Identification of Repeated Attacks Using Network Traffic Forensics

Alefiya Hussain John Heidemann Christos Papadopoulos
USC/Information Sciences Institute
{hussain,johnh,christos}@isi.edu

ABSTRACT

Denial-of-service attacks on the Internet today are often launched from zombies, multiple compromised machines controlled by an attacker. Attackers often take control of a number of zombies and then repeatedly use this army to attack a target several times, or to attack several targets. In this paper, we propose a method to identify repeated *attack scenarios*, that is, the combination of a particular set of hosts and attack tool. Such identification would help a victim coordinate response to an attack, and ideally would be a useful part of legal actions. Because packet contents can be forged by the attacker, we identify an attack scenario by spectral analysis of the arrival stream of attack traffic. The attack spectrum is derived from the characteristics of the attack machines and can therefore be obscured only by reducing attack effectiveness. We designed a multi-dimensional maximum-likelihood classifier to identify repeated attack scenarios. To validate this procedure we apply our approach on real-world attacks captured at a regional ISP, identifying similar attacks first by header contents (when possible) and comparing these results to our process. We conduct controlled experiments to identify and isolate factors that affect the attack fingerprint.

1. INTRODUCTION

Denial of service (DoS) attacks occur every single day on the Internet [16]. To launch a DoS attack, the attacker relies on the existence of attack tools on compromised machines that enable him to attack any target on command. Typically, an attacker compromises a machine (also called a *zombie*), installs the attack tools there, and then organizes one or many compromised machines into *attack troops*. The attacker then repeatedly deploys the same attack troop to flood different targets. Every invocation of the attack troop to target a new victim is identified as a DoS attack and the combination of the attack troop and the attack tool is defined as an *attack scenario*.

Approaches to network security are focused on attack prevention, detection, and resolution. Prevention of DoS attack encompass techniques that ensure integrity of the hosts [24, 26] and techniques to detect and rate-limit abnormal network activity [14, 15]. The difficult task of detecting a DoS attack has attracted a great deal of research attention in the last few years. The prevailing approach of most of these efforts has been to focus on packet content signature matching [19, 20] or anomaly detection by correlating network traffic [8, 17, 25]. However, without large-scale deployment of the attack prevention strategies, it is unlikely that malicious activity will be eliminated on the Internet. Approaches for the resolution of ongoing attacks range in difficulty. One simple and effective technique adopted by many systems is to block malicious packets, whereas more complex traceback techniques [21, 23] attempt to isolate the source machines.

When resolving DoS attacks, it would be helpful to know if the current attack has been observed previously and if it originated from the same hosts. In criminal courts of law, forensic evidence is used to investigate and establish facts, and consequences often depend on the severity of the attackers actions. We believe that network traffic can provide evidence to identify repeated DoS attacks from the same attack scenario, and, by inference, the same attacker, much as ballistics studies of firearms can trace multiple uses of a weapon to the same gun. Such evidence of repeated attacks would help establish the maliciousness of a given attacker prompting legal response. It might also help guide practical response, such as the amount of effort spent tracing back attack hosts or improving filtering.

In this paper we explore the possibility of applying *network traffic forensics* to identify patterns in attack traffic and build an *attack fingerprinting* system to passively monitor attack traffic on the Internet. The goal of the proposed attack fingerprinting system is not to design yet another intrusion detection system based on attack packet content, but to design a system that can identify attack patterns encoded in the *attack stream*, the sequence of attack packets created by the host machine and the attack tool. The attack stream is shaped by many factors: number of attackers, attack tool, operating system, host CPU, network speed, host load, and network cross-traffic. Since we define an attack scenario as a combination of the attacker and attack tool, the fingerprinting techniques should be robust to variability in host load and network cross traffic.

The contribution of this paper is an automated attack fingerprinting algorithm to identify repeated attack scenarios. The algorithm is based on detailed analysis of the effects of host load and cross traffic on the attack signature. The attack scenario fingerprints are learned by observing the attacks on the Internet. Once an attack is detected, the spectral profile of the attack is compared to the previously registered fingerprints in the database to look for similar attacks. The spectral profile is generated by identifying the dominant frequencies from the power spectral density. We found that even though there is sensitivity with respect to host load and cross traffic, the dominant frequencies remain nearly invariant and can be used to uniquely identify an attack scenario To our knowledge, there have been no previous attempts to identify or analyze attack scenarios for forensic purposes. We describe these techniques in detail in Section 3. In Section 5, we carefully study how traffic is influenced by changes at the source (in application software, OS, or CPU) and in the network (cross-traffic). Although, we undertook these studies to validate our forensic work, the observations apply to Internet traffic in general.

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

Create Scenario Fingerprint for Attack A

- Create multiple attack segment time-series $x_k(t)$ of the filtered attack packets where $k = 1 \ldots N_A$ and $0 \leq t \leq 2$ seconds,
- Estimate the power spectral density $S_k(f)$ for each $x_k(t)$ (Eq 3),
- Extract dominant frequencies from each segment $S_k(f)$ to form a attack feature segment X_k, where $k = 1 \ldots N_A$
- Define the attack fingerprint F_A consisting of all X_k
- Create attack digest estimating distribution parameters, M_A and C_A(Eq 4 and 5),
- Register attack A in the database

Compare Current Attack C with Registered Fingerprint for Attack A

- Create multiple attack segment time-series $x_l(t)$ of the filtered attack packets where $l = 1 \ldots N_C$ and $0 \leq t \leq 2$ seconds,
- Estimate the power spectral density $S_l(f)$ for each $x_l(t)$ (Eq 3),
- Extract dominant frequencies from each segment $S_l(f)$ to form a attack feature segment X_l, where $l = 1 \ldots N_C$
- Compute the divergence, $l_{CA,l}$ for each attack feature segment X_l against attack digest A (Eq 6),
- Define L_{CA} as set of N_C matches between attack C and A
- Test for accuracy and precision of the attack match by computing low_{CZ} and $range_{CZ}$,

Table 1: Algorithm used to register and compare attack scenarios

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 4. We further investigate our methodology by conducting controlled experiments on a testbed with real attack tools. The testbed environment enabled testing the robustness of the attack scenario fingerprints with changes in host load and cross traffic. We discuss experimentation details in Section 5

2. RELATED WORK

Pattern recognition has been applied extensively in character, speech, image, and sensing applications [11]. 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 [4] and Katabi and Blake apply pattern clustering to detect shared bottlenecks [12]. 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. Cheng et al. apply spectral analysis to detect high volume DoS attack due to change in periodicities in the aggregate traffic [5] whereas Barford et al. make use of flow-level information to identify frequency characteristics of DoS attacks and other anomalous network traffic [1]. Prior research in the DoS area has applied signal processing to detect ongoing attacks. In a broader context, researchers have used spectral analysis to extract information about protocol behavior in encrypted wireless traffic [18]. 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 [8, 17, 25] or signature-scan techniques [19, 20]. Both these techniques have their disadvantages. Anomaly-detection systems first need to learn what normal traffic consists before it can identify unusual network events. making it prone to many false positives. Signature-scan techniques require a large set of frequently updated rules to identify attacks and are not an effective defense for novel attacks. While both types of IDS can provide hints regarding if a particular attack was seen before, they do not have a techniques to identify if it originated from the same set of attackers.

3. ATTACK SCENARIO FINGERPRINTING

In this section we develop the algorithm used to identify similar attack scenarios. We first provide an intuitively explanation of how the detection of repeated attacks works. We then detail the algorithm with the help of an example.

3.1 Our Approach in a Nutshell

Given an attack, we wish to test if this scenario occurred previously. To make this identification, we filter the attack packets and create an attack fingerprint. Intuitively, the fingerprint of each attack scenario can be uniquely mapped as a set of points in space generating a density of points at a particular location in space, or in other words, a multivariate probability density function corresponding to the attack scenario. We define attack scenario fingerprinting as categorizing attacks based on the distance between their probability densities. The detailed methodology to compare two attack scenarios is outlined in Table 1 and described in Section 3.3 and Section 3.4.

Figure 1 can be used to show how such a system would work. Assume the attack fingerprint database consists of two attacks O and H. We now want to compare a current attack M with both these attacks in the database to identify if the attack M has occurred earlier. When visually comparing attack M with O (top plot in Figure 1), we observe that the spectra overlap indicating very similar behavior. Thus the algorithm (discussed in Section 3.3) would return good precision and accuracy distance values for attack M compared with O that can then be used to indicate a repeated attack. Next, we will compare current attack M with attack H in the database to distinguish dissimilar attacks. Looking at the spectral fingerprints of attack M and H in the bottom plot of Figure 1 we observe that the spectral fingerprints are distinct and do not overlap. Therefore the algorithm would return poor precision and accuracy distance values for attack M compared with H that can be used to indicate no match. We present more examples with real attacks in Section 4. Next, we elaborate on the approach outlined in Table 1.

3.2 Creating the Attack Fingerprint

Before we can generate the attack fingerprint, we first need to extract fingerprint features from the attack stream. To extract feature data, we first convert the packet trace into a time series. We assume a given sampling bin of p second and define the arrival process $x(t)$ as the number of packets that arrive in the bin $[t, t + p)$. For a T second long packet trace, we 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. Therefore, based on the frequency range of interest, researchers use different bin sizes [9, 5]. In this paper, 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 $S_k(f)$. Since initial ramp-up or abrupt changes in the attack stream can bias the spectral analysis, we do not include such segments in the analysis. We compute the power spectral density of each attack segment by performing the discrete-time Fourier transform on the autocorrelation function (ACF) of the attack segment.

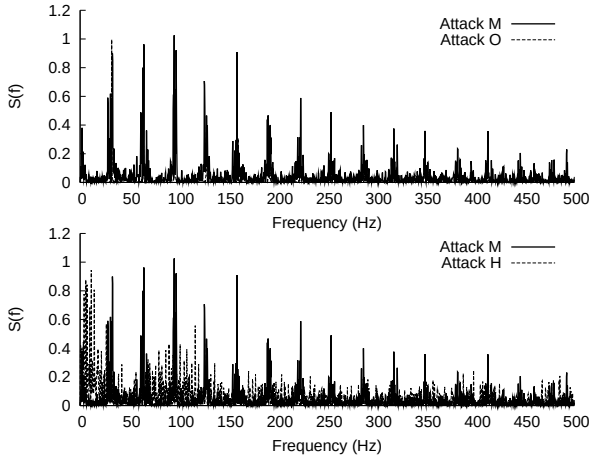


Figure 1: Top plot: Frequency Spectra of two similar attacks overlap, Bottom plot: Frequency Spectra of two dissimilar attacks are distinct

The ACF is a measure of how similar the traffic is to itself shifted in time by offset ℓ [2, 3]. When $\ell = 0$ we compare the traffic stream to itself, and the autocorrelation is maximum and equal to the variance of the traffic stream. When $\ell > 0$ we compare the traffic stream with a version of itself shifted by lag ℓ. Therefore, for every attack segment k we calculate the ACF as:

$$c_k(\ell) = 1/M \sum_{t=0}^{M-\ell} (x_k(t) - \bar{x_k})(x_k(t+\ell) - \bar{x_k}); \quad (1)$$

$$r_k(\ell) = c_k(\ell)/c_k(0) \quad (2)$$

where $\bar{x_k}$ is the mean of $x_k(t)$ and M is the length of the attack segment $x_k(t)$.

The power spectrum $S_k(f)$ of the attack is obtained by the discrete-time Fourier transform of the ACF to gives the frequency spectra for each attack segment, as shown in Figure 1. Formally:

$$S_k(f) = \sum_{\ell=0}^{2M-1} r(\ell) e^{-\imath \ell 2\pi f} \quad (3)$$

Next we need to define a technique to quantitatively compare each attack segment. Therefore, we define an attack segment fingerprint X_k to be the frequency representation for each segment k (where $k = 1 \ldots N_A$), consisting of the twenty dominant frequencies in $S_k(f)$. 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. The dominant twenty frequencies provide a good estimate of the important periodic events that constitute the attack stream. In Section 6, we perform a detailed analysis of the effect of signature size on the matching algorithm.

Next for each attack A, we define F_A as the attack fingerprint consisting of all the segment fingerprints $X_k (k = 1 \ldots N_A)$. 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:

$x_k(t)$	Segment of the times series of attack packet traces
p	Sampling bin size used to create time series
N_A	Number of attack segments available for attack A
M	Length of each attack segment
$S_k(f)$	Spectral density of the attack segment $x_k(t)$
X_k	Attack feature segment where $k = 1 \ldots N_A$
F_A	Attack fingerprint consisting of all X_k
M_A	Mean vector of F_A
C_A	Covariance vector of F_A
$l_{CA,k}$	Match value between segment k of attack C and attack digest A
L_{CA}	Set of all matches between attack C and A
low_{CA}	5% quantile of L_{CA}
$high_{CA}$	95% quantile of L_{CA}
$range_{CA}$	The difference $high_{CA}$ - low_{CA}

Table 2: Notation used in the attack fingerprinting algorithms

$$M_A = 1/N_A \sum_{k=1}^{N_A} X_k \quad (4)$$

$$C_A = 1/N_A \sum_{k=1}^{N_A} (X_k - M_A)(X_k - M_A)^T \quad (5)$$

A minimum ratio of 10 for the the number of attack segments N_A, to the size of the feature segment X_k, is required to ensure robust estimates for the mean and covariance of F_A. Since the feature segment consists of twenty dominant frequencies, we therefore consider attacks that consist of at least 200 segments, ($N_A = 200$) each of 2 second, duration making the minimum attack duration of 400seconds. As a result, the attack fingerprint F_A is defined as a 20x200 matrix. The attack digest M_A is defined as 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.

Table 2 summarizes the notation used to estimate the distribution parameters. 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 separate attacks when we were working on our previous paper [10].

3.3 Comparing Two Attacks

Once we have a database of registered attack fingerprints, we can test if the current attack scenario, C, has been previously observed by applying the Bayes maximum-likelihood classifier [7]. 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 first verify that the attack segment fingerprint F_A has an approximately normal distribution for each dominant frequency represented in each segment X_k where $k = 1 \ldots N_A$. To test this assumption, we plot the distribution of the first element from X_k of attack A in Figure 2. The first element has a mean of 15Hz with good spread of values around the mean,

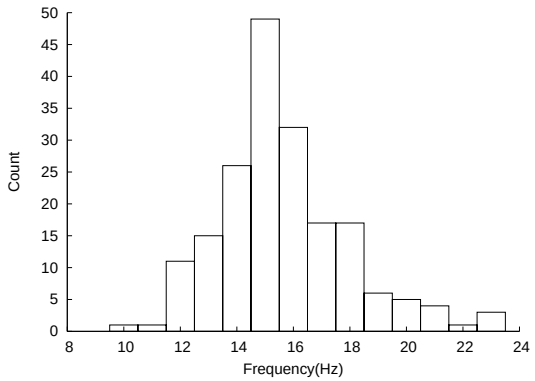


Figure 2: The distribution of the first dominant frequency in F_A for 200 attack segments is approximately normal.

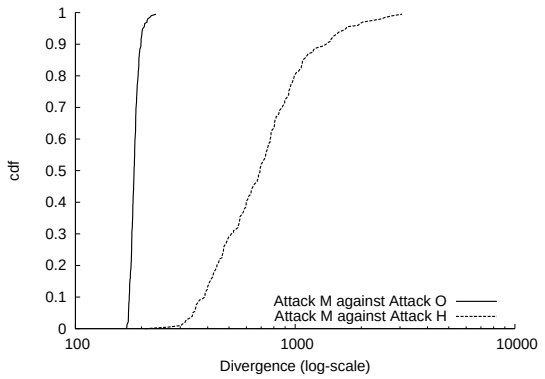


Figure 3: The maximum-likelihood values when comparing the attack M with attacks O and H.

indicating that the spectral profiles are approximately normal. Additionally, we the the χ^2 test at 90% significance level indicate normal distribution of the first element of X_k. We verify all the dominant frequencies of the attack have normal distribution and repeat the same procedure for all attacks incorporated into the database. 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 assumption as discussed in Section 6.

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 less than 200 used for N_A. Since for making attack comparison, we are not estimating distribution parameters and therefore 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 attack A using:

$$l_{CA,l} = (X_l - M_A)^T C_A^{-1} (X_l - M_A) - log|C_A| \qquad (6)$$

where X_l represents each attack feature segment in F_C, $l = 1 \ldots N_C$. Intuitively, Equation 6 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 be generated for all the attacks in the database.

3.4 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? ie: Does attack C match well with the attack digest A?

2. Are the comparisons precise? ie: Does attack C consistently have a small divergence with attack digest A?

To measure accuracy, 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 measure precision, 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.

For example, Figure 3 shows cumulative plots of the matches for attacks M, O and H indicated in Figure 1. Since attack M and O have similar spectra, the set of matches L_{MO} are both accurate, indicated by a small low_{MO} value of 174, and precise, indicated nearly vertical line since the $range_{MO}$ value is 28. On the other hand, when comparing M with H, we obtain a low_{MH} value of 342 and a $range_{MH}$ value of 1700 indicating a poor match.

To automate the matching procedure, we now need to identify what values of low and $range$ 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 $range_{CA} < threshold$ AND $low_{CA} < low_{CB} \forall B \neq A$ (Condition 1).

We empirically derive the values of the $range$ threshold in Section 4.2 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 $low_{CA} > threshold$ (Condition 2). The identification of such a threshold if more difficult since we would need observe completely new attacks in the wild. We believe such a threshold will emerge as the database increases in size.

4. ANALYZING ATTACK TRAFFIC

We now evaluate our comparison technique on attacks captured at Los Nettos, a moderate size ISP located in Los Angeles [13]. This approach was motivated by observing similar spectra during attack classification [10]. We observed that the spectral content of several attacks, even though they occurred at different times, was remarkably similar. Since the trace data is from a live network,

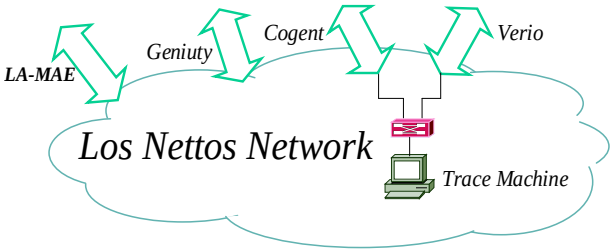


Figure 4: The trace machine monitors two of the four peering links at Los Nettos.

we cannot prove that independent attacks are from the same hosts. Instead, in Section 4.2 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 4.3.

4.1 Trace Methodology

Los Nettos has four major peering links with commercial ISP providers. Due to lack of available mirroring capacity, we were able to monitor only two links. Los Nettos has a diverse clientele including academic and commercial customers. The trace machine is an off-the-shelf Intel P4 1.8GHz, with 1GB of RAM running FreeBSD 4.5. We use a Netgear GA620 1000BT-SX NIC, and modified the driver to support partial packet transfer from the NIC to the kernel. Typical daytime load is 140Mb/s with a mean of 38Kpackets/s. Measurement drops (as reported by tcpdump) are usually below 0.04% during normal operation, rising to 0.6% during attacks that increase packet rates to 100Kpackets/s.

For each monitored link, we continuously capture packet headers using tcpdump, creating a trace file every two minutes. Each trace is processed and flagged as containing a potential attack if either of two thresholds are reached: (a) the number sources that connect to the same destination within one second exceeds 60, or (b) the traffic rate exceeds 40Kpackets/s. These thresholds were determined by observing the traffic on the network. Traces that are not flagged as an attack are discarded. We identify and ignore known servers that would trigger these thresholds through normal traffic. Finally, we manually verify each flagged trace to confirm the presence of an attack. The automated thresholding works reasonably well, but provides a false positive rate of 25–35%.

We applied the attack fingerprinting system on 18 attacks captured at Los Nettos. Although, we have observed 80 attacks at Los Nettos when monitoring, we chose to use only longer attacks (at least 400s) to generate fingerprint digests to capture steady-state behavior. This requirement leaves only 18 eligible attacks. This threshold is probably overly pessimistic; evaluating attack duration needed for fingerprint generation is an area of future work.

Table 3 summarizes the the packet header content for each attack captured 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.

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

Table 3: Packet header content observed in the captured attacks

4.2 Emulating the Same Attack Scenario

We have extremely limited amount of information regarding the attack scenario for attacks captured at Los Nettos. The attacks listed in Table 3 are "from the wild", therefore the comparisons among them can only suggest, but not prove, reuse of the same attack hosts and tools. Therefore, 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's tail 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 low and $range$ indicate a good match for the matching conditions described in Section 3.4.

We divide each attack (A–R from Table 3) 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 3.2. The tail section is made up of at least 20seconds of the remaining attack to ensure reasonable number of matches that will give the low and $range$ values.

For each of the tail section, we compare the attack against the registered fingerprints, using the technique outlined in Section 3.3 and Section 3.4 to produce the comparison matrix of low and $range$ values given in Table 4 that indicate the accuracy and precision of the match. For example, we compute low_{AA} and $range_{AA}$ for the set of matches L_{AA}, obtained when comparing the tail of attack A with the attack digest of attack A. Since values greater than 1000 indicate large divergence and hence poor matches, we represent such value by an asterix ($*$).

If our match condition holds, we should get small values for the low and $range$ in the diagonal elements of the comparison matrix. The results presented in Table 4 indicate that when comparing against the same attack scenario, low_{XX} is the lowest provided the $range_{XX}$ being less than 100. For example, in the row of Attack H, although Attack L has a smaller low_{HL} than low_{HH}, the $range_{HL}$ is large than 100. We consistently observe comparing the head and tail sections of the same attack provide the closest

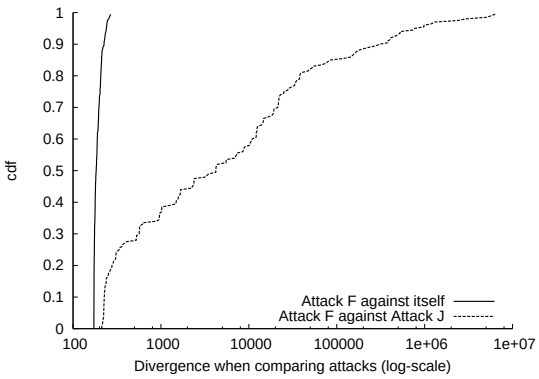


Figure 5: The cumulative distribution of the maximum-likelihood values when comparing the same attack scenario and when comparing different attacks.

matches for all attacks (exceptions are discussed in Section 4.3), validating our comparison techniques. Thus, the comparisons of the head and tail sections provides us with the estimate for the $range$ threshold. We subsequently use a $range$ threshold of 100 in match Condition 1 (Section 3.4) to indicate a good match.

Table 4 compares 18x18 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 ($low_{FF} = 172$ and $range_{FF} = 57$) and the registered fingerprint of attack J ($low_{FJ} = 223$ and $range_{FJ} = 768333$), and visually analyze the difference in 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 $range_{FF}$ 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 range 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.

We also evaluate the false negative, that is attacks that have their tail section match better with a separate attack head section. The low_{MM}, low_{NN}, and low_{OO} diagonal elements are about three points higher than the non-diagonal elements low_{MP}, low_{MO}, and low_{OP} indicating closer matches with separate attach head sections. We believe this is due to the close match in these attacks. The false positives in the attack matching algorithm, indicated by detecting an attack has occurred previously, when actually it not present in the fingerprint database is more difficult to identify in the real world attacks without well defined threshold match Condition 2 (Section 3.4).

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 $range$ 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.

4.3 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 Table 4 provides the low_{XY} and $range_{XY}$ 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 3.4), with the $range$ threshold of 100, established in the previous section. The packet contents in Table 3, provide insight into plausible repeated attack scenarios. We expect the low and $range$ values to be small for similar attacks. We observe four sets of very similar scenarios.

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 low_{FG}, low_{GF} are the smallest in the non-diagonal elements with $range_{FG}$, $range_{GF}$ less than 100. Further, small low_{RF}, and low_{RG} with small $range_{RF}$, and $range_{RG}$ statistics indicate attack R is similar to attacks F and G. We did not obtain sufficiently small low_{FR} and low_{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 low and $range$ statistics in the non-diagonal elements providing a good four-way match with each other. Due to the close match, the low_{MM}, low_{NN} and low_{OO} diagonal elements are approximately three points higher than the non-diagonal elements low_{MP}, low_{MO} and low_{OP} respectively. These attacks therefore are an exception of the rule indicating smallest low values are seen in the diagonal elements and discussed in Section 4.2. 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 with same subnet and occur more than 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 [10]. 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.

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

5. EXPERIMENTAL EVALUATION

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

The key question to the utility of our approach is, what factors influence a fingerprint? Our prior experience working with power spectra [10] suggests that number of attackers, host CPU speed,

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
A	201(15)	210(123)	205(18)	211(172)	201(30)	237(133)	237(110)	*(*)	363(*)	223(*)	246(143)	*(*)	351(405)	423(658)	341(311)	396(493)	230(244)	216(117)
B	211(19)	198(19)	209(14)	211(63)	216(18)	235(83)	234(56)	*(*)	221(289)	222(*)	230(90)	*(*)	307(127)	358(191)	288(139)	338(203)	276(178)	239(90)
C	207(32)	209(167)	203(20)	198(161)	211(68)	205(151)	200(133)	354(*)	225(833)	219(*)	206(160)	*(*)	216(371)	231(506)	215(313)	221(441)	213(298)	203(174)
D	208(38)	215(213)	202(33)	188(38)	206(96)	172(170)	185(120)	130(*)	251(813)	213(*)	173(210)	92(*)	186(324)	198(521)	185(347)	183(429)	159(396)	167(199)
E	189(40)	221(23)	197(31)	231(76)	185(42)	239(97)	250(71)	*(*)	324(558)	187(*)	243(165)	*(*)	376(169)	423(312)	391(87)	460(175)	182(342)	180(177)
F	211(23)	248(157)	203(22)	187(58)	240(54)	172(57)	185(69)	135(*)	344(636)	223(*)	173(125)	256(*)	188(109)	203(109)	185(120)	181(169)	160(231)	167(128)
G	203(26)	237(108)	198(23)	189(51)	221(64)	180(91)	180(42)	330(*)	253(422)	203(*)	183(93)	*(*)	172(127)	173(116)	172(152)	171(179)	178(233)	176(119)
H	211(8)	377(49)	202(6)	188(15)	275(25)	172(25)	186(15)	129(80)	881(180)	223(*)	172(27)	101(*)	196(21)	218(31)	187(31)	185(21)	159(81)	166(35)
I	206(19)	207(13)	203(19)	204(27)	209(19)	220(50)	211(49)	*(*)	175(50)	0(*)	236(53)	*(*)	262(77)	286(146)	255(91)	286(107)	226(149)	224(59)
J	212(77)	340(124)	217(49)	298(147)	230(87)	338(150)	345(77)	*(*)	660(*)	188(49)	361(96)	*(*)	455(658)	439(*)	505(408)	529(768)	282(496)	249(305)
K	210(19)	270(155)	203(15)	188(31)	250(50)	173(78)	186(48)	129(*)	350(697)	219(*)	172(49)	79(*)	185(66)	187(65)	186(78)	185(86)	159(203)	167(107)
L	213(1)	412(8)	203(1)	188(1)	295(4)	172(1)	188(1)	128(3)	990(34)	223(*)	172(1)	68(21)	201(3)	237(6)	192(3)	189(3)	159(1)	166(1)
M	204(7)	257(50)	198(6)	190(13)	224(32)	180(24)	182(13)	342(*)	336(251)	206(*)	180(29)	*(*)	174(18)	176(22)	174(28)	171(38)	182(64)	178(33)
N	204(7)	261(41)	199(5)	190(11)	227(29)	182(20)	182(13)	372(*)	335(210)	209(*)	182(25)	*(*)	175(26)	175(18)	173(27)	174(40)	180(55)	179(26)
O	204(7)	256(60)	198(5)	188(17)	228(30)	177(27)	180(18)	298(*)	326(280)	204(*)	178(28)	*(*)	172(27)	174(26)	170(26)	168(39)	172(71)	173(37)
P	205(6)	271(58)	198(5)	188(11)	232(30)	176(22)	180(15)	249(*)	397(242)	207(*)	178(23)	*(*)	172(20)	175(19)	170(24)	167(25)	171(64)	172(32)
Q	212(16)	222(202)	203(17)	187(80)	217(82)	172(130)	187(98)	129(*)	308(747)	223(*)	172(138)	85(*)	198(266)	229(335)	190(255)	187(351)	159(96)	166(133)
R	194(30)	201(226)	193(24)	188(41)	202(99)	172(88)	180(70)	130(*)	192(874)	206(*)	173(132)	97(*)	182(173)	183(215)	177(160)	177(237)	159(205)	166(72)

Table 4: The comparison matrix of low_{XY} ($range_{XY}$) statistics for 18 Los Nettos attacks (value greater than 1000 is indicated as $*$). The columns indicate registered attack fingerprints and the rows indicate trial attack segments.

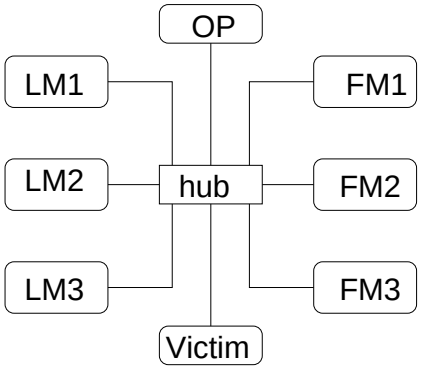


Figure 6: The testbed setup used to evaluate attack spectral stability.

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 4 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. First, we observe how the spectral behavior of an attack tool varies across different operating systems and hardware configurations and analyze spectral behavior of different attack tools. Finally we study the effect of host load and cross traffic on the attack spectral behavior. In recent work, He et al explore the effect of cross traffic on spectral behavior in context of congested links [9]. The experiments suggest that the attack fingerprint is primarily defined by the host and attack tool characteristics.

5.1 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 as shown in Figure 6. 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 Athelon CPU with 512MB of memory. These machines are the previous generation CPU and they also helps test for differences between Intel and Athelon 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 wanted to vary, are we 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

Type of tool	Testbed Machine					
	M1	*M1 w/ load*	*M2*	*M2 w/ load*	*M3*	*M3 w/ load*
I	15Kpkt/s	10Kpkts/s	15Kpktss/s	10Kpkts/s	15Kpkts/s	10Kpkts/s
II	9-11Kpkts/s	6Kpkts/s	15Kpkts/s	10Kpkts/s	15Kpkts/s	10Kpkts/s
III	50pkts/s	50pkts/s	50pkts/s	50pkts/s	50pkts/s	50pkts/s

Table 5: Attack tool categories exercised on the testbed with 50B packets

have a static route to the victim machine and the victim machine is configured to not generate additional ARP or ICMP messages.

The above setup allows us to test attack fingerprint sensitivity to factors such as operating system, host CPU, and host load. 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) 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) 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) tools that have a fixed packet rate that is not affected by 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 5.7, 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 for and while loops 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 5 provides information regarding the packet generation capabilities of each attack tool category.

5.2 Comparing the Experiment 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 were 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. We are currently working on how to improve the sampling technique 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 3.3.

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. We assess the accuracy and precision of the match by computing the low and $range$ values for the set of matches.

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 en route to the victim, the spectrum get significantly altered, and extracting the fingerprint from resulting spectrum may not be possible.

5.3 Experimenting with different OS

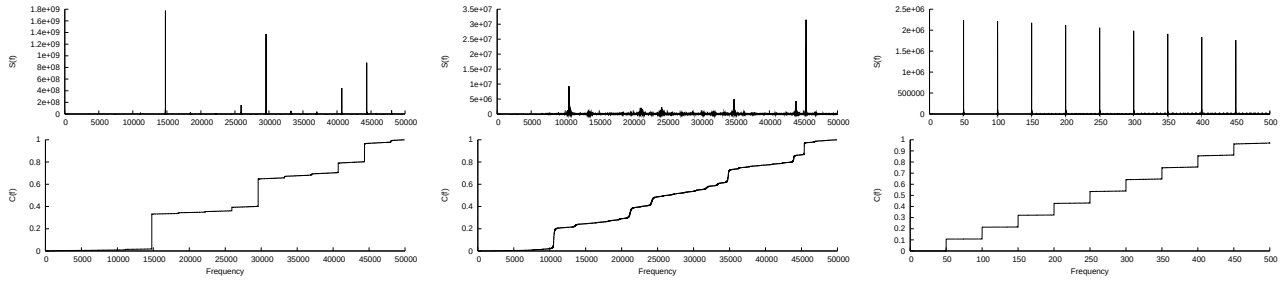
First we evaluate if different operating system can alter the attack stream in different ways when all other factors are constant. If we find that the operating system significantly alters the attack spectrum, then it will be a an important aspect of the attack fingerprint.

We conduct experiments with all the attack categories on each pair of the testbed machines. Figure 7 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. The top plot shows the power spectral density, while the bottom plot shows the cumulative power spectral density. We observe that both operating systems produce nearly identical spectra for type I and type III tools on all three pairs of testbed machines (note: only M1 testbed machines as shown in Figure 7). Specifically, both Figure 7(a) and Figure 7(d) have a sharp peak at 15KHz and then we observe equally spaced harmonics at 30KHz and 45KHz. Figure 7(c) and Figure 7(f) have peaks at 50Hz and then harmonics at the remaining frequencies.

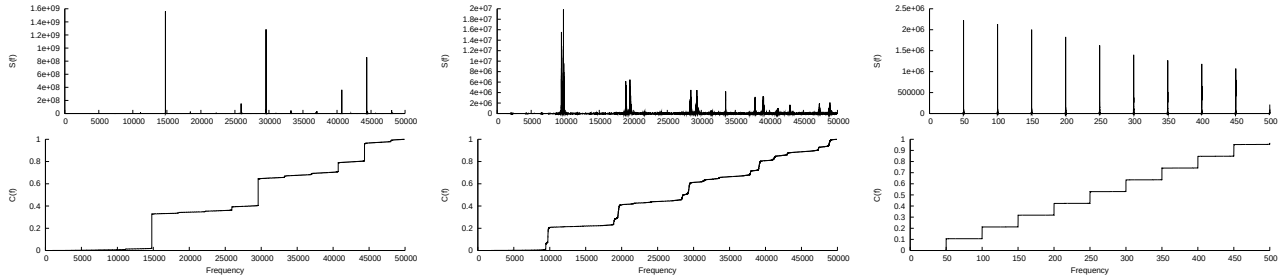
However, when comparing Figure 7(b) and Figure 7(e), we observe difference in the spectra for type II tools. We observed that type II tools generate packets at a slightly higher rate on FreeBSD (11Kpkts/sec) than Linux (9Kpkts/s) for M1 machines resulting in different spectra. For the other two sets 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 scenario signature. Table 6 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 $range_{FM1(II)LM1(II)}$ 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.

We believe that since testbed machines M1 are CPU bound, the operating system can influence the efficiency of packets generation and thus the attack spectrum created by the attack stream. We ex-



(a) FM1 Type I (b) FM1 Type II (c) FM1 Type III



(d) LM1 Type I (e) LM1 Type II (f) LM1 Type III

Figure 7: Effect of the operating system on the attack fingerprint

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)

Table 6: Effect of operating systems on the attack fingerprint

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 7: Effect of CPU speed on the attack fingerprint

plore this hypothesis in the next section.

5.4 Experimenting with different CPU speeds

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 signatures based on the speed of the CPU.

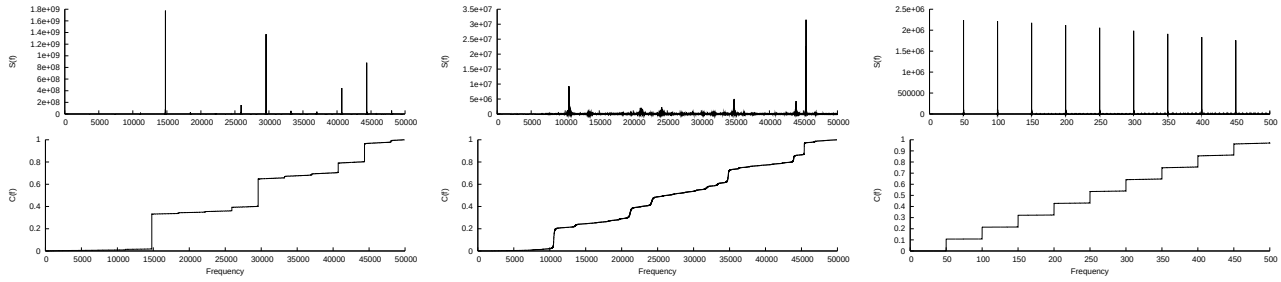
The results shown in Figure 8 compare all three attack tool categories on the slowest machines, FM1, against the fastest machines, FM3, on the testbed. The top plot show the power spectral density, while the bottom plot shows the cumulative power spectral density.

If the CPU speed did not matter, then we would observe no difference in all the spectra. However, when looking across Figure 8 we observe two things. First, type I and type III have identical spectra on both testbed machines indicating that the CPU speed does not alter the attack spectra significantly; Figure 8(a) and Figure 8(d) have a sharp peak at 15KHz and then we observe equally spaced harmonics at 30KHz and 45KHz. Figure 8(c) and Figure 8(f) have peaks at 50Hz and then harmonics at the remaining frequencies. Second, type II tools have different spectral behavior on FM1 machines compared to FM3. The Figure 8(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 (Figure 8(e)) that has a sharp peak at 15KHz. These two behaviors combined indicate that the CPU speed affects the spectrum if the attack tool is limited by the computation power of the CPU.

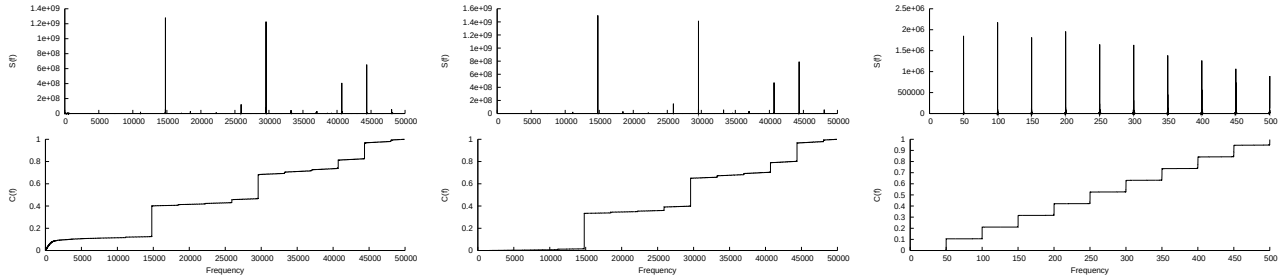
Table 7 summarizes the results of the effects of the CPU on the attack fingerprint. Each entry in the table indicates the comparison of the attack spectrum using the attack tool on FM1 as compared to FM2 and FM3 respectively. The results are similar when machines LM1, LM2, and LM3 are compared. Observe that the type II tools have large $range$ values indicating a poor match.

5.5 Experimenting with 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 5.8), is ephemeral (since it changes with time) and thus ideally should not contribute to the attack signature. Our results indicate that the proposed algorithms are robust to changes in the attack fingerprint due to host load.



(a) FM1 Type I (b) FM1 Type II (c) FM1 Type III



(d) FM3 Type I (e) FM3 Type II (f) FM3 Type III

Figure 8: Effect of CPU on the attack fingerprint

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 [22]. 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 5, 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 Figure 9 we compare the attack spectral fingerprints for type I tools on the Linux machines with and without 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 are more sensitive to 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.

Looking across Figure 9, we observe all the testbed machines have the same dominant frequency at 15KHz for both no load and load conditions. However, observe that the addition of host load increases the power in in low frequencies by about 10%, clearly visible in the cumulative spectrum. Although, the load changes the lower frequency content, it does not add any dominant frequencies and therefore the spectral signature is stable.

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)

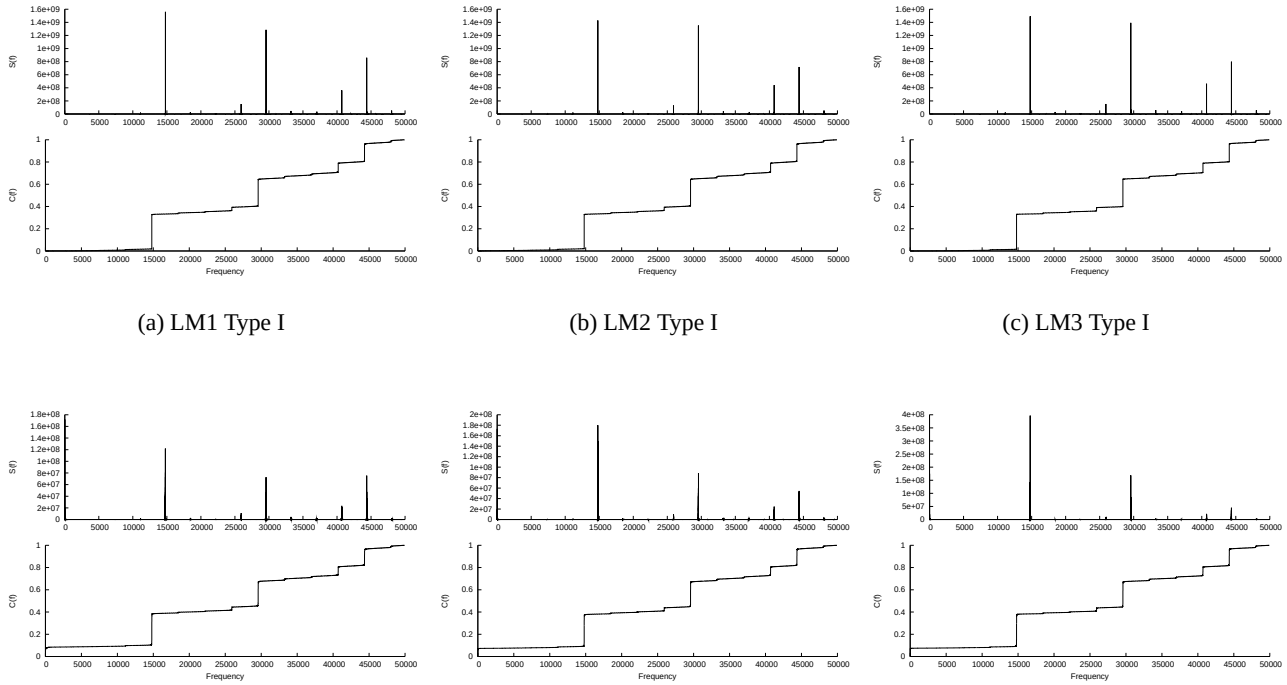
Table 8: Effect of load on spectrum

The above observation indicates that although the load reduces the overall packet rate by the attack tool our technique for generating spectral signature is robust to load and thus can be used to identify repeated attacks. Table 8 summarizes the quality of the signature matches under load conditions. The entries in the table match the spectral signatures 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.

5.6 Experimenting with different 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, and host load. Therefore, we know that the attack tools spectral signature does not survive in the packet stream in all cases partially answering the above question. In this section, we present results that indicate that



(a) LM1 Type I (b) LM2 Type I (c) LM3 Type I

(d) LM1 Type I and Load (e) LM2 Type I and Load (f) LM3 Type I and Load

Figure 9: Effect of host load on the attack spectra

the attack tool defines the spectrum provided the attack tool is not limited by any other resource.

Referring to Figure 7 and Figure 8 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 signature that is a function of how long the tool waits between two packets. Therefore the graphs lead us to believe that as long as the attack tool is not limited by the available resources, the attack signature will survive.

5.7 Experimenting with different packet sizes

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.

Figure 10 plots the attack spectra of type I tools on both FreeBSD and Linux machines. The increase in packet size resulted in the dominant frequency to move from 31KHz to 2.1KHz supporting our hypothesis that a saturated network can completely alter the attack spectrum.

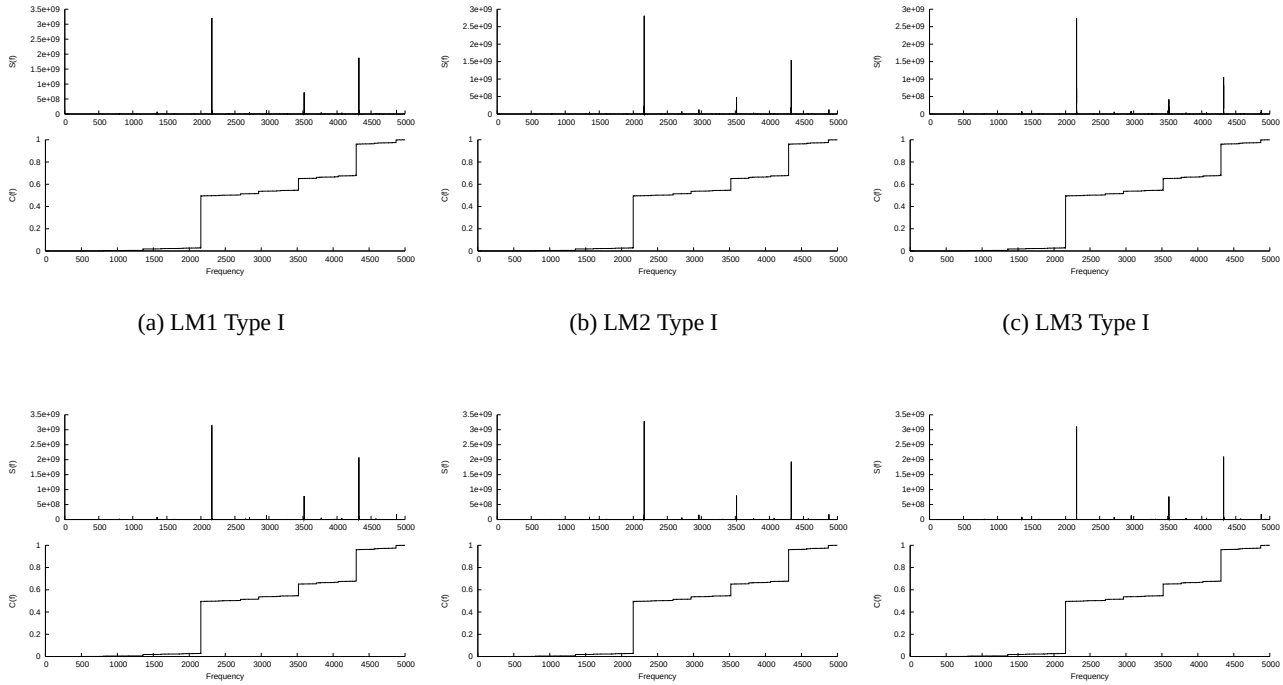
Further, since the packet size is large in this set of experiments, the attack spectra are not susceptible to host load. However, the Type II tools on the other hand can generate more packets when there is extra computational resources available, thus when load is added the attack rate reduces. Type 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.

5.8 Experimenting with network cross-traffi c

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 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 [6]. 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



(a) LM1 Type I (b) LM2 Type I (c) LM3 Type I

(d) M1 Type I (e) M2 Type I (f) M3 Type I

Figure 10: Effect of packet size on the attack spectra

its effect on the attack spectral signature. 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 signature.

In Figure 11 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. Once the cross-traffic increases to 60% of the traffic on the network, the signature shifts to a sharp peak at 32KHz. 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.

The battery of experiments suggest that the spectral fingerprint is defined by the attack tool and can be altered by the following components: operating system, the host speed and load, and the network capacity and cross-traffic. The operating system and host speed alters the spectrum only in type II tools. Although the host load increases the energy in the lower frequencies, it does not change the attack fingerprint and therefore gives good matches. The network influences the fingerprint completely when it is saturated. Thus when the cross-traffic is increased to the network capacity then the fingerprint is dominated by the frequencies present in the cross-traffic. All the experiments thus collectively support our hypothesis that the attack scenario is primarily defined by the attacker host and the attack tool.

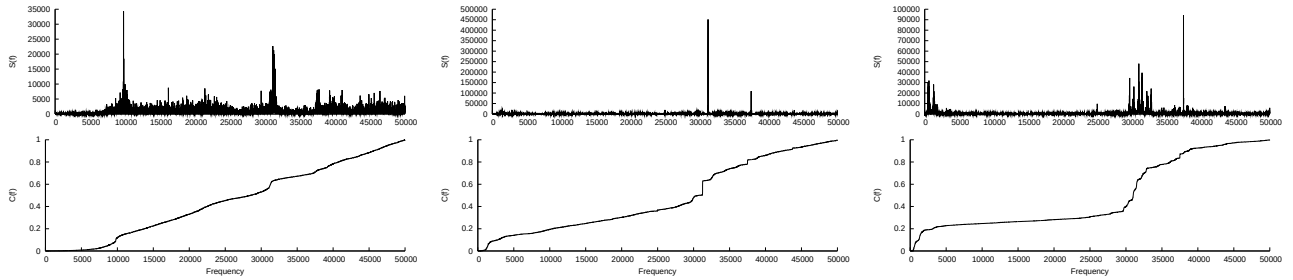
6. FUTURE WORK

Our system uses statistical techniques and pattern matching to identify repeated attacks. As such the quality if its results depend algorithm parameters and environmental factors. Although we have explored sensitivity of the results to many factors above, several areas remain.

Number of features: The success of the matching algorithm proposed in Section 3.3 depends largely on the feature data. In this paper, we use dominant twenty spectral frequencies as the features that identify the attack fingerprint. We have a preliminary exploration of varying the number of frequencies used as features, finding that we will get more accurate match values are the feature set increases and the converse is true for a smaller feature set. We tested our algorithm with feature sets of 5 and 30 frequencies on the testbed attacks and obtained varying match results. The top 5 feature produced significantly lower quality matches (evaluated using precision and accuracy statistics) while the top 30 frequencies did not improve the quality of our matches. The current approach of using the dominant twenty frequencies seems to capture most of the important features in the attack spectra. 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 could group adjacent frequencies as one feature. This approach may be more robust to noisy data.

Alternate feature definitions: Rather than simply varying the number of frequencies, alternative definitions of what a feature is may be helpful. Other features might include the complete spectra, certain fields of the packet header, or inter-arrival rate, could also be used to create unique fingerprints. These fingerprints may be more robust to single-source attacks, that our current technique cannot handle. Previous work has shown that usually there are multiple disjoint features that can be extracted from the data. For example, there are six popular feature sets for 0–9 handwritten numerals [11]. Exploration of other kinds of features is an area for future work.



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

Figure 11: Effect of cross traffic on the attack spectra

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, most "interesting" behavior will be at high frequencies. Sampling at a higher rate may improve identification of such attacks.

Temporal stability: Another important research question we need to explore when creating a 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 quantify 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.

Other applications: In the past, studies have been done to quantify attack prevalence on the Internet. Moore et al. used backscatter analysis and detected 12,805 attacks during a period of 3 weeks [16]. However, there are no studies on how many attacks are launched from the same attack troops and are indeed repeated attacks. As our database of attacks grow, we would like to analyze how many repeated attacks occur on the Internet every day. This study would also help understand independence and likelihood of repeated attack occurrences (Section 3.3). Such a study could maybe provide insight into attacker motivation and psyche.

Further, we would also like to explore how this research be applied to different network traffic problems. Such techniques may be applicable to other systems with regular traffic patterns. In Section 2, we list other efforts that analyze network traffic applying pattern recognition techniques. We believe the application of pattern recognition to network traffic is a new area of research that can provide valuable insights into network traffic behavior.

7. CONCLUSION

In this paper we presented techniques to identify attack streams in network traffic. We developed an attack fingerprinting system to identify instances of repeated attack scenarios on the network. We applied unique 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.

We also performed a systematic experimental study of the 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 completely when it is saturated. We believe such a system would enhance network traffic forensic capabilities and aid in investigating and establishing attribution of the DoS attacks seen on the Internet.

Acknowledgements

We would like to thank Jim Pepin, Walter Prue and Sanford George of Los Nettos, and Brian Yamaguchi of USC, for helping setup the trace machines and discussions about handling DDoS attacks. We would also like to thank Xinming He and Ramachandran Ramani for their discussions about spectral analysis.

8. 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] George Box, Gwilym Jenkins, and Gregory Reinsel. *Time series analysis: forecasting and control*. Prentice-Hall, Upper Saddle River, New Jersey, 1994.

[3] Ronald Bracewell. *The Fourier Transform and Its Applications*. Series in Electrical Engineering. McGraw-Hill, New York, NY, 1986.

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

[5] 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.

[6] K 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.

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

[8] Thomer M. Gil and Massimiliano Poletto. MULTOPS: A Data-Structure for bandwidth attack detection. In *Proceedings of the USENIX Security Symposium*, pages 23–38, Washington, DC, July 2001.

[9] Xinming He, Christos Papadopoulos, John Heidemann, and Alefiya Hussain. Spectral characteristics of saturated links. Under Submission.

[10] 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.

[11] 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.

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

[13] Los nettos-passing packets since 1988. http://www.ln.net.

[14] 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.

[15] Jelena Mirkovic, Greg Prier, and Peter Reiher. Attacking DDoS at the source. In *Proceedings of the IEEE International Conference on Network Protocols*, Paris, France, November 2002.

[16] 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.

[17] 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.

[18] 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.

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

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

[21] 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.

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

[23] 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.

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

[25] Haining Wang, Danlu Zhang, and Kang Shin. Detecting SYN flooding attacks. In *Proceedings of the IEEE Infocom*, New York, NY, June 2002. IEEE.

[26] E. Zwicky, S. Cooper, D. Chapman, and D.Ru. *Building Internet Firewalls*. 2nd Edition. O'Reilly and Associates, 2000.