Hindawi
Security and Communication Networks
Volume 2018, Article ID 9463653, 8 pages
https://doi.org/10.1155/2018/9463653

Research Article

A Classification Detection Algorithm Based on Joint Entropy Vector against Application-Layer DDoS Attack

Yuntao Zhao,[1,2] **Wenbo Zhang**,[1] **Yongxin Feng**,[1] **and Bo Yu**[1]

[1]*School of Information Science and Engineering, Shenyang Ligong University, Shenyang 110159, China*
[2]*College of Information Science and Engineering, Northeaster University, Shenyang 110819, China*

Correspondence should be addressed to Yuntao Zhao; zhaoyuntao2014@163.com

Received 12 September 2017; Revised 22 February 2018; Accepted 28 March 2018; Published 3 May 2018

Academic Editor: Jesús Díaz-Verdejo

The application-layer distributed denial of service (AL-DDoS) attack makes a great threat against cyberspace security. The attack detection is an important part of the security protection, which provides effective support for defense system through the rapid and accurate identification of attacks. According to the attacker's different URL of the Web service, the AL-DDoS attack is divided into three categories, including a random URL attack and a fixed and a traverse one. In order to realize identification of attacks, a mapping matrix of the joint entropy vector is constructed. By defining and computing the value of EUPI and jEIPU, a visual coordinate discrimination diagram of entropy vector is proposed, which also realizes data dimension reduction from N to two. In terms of boundary discrimination and the region where the entropy vectors fall in, the class of AL-DDoS attack can be distinguished. Through the study of training data set and classification, the results show that the novel algorithm can effectively distinguish the web server DDoS attack from normal burst traffic.

1. Introduction

Distributed denial of service (DDoS) attack [1–3] is one of the most serious threats to today's networks and has aroused great concern in various countries around the world [4, 5]. DDoS attack refers to consumption of the victim server resource and keeping the targets from providing services for legitimate users. DDoS attack is categorized into two classes: network-layer DDoS (NL-DDoS) attack and application-layer DDoS (AL-DDoS) [6, 7]. The early DDoS attack was a network-layer attack. In NL-DDoS, attackers send a large number of bogus packets towards the victim host with vulnerability exploitation that exists only on the network and transport layer. For example, IP spoofing uses fake connections to quickly consume the server's bandwidth and hide its location; SYN Flood attackers keep sending unused connections requests with only SYN flags to the server, which would exhaust the bandwidth and resources of the server on a massive number of TCP half-connections. In NL-DDoS attack, the victim server or IDS can easily distinguish legitimate packets from DDoS packets [8]. In contrast, in AL-DDoS, perpetrators attack the victim server through a flood of legitimate requests. AL-DDoS attack does not saturate the bandwidth of the victim server through inbound traffic but through outbound traffic. Because AL-DDoSs behave very much like flash crowd, a legitimate behavior where a very large number of users simultaneously access a website, it is not easy to distinguish them. Consequently, due to universality and variety of web service in application layer, AL-DDoS may be stealthier and more dangerous than the traditional NL-DDoS attack.

Considering that impacts of the AL-DDoS attacks are becoming great, researchers at home and abroad have done a lot of related work in this field. Jung et al. [9] deeply analyzed the difference between the AL-DDoS and the flash crowd. When flash crowd occurs, a large number of address clusters recur, while a large number of new address clusters will appear with AL-DDoS attacks. The distribution of access addresses of flash crowd is uneven, while the distribution of access addresses is more uniform when DDoS attacks. Li and others [10] use a mixed measure to detect the shifting of the flow distribution, thus distinguishing between DDoS attacks and flash crowd. Yu and others [11] use the Sibson distance to measure the similarity between the flows and realize the distinction between the DDoS attack flow and the flash crowd flow. Oikonomou and Mirkovic [12] built a normal behavior model to distinguish between attackers and normal visitors. Lee et al. [13] proposed a detection algorithm of AL-DDoS attack based on information entropy. According to the information entropy of URL access rate, the algorithm can detect DDoS attack, but can not distinguish between DDoS attacks and flash crowd. Rathika et al. [14] put forward a method to detect attacks based on the average number of requests per unit of time for each session (ANRS). When the DDoS attack flow is small and occurred in the low-rate, the value of ANRS neither rises significantly nor descends. Xie and Yu [15] simulated the users' access and page request behaviors with Markov chain model. According to the jump probability of each session, the behaviors model takes the degree of deviation as detection indicator.

In this paper, based on the characteristics of user access behavior in application layer, the attacks are classified in terms of access mode of URL. The matrix from IP to URL maps a joint entropy vector, which realizes data dimension reduction. Through defining and computing the EUPI and jEIPU, the coordinate discrimination diagram of entropy vector is constructed. Also by the region where the entropy vector falls in, the type of AL-DDoS attack can be discriminated. The simulation experiment shows that the algorithm can effectively distinguish between DDoS attacks from normal traffic.

2. Behavior of AL-DDoS Attack

On the web, user behaviors of accessing to a URL include three steps: visiting, lingering, and abandoning. Because different users are interested in the different content and pages, the legitimate access behaviors from users' IP address to URL are random. In contrast, AL-DDoS attacks are usually launched by a specific tool or bot-nets, which make those collaboration-based behaviors more regular and nonrandomized.

According to the attacker's selection of attacked URL, the AL-DDoS attack is divided into three categories. The first category is a fixed URL attack, in which the attacker initiates and determines one or a few of URLs. In order to achieve a better attack effect, the attacker often chooses to download a large picture or file request, which is the most common and easy to implement. For example, the SOAP replay attackers [16] send the soap request message repeatedly to a fixed URL, which exhaust the source of the victim server through outbound traffic.

The second category is a random URL attack, which scans the attack list of site and randomly selects URLs in every attack. The random URL attacks masquerade as normal web access behaviors in a low-density manner. It outwardly likes that a very large number of users simultaneously access few popular websites. Therefore, the random URL attack is stealthier than the fixed URL attack.

The third category is a traversal URL attack, which is similar to the web crawler. This mode of attack is in the form of web crawler, grabbing URL and selecting the URL request. The attacker starts from the home page URL and selects a URL as the next request. Then the process is repeated and cycled until no new URL is obtained.

3. A Novel Detection Algorithm against AL-DDoS

3.1. Mapping from IP to URL. For the websites, users' traffic that converges to a URL is a stream of successive URL requests. The number of requests belonging to the kth URL from the ith source IP address in a constant time window (interval) is $x_i^{(k)}$. The matrix $[x_i^{(k)}]_{N\times K}$ from source IP to URL is the map of web access. Let i be equal to N and k be equal to K. The map from IP to URL is shown in Table 1.

Thereinto, $x_i^{(k)}$ is element of the matrix in k row and i column:

$$x_i \in X = \{x_1, x_2, \ldots, x_N\}, \tag{1}$$

where $x_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(K)}) \in R^K$ and x_i is K dimension vector.

3.2. Definition of EUPI and EIPU. The entropy of URL request per IP address (EUPI) is defined as

$$\begin{aligned}\text{EUPI}\left(i \mid x_i^{(k)} : x_i \in R^K\right) \\ = -\sum_{k=1}^{K} P_i^U(x_i)\log\left[P_i^U(x_i)\right],\end{aligned} \tag{2}$$

where $P_i^U(x_i)$ is the probability of kth URL request per source IP address. $P_i^U(x_i)$ indicates the percentage of different URL requests that are accessed by one IP address:

$$P_i^U(x_i) = \frac{x_i^{(k)}}{\sum_{k=1}^{K} x_i^{(k)}}. \tag{3}$$

The entropy of IP address per URL (EIPU) is defined as

$$\text{EIPU}\left(k \mid x_i^{(k)} : x_i \in R^K\right) = -\sum_{i=1}^{N} P_i^I(x_i)\log\left[P_i^I(x_i)\right], \tag{4}$$

where $P_k^I(x_i)$ is the occurrence probability of ith source IP address which accesses the kth URL:

$$P_k^I(x_i) = \frac{x_i^{(k)}}{\sum_{i=1}^{N} x_i^{(k)}}. \tag{5}$$

Entropy, not only EUPI but also EIPU, represents the probability of occurrence of discrete random events. In other words, information entropy is low in an orderly system. On the contrary, the more disordered and random the system is, the higher the information entropy would become. So it is able to be a measure of the ordering degree of the system. Typically, legitimate users that access to the site have certain randomness. Users will access web pages based on their interests. According to statistics, 80% of users visit 20% of the hot web pages [17].

Table 1: The map from IP to URL.

	Sequence number of URL (from 1 to K)
Source IP address (from 1 to N)	$\underbrace{\begin{matrix} \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(k)} & \cdots & x_1^{(K)} \\ \vdots & \cdots & \cdots & \cdots & & \\ x_i^{(1)} & x_i^{(2)} & \cdots & x_i^{(k)} & \cdots & x_i^{(K)} \\ \vdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_N^{(1)} & x_N^{(2)} & \cdots & x_N^{(k)} & \cdots & x_N^{(K)} \end{bmatrix} \\ \cdots \ \mathrm{EIPU}\left(j = k \mid x_i^{(k)}\right) \ \cdots \end{matrix}}$ $\left.\begin{matrix} \mathrm{EUPI}\left(i = 1 \mid x_1^{(k)}\right) \\ \cdots \\ \mathrm{EUPI}\left(i \mid x_i^{(k)}\right) \\ \cdots \\ \mathrm{EUPI}\left(i = N \mid x_N^{(k)}\right) \end{matrix}\right\}$

For a fixed URL attack whose IP address is represented as i, $P_i^U(x_i)$ would be significantly increased in the victim URLs, namely, to converge those access requests to the one or a few of attacked URLs. Therefore, the value of EUPI would reduce. For both a random URL and a traversal URL attack, EUPI of them would increase. In particular, because of the equal probability characteristic of traversal URL behavior, EUPI of a traversal URL attack would be close to maximum entropy of its value.

The space complexity of the algorithm in the article is $O(n^2)$, which increases with matrix dimension. Because the detection model of the article only needs to calculate the conditional entropy of the matrix, the time complexity of the algorithm is $O(n \log n)$.

However, a situation, on which a lot of sudden hits and needs (e.g., hot events, festival online shopping, and centralized e-ticketing) will lead to a sharp increase of URL traffic in a certain time, must be considered. The situation is named flash crowd, whose burst traffic and high volume are the common characteristics of AL-DDoS attack. If simply relying on EUPI, there would have a higher false alarm rate. So in order to optimize the detection method and reduce false alarm rate, the entropy of IP address per URL (EIPU) is considered and applied to distinguish between AL-DDoS attacks and flash crowd. When a URL is accessed from many IP addresses, there is approximated uniformly distributed traffic of each IP source address under the event of flash crowd. Thus, with characteristics of EUPI and EIPU, a matrix transform of $Z(\cdot)$ is constructed, which is defined as the following formula:

$$Z\left(\left[x_i^{(k)}\right]_{N\times N}\right) = \left(z_1, z_2, \ldots, z_N\right)^T. \tag{6}$$

Thereinto, the transform is needed to satisfy the condition of $N = K$. z_i stands for a joint entropy vector:

$$\begin{aligned} z_i &= z_i^{(1)} + jz_i^{(2)} \\ &= \mathrm{EUPI}\left(i \mid x_i^{(k)}\right) + \mathrm{jEIPU}\left(k = i \mid x_i^{(k)}\right). \end{aligned} \tag{7}$$

So we can obtain formulas of $z_i^{(1)} = \mathrm{EUPI}(i \mid x_i^{(k)})$ and $z_i^{(2)} = \mathrm{EIPU}(k = i \mid x_i^{(k)})$.

$$\begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(k)} & \cdots & x_1^{(N)} \\ \vdots & \cdots & \cdots & \cdots & & \\ x_i^{(1)} & x_i^{(2)} & \cdots & x_i^{(k)} & \cdots & x_i^{(N)} \\ \vdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_N^{(1)} & x_N^{(2)} & \cdots & x_N^{(k)} & \cdots & x_N^{(N)} \end{bmatrix} \longrightarrow \begin{bmatrix} z_1^{(1)} & z_1^{(2)} \\ \cdots & \cdots \\ z_i^{(1)} & z_i^{(1)} \\ \cdots & \cdots \\ z_N^{(1)} & z_N^{(2)} \end{bmatrix}. \tag{8}$$

In the condition of $N \neq K$, extended processing of matrix is used to satisfy equal conditions.

(a) $N > K$. When N is greater than K, the matrix needs to be extended to N order square matrix. The data from the $K + 1$ to N column comes from the normal access traffic of training data set. The extended URL from the $K + 1$ to N is named as virtual URL.

$$\left[x_{ij}\right]_{N\times K} \longrightarrow \left[\begin{array}{cccc|ccc} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(K)} & x_1^{(K+1)} & \cdots & x_1^{(N)} \\ \vdots & \cdots & \cdots & & \vdots & \cdots & \cdots \\ x_i^{(1)} & x_i^{(2)} & \cdots & x_i^{(K)} & x_1^{(K+1)} & \cdots & x_i^{(N)} \\ \vdots & \cdots & \cdots & & \vdots & \cdots & \cdots \\ x_N^{(1)} & x_N^{(2)} & \cdots & x_N^{(K)} & x_1^{(K+1)} & \cdots & x_N^{(N)} \end{array}\right]. \tag{9}$$

(b) $N < K$. When N is less than K, the matrix needs to be extended to K order square matrix. The data from the $N + 1$ to K row comes from the normal access traffic of training data

set. The extended IP address from the $N + 1$ to K is named as virtual IP.

$$\left[x_{ij}\right]_{N\times K} \longrightarrow \left[\begin{array}{cccccc} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(k)} & \cdots & x_1^{(K)} \\ \vdots & \cdots & \cdots & \cdots & & \\ x_i^{(1)} & x_i^{(2)} & \cdots & x_i^{(k)} & \cdots & x_i^{(K)} \\ \vdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_N^{(1)} & x_N^{(2)} & \cdots & x_N^{(k)} & \cdots & x_N^{(K)} \\ \hline x_{N+1}^{(1)} & x_{N+1}^{(2)} & \cdots & x_N^{(k)} & \cdots & x_N^{(K)} \\ \vdots & \cdots & \cdots & \cdots & & \\ x_K^{(1)} & x_K^{(2)} & \cdots & x_K^{(k)} & \cdots & x_K^{(K)} \end{array}\right]. \qquad (10)$$

Because the extended rows or columns is from normal access traffic of training data set, it does not affect the judgment of attack behaviors or abnormal traffic. For the algorithm of joint entropy vector, the extended processing of the matrix only adds the number of normal entropy vectors and does not change the distribution or the number of attack vector.

So the mapping from $\max(N, K)$-dimensional space to two-dimensional space is expressed as

$$X \longrightarrow Z : z_i \in Z = \left\{z_1, z_2, \ldots, z_{\max(N,K)}\right\}. \qquad (11)$$

Thereinto, $z_i = (z_i^{(1)}, z_i^{(2)})$ is two-dimensional entropy vector.

3.3. Boundary Discriminant. Let T be training data set. T is defined as

$$T = \left\{(z_1, y_1), (z_2, y_2), \ldots, (z_N, y_N)\right\}. \qquad (12)$$

Thereinto, $y_i \in Y = \{c_1, c_2, \ldots, c_K\}$ is a set of types of access behaviors. In the instance, c_1 stands for the fixed attack. c_2 stands for the traversal attack. c_3 stands for flash crowd. c_4 stands for normal access.

The classification decision rule is defined as

$$\arg\max_{c_j} \sum I\left(y_i = c_j\right), \quad j = 1, 2, \ldots, K. \qquad (13)$$

$I(y_i = c_j)$ is indicating function, which is defined as

$$I(x) = \begin{cases} 1, & x = \text{true} \\ 0, & x = \text{flase}. \end{cases} \qquad (14)$$

In summary, detecting AL-DDoS attacks is transformed into classifying points that represent entropy vectors in the coordinate system. According to the respective characteristics of different attack behavior types, the implementation of entropy vector detection algorithm is as follows.

In terms of training data set T, the point number of each class is calculated.

$$K_i = \sum_T I\left(y_i = c_i\right) \qquad (15)$$

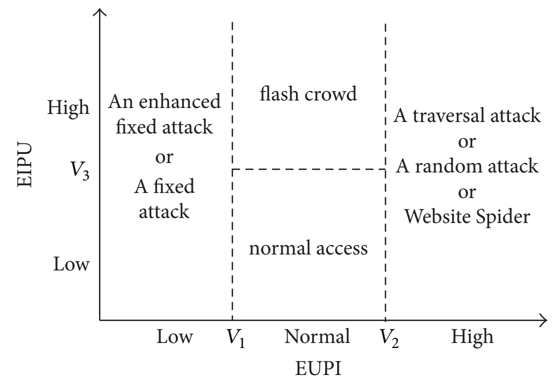

High
V_3
Low
EIPU
An enhanced fixed attack or A fixed attack
flash crowd
normal access
A traversal attack or A random attack or Website Spider
Low
V_1
Normal
V_2
High
EUPI

Figure 1: The coordinate discrimination diagram of entropy vector.

K_i stands for the point number of each class.

The boundary discriminant rule is defined as

$$V_j = \arg\left[\sum_T I\left(z_i^{(k)} \leq V_j\right) = K_i\right]$$
$$\text{or } V_j = \arg\left[\sum_T I\left(z_i^{(k)} \geq V_j\right) = K_i\right]. \qquad (16)$$

According to the characteristics of AL-DDoS entropy vector, the coordinate plane is divided into different regions by the boundary of V_j, in which the region decides what class it belongs to. The coordinate discrimination diagram of entropy vector is shown in Figure 1.

From Figure 1, c_1, c_2, c_3, and c_4, respectively, stand for class of a fixed URL attack, a traversal URL attack, flash crowd, and normal access. On the basis of training data set, K_1, K_2, and K_3, respectively, stand for the point number of three classes of attacks. The boundary V_1 is able to be calculated as

$$V_1 = \arg\left[\sum_T I\left(z_i^{(1)} \leq V_1\right) = K_1\right]. \qquad (17)$$

The boundary V_2 is able to be calculated as

$$V_2 = \arg\left[\sum_T I\left(z_i^{(1)} \geq V_2\right) = K_2\right]. \qquad (18)$$

The boundary V_2 is able to be calculated as

$$V_3 = \arg\left[\sum_T I\left(z_i^{(2)} \geq V_3\right) = K_3\right]. \qquad (19)$$

Accordingly, AL-DDoS attack would inevitably cause the change of entropy. We take the entropy vector z_i as an index. In terms of where z_i falls in, it can be found whether AL-DDoS attack has occurred and what type AL-DDoS attack could be.

When multiple training data sets are collected, the optimum classification boundary value is able to be obtained. A precision rate of class discrimination is defined as R_{pre}:

$$R_{\text{pre}} = \frac{\sum_T I\left(z_i^{(k)} \le V_j \&\& y_i = c_i\right)}{\sum_T I\left(y_i = c_i\right)}$$
$$\text{or } R_{\text{pre}} = \frac{\sum_T I\left(z_i^{(k)} \ge V_j \&\& y_i = c_i\right)}{\sum_T I\left(y_i = c_i\right)}. \tag{20}$$

Suppose the total training data set is T:

$$T = \{T_1, T_2, \ldots, T_M\}. \tag{21}$$

Thereinto, $T_i = \{(z_1^i, y_1^i), (z_2^i, y_2^i), \ldots, (z_N^i, y_N^i)\}$ is subset, which is training data set in one sampling time Δt. The optimum classification boundary value V_{opt} is defined as

$$V_{\text{opt}} = \arg\max \left\{ \sum_{i=1}^{M} \sum_{T_i} \frac{I\left(z_j^{(k)} \le V_{\text{opt}} \&\& y_j^i = c_i\right)}{I\left(y_j^i = c_j\right)} \right\} \tag{22}$$

or

$$V_{\text{opt}} = \arg\max \left\{ \sum_{i=1}^{M} \sum_{T_i} \frac{I\left(z_j^{(k)} \ge V_{\text{opt}} \&\& y_j^i = c_i\right)}{I\left(y_j^i = c_j\right)} \right\}. \tag{23}$$

4. Simulation Experiment Verification and Analysis

4.1. Experimental Conditions and Processes. Based on the open website log and MIT Lincoln Laboratory data sets [18], we use MATLAB software to simulate the access of the web server under the normal condition. Set up a website with 200 URLs, 10% of which are hot pages. There are about 800 visits per simulation time. Under normal circumstances, EUPI is shown in Figure 2.

When the fixed URL attack occurs, the change of EUPI of Web server is shown in Figure 3. At 30th time units, the fixed URL attack started, which made entropy obviously decreased.

As shown in Figure 4, EUPI instantly increases when the random URL attack occurs suddenly. A large number of random URL request makes the traffic of the server more disorder and chaos, so the corresponding URL request entropy will accordingly increase.

For traversal URL attacks, if the attacks started at 30th simulation time, the request entropy would suddenly rise as shown in Figure 5. On these attacks the URLs are relatively random in a single time unit and the detection results are also consistent with the results of random URL attack.

4.2. Analysis and Optimization of Approach. Through above experiments, it can be seen that when those attacks have occurred the URL request entropy instantly changes, which are very obvious such as fixed, random, and traverse URL attacks. So it shows that the change of the value of EUPI can effectively detect the abrupt changes of traffic that are caused by DDoS attack.

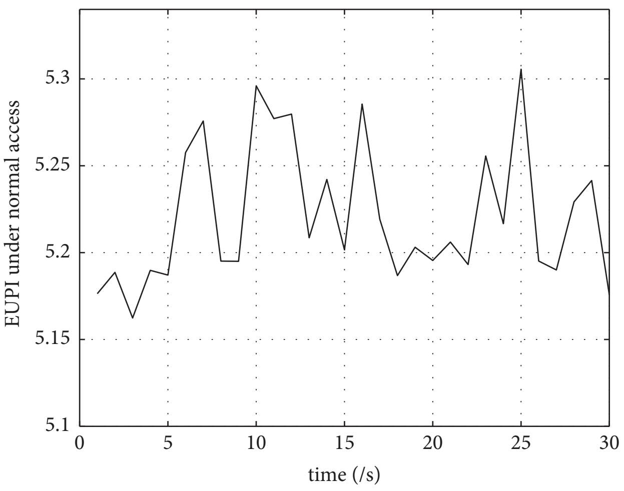


Figure 2: EUPI under the normal access.

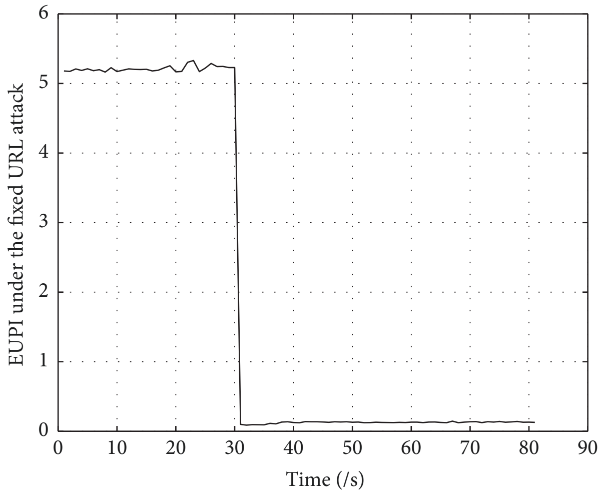


Figure 3: EUPI under fixed URL attack.

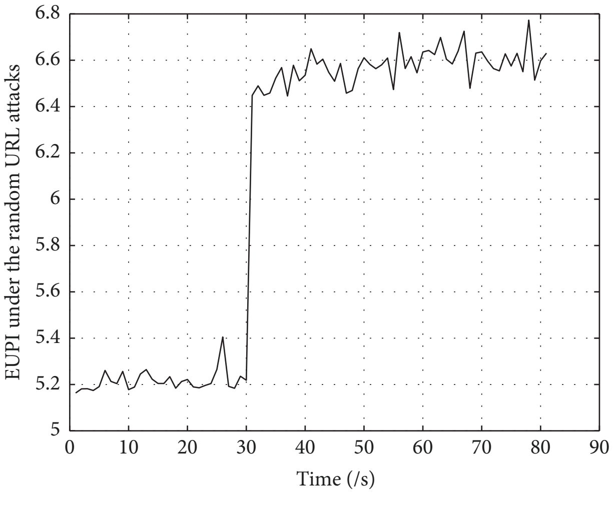


Figure 4: EUPI under random URL attack.

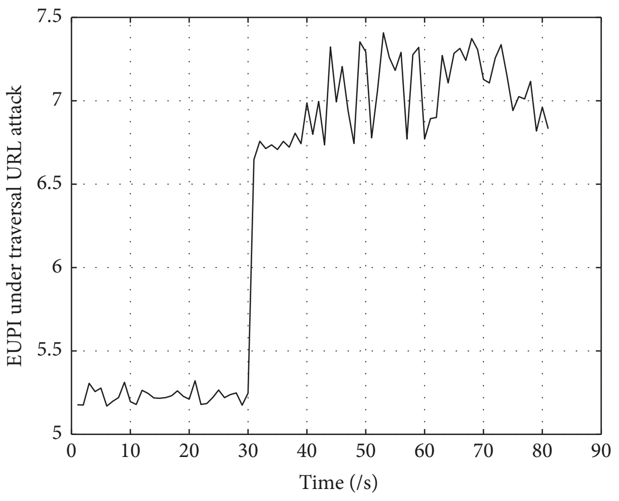


Figure 5: EUPI under traverse URL attacks.

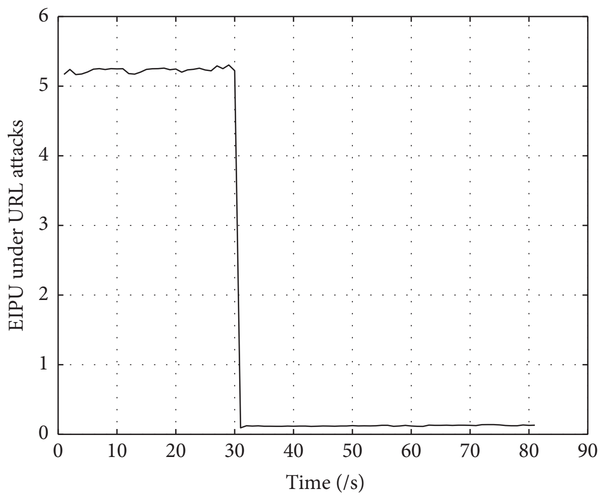


Figure 7: EIPU under URL attack.

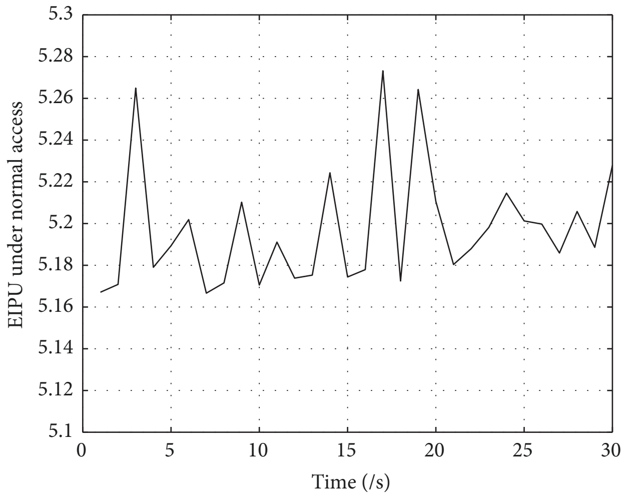


Figure 6: EIPU under normal access.

In order to discriminate between attacks and flash crowd on which a lot of sudden hits and needs (e.g., hot events, festival online shopping, and centralized e-ticketing) will lead to a sharp increase of URL traffic in a certain time. It is difficult to discriminate between attacks and flash crowd only through EUPI, so EIPU is considered and applied to detect whether AL-DDoS attacks exist.

Under normal circumstances, EIPU is shown in Figure 6.

From Figure 6, EIPU is between 5.16 to 5.28 under normal access.

At 30th time units, the URL attack started, which made EUPI obviously decreased in Figure 7. In order to detect whether AL-DDoS attack exists and discriminate what type the attack is, a simulation experiment of joint entropy vector algorithm is designed. The simulation parameter is shown in Table 2.

The simulation experiment scenario is constructed by the interaction process from 200 IP source addresses to 200 URLs access addresses in the matrix of $[x_i^{(k)}]_{N\times N}$. There are 6 IP nodes of the fixed URL attack, 20 IP nodes of the traverse URL attack, and 194 legitimate nodes in simulation experiment, whose proportion of all nodes, respectively, is 3%, 10%, and 87%. There are 2 URLs on flash crowd, whose proportion of all URLs is 1%. In the experiment scenario, we use SOAP replay attacks to simulate DDoS. SOAP replay attackers from distributed nodes send the soap request message repeatedly to URLs, which exhaust the source of the victim server. The number of requests is in proportion to the attack strength and is taken as a measure of attack strength. The experiment data is from Lab website log and MIT Lincoln Laboratory data sets [18]. Matlab is used to integrate data and construct the matrix of $[x_{ij}]_{N\times N}$. The nodes are divided into four categories: c1, c2, c3, and c4, respectively, stand for class of a fixed URL attack, a traversal URL attack, flash crowd, and normal access. The corresponding access or attack behaviors are described in the paper (in Section 2). The server records the interactive process and saves it. And we refer the data format of MIT Lincoln Laboratory data sets. We change the distribution, attack strength, and proportion of the four kinds of nodes, and let it run many times. The data with labeled category is used as the training data. According to the boundary criterion (in Section 3.3), the optimal boundary value of the satisfied formulas (22) and (23) is obtained.

The simulation results is shown in the coordinate plane consisting of the EUPI as x-axis and the jEIPU as y-axis. From Figure 8, vector dots belonging to different attack types are distributed in different regions of the coordinate plane. According to the analysis of the third chapter and Figure 7, the position feature of the entropy vector can reflect the characteristics of the AL-DDoS attack.

In order to verify the effectiveness of the algorithm on different cases, AL-DDoS attack strength is defined:

$$A_{\text{stre}}(j) = \frac{1}{N_j M_j} \sum_{(j=1|y_j=c_j)}^{N_j} \sum_{k=1}^{M_j} x_j^{(k)}. \qquad (24)$$

Table 2: Table of simulation parameter.

Class	Number (K_i)	Proportion	V_j
The fixed	6 (attacks)	3% (source IP address)	$z_i^{(1)} \le 0.69$
The traverse	20 (attacks)	10% (source IP address)	$z_i^{(1)} \ge 4.89$
Flash crowd	2 (legitimate)	1% (URL)	$z_i^{(2)} \ge 4.55$
Legitimate IP	194	87%	other

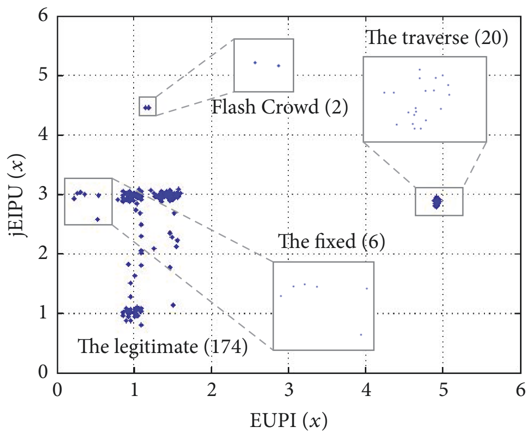


Figure 8: The simulation distribution figure of entropy vector.

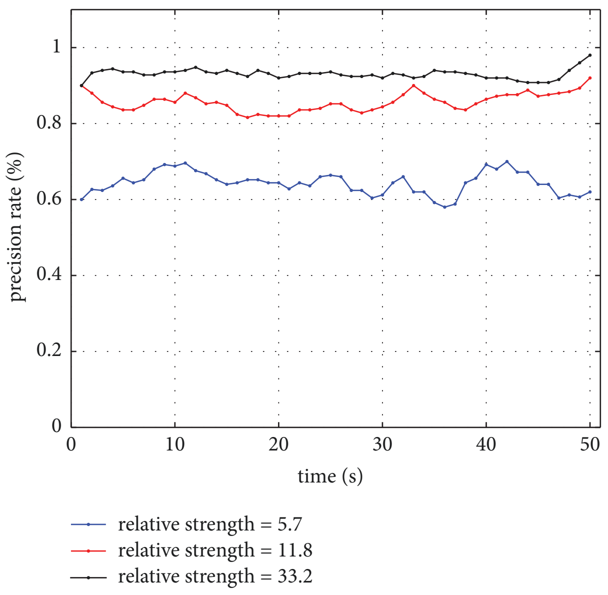


Figure 9: Comparisons of detection precision rate under different relative strength.

In terms of formula (15), $N_j = K_j = \sum_T I(y_j = c_j)$. M_j only counts $x_j^{(k)}$ that is not equal to 0:

$$M_j = \sum_{k=1}^{K} I\left(x_j^{(k)} \neq 0\right). \tag{25}$$

Also, we can define *relative strength* of AL-DDoS attack:

$$R_{\text{stre}} = \frac{A_{\text{stre}}(j)\big|_{cj=\{\text{attack}\}}}{A_{\text{stre}}(i)\big|_{ci=\{\text{normal}\}}}. \tag{26}$$

In Figure 9, 500 sets of data $T = \{T_1, T_2, \ldots, T_{500}\}$ are collected. At the same time, the detection precision rate R_{pre} under three kinds of relative strength R_{stre} is compared. As can be seen from the Figure 8, with the increase of relative strength, the precision rate of the algorithm is be able to increase and reach more than 90%.

In Figure 9, the detection precision rate under the different relative strength is compared. The joint entropy vector can be used to quickly judge what class of DDoS attacks has happened. It also can effectively distinguish the web server DDoS attacks and flash crowd and improve the detection precision rate with the increase of relative strength.

5. Conclusions and Future Work

With the popularity of the network and the rapid growth of network traffic, the burst traffic caused by hot events and centralized access often leads to the service congestion and even paralysis. This burst of traffic is usually called "flash crowd." Flash Crowd and DDoS attacks are essentially different. In this paper, based on URL access entropy, an anomaly detection algorithm is proposed. The novel method can effectively distinguish between AL-DDoS attacks, which has great reference value for further analysis of DDoS attack and its effective detection. As we have discussed, there is currently a lack of analysis on a big bot-net attacks, for example, hundreds of thousands of zombie machines. (The experiment scenario of this article is constructed by the interaction process from 200 IP source addresses to 200 URLs access addresses.) Focusing more closely on the application layer, we plan to further detect attacks on a larger network scale and more nodes in future. With improving experimental conditions and environment, in subsequent studies, we will further analyze the complexity of the defense technique under increasing the number of simulation nodes. As for future work, we also plan to extend the detection capabilities of the framework, namely, by supporting detection of other indicators, which can be used as a measure of the attack strength, such as the amount of traffic and the number of packets.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors' Contributions

Yuntao Zhao is the main contributor of this work, given that he originated the idea, provided the general design, and wrote most of the paper. Wenbo Zhang contributed to the implementation and testing of the simulation. All authors read and approved the final manuscript.

Acknowledgments

This work was supported by China Postdoctoral Science Foundation (2016M590234), General Project of Liaoning Provincial Department of Education (LG201611), the Open Foundation of Key Laboratory of Shenyang Ligong University (4771004kfs32), Postdoctoral Fund of Shenyang Ligong University, Project of Applied Basic Research of Shenyang (18-013-0-32), and 2017 Distinguished Professor Project.

References

[1] R. K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 42–51, 2002.

[2] W. Zhou, W. Jia, S. Wen, Y. Xiang, and W. Zhou, "Detection and defense of application-layer DDoS attacks in backbone web traffic," *Future Generation Computer Systems*, vol. 38, pp. 36–46, 2014.

[3] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[4] S. Rajesh, "Protection from application layer DDoS attacks for popular websites," *International Journal of Computer and Electrical Engineering*, vol. 5, no. 6, pp. 555–558, 2013.

[5] S. Wen, W. Jia, W. Zhou, and C. Xu, "CALD: Surviving various application-layer DDoS attacks that mimic flash crowd," in *Proceedings of the 4th International Conference on Network and System Security (NSS '10)*, pp. 247–254, September 2010.

[6] H. Beitollahi and G. Deconinck, "ConnectionScore: A statistical technique to resist application-layer DDoS attacks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 5, no. 3, pp. 425–442, 2014.

[7] H. Beitollahi and G. Deconinck, "Tackling application-layer DDoS Attacks," in *Proceedings of the 3rd International Conference on Ambient Systems, Networks and Technologies, ANT 2012 and 9th International Conference on Mobile Web Information Systems, MobiWIS 2012*, pp. 432–441, Canada, August 2012.

[8] G. Zhao and S. Yu, "Detecting application-layer DDoS attack based on analysis of users' behaviors," *Application Research of Computer*, vol. 28, no. 2, pp. 717–719, 2011.

[9] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites," in *Proceedings of the 11th International Conference on World Wide Web, WWW '02*, pp. 293–304, May 2002.

[10] K. Li, W. Zhou, P. Li, J. Hai, and J. Liu, "Distinguishing DDoS attacks from flash crowds using probability metrics," in *Proceedings of the 2009 3rd International Conference on Network and System Security, NSS 2009*, pp. 9–17, October 2009.

[11] S. Yu, T. Thapngam, J. Liu, S. Wei, and W. Zhou, "Discriminating DDoS flows from flash crowds using information distance," in *Proceedings of the 2009 3rd International Conference on Network and System Security, NSS 2009*, pp. 351–356, October 2009.

[12] G. Oikonomou and J. Mirkovic, "Modeling human behavior for defense against flash-crowd attacks," in *Proceedings of the IEEE International Conference on Communications (ICC '09)*, June 2009.

[13] S. Lee, G. Kim, and S. Kim, "Sequence-order-independent network profiling for detecting application layer DDoS attacks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, no. 1, article no. 50, 2011.

[14] R. Rathika, B. Dharanya, and KK. Devi, "Detecting the DDOS Attacks in Application Layer," *Smart Structures and Systems*, vol. 14, no. 5, pp. 765–784, 2011.

[15] Y. Xie and S.-Z. Yu, "Monitoring the application-layer DDoS sttacks for popular websites," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 15–25, 2009.

[16] J.-Q. Shi, B.-X. Fang, L. Guo, and L.-H. Wang, "Hybrid-structured onion scheme against replay attack of MIX," *Tongxin Xuebao/Journal on Communication*, vol. 30, no. 3, pp. 21–26, 2009.

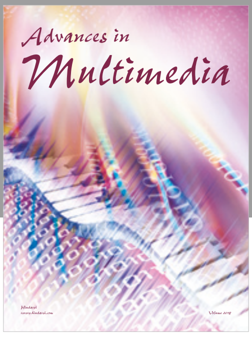
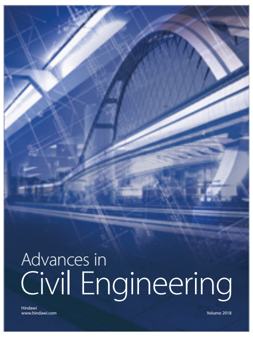
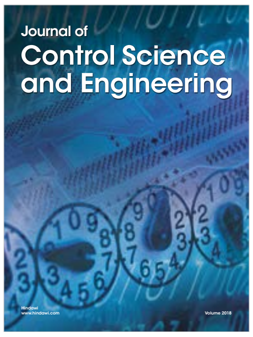
[17] C. Anderson, *The long trail*, CITIC Press Group, 2016.

[18] MIT Lincoln Laboratory, 2000, http://www.ll.mit.edu/ideval/data/2000data.html.

Journal of
Engineering

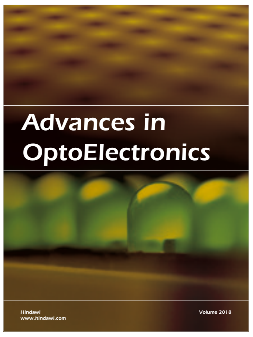
The Scientific World Journal

International Journal of
Rotating Machinery

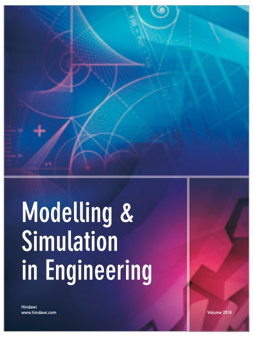
Journal of
Sensors

Advances in
Multimedia

Advances in
Civil Engineering

Journal of
Control Science and Engineering

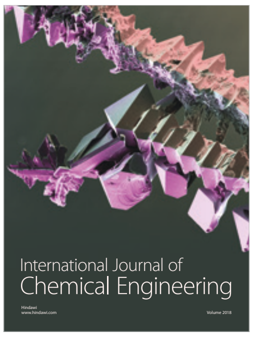
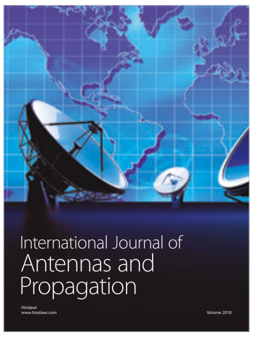
Journal of
Robotics

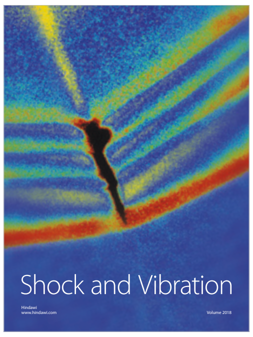
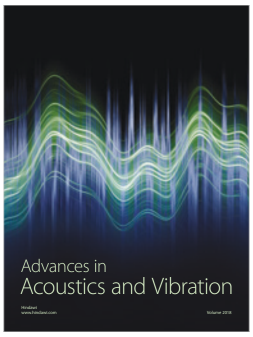
Journal of
Electrical and Computer Engineering


Advances in
OptoElectronics

VLSI Design

International Journal of
Navigation and Observation

Modelling & Simulation in Engineering

International Journal of
Aerospace Engineering

International Journal of
Chemical Engineering

International Journal of
Antennas and Propagation

Active and Passive Electronic Components

Shock and Vibration

Advances in
Acoustics and Vibration