to to indicate good matches for different attacks. We next compare different attacks to see if it is plausible that any two observed attacks represent the same scenario.

C. Testing with Different Attacks

We now attempt to identify similar attack scenarios by comparing different attacks against the fingerprint registered in the attack database. The comparison matrix presented in Figure 4 provides the TA_{XY} and TP_{XY} statistics for all the attacks compared with each other in the non-diagonal elements. To test for similarity, we use the match Condition 1 (Section III-E), with the TP threshold of 100, established in the previous section. The packet contents in Table I, provide insight into plausible repeated attack scenarios. We expect the TA and TP values to be small for similar attacks. We observe four sets of very similar scenarios. We have ordered the rows and columns to place these adjacent to each other, and we surround their comparisons with a dashed box.

The first set consists of three attacks F, G, and R. All three attacks have the protocol field in the IP header set to 255, and a TTL value of 123, and the source IP addresses originate from the same subnet but vary in number. Attacks F and G occur approximately 31 hours apart, whereas attack R occurs 75 days later. Comparing the statistics we observe that the values of TA_{FG}, TA_{GF} are the smallest in the non-diagonal elements with TP_{FG}, TP_{GF} less than 100. Further, small TA_{RF}, and TA_{RG} with small TP_{RF}, and TP_{RG} statistics indicate attack R is similar to attacks F and G. We did not obtain sufficiently small TA_{FR} and TA_{GR} statistics. These the statistical values indicate a strong similarity between the attack scenarios.

The next set consists of attacks M, N, O, and P. All four attacks originate from reflectors belonging to the same subnet. These attacks occur within 6 hours of each other. The attacks have very small TA and TP statistics in the non-diagonal elements providing a good four-way match with each other. Due to the close match, the TA_{MM}, TA_{NN} and TA_{OO} diagonal elements are approximately three points higher than the non-diagonal elements TA_{MP}, TA_{MO} and TA_{OP} respectively. These attacks therefore are an exception of the rule indicating smallest TA values are seen in the diagonal elements and discussed in Section IV-B. We believe the small difference in the statistics is due to close matches with the similar attack scenarios and validates the conclusions made earlier.

The statistics do not provide a good matching criteria for the two sets of attacks. Attacks I and J are mixed attacks from the same subnet occurring approximately 33 hours apart. The statistics for comparing these attacks are more than 1000 points apart indicating no match. The last set consists of attacks C and Q and they occur approximately 3 months apart. The statistics do not provide a good match for attacks C and Q. Due to the limited information available for the captured attacks, it is very difficult to assess why the techniques do not work. However, two these sets of attacks are single-source attacks that have a very noisy spectrum when observed at 1ms sampling bins [12]. The comparison approach tries to identify dominant frequency patterns when comparing two attacks, therefore it can not make good matches for noisy spectra indicating these techniques can be applied only to attacks that have distinct dominant frequencies. We are exploring how to estimate frequency spectra more robustly especially for single-source attacks as future work.

TABLE II
TOOL CATEGORIES WITH ATTACK RATES WITH NO LOAD AND LOAD CONFIGURATIONS IN KPKTS/S

Type of tool	**Testbed Machine**					
	M1	*w/load*	*M2*	*w/load*	*M3*	*w/load*
Network Limited	15	10	15	10	15	10
Host Limited	9-11	6	15	10	15	10
Self Limited	0.05	0.05	0.05	0.05	0.05	0.05

Hence we observed highly probable repeated attack scenarios, that were detected by the attack fingerprinting system. In the next section, we investigate factors that affect the attack fingerprint, we conducting controlled experiments and isolating one factor at a time.

V. UNDERSTANDING CAUSES OF SPECTRA

In the previous section we showed that real attack traces can be used to build a database of attack fingerprints, and that they can statistically identify multiple attacks representing the same attack scenario. But to trust these results we must *understand* what network phenomena cause these fingerprints, and particularly how robust this technique is to environmental interference. We cannot do this with observations of real attacks because they do not provide a controlled environment.

The key question to the utility of our approach is, what factors influence a fingerprint? Our prior experience working with power spectra [12] suggests that number of attackers, host CPU speed, host load, network link speed, attack tool, and cross-traffic, all affect the dominant frequencies of traffic. Our definition of attack scenario is the combination of a set of hosts and the attack tool. Our hypothesis is that the primary factors that define and alter the frequency spectra are characteristics of an individual attack host (OS, CPU speed, and network link speed) and the attack tool; such a definition of attack scenario would provide a useful tool for network traffic forensics.

If other factors affect the attack traffic, we will require a broader or narrower definition of attack scenario. A broader, less restrictive, definition of attack scenario might be the attack tool alone, if spectral content is largely independent of host characteristics and network characteristics. Such a definition may still be useful for identifying new attack tools, but it would lose the value of applying this approach for forensic purposes. Alternatively, fingerprints may be more strongly dependent on other factors such as network cross-traffic. If fingerprints are strongly influenced by cross-traffic then a fingerprint may be very specific to a point in time and space, thus our approach may lose its value to track a single host/tool pair.

We believe the trace data presented in Section IV is consistent with our hypothesis, since self-comparison argues against a broad interpretation, yet repeated examples of similar fingerprints at different times argues against a narrow interpretation. But we cannot truly verify our definition from trace data because it does not provide a controlled environment.

To validate our definition of the attack scenario, we conduct a battery of controlled experiments on a network testbed testing fingerprint sensitivity to environmental perturbation. First, we observe how the spectral behavior of an attack tool varies due to systematic changes in the environment, such as different operating systems and hardware configurations and analyze spectral behavior of different attack tools. We then study how environmental noise, such as the variations of host load and cross traffic change the attack spectral behavior. The experiments suggest that the attack fingerprint is primarily defined by the host and attack tool characteristics.

A. Testbed Setup

To study the effect of various factors such as OS, attack tool, CPU speed, host load, and cross traffic, on the attack fingerprint, we conduct a battery of controlled experiments on a network testbed. During each experiment, we isolate one parameter of interest, for example, operating system behavior, and study the stability of packet stream fingerprints.

To perform these experiments, we constructed a symmetrical testbed consisting of eight machines connected in a star topology.

The testbed machines are chosen such that there are three sets of two identical machines, the LMx machines have Linux 2.4.20 installed whereas the FMx machines have FreeBSD 4.8. This allows us to keep all hardware configurations exactly the same, when studying the effects of software, such as operating system and attack tools. The testbed includes different hardware architectures and operating speeds to stress our algorithm to the maximum and validate it works in most conditions.

Each pair of machines on the testbed represents increasingly more powerful computers. The first pair of machines, LM1 and FM1, collectively called M1 testbed machines are the slowest machines on the testbed. They have 266MHz Intel PII CPU with 128MB of memory. These machines represent the old generation CPUs on the Internet machines. The next pair of machines, LM2 and FM2, collectively addressed as the M2 testbed machines have 1.6GHz Athlon CPU with 512MB of memory. These machines are the previous generation CPU and they also helps test for differences between Intel and Athlon hardware. The last pair, LM3 and FM3, collectively called as M3 testbed machines, are the current generation of machines and have a 2.4GHz Intel P4 with 1GB of memory.

Great care was taken while setting up the testbed to ensure that all factors, other than the one we want to vary, are kept constant. For example, we ensured all the testbed machines have identical 3Com 3c905C network cards. We constructed a 10Mbit/s network with all the testbed machines connected together with a hub to allow traffic observation. In addition to the symmetrical machines that are used to generate packet stream, we use two additional machines; a observation point machine, which is a 1GHz Intel PIII with 512MB of memory, to gather tcpdump network traces during the experiments, and a victim machine, which is a 600MHz Intel PII with 256MB of memory, that is used as the target for all attack traffic on the testbed. Additionally, we try to minimize local network traffic such as ARPs by ensuring all the testbed machines have a static route to the victim machine and the victim machine is configured to not generate additional ARP or ICMP messages.

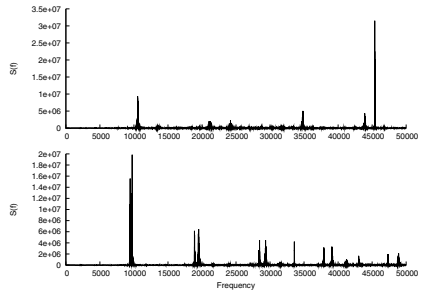


Fig. 6. The effect of the operating system on the attack fingerprint

We conduct all the experiments using using six different attack tools: mstream, stream, punk, synful, synsol, and synk4. We categorize the attack tools into three groups:

(I) Network limited tools that can generate packets at their maximum capacity even when deployed on slow testbed machines such as M1, for example, mstream and stream.

(II) Host limited tools that can generate more attack packets when deployed on a fast testbed machines such as M2 and M3, for example, punk and synful.

(III) Self-limited tools that have a fixed packet rate irrespective of the testbed machine for example, synsol and synk4.

We selected our attack tools such that each category above has two attack tools. All the attack tools generate 40 byte packets and consist of packet headers only. In Section V-G, we modify the attack tools to generate 500B packet to evaluate how a saturated network modifies the fingerprint.

Although all the attack tools generate the same size packets, the different behaviors categorized above is due to the way the tools are programmed. The type I tools have efficient loop structures that can rapidly generate packets without requiring much computational power. Additionally these tools do not randomize many fields in the packet headers. Whereas the type II tools require more computational power usually because they randomize most of the header fields and invoke multiple functional calls between each packet generation. The type III tools are not CPU bound, that is, they do not generate high packet rates as they deliberately introduce delays between packet generation to evade detection. Table II provides information regarding the packet generation capabilities of each attack tool category.

TABLE III

COMPARING THE EFFECT OF OPERATING SYSTEMS ON THE ATTACK FINGERPRINT USING $TA_{XY}(TP_{XY})$.

Type of tool	Testbed Machine		
	M1	*M2*	*M3*
I	1(35)	101(57)	22(57)
II	131(814)	34(87)	7(1)
III	1(1)	2(1)	1(1)

B. Comparing the Spectra

We conduct more than 1000 experiments to explore the factors the affect the attack spectra. While exploring each factor, we conducted experiments on all pairs of testbed machines using all the attack tools. Further, to make sure our results were stable, we performed each experiment at least three times. In all cases the spectral fingerprint estimates are nearly identical.

For each experiment, we observe detailed spectral information using a sampling bin size of $p = 10\mu$s which provides a frequency range up to 50KHz. Since some of the attack tools generate packets at very high rates the increased resolution allows observation of all the frequencies present in the spectrum without losing any information. When the attack tool generates packets at a slower rate, we reduce the sampling rate to minimize the effect of harmonics.

Although, the testbed setup allows us to systematically explore all the factors that effect the fingerprint, we next need to quantitatively compare each set of attack fingerprints. In addition to comparing the spectral plots visually, we find the match set defined in Section III-D.

Specifically, we first need to create a fingerprint database. Since all our experiments were repeated three times, we use one set of the experiment results to generate the fingerprint digests and register them to create the fingerprint database. We then use 100 attack segments from the remaining experiment runs to compare the two spectral fingerprints and test for accuracy and precision of each experiment run.

In the next sections, we present both results, that is, the attack spectral plots as well as the match quality data for each comparison. The results indicate that the attack fingerprint is primarily governed by host and attack tool characteristics. However, if a network link gets completely saturated with cross traffic en route to the victim, the spectrum is significantly altered, and extracting the fingerprint from resulting spectrum may not be possible.

C. Varying the OS

First we evaluate if different operating system can alter the attack stream in different ways when all other factors are constant. If we find that the operating system significantly alters the attack spectrum, then it will be an important aspect of the attack fingerprint.

We conduct experiments with all the attack categories on each pair of the testbed machines. Table III compares the spectra fingerprint for all three categories of attack tools on testbed machines M1 by comparing the attack spectrum of the attack tool on a FreeBSD machine to a Linux machine. We observe that both operating systems produce nearly identical spectra for type I and type III tools on all three pairs of testbed machines. Specifically, both FreeBSD and Linux generate very similar spectra for type I and type III tools results in low TA and TP values.

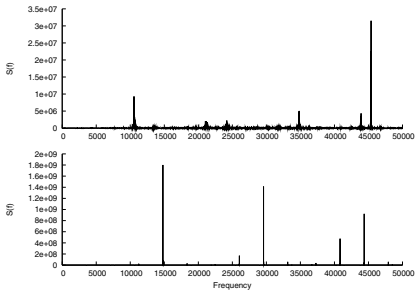


Fig. 7. The effect of CPU speed on the attack fingerprint

However, when comparing the plots in Figure V-A, we observe difference in the spectra for type II tools. We observed that type II tools generate packets at a slightly higher rate on FreeBSD (11Kpkts/sec) than Linux (9Kpkts/s) for M1 machines resulting in different spectra. For the other two sets of testbed machines, since type II tools manage to generate packets at their maximum capacity (15Kpkts/s), they have identical spectra.

Because of the difference in spectra observed for type II tools on M1 machines leads us to conclude that the operating system does effect the attack fingerprint. Table III summarizes the results. Each entry in the table indicates the quality of the comparison on the attack fingerprint when using same attack tool on a FreeBSD machine compared to a Linux machine. As expected the $TP_{FM1(II)LM1(II)}$ for type II attacks on testbed machines LM1 and FM1 is extremely high (814) indicating a poor match. All the other match values indicate a good match between the two attack fingerprints since their values are below the threshold of 100.

This experiment clearly suggests that if the attacker uses a host bound tool, the operating system can influence the efficiency of packet generation and thus create different spectra.

D. Varying the CPU Speed

We now evaluate if CPU speed differences produce a different spectral behavior when keeping all other factors constant. In the earlier section, we saw that the operating system can influence the attack fingerprint, especially on M1 testbed machines. In this section, we demonstrate that when using the same operating system (we use FreeBSD in this example) we observe different attack spectral behavior based on the speed of the CPU. The results in Table IV compare all three attack tool categories on the slowest machines, M1, against the faster machines, M2 and M3, on the testbed.

If the CPU speed did not matter, then we would observe no difference in all the spectra. However, when looking at the TA and TP values, we observe two things. First, type I

TABLE IV

COMPARING THE EFFECT OF CPU SPEED ON THE ATTACK FINGERPRINT USING $TA_{XY}(TP_{XY})$.

Type of tool	Testbed Machines	
	M1:M2	*M1:M3*
I	6(23)	71(35)
II	78(472)	40(436)
III	1(1)	2(1)

TABLE V

COMPARING THE EFFECT OF HOST LOAD ON THE ATTACK FINGERPRINT USING $TA_{XY}(TP_{XY})$.

Type of tool	Testbed Machines		
	M1	*M2*	*M3*
I	2(29)	201(25)	2(1)
II	390(485)	25(174)	1450(2)
III	9(1)	34(1)	2(1)

and type III have identical spectra on both testbed machines indicating that the CPU speed does not alter the attack spectra significantly. Second, type II tools have different spectral behavior on FM1 machines compared to FM3. The Figure V-B shows that since FM1 has a slower CPU, it cannot generate packets at the network speed and has a frequency at 11KHz as compared to machine FM3 that has a sharp peak at 15KHz.

We observe similar results when machines LM1, LM2, and LM3 are compared. Observe that the type II tools have large TP values indicating a poor match.

Similar to our previous conclusion, this experiment also suggests that when using host bound attack tools, the CPU speed affects the attack fingerprint since the packet generation capability is limited by the computation power of the CPU.

E. Varying the Host Load

We have previously observed that CPU speed has a strong influence on spectrum. This suggests that other programs competing for the host CPU may alter an attack spectrum. Therefore in this section, we evaluate the effect of host load on the spectral behavior of the attack stream. If we find that the fingerprint is sensitive to host load changes during the attack, it would make this technique more restrictive in its application. Host load, similar to cross traffic on the network (Section V-H), is ephemeral (since it changes with time) and thus ideally should not contribute to the attack fingerprint. Our results indicate that the proposed algorithms are robust to changes in the attack fingerprint due to host load.

To perform this set of experiments, we first need to generate host load on the testbed machines. We therefore launch the attack tools along with a command-line instance of Seti@home [27]. Seti@home is a widely available, non-trivial application that generates large amounts of computational load. For our experiments, we execute a single copy of Seti@home in the foreground at normal priority, unlike it's usual configuration where it runs in the background. Seti@home forces the CPU usage of the attack tool to drop to a 45–60% range. When the attack tools are executed exclusively on the testbed machine the CPU usage ranges between 85-95% as reported by top. These CPU usage values indicate a significant difference in the performance of the attack tool with and without Seti@home.

Referring to Table II, observe that both type I and type II tools experience a drop in the aggregate attack packet rates when load is added. Due to the extra load on the testbed machine, the attack tool get scheduled less frequently and hence can no longer generate packets at its peak attack rate.

In Table V we compare the attack spectral fingerprints for type I tools on the Linux machines without load and with load. In this example, we compare only type I attack tools, since both type II and type III tools are not good candidates for such comparisons. Type II tools do not have the same spectra on different CPU speeds and hence cannot be compared across testbed machines whereas type III tools generate packets at such low rates that they are not affected by the increased load.

We observe all the testbed machines have the same dominant frequency at 15KHz for both no load and load conditions. However, the addition of host load increases the power in low frequencies by about 10%. Although, the load changes the lower frequency content, it does not add any dominant frequencies and therefore the spectral behavior is stable.

Table V summarizes the quality of the fingerprint matches under load conditions. The entries in the table match the spectral fingerprint of the Linux testbed machines, with and without load. Type I tool provides a good match across all testbed machines indicating that the host load does not affect the spectral fingerprint significantly.

This experiment indicates that although the load reduces the overall packet rate by the attack tool our technique for generating attack fingerprints is robust to load and can be used to identify repeated attacks even in case of variation in host load during the attacks.

F. Varying the Attack Tool

Next we evaluate how much does the attack tool contribute to the attack spectral fingerprint. In this section, we try to answer the question, is it possible to identify each attack tool by their spectral behavior observed in the attack stream? If it is possible to do so then each attack tool can have its own spectral fingerprint and it will allow us to understand the deployment and usage of specific attack tools on the Internet.

When comparing the attack fingerprints in the previous sections, we observe that the attack stream is strongly influenced by host parameters such as operating system, CPU speed. Therefore, we know that the attack tool spectral behavior does not survive in the packet stream in all cases partially answering the above question. In this section, we present results that indicate that the attack tool defines the spectrum provided the attack tool is not limited by any other resource.

Referring to Table III, Table IV, Table V we observe that type I and type III attack tools have identical spectra when seen across all the hardware platforms. Both these tool categorizes are not limited by the available resources since they require low resources due to the way they are programmed. Type I

tools are efficiently written and thus do not have a high packet generation overhead and creates the same spectra on all the testbed machines. Type III attack tools on the other hand, have their own distinct fingerprint that is a function of how long the tool waits between two packets.

These results lead us to believe that the attack tool on each attack host creates a distinct pattern that can be fingerprinted to identify repeated attacks.

G. Varying the Attack Packet Size

All the above experiments suggest that the host characteristics (such as operating system, CPU speed) and the attack tool defines the spectral behavior provided the network is not saturated. For Type I and Type II attacks tools, the spectra is influenced by the available network capacity. These tools saturate the network by generating packets at 15Kpkts/s which results in a sharp peak at 15KHz in their respective spectrum. We believe, that if we modify the packet rate by increasing the packet size, then the attack tools will produce a different spectra.

To verify if this is true, we rerun the above set of experiments by increasing the packet size in the attack tools to 500B and observe how the change affects the spectral behavior. Type I tools now generate packets at 2100pkt/s across all testbed machines and are not affected by the load on the machine. Type II tools also generate packets at 2100pkts/s across all testbed machines but the packet rate reduces to 1700pkts/s when load is increased using Seti@home instances. Type III tools still generate packets at 50pkts/s.

Due to space constraints we omit plots that show the changed spectra. However, as expected the increase in packet size resulted in the dominant frequency to move from 31KHz to 2.1KHz for both FreeBSD and Linux machines. Further, since the packet size is large in this set of experiments, the attack spectra are not susceptible to host load. However, the Type II tools on the other hand can generate more packets when there is extra computational resources available, thus when load is added the attack rate reduces. Type III attacks generate a very low volume of packets that can keep up with the slowest machine on the testbed and are thus not affected by the load and have a fixed packet rate.

This experiment suggests that the attack fingerprint is altered by a bottleneck link. In most cases the Internet access link is the bottleneck and is present at the first hop of the path. We have seen that Type I tools that are network bound always saturate the access link and, if computation power is available, Type II tools also saturate the access link leading us to believe that the attack fingerprint is robust is most network path topologies. Next, we explore the effect of cross traffic on the attack fingerprint.

H. Varying the Network Cross Traffic

The above set of experiments provide insight into how software and hardware characteristics contribute to the attack fingerprint. In this section, we explore the effect of cross traffic on the attack spectra.

To understand the impact of the network cross-traffic, we propose a simple model that simulates the network using exponential packet arrivals. A packet is transmitted with a probability $prob$, which ranges from 5–100%. If a decision is made not to transmit a packet, during any time instance, it delays transmission for an exponential amount of time before attempting transmission again. The mean exponential inter-arrival time is the transmission time for smallest packet on the network. The network cross-traffic consists of a mix of different packet sizes. The cumulative distribution of the packet sizes models traffic seen on the Internet [4]. In particular, 50% of the packets are 40bytes, 25% packets are 560bytes, and 25% of the packets are 1500bytes.

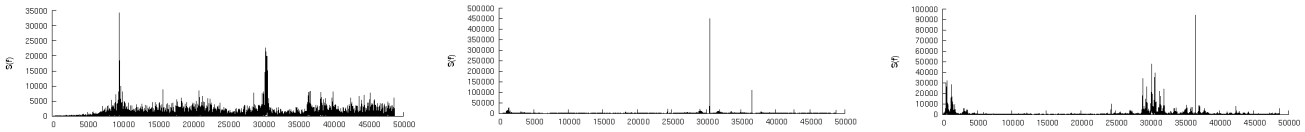
The cross-traffic is then combined with the attack traffic to see its effect on the attack fingerprint. Since we are interested in observing at what point the attack spectrum is affected by the cross-traffic, we progressively increase the cross-traffic rate to see what maximum ratio of cross traffic to attack traffic will still preserve the attack fingerprint.

In Figure 8 we observe how the attack spectrum of type II attacks on LM1 changes as the amount of network cross-traffic increases from 5–100%. When there is less than 60% cross-traffic, a sharp peak can still be observed at 10KHz and the comparison algorithm indicates a good match with TA values of 1–75 and TP values of 35–94. Once the cross-traffic increases to 60% of the traffic on the network, the spectral behavior shifts to a sharp peak at 32KHz and the fingerprint no longer matches (TA=97, TP=583). The sharp peak at 32KHZ reflects that the network is saturated and corresponds to the frequency created by 40byte packets on the network. As the rate of cross-traffic increases further, we can observe other dominant frequencies corresponding to 560bytes and 1500bytes appear in the spectrum.

This experiment indicates that cross traffic of more than 60% network capacity will affect the fingerprint. However, backbone network links rarely operate at capacity and thus the possibility of traversing a saturated link is very minuscule. Thus we believe our attack fingerprinting technique can be used in most network conditions.

The battery of experiments presented in this section suggest that the spectral fingerprint is defined by the attack tool and attacking host (operating system and host CPU) and can be altered only by network paths that are saturated by cross-traffic. When the cross-traffic is increased to more than 60% of the network capacity then the fingerprint is dominated by the frequencies present in the cross-traffic. Additionally, although the host load increases the energy in the lower frequencies, it does not change the attack fingerprint and therefore provides good matches when using the proposed algorithms.

The experiments collectively support our hypothesis that the attack scenario is primarily defined by the attacker host and the attack tool. We have shown as long as the some link (usually the first hop) in the path remains saturated, the spectral behavior will not change. Therefore, the attacker must reduce the attack rate to below saturation for *each* zombie individually in order to alter the attack fingerprint.



(a) 50% Traffic (b) 60% Traffic (c) 100% Traffic

Fig. 8. Effect of cross traffic on the attack spectra

VI. Robustness to Countermeasures

In the previous section we performed a detailed evaluation of how systematic environmental factors and noise effect the fingerprint. We showed that the attack spectra is robust to most environmental changes and hence can be used effectively to fingerprint an attack. However, a DoS attacker is adversarial, so we next consider how a sophisticated attacker might change their attacks to alter their fingerprint.

Similar to most protection systems, our fingerprinting systems is also vulnerable to active countermeasures. We show our system is robust to small adversarial countermeasures, and that the deployment of such a system raises the bar on the amount of effort required on part of the adversary to evade identification.

Attack root-kits are easy to create and and could consists of a number of different attack tools with configurable options to control parameters such as:

- Attack tools
- Number of zombies
- Attack send rate
- Location of zombies
- Start time
- Packet size

We next consider how each of these parameters affect the performance of our fingerprinting system.

a) Change in attack tool: A common theme in our studies from Section V is that the *limiting resource* dominates the attack spectra. We observe that if the network is the limiting resource and the attacker does not change the packet size, then the fingerprint is insensitive to change in the attack tool, attack rate, or the number of zombies. However, if the attack tool is the limited resource, then the change would create different spectra and we must treat the new attack as a different attack scenario with a different fingerprint.

b) Change in number of zombies: The attacker may invoke different number of zombies for a repeated attack to increase or reduce the attack rate. We believe that this will affect the attack fingerprint only if the size of the attack troop is changed significantly. We use simulation to test the effect of small changes in the attack troop on the fingerprint. We first create multiple attack streams each consisting of 40 byte UDP packets at an approximately rate 450pkts/sec (saturation capacity of a 128kb/s ADSL uplink). We then create a large attack by merging 50 such attack streams using techniques described by Kamath and Rupp et al [15], [25]. The attack fingerprint has a peak frequency at 22500Hz. We then randomly remove 1–3 streams from the aggregate attack stream and test the resulting attack fingerprint for a match (figure omitted due to space constraints). We observe that good matches with low accuracy and precision values of 2(6) for 49 zombies, 5(9) for 48 zombies, and 7(12) for 47 zombies when compared to the attack fingerprint of the original attack. When we remove more than five zombies, which is equal to 10% change in the number of zombies, we observe poor match values.

Thus the system is robust to small changes in the attack troop. However, if there are large changes in the attack troop then we must treat the new attack as a different attack scenario with a different fingerprint.

c) Change in attack send rate: Fine control over attack send rate is not necessarily easy. Most attack tools are defined to send as fast as possible. Assuming the attacker is willing to reduce attack effectiveness by reducing the attack rate, we observed that by simply adding a microsecond sleep can substantially reduce the packet send rate by 3000-5000pkts/sec. For attack tools already designed to control rate, we expect that minor changes correspond to minor shifts in the spectra as when we consider changes in packet size below.

d) Change in zombie location: Next, we consider the effect of the attacker changing zombie location, but keeping the number of zombies approximately the same. We believe this will affect the signature only if a substantial number of the replacement zombies have a different limiting resource, such as additional network capacity or CPU power (whichever is limiting). If the limiting resource changes then we must treat the new attack as a different attack scenario with a different fingerprint.

e) Change in start time: Changing the location or start time of the attack could affect the fingerprint by changing interferences from cross traffic. If cross-traffic were a limiting resource this would change the fingerprint, or traffic might cause enough noise to make matches unlikely. In our evaluation of real-world attacks (Section IV-C) we showed successful matches from several several attacks occurring at different times of the day, for example, attacks M, N,O, and P occur over a period of six hours with attack M starting at 1pm and attack P starting a 7pm. This example suggests that, at least in some cases, cross traffic is not the limiting resource. Additionally, in Section V-H we conduct testbed experiments

to show that the attack fingerprint does not change when the cross-traffic is less than 60% of the link capacity. Since current ISP operating practices are to run the core network at low utilization, it seems unlikely that cross-traffic will reach these levels at the core. If a new attack location causes saturation of different link, the link will likely be near the source or the victim. Should a new link near the source be saturated it will bring the attention of the network operator at the source, reducing any stealthiness of the attack. Often times saturating the victim's network connection is a goal; we expect that many fingerprints will include a saturated victim link.

f) Change in packet size: Finally, the attacker can easily change the packet size in the attack streams. Doing so alters the signature. An attacker would therefore like to try as many packet sizes as possible. However, an attacker's options are somewhat limited for several reasons. First, *small* changes in packet size correspond to only small shifts in the fingerprint. We show this by conducting a set of testbed experiment using the setup described in Section V. We programmed a Type I attack tool to control the attack packet size and then conducted three sets of experiments on FM1 machines to change the default attack packet size of 40 bytes to 45, 47, and 50 byte packets. Due to the increase in packet size, the Type I tool now generates a slightly lower attack rate of 14600-14200pkts/sec. This results in a small shift to a lower peak frequency of 14500Hz (figure omitted due to space constraints). We then applied the attack fingerprinting algorithm on the new fingerprints and observed good matches with small low and $range$ values of 5(11) for 45B attack packet, 7(20) for 47B attack packet, and 10(32) for 50B attack packets when compared with the default packet size of 40B. The values indicate an accurate and precise match and therefore imply that the fingerprinting technique is not sensitive to small variations in packet size.

Therefore attackers must make large changes in packet sizes to generate a new fingerprint. Second, distribution of packet sizes in the Internet is trimodal, with packets around 40, 550, and 1500 bytes common and intermediate sizes much rarer [10]. Streams of unusual packet sizes (say, 1237B) could be easily detected through other means, should they become commonly used to spoof fingerprints. Therefore there are relatively few choices for an attacker to change to. Should this countermeasure become common, we would need to log three times the number of fingerprints, one for each packet size.

The discussion above clearly suggests that our system is robust to small changes in attack parameters. In the worst case, for large changes, we must build up separate fingerprints for each attack configuration. Our approach there will raise the bar and force much more sophisticated attack approaches.

There is an inherent tension between the ability to be robust to noise or countermeasures and being sensitive enough to distinguish between different attack groups. An addition contribution of this work is to begin to explore this trade-off and highlight it as an area of potential future exploration.

VII. Future Work

Our system uses statistical pattern matching techniques to identify repeated attacks. As such the quality of the results depend on environmental factors and algorithm parameters. In this section we discuss techniques we would like to explore in the future that could strengthen our algorithm.

Number of features: The success of the matching algorithm depends largely on the feature data. In Section III-C, we use dominant twenty spectral frequencies as the features and discuss the effect of feature size on the quality of the match results. This approach seems to capture most of the important features in the attack spectra, however, as future work we hope to re-evaluate the feature data once again when the attack database increases in size. In addition to varying the number of frequencies, we would also like to group adjacent frequencies as one feature. This approach may be more robust to noisy data.

Alternate feature definitions and classification algorithms: Alternative definitions should also be explored. Other features might include the complete spectra, wavelet-based energy bands, certain fields of the packet header, or inter-arrival rate, to create unique fingerprints. These fingerprints may be more robust and be able to handle a larger variety of attacks, that our current technique cannot handle. Additionally, there are many additional statistical clustering techniques that can be applied to identify repeated attacks [6]. We are currently evaluating wavelet-based feature algorithms and automated clustering algorithms for classification.

Higher sampling rates: We currently compute spectra from timeseries evaluated based on sampling bins of fixed size p. Changing p will affects the algorithms since more detailed sampling will generate higher frequency spectra. Particularly with single-source attacks, more "interesting" behavior will be at high frequencies. Sampling at a higher rate may improve identification of such attacks.

Stability and Portability: Another important research question we need to explore when creating an attack fingerprint database is the level of temporal stability that is required for fingerprinting. Traffic usage patterns and volume change dynamically in the Internet varying the composition and quantity of cross traffic. If the fingerprint is sensitive to this variability, the database will need to be recycled periodically and will not provide accurate results. We will attempt to answer such questions by gathering more real-world attacks over a longer period. Also ideally, fingerprints could be "portable", so that fingerprints taken at different monitoring sites could be compared to identify the attack scenarios with victims in different edge networks. It is plausible that signatures generated at two different monitoring sites would be similar if the sites were "similar enough". Characterization of "enough" is an open area.

VIII. Conclusion

In this paper we proposed an attack fingerprinting system to identify instances of repeated attack scenarios on the network. We applied pattern matching techniques making

use of the maximum-likelihood classifier to identify repeated attack scenarios in 18 attacks captured at a regional ISP. We observed seven attacks that are probably repeated attack scenarios and our hypothesis is also corroborated with packet header information gathered from the attack stream.

Additionally, we performed a systematic experimental study of environmental factors that affect the attack stream. We conducted a battery of controlled experiments that allow us to isolate various factors, such as, attack tool, OS, CPU speed, host load, and cross traffic, that may affect the attack fingerprint. Our study indicates that spectral fingerprint is primarily defined by the attacking host and the tool, however, the network influences the fingerprint when it is saturated.

We also performed a detailed analysis of the robustness of the attack fingerprint to active adversarial countermeasures, such as, change in attack send rate, number of zombies, location of zombies, start time, and packet size. The analysis suggests that our system is robust to small changes in attack parameters and in the worst case, for large changes, we must build separate fingerprints for each attack configuration.

Denial of service attacks today are used for extortion, cybervandalism, and at times even to disrupt competitor websites. We believe our system provides a new tool that can be used to assist in criminal and civil prosecution of the attackers. Such a system would greatly enhance network traffic forensic capabilities and aid in investigating and establishing attribution of the DoS attacks seen on the Internet.

Acknowledgments: This material is based on work partially supported by the United States Department of Homeland Security contract number NBCHC040137 (”LANDER”). All conclusions of this work are those of the authors and do not necessarily reflect the views of DHS. We would like to thank Los Nettos for helping setup the trace machines and discussions about handling DDoS attacks. We would also like to thank the members of the ANT research group for their discussions about spectral analysis and evaluation of classification schemes. Finally, we are indebted to Jim Kurose for discussions clarifying our problem formation and discussion of motivation.

REFERENCES

[1] Paul Barford, Jeffery Kline, David Plonka, and Ron Amos. A signal analysis of network traffic anomalies. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, Marseilles, France, November 2002.

[2] Andre Broido, Evi Nemeth, and kc Claffy. Spectroscopy of DNS Update Traffic. In *Proceedings of the ACM SIGMETRICS*, San Diego, CA, June 2003.

[3] Chen-Mou Cheng, H.T. Kung, and Koan-Sin Tan. Use of spectral analysis in defense against DoS attacks. In *Proceedings of the IEEE GLOBECOM*, Taipei, Taiwan, 2002.

[4] kc Claffy, G Miller, and K Thompson. The nature of the beast: Recent traffic measurements from an internet backbone. http://www.caida.org/outreach/resources/learn/packetsize, April 1998.

[5] United States Code. Fraud and related activity in connection with computers. USC, Title 18, Part I, Chapter 47, Section 1030.

[6] Richard Duda, Peter Hart, and David Stork. *Pattern Classification*. Wiley Interscience, New York, NY, 2000.

[7] N.G. Duffield, J. Horowitz, F. Presti, and D Towsley. Network delay tomorgraphy from end-to-end unicast measurements. In *Lecture notes in Computer Science*, 2001.

[8] Anja Feldmann, Anna C. Gilbert, Polly Huang, and Walter Willinger. Dynamics of IP traffic: a study of the role of variability and the impact of control. *SIGCOMM Comput. Commun. Rev.*, 29(4):301–313, 1999.

[9] Xinwen Fu, B Graham, D Xuan, R Bettati, and Wei Zhao. Empirical and theoretical evaluation of active probing attacks and their countermeasures. In *6th Information Hiding Workshop*, Toronto, Canada, May 2004.

[10] Nanog: North American Network Operators Group. Internet packet size samples. http://www.merit.edu/mail.archives/nanog/2000-07/msg00691.html.

[11] Ian Hopper. Mafiaboy faces prison term. ”http://archives.cnn.com/2000/TECH/computing/04/19/dos.charges/”, February 2000.

[12] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. A Framework for Classifying Denial of Service Attacks. In *Proceedings of ACM SIGCOMM 2003*, Karlsrhue, Germany, August 2003.

[13] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. Identification of Repeated Denial of Service Attacks. Technical Report ISI-TR-2003-577, USC/Information Sciences Institute, February 2005.

[14] Anil Jain, Robert Duin, and Jainchang Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[15] Purushotham Kamath, Kun Chan Lan, John Heidemann, Joe Bannister, and Joe Touch. Generation of high bandwidth network traffic traces. In *In Proceedings of MASCOTS*, pages 401–410, Fort Worth, Texas, USA, October 2002. IEEE.

[16] Dina Katabi and Charles Blake. Inferring congestion sharing and path characteristics for packet interarrival times. Technical report, MIT-LCS, 2001.

[17] Gregg Keizer. DOJ Accuses Six Of Crippling Rivals' Web Sites. ”http://news.bbc.co.uk/1/hit/technology/3549883.htm”, August 2004.

[18] Los Nettos Passing Packets since 1988. http://www.ln.net.

[19] Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. Controlling high bandwidth aggregates in the network. In *ACM Computer Communication Review*, July 2001.

[20] David Moore, Geoffrey Voelker, and Stefan Savage. Inferring Internet Denial of Service activity. In *Proceedings of the USENIX Security Symposium*, Washington, DC, USA, August 2001. USENIX.

[21] Christos Papadopoulos, Robert Lindell, John Mehringer, Alefiya Hussain, and Ramesh Govindan. COSSACK: Coordinated Suppression of Simultaneous Attacks. In *In Proceeding of Discex III*, Washington, DC, USC, April 2003.

[22] Craig Partridge, David Cousins, Alden Jackson, Rajesh Krishnan, Tushar Saxena, and W. Timothy Strayer. Using signal proceesing to analyze wireless data traffic. In *Proceedings of ACM workshop on Wireless Security*, pages 67–76, Atlanta, GA, September 2002.

[23] Vern Paxson. Bro: A system for detecting network intruders in real-time. *Computer Networks*, 31(23–24):2435–2463, Decemeber 1999.

[24] Martin Roesch. Snort - lightweight intrusion detection for networks. http://www.snort.org.

[25] Andy Rupp, Holger Dreger, Anja Feldmann, and Robin Sommer. Packet trace manipulation framework for test labs. In *Proceedings of ACM SIGCOMM Internet Measurement Conference 2004*, Sicily, Italy, October 2004.

[26] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson. Practical network support for IP traceback. In *Proceedings of the ACM SIGCOMM Conference*, pages 295–306, Stockholm, Sweeden, August 2000. ACM.

[27] Seti@home. Search for extraterrestrial intelligence. http://setiathome.ssl.berkeley.edu/.

[28] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio Stephen T. Kent, and W. Timothy Strayer. Hash-based ip traceback. In *Proceedings of the ACM SIGCOMM*, pages 3–14, San Deigo CA, August 2001. ACM.

[29] BBC Online Technology. DDoS extortion in the uk gambling industry. ”http://news.bbc.co.uk/1/hit/technology/3549883.htm”, March 2004.

[30] Los Angeles Times. Deleting Online Extortion. ”http://www.latimes.com/business/la-fi-extort25oct25,1,5030182.story?coll=la-home-headlines”, October 2004.

[31] Tripwire. http://www.tripwire.com.

[32] Vnunetwork. WorldPay hit by malicious denial of service attack. ”http://www.vnunet.com/news/1158559”, October 2004.