A Game-Theoretic Framework for Bandwidth Attacks and Statistical Defenses

Mark E. Snyder
Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65409, USA
Email: mark.snyder@umr.edu

Ravi Sundaram
Department of Computer and Information Science
Northeastern University
Boston, MA 02115, USA
Email: koods@ccs.neu.edu

Mayur Thakur
Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65409, USA
Email: thakurk@umr.edu

***Abstract*—We introduce a game-theoretic framework for reasoning about bandwidth attacks, a common form of distributed denial of service (DDoS) attacks. In particular, our *traffic injection game* models the attacker as a rational but limited-resource entity who uses limited knowledge of traffic patterns to launch IP spoofing based bandwidth attacks on a server. We model the defender as a coarse-grained, relative volume based statistical filter.**

We analyze the effectiveness of the defender against the attacker by analyzing the payoffs of various strategies in the traffic injection game. Furthermore, we analyze how these payoffs change in the presence of random noise.

Our results show that there is potential for using statistical methods for creating defense mechanisms that can detect a DDoS attack and that even when an attacker has a priori knowledge of the expected traffic volume for the dimension and divisions employed in the attack, the attack traffic can still be exposed to the defender.

I. Introduction

IP spoofing—changing the source address of IP packets—has been used in DDoS attacks on popular websites (Yahoo!) and root DNS servers. Examples of DDoS attacks that use IP spoofing are Smurf and SYN attacks. IP spoofing-based attacks are dangerous because they can bypass filters that drop packets based on IP addresses. Even though techniques such as backscatter can detect IP spoofing-based attacks, one needs more sophisticated filtering mechanisms to detect more intelligent attacks. However, these filtering mechanisms cannot be too fine-grained because they have to be installed on routers. Also, simple volume-based detection mechanisms are unlikely to be effective because of the busyness of the Internet traffic. We thus study the effectiveness of *relative volume* and *coarse-grained* measures against IP spoofing attacks.

During a Distributed Denial of Service (DDoS) attack, the attacker increases the amount of illegitimate traffic originating from machines under his control. This results in a positive increase by some ratio relative to the traffic that was present in the system to begin with. We will refer to this ratio as α, where $0 \leq \alpha \leq 1$.

We view the attacker as choosing a distribution by which traffic is added to the system by the machines under his control. The attacker can view this from a number of directions, which we will call dimensions. For example, an attacker might choose to have all controlled machines use only spoofed IP addresses that are even-numbered. In this case, divisibility by 2 becomes a dimension, and there are 2 divisions within this dimension, even and odd. Traffic can be increased using a geographical distribution. The attacker may decide to increase traffic only to target machines in New York, or to add twice as much traffic coming from New York machines as from Los Angeles machines. Thus the geographical location of machines becomes a dimension and there could be numerous divisions within this dimension. The attacker is able to choose a distribution across the divisions of any dimension he envisions. In any set the number of groupings is only limited by the power set, which, while very large is still finite. In each case, the attacker identifies a dimension and then chooses a distribution across the divisions within that dimension.

The defender views the traffic in much the same way. As traffic is received, the defender counts it and monitors how the traffic is distributed amongst the divisions of some dimension. If the defender can detect a change from the base distribution, then it knows that an attack may be underway. The challenge for the defender, however, is that even though it can detect that an attack may be happening, there is still the matter of sorting the legitimate requests from the illegitimate. The solution that we will propose is to merely tag each division within the observed dimension with a measurement of suspiciousness. The more suspicious a group of traffic is, the more scrutiny it will be subjected to and thus the longer it will take to process, but it will not be dropped. Whereas traffic that is not suspicious is not delayed at all. Therefore, the defender in our game must only identify the groups of traffic that are more suspicious than other groups of traffic.

This is a very different approach than, say, that employed by synkill[8] where the goal is to spend as few resources as possible on suspicious traffic, but our approach is meant as a first line filter to incoming traffic, and the goal is to advise subsequent processes about which traffic merits additional scrutiny using more definitive techniques.

Since neither the attacker nor defender has perfect information about all dimensions and divisions within those dimensions, we define a "hidden" distribution of traffic to represent these hidden dimensions. We take care to note that our model is equivalent to one that allows the attacker to

control, defender to monitor, multiple dimensions as these multiple dimensions can be projected onto a new dimension so that it can be represented as a single dimension again. When an attacker adds traffic according to its chosen distribution, the hidden distribution is applied to the traffic across all other dimensions to produce the counts that the defender views from the perspective of his chosen dimension. Thus, even if the attacker is aware of or tries to approximate the base distribution with the attack distribution, the hidden distribution disseminates the traffic in such a way that the defender can still observe an anomaly with a careful choice of dimension to observe.

The Traffic Injection Game is a two person, zero sum, competitive game with imperfect knowledge. We represent the game in normal form as a sensitivity matrix in the form of a bimatrix of summed difference of new traffic to base traffic once the attack traffic is added [9].

Our game consists of two d-dimensional matrices—base distribution (representing the relative volume of traffic distribution) and a hidden distribution (representing the relative entity distribution). The attacker and defender (the two players in the game) each choose (independently) a dimension. We assume that the attacker and the defender both have knowledge of the base distribution along the dimension of their choice. The attacker chooses to inject traffic in the network (based on its knowledge of the distribution along a dimension), while the defender uses statistical means to measure the change in distribution from the normal distribution. The change in distribution per unit of attack traffic injected is the payoff to the defender.

We show that knowledge of traffic distribution helps both the attacker and the defender. When an attacker chooses a strategy (the amount of attack traffic that it generates per division of the attack dimension) that closely matches the base distribution for the attack dimension, then the payoff is best for the attacker. As the strategy deviates more and more from the base distribution, the attack traffic becomes more exposed to the defender.

We also show that even when the attacker knows the base distribution for the attack dimension and utilizes it to its best advantage, the attack is still detectable, given favorable signal-to-noise ratio and a careful choice of dimension by the defender in which to monitor traffic.

II. Related Work

A variety of methods of dealing with the problem of denial of service (DoS) attacks have been explored. The idea of comparing incoming to outgoing flows of network traffic was explored in [7]. Using game theory to identify another kind of DoS attack, that attempted by nodes in wireless sensor networks, was investigated in [1]. Cominetti, et al., examined the cost of anarchy in network games[2], following work by Roughgarden and Tardos[6]. Many of the defense mechanisms that have been proposed to combat these attacks are described, and advantages and disadvantages of each proposed scheme are outlined in [3]. In [4] techniques to thwart IP address spoofing in DNS DDoS attacks are detailed in the form of a firewall process, but this work also highlights the disadvantage to such definitive techniques, in that they have the potential to create a bottleneck themselves. The danger of indiscriminately dropping suspected traffic is described in [5]. They compare DDoS attacks to *flash crowds* which look similar to a DDoS attack but constitute legitimate traffic. Schuba, et al., contributed a detailed analysis of the SYN flooding attack and a discussion of existing and proposed countermeasures[8]. Calculating Nash equibria in bimatrix games in normal form was detailed by von Stengel[9].

III. Traffic Injection Game

We now define a game which would allow us to analyze IP-spoofing based bandwidth attacks. Our game is a 2-player game. We call the two players *attacker* and *defender*. We will refer to the server under consideration as *the server*. The attacker's motive is to attack the server using an IP-spoofing based bandwidth attack, while the defender's motive is to detect these attacks using coarse-grained statistical filtering.

A. Base and Hidden Distributions

The coarse granularity of filtering is modeled using a set of *dimensions*. Each dimension is a partition of the IP address space, and a corresponding partition of the total *typical* IP traffic reaching the server. We call the former the *hidden distribution* and the latter the *base distribution*. (The reason for these names will become clear shortly.) For example, if our dimension is the "autonomous system" (AS) dimension, then the ith component of the hidden distribution is the fraction of IP addresses in the ith AS, and the ith component of the base distribution is the fraction of traffic reaching the server that originates in the ith AS.

First, we need to model typical (or *base*) traffic, which we do by defining an n-dimensional matrix of traffic values T. Let d_i denote the number of divisions (or buckets) along the ith dimension. Each value in this matrix represents a fraction of the overall traffic. Thus, the sum of all entries in T is 1.

We also define matrix H as what we call the *hidden distribution*. Each cell in H represents a fraction of traffic generating entities (IP addresses). The attacker specifies how traffic is spread across the divisions of the attack dimension, but for all other dimensions, the hidden dimension is used to distribute attack traffic. The matrices T and H represent the board setup for a single game.

Example 1. *Say there are two dimensions* 1 *and* 2 *(each with 2 divisions). The base and hidden distributions can each be represented as* 2×2 *matrices. Let the base distribution be*

$$T = \begin{bmatrix} 0.1 & 0.3 \\ 0.4 & 0.2 \end{bmatrix}.$$

Note, for example, that the amount of traffic reaching the server from IP addresses that fall in division 1 *of the first dimension (rows) and division* 2 *of the second dimension*

(columns) is 0.3 *(*$= T[1,2]$*) fraction of the total traffic reaching the server. Let the hidden distribution be*

$$H = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix}.$$

Note, for example, that the number of IP addresses that fall in division 1 *of the first dimension and division* 2 *of the second dimension is* 0.2 *(*$= H[1,2]$*).*

B. Attacker's and Defender's Knowledge

We will assume limited knowledge for both the attacker and the defender. The defender can know the typical (non-attack) distribution for each IP address. However, for two reasons the defender cannot apply statistical filtering at this fine-grained level. First, the amount of noise at this fine-grained level is significant. Second, even if the noise was negligible, there is a significant price to be paid for implementing a statistical filter at this level. For example, in the simplest scheme, 2^{32} numbers (one per IP address) must be stored and looked up for each packet. For these reasons, we will assume that the defender uses only aggregate information. In particular, the defender uses the base distribution along one, say the jth dimension.

We will assume that the attacker has knowledge of one dimension, say the ith dimension. What this means is that the attacker knows how much traffic is typically generated by nodes in each division of dimension i, and the attacker knows the set of IP addresses in each *division* of dimension i. In particular, it can use this knowledge to inject traffic with spoofed IP addresses such that the distribution of traffic (or relative traffic) along dimension i does not change from its base distribution.

C. Injecting Traffic

Continuing with Example 1 above, say the attacker has a priori knowledge about the typical historical distribution for dimension 1. Suppose he decides to inject a total α amount of traffic (relative to the base traffic) with distribution $[\beta, 1-\beta]$ along dimension 1 (that is, the first division of dimension 1 gets $\alpha * \beta$ additional traffic, etc). How much traffic is injected in the cell $[1,1]$, for example? More formally, let i be the dimension that the attacker knows. Let δ_j be the amount of attack traffic that the attacker decides to add to division j of dimension i. Then we assume that the attacker uniformly spreads δ_j across all IP addresses that fall in division j of dimension i. This is a reasonable assumption because the attacker has no knowledge about the other dimensions. Thus, in our example above, cell $[1,1]$ will get $\alpha * \beta * \frac{0.3}{0.3+0.2}$ amount of additional traffic. After new values for each cell are calculated and the matrix is normalized, we calculate the sensitivity matrix.

D. Payoff: Sensitivity Matrix

We analyze the game by analyzing the payoff matrix of the game. Let the dimension that the attacker knows be i and let the dimension that the defender uses be j. The attacker's goal is to inject one unit of traffic such that the change in the distribution along dimension j is minimized. The defender wants to detect such a change. The payoff to the defender is the change in the distribution along the jth dimension if a unit amount of attack traffic is introduced using the ith dimension. We call this quantity *sensitivity to the* i*th dimension along the* j*th dimension* and denote it as $S[T, H, i, j]$. When T and H are clear from the context, we refer to $S[T, H, i, j]$ as $S[i, j]$. S is called the *sensitivity matrix*. Periodically we will also refer to simply the *sensitivity* of an experiment, in which case we mean the maximum sensitivity value of a sensitivity matrix. This value represents the best payoff for the defender. Since the game is a zero-sum game, the payoff to the attacker is the negation of the payoff to the defender.

It is easy to see that in general the sensitivity function is asymmetric with respect to dimensions. That is, for a given attack A, and dimensions i and j, $S[i, j]$ is not necessarily equal to $S[j, i]$.

E. Effect of Injecting Attack Traffic

Given the base and hidden matrices, we now show how to analytically compute the effect of injecting attack traffic. As a special case we get how to compute the sensitivity matrix.

Let T be the base distribution and let H be the hidden distribution on n dimensions such that dimension i has d_i divisions. Let α be the amount of traffic to be injected relative to the total amount of traffic represented by the base distribution. Let q be the attacker's dimension and let r be the defender's dimension. Let A be a d_q-dimensional vector representing the attack traffic distribution. (The rth entry in A denotes the fraction of attack traffic that is added to the rth division of dimension q.)

Let $H[1\!:\!i_1, 2\!:\!i_2, \ldots, n\!:\!i_n]$ denote the cell of H whose label along dimension j is i_j, for each $j \in \{1, 2, \ldots, n\}$. Let $H[k\!:\!\ell]$ denote an $n-1$ dimensional matrix H' with dimensions $\{1, 2, \ldots, k-1, k+1, \ldots n\}$ such that the $H'[1\!:\!i_1, 2\!:\!i_2, \ldots, k-1\!:\!i_{k-1}, k+1\!:\!i_{k+1}\!:\!i_n]$ is $H[1\!:\!i_1, 2\!:\!i_2, \ldots, k-1\!:\!i_{k-1}, k\!:\!\ell, k+1\!:\!i_{k+1}, \ldots, n\!:\!i_n]$. Thus, informally speaking, $H[k\!:\!\ell]$ denotes a $n-1$ dimensional *slice* of H containing all cells for which the dimension k label is ℓ. Note that $0 \le i_j < d_i$, for each dimension i in H.

We also define operator $H_{|j}$ (the *projection of* H *along dimension* j) as a vector of size d_i such that the i-th component is the *sum* of entries in the matrix $H[i\!:\!j]$. Next, we define the operator $agg(H)$ which is the set of all n sum vectors of H, where vector i of $agg(H)$ is $H_{|j}$ and has d_i values.

We calculate the fraction of attack traffic that will be added to a cell C labeled $[1\!:\!i_1, 2\!:\!i_2, \ldots, n\!:\!i_n]$:

$$\Delta[1\!:\!i_1, 2\!:\!i_2, \ldots, n\!:\!i_n] = A[i_q] \frac{H[1\!:\!i_1, 2\!:\!i_2, \ldots, n\!:\!i_n]}{agg(H[q\!:\!i_q])}$$

Note that the $A[i_q]$ term denotes the fraction of attack traffic added to the i_qth division of dimension q and the fractional term is the number of IP addresses that lie in cell C as a fraction of the number of IP addresses that lie in the i_qth division of dimension q. Note that $agg(\Delta) = 1$.

Finally, we can calculate the new base matrix T' that results when the attack traffic is added to the original base matrix T:

$$T' = \frac{T + \alpha\Delta}{1 + \alpha} \tag{1}$$

The new distribution along dimension r is $T'_{|r}$. The *effect of attack A volume α in the qth dimension as observed in the rth dimension* is

$$E(A, \alpha, q, r) = \sum_{1 \le i \le d_q} |T_{|r}[i] - T'_{|r}[i]|.$$

The *sensitivity of attack A in the qth dimension as observed in the r dimension* is defined as

$$S(A, q, r) = E(A, 1, q, r).$$

Note that the sensitivity is the effect of a unit attack traffic.

We show below how the effect of of an attack A with an arbitrary amount of traffic is related to the sensitivity of the same attack.

Lemma 2. *For each $\alpha \ge 0$, $E(A, \alpha, q, r) = \frac{2\alpha}{1+\alpha} S(A, q, r)$.*

Proof omitted due to space considerations.

F. Modeling Noise

The base distribution represents a historical pattern developed from a snapshot of traffic, and the traffic injection game is played based on something assumed to resemble this historical pattern, thus we need to model the effect that noise has on the game and calculate the new traffic matrix T' that results when the noise and attack traffic are added to the original base matrix T.

Let N be the noise distribution on n dimensions such that dimension i has d_i divisions. Thus the noise distribution is the same shape as the base and hidden distributions. We will consider the noise as a normal distribution that is independent of the base distribution, and we define ν as the noise level relative to the base traffic volume.

Using this definition of noise, we can now calculate the new traffic matrix T' as:

$$T' = \frac{T + \nu N + \alpha\Delta}{1 + \nu + \alpha} \tag{2}$$

If we set $\nu = 0$ then Equation 2 is equivalent to Equation 1.

G. Metrics

The sum of the positive differences between the new volume and the base volume when the volumes are projected onto each dimension constitute a sensitivity measurement for a given defense dimension. A table of sensitivity measurements by attack dimension and defender dimension constitutes the $n \times n$ sensitivity matrix. From the sensitivity matrix, we calculate the six metrics described below.

We identified a number of aggregations we want to capture about this game. For a set of sensitivity matrices, we want to know what the max payoff (plus mean, stdev), min payoff (plus mean, stdev), maxmin payoff (plus mean, stdev), and minmax payoff (plus mean, stdev) are across all the cells of the sensitivity matrices. This should allow us to identify the expected payoff under various scenarios.

We have taken six key measurements from the sensitivity matrices generated by the game.

- *BIMA:* Best independent move for the attacker. This is the max-minimum value for any row in the sensitivity matrix.
- *BIMD:* Best independent move for the defender. This is the min-maximum value for any column in the sensitivity matrix.
- *AOS:* Average overall sensitivity. Simple average of all values in the sensitivity matrix.
- *AOP:* Average overall payoff. This is the average of the value at the intersection of the sensitivity matrix of the BIMA row and the BIMD column.
- *DDA:* Delta defection to best move for the attacker. Starting at the intersection of the BIMA row and BIMD column in the sensitivity matrix, find the lowest value in that column that the attacker could choose if he changed moves.
- *DDD:* Delta defection to best move for the defender. Starting at the intersection of the BIMA row and BIMD column in the sensitivity matrix, find the highest value in that row that the defender could choose if he changed moves.

H. Attack Types

We have analyzed four kinds of attacker:

- *Random:* The ratio of attack traffic for each division of the attack dimension is a randomly chosen normalized distribution.
- *Base:* Attack traffic is spread so that it matches the distribution for divisions in the base traffic distribution for the attack dimension.
- *Uniform:* Attack traffic is spread evenly amongst the divisions in the attack dimension. For example, if there are 10 divisions, each gets 10% of the attack traffic.
- *Loaded:* This kind of attacker directs all of the attack traffic at division 0 of the attack dimension.

For each of the four attacker strategies, our analysis consisted of generating many base and hidden distributions, and for each of these an attack distribution and noise distribution are generated many times. For each of these, a sensitivity matrix was generated so that we could analyze all possible moves for attacker and defender. The results were aggregated and the average and standard deviation reported by the program.

I. Equilibrium and Distribution Difference Analysis

One important aspect of the the traffic injection game to discuss is the equilibrium of the game. Assuming full information and no noise given that there are n dimensions or actions for each party then it is easy to go exhaustively over all n^2 possibilities to check for pure Nash Equilibrium (NE) in polynomial time. Mixed NE can be computed exactly using linear programming. However our goal here is not to solve for NE since that assumes full information and in reality the

attacker and defender may be unaware of H, T, etc. We are after an understanding in the imperfect information model, so we study how the sensitivity varies with the *difference* between the distributions in our model. There are two basic types of distributional differences:

Interdimensional distributional differences occur between distributions along two dimensions of the base distribution or they occur between distributions along two dimensions of the hidden distribution. Intuitively speaking, if there is a large interdimensional difference between the attacker's and defender's dimensions, then the sensitivity is high.

Base-Hidden distributional differences occur between two distributions—a projection along a dimension of the base distribution and a projection along a dimension of the hidden distribution.

Because of the complex relationship between the base and the hidden distributions, a complete analysis of these differences and the equilibrium of the game are outside the scope of this paper (and is left as future work). However, we present an analysis of a special case of base-hidden difference. In particular, we study the change in sensitivity in the following case. We are given a base distribution $\hat{T}$. The attacker's dimension is 1 and the defender's dimension is 2. (Note that this is not a restriction because we can rename dimensions.) We want to find hidden distributions H such that the sensitivity is maximized and the following hold for each dimension $j \neq 2$:

$$H_{|j} = \hat{T}_{|j}.$$

That is, the base and hidden distributions match in all dimensions except the defender's dimension.

This problem can be expressed as the following optimization problem:

$$\max S[\hat{T}, H, 1, 2]$$
$$\text{s.t.}$$
$$H_{|j} = \hat{T}_{|j}, \quad \forall j = 1, 3, 4, \ldots, n.$$

Formally, a *distribution with k divisions* is a k-dimensional vector $[x_1, x_2, \ldots, x_k]$ such that each $0 \leq x_i \leq 1$ and $\sum_{i=1}^{k} x_i = 1$. The difference between two distributions $X = [x_1, x_2, \ldots, x_k]$ and $Y = [y_1, y_2, \ldots, y_k]$ (each with k divisions) is

$$\sum_{i=1}^{k} |x_i - y_i|.$$

We show in the results section that the optimal solution to this problem increases linearly with the difference between $T_{|2}$ and $H_{|2}$. Intuitively, it is safest for the attacker not to attack, if his goal is not to get detected. As the amount of attack traffic increases (as shown for the special case above), the higher the chance of the attacker being detected by the defender.

IV. TIGGER: Software

Traffic Injection Game Graphical Engine for Results (TIGGER) is a software tool developed for performing reproducible simulation and analysis required by this research. The parameters that are recognized by the program consist of the following:

- Dimensions: set of comma-delimited numbers indicating how many divisions make up each dimension. Example: "2,2", "10,10,10", "5,2,8*". Suffixing a number with an asterisk directs the values in that dimension to be generated using a power law distribution. Otherwise, the values are a randomly chosen, normalized distribution.
- Attacker type: Base, Random, Uniform, or Loaded.
- Noise Level: decimal number specifying the noise ratio relative to the base traffic. The base traffic level is always considered to be 1.0.
- Attack Traffic: decimal number specifying the ratio of attack traffic volume relative to the amount of base traffic.
- Random Seed: integer seed value (for reproducibility).

The program also supports iteration by repeating experiments for various values for noise level and attack traffic volume.

V. Results

Based on the results of the two previous examples, we can design a statistical filter that analyzes traffic patterns, identifies the division within the observed dimension that represents the most suspicious group of traffic, and recommend that group of traffic for more rigorous, albeit time-consuming, verification techniques. It is not necessary to limit the functioning of such a filter to just the most suspicious group of traffic. The filter could also be designed to send any group of traffic whose traffic increases above a certain threshold to be scrutinized more carefully.

The base traffic pattern in a real-life scenario would need to account for per hour patterns, since the dynamics of many dimensions from which a defender could observe will change based on the time of day as found in our research.

If the statistical filter were combined with a feedback loop that processes emerging traffic patterns back into the expected base traffic values, the benefit of the filter might even be able to be improved.

The rest of this section describe the results that our implementation of the traffic injection game produced, using various attacker types, noise levels and attack traffic volumes. This allows us to analyze the effect of such parameters on the 6 metrics of which we have been discussing.

A. Variation with Attack Traffic and Noise

The best independent move for the attacker (BIMA) is a maximin value that identifies the best that the attacker could do if the defender chose the worst dimension to watch based on the attacker's move. If the attacker knows the dimension that the defender will observe, then the attacker is always going to choose the dimension that generates the minimum sensitivity value for the defense dimension. This means that the defender's suspicion was minimally heightened.

Figure 1 shows the effect of varying the attack traffic ratio given a fixed level of noise (0.3), while Figure 2 shows how the

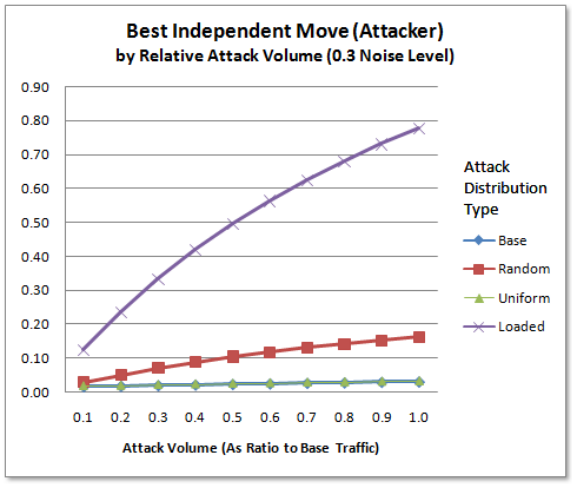

Fig. 1. Plot of best independent move for the attacker versus attack volume on a 10x10x10 matrix with 0.3 noise level.

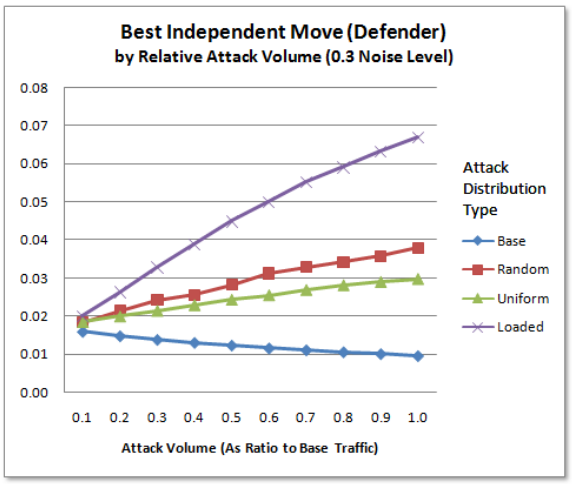

Fig. 3. Plot of best independent move for the attacker versus attack volume on a 10x10x10 matrix with 0.3 noise level.

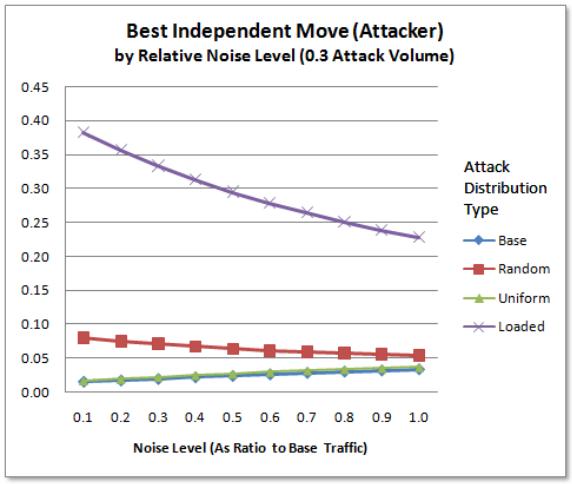

Fig. 2. Plot of best independent move for the attacker versus noise level on a 10x10x10 matrix with 0.3 attack volume.

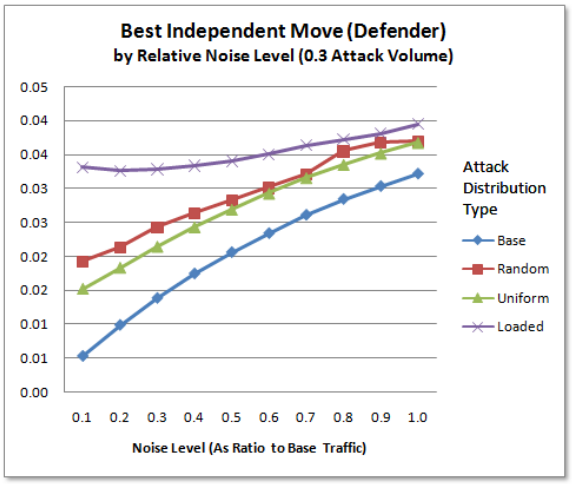

Fig. 4. Plot of best independent move for the attacker versus noise level on a 10x10x10 matrix with 0.3 attack volume.

BIMA is affected by fixing the attack traffic at 0.3 and varying the noise level. As we see, the best strategy for the attacker is observed when the attack traffic is distributed randomly or when the attacker is aware of and employs a distribution that matches the base distribution for the attack dimension. The random strategy does as well because of the stochastic nature of the generated distributions. The worst performance comes when the attacker blindly, evenly distributes traffic or blindly applies all its resources at a single distribution, illustrating that a powerful, wise attacker performs better.

Figure 1 also illustrates the fact that as the attack traffic increases, the uniform and loaded strategies fare worse as the attack traffic becomes more and more exposed relative to the base traffic and noise, while the attacker's best move gets no worse by increasing his attack volume. Figure 2 shows that as the noise level increases relative to the base and attack traffic, the poorer strategies are hidden a little better. This would support the idea that as observed traffic deviates more and more from established patterns, even bad attack strategies become harder to detect.

The best independent move for the defender is analyzed similarly, however in this case the values are calculated based on the best that the defender can do given that the attacker can choose the dimension that presents the worst max value for the defender, or a minimax value. As can be seen in Figure 3 and Figure 4, the strategies that worked well from the attacker's point of view also present the biggest problems for the defender, and vice-versa.

The average overall sensitivity (AOS) values, as seen in Figure 5 and Figure 6, is a metric of the best potential payoff that the defender could achieve in any dimension for a particular experiment. The base and uniform attacker types perform with this metric almost identically, with the figures isolating only these two. As the noise level and the attack volume increase, so does the average overall sensitivity measurement. This also shows that as an attacker, using knowledge about the expected

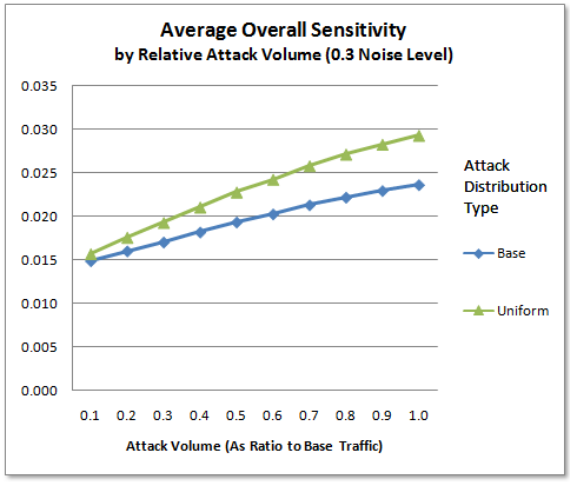

Fig. 5. Plot of average overall sensitivity versus attack volume on a 10x10x10 matrix with 0.3 noise level.

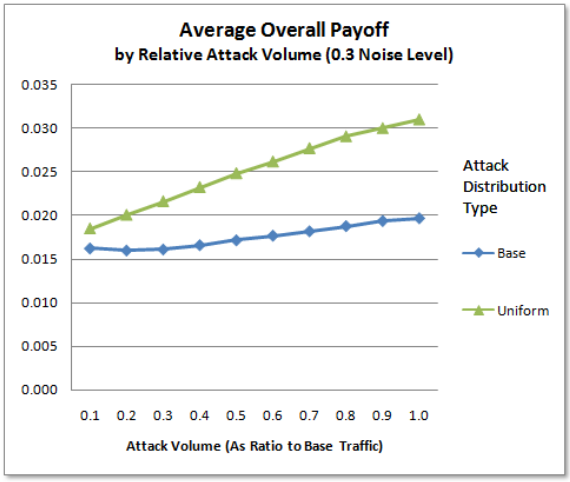

Fig. 7. Plot of average overall payoff versus attack volume on a 10x10x10 matrix with 0.3 noise level.

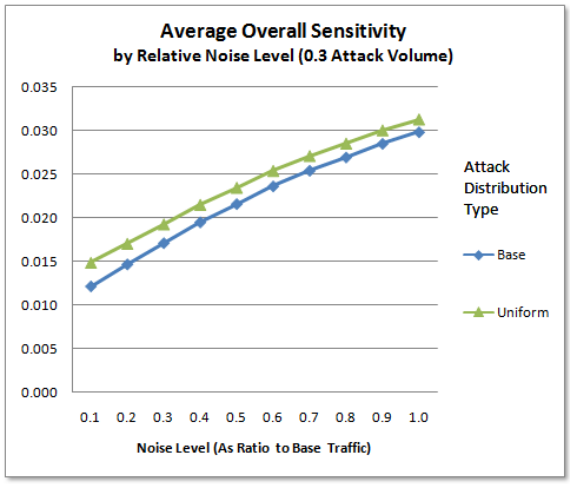

Fig. 6. Plot of average overall sensitivity versus noise level on a 10x10x10 matrix with 0.3 attack volume.

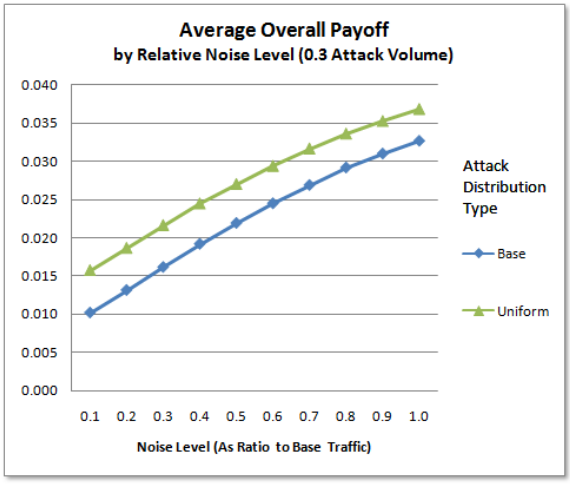

Fig. 8. Plot of average overall payoff versus noise level on a 10x10x10 matrix with 0.3 attack volume.

distribution of traffic in the dimension being attacked is a powerful tool for hiding attack traffic.

The average overall payoff (AOP) value, as seen in Figure 7 and Figure 8 shows what the average of the expected payoff would be, given that the attacker chooses the best independent move for the attacker, and the defender chooses the best independent move for the defender. The value in the sensitivity matrix at this junction is the overall payoff, and the AOP is the average value as measured in each experiment. As can be seen in the figures, the pattern followed is similar to the sensitivity for this game. As the noise level and the attack volume increase, so does the average overall payoff measurement. An attacker wishing to optimize this metric would be best served by a strategy taking advantage of knowledge about the expected distribution of traffic in the dimension being attacked.

Another conclusion illustrated by these results is that a defender that has knowledge of certain dimensions within the system that the attacker does not is able to better expose that an attack is occurring and from where that attack is coming. This fact is shown by examining the loaded attacker type. In this type of attack, the attacker is applying attack traffic to a single division within its attack dimension, and regardless which dimension the defender is watching, the attacker exposes its traffic the most of any of the attack strategies we examined.

B. Distribution Difference Analysis-Interdimensional

As discussed above in Section III-I, interesting things happen when we vary the distance between distributions. There are several components we may consider when analyzing the data in this way. First, we can vary the distance between the aggregate distributions for the divisions of two dimensions of the base distribution, fixing the attack distribution. We can also fix the base and attack distributions, and vary the distance between the base and hidden distributions.

First, we consider a game consisting of symmetric randomly chosen distributions, fix the attacker type as a Base attacker,

and then analyze the sensitivity matrices that result. As shown in Figure 9, the sensitivity grows as the distributions used by the attacker and defender approach zero. This means that the defender has chosen a dimension whose distribution perfectly isolates the attack traffic. As the two distributions diverge, the attack traffic becomes more hidden from the defender.

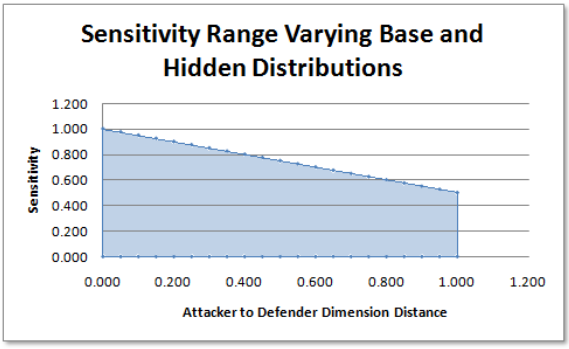

Fig. 9. Plot of sensitivity values as the distance between the attacker and defender dimensions vary.

C. Distribution Difference Analysis-Base to Hidden

When the base and hidden distributions coincide, we say the distance between the two distributions is zero. In this case, and when the attacker is allowed awareness of and he emulates the base distribution for the attack dimension, then the sensitivity matrix contains all zeroes. In other words, the attack is perfectly hidden from the defender. On the other hand, when the distance between the base and hidden distributions is the greatest, the sensitivity is maximal. These observations are illustrated in Figure 10 at the bottom left and top right edges of the shaded areas.

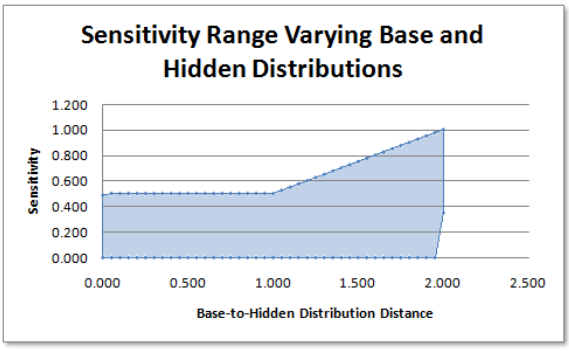

Fig. 10. Plot of max sensitivity values varying the distance between the attacker and defender dimension aggregate values.

One interesting aspect of our research is the fact that even when the attacker mimics the base distribution in the attack dimension, there are still many hidden distributions that result in an aggregate distribution in the defender's chosen dimension that match the base distribution in that dimension.

We next further examine what happens when we fix the base distribution and show the effect of changing the hidden distribution. As shown in Figure 11, varying the hidden distribution so that the distance between it and the fixed base distribution grows causes the sensitivity to increase.

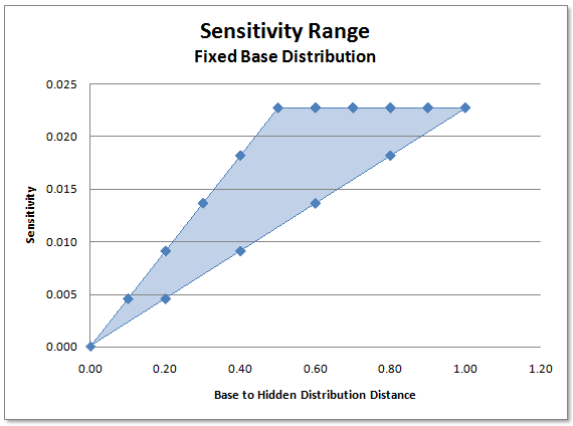

Fig. 11. Plot of sensitivity values for a fixed base distribution varying the hidden distribution.

VI. Conclusion

As we have shown, our research indicates that there is potential for using statistical methods such as ours for creating defense mechanisms that can detect a DDoS attack, given that favorable signal-to-noise ratio exists. Our work also shows that an attacker is more exposed as the frequency of attack traffic increases, but that an attacker that has a priori knowledge of the expected traffic volume for the dimension and divisions employed in the attack has the best ability to hide attack traffic when such statistical methods are used by the defender. Future work seeks to show similar results when real world distributions such as those modeled in a power law are integrated into the sensitivity matrix and employed by the attacker and defender.

References

[1] Afrand Agah, Mehran Asadi, and Sajal K. Das. Prevention of dos attack in sensor networks using repeated game theory. In *ICWN*, pages 29–36, 2006.

[2] Roberto Cominetti, José R. Correa, and Nicolás E. Stier Moses. Network games with atomic players. In *ICALP (1)*, pages 525–536, 2006.

[3] Christos Douligeris and Aikaterini Mitrokotsa. Ddos attacks and defense mechanisms: classification and state-of-the-art. *Comput. Networks*, 44(5):643–666, 2004.

[4] Tzi-cker Chiueh Fanglu Guo, Jiawu Chen. Spoof detection for preventing dos attacks against dns servers. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS'06)*, page 37, 2006.

[5] Fanglu Guo, Jiawu Chen, and Tzi cker Chiueh. Spoof detection for preventing dos attacks against dns servers. *icdcs*, 0:37, 2006.

[6] Tim Roughgarden and Eva Tardos. How bad is selfish routing? In *IEEE Symposium on Foundations of Computer Science*, pages 93–102, 2000.

[7] Abdelsayed S. Glimsholt D. Leckie C. Ryan S. Shami S. An efficient filter for denial-of-service bandwidth attacks. In *Proc. IEEE GLOBECOM '03.*, volume 3, pages 1353 – 1357, December 2003.

[8] Christoph L. Schuba, Ivan V. Krsul, Markus G. Kuhn, Eugene H. Spafford, Aurobindo Sundaram, and Diego Zamboni. Analysis of a denial of service attack on TCP. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 208–223. IEEE Computer Society, IEEE Computer Society Press, May 1997.

[9] B. von Stengel. Computing equilibria for two-person games.