Mining Web User Behaviors to Detect Application Layer DDoS Attacks

Chuibi Huang[1]
[1]Department of Automation, USTC
Key laboratory of network communication system and control
Hefei, China
Email: hcb@mail.ustc.edu.cn

Jinlin Wang[1, 2], Gang Wu[1], Jun Chen[2]
[2]Institute of Acoustics, Chinese Academy of Sciences
National Network New Media Engineering Research Center
Beijing, China
Email: {wangjl, chenj}@dsp.ac.cn

***Abstract*— Distributed Denial of Service (DDoS) attacks have caused continuous critical threats to the Internet services. DDoS attacks are generally conducted at the network layer. Many DDoS attack detection methods are focused on the IP and TCP layers. However, they are not suitable for detecting the application layer DDoS attacks. In this paper, we propose a scheme based on web user browsing behaviors to detect the application layer DDoS attacks (app-DDoS). A clustering method is applied to extract the access features of the web objects. Based on the access features, an extended hidden semi-Markov model is proposed to describe the browsing behaviors of web user. The deviation from the entropy of the training data set fitting to the hidden semi-Markov model can be considered as the abnormality of the observed data set. Finally experiments are conducted to demonstrate the effectiveness of our model and algorithm.**

***Index Terms*—HsMM, web user behaviors, DDoS, DDoS Attacks , clustering**

I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks have become a serious problem in recent years. Generally, DDoS attacks are carried out at the network layer, such as ICMP flooding, SYN flooding and UDP flooding. Without advance warning, a DDoS attack can easily exhaust the computing and communication resources of its victim within a short period of time [1]. Because of the seriousness and destructiveness, many studies have been conducted on this type of attacks. A lot of effective schemes have been proposed to protect the network and equipment from bandwidth attack, so it is not as effortless as in the past for attackers to launch the network layer DDoS attacks. In order to dodge detection, attackers shift their offensive strategies to application-layer attacks and establish more sophisticated types of DDoS attacks. They utilize legitimate application layer HTTP requests from legitimately connected network machines to overwhelm web server. These attacks are typically more efficient than TCP or UDP-based attacks, requiring fewer network connections to achieve their malicious purposes. The MyDoom worm and the CyberSlam are all instances of this type attack [2, 3].

The challenges of detecting the app-DDoS attacks can be summarized as the following aspects:

- The app-DDoS attacks make use of higher layer protocols such as HTTP to pass through most of the current anomaly detection systems designed for low layers.
- Along with flooding, App-DDoS traces the average request rate of the legitimate user and uses the same rate for attacking the server, or employs large-scale botnet to generate low rate attack flows. This makes DDoS attacks detection more difficult.
- Burst traffic and high volume are the common characteristics of App-DDoS attacks and flash crowds. "Flash crowd" refers to the situation when a very large number of users simultaneously access a popular Website, which produces a surge in traffic to the Website and might cause the site to be virtually unreachable [4]. It is not easy for current techniques to distinguish APP-DDoS attacks from flash crowds.

From the literature, few exiting researches focus on the detection of app-DDoS attacks during the flash crowd event. This paper introduces the hidden semi-Markov model (HsMM) to capture the user browsing patterns during flash crowd and to implement the app-DDoS attacks detection. Our contributions in this paper are threefold: 1) We use the clustering method to extract the web objects' access features, which can well portray current web user browsing behaviors. 2) We apply hidden semi-Markov model (HsMM) to describe the dynamics of access features and to implement the detection of app-DDoS attacks. 3) Experiments based on real traffic are conducted to validate our detection scheme.

The organization of the paper is as follows. In the next section, we review the related work of our research. In

doi:10.4304/jsw.9.4.985-990

Section 3, we explain our model and algorithm to detect the app-DDoS attacks. Experiment results are presented in section 4. Finally, we conclude our paper in Section 5.

II. Related Work

Most current DDoS-related researches are conducted on the IP layer or TCP layer instead of the application layer. These mechanisms typically utilize the specific network features to detect attacks. Mirkovic et al. [5] proposed a defense system called D-WARD located in edge routers to monitor the asymmetry of two-way packet rates and to detect attacks. Yu et al. [6] discriminated DDoS attacks from flash crowds by using the flow correlation coefficient as a similarity metric among suspicious flows. Zhang et al. [7] proposed the Congestion Participation Rate (CPR) metric and a CPR-based approach to detect and filter the TCP layer DDos attacks. Cabrera et al. [8] depend on the MIB traffic variables collected from the systems of attackers to achieve the early detection. Yuan et al. [9] capture the traffic pattern by using cross correlation analysis and then decide where and when a DDoS attack possibly arises. Soule et al. [10] used the traffic matrix, which represents the traffic state, to identify various dynamic attacks at early stage. In [11], DDoS attacks were discovered by analyzing the TCP packet header against the well-defined rules and conditions and distinguished the difference between normal and abnormal traffic. In [12], attackes are detected by computing the rate of TCP flags to TCP packets received at a web server. However, there are little DDoS defense methods that utilize the application layer information. In [19], a suspicion assignment mechanism Ranjan et al. [19] used statistical method to detect time related characteristics of HTTP sessions, such as request interarrival time, session inter-arrival time and session arrival time. Yen et al. [21] defended the application DDoS attacks with constraint random request attacks by the statistical methods. Kandula et al. [3] designed a system to protect a web cluster from DDoS attacks by (i) optimally dividing time spent in authenticating new clients and serving authenticated clients (ii) using CAPTCHAs designing a probabilistic authentication mechanism, but the task of requiring users to solve graphic puzzles causes additional service delay. As a result, the graphic puzzle cause annoying legitimate users as well as act as another DDoS attack points. [16] introduced a web browsing model represented by the transition of the click pages. In a few previous studies, analysis of user behaviors have been applied in many research fields [13, 15, 17, 20, 22].

III. Model Preliminaries And Assumptions

When detecting the app-DDoS attacks, we are faced with the following challenges: 1) attacker may launch an app-DDoS attack by mimicking the normal web user access behavior, so the malicious requests differ from the legitimate ones but not in traffic characteristics. Therefore, most current detection mechanisms based on traffic characteristics become invalid 2) Both the flash crowd and app-DDoS attacks are unstable, bursty and huge traffic volume. The app-DDoS attackers are increasingly moving away from pure bandwidth flooding to more surreptitious attacks that hide in normal flash crowd of the website. It is a challenge to detect the app-DDoS attacks when they occur during a flash crowd event.

To meet the above challenges, we focus on analysis of the user behaviors. In this paper, we assume that it is impossible for app-attacks to completely mimic the normal web user behaviors. This assumption is based on the following consideration. Generally, web user access behaviors can be described by three factors: HTTP request rate, page retention time, and request sequence. The attackers can mimic the normal access behavior by launching attacks with similar HTTP request rate and retention time as the normal users. However, the app-DDoS attacks cannot simulate the dynamic process of the web user behavior or the request sequence, because it is a pre-designed routine and cannot capture the dynamics of the users and the networks. Another assumption of our paper is that the user behavior can be described by the distribution of web objects popularity. Then the variation of web objects popularity could represent the dynamic changing process of user behavior. Since app-DDoS attacker is unable to obtain historical access records from the victim server, it cannot mimic the dynamic of normal user behaviors. We consider the app-DDoS attacks as anomaly browsing behavior. Moreover, we also assume that normal users always access the "hot webpages". However, the attackers access the "cold webpages". Therefore, we could build the browsing behavior model for both attacks and normal user by monitoring the dynamic change of the web objects popularity. Using this model, we could distinguish the app-DDoS attack from normal user.

IV. Model And Detection Principle

A. Hidden Semi-Markov Model

Hidden semi-Markov model (HsMM) [14] is an extension of hidden Markov model (HMM) [23] with variable state duration. It is a stochastic finite state machine, specified by $\lambda = (Q, \boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{P})$ where:

- Q is a discrete set of hidden states with cardinality M, i.e., $Q = \{1, \ldots, N\}$. $q_t \in Q$ denotes the state that the system takes at time t;
- $\boldsymbol{\pi}$ is the initial state probability distribution, i.e., $\boldsymbol{\pi} = \{\pi_m \mid m \in Q\}$, $\pi_m \equiv \Pr[q_1 = m]$. q_t denotes the state that the system takes at time t and $m \in Q$. The initial state probability distribution satisfies $\sum_m \pi_m = 1$;
- $\mathbf{A}$ is the state transition matrix with probabilities: $a_{mn} \equiv \Pr[q_t = n \mid q_{t-1} = m]$, $m, n \in Q$, and the state transition coefficients satisfy $\sum_n a_{mn} = 1$;

- B is the output probabilities matrix $b_m(k) \equiv \Pr\left[\overline{o_t} = \overline{v_k} \mid q_t = m\right]$, $m \in Q$ and $\overline{o_t}$ denotes the observed vector at time t, taking values from $\left\{\overline{v_1}, \ldots, \overline{v_K}\right\}$, K is the size of the observable output set. For a given state m, $\sum_k b_m(k) = 1$;
- P is the state duration matrix with probabilities: $p_m(d) \equiv \Pr\left[\tau_t = d \mid q_t = m\right]$, τ_t denotes the remaining time of the current state q_t, $m \in Q$, $d \in \{1, \ldots, D\}$, D is the maximum interval between any two consecutive state transitions, and the state duration coefficients satisfy $\sum_d p_m(d) = 1$.

Then, if the pair process (q_t, τ_t) takes on value (m, d), the semi-Markov chain will remain in the current state m until time $t + d - 1$ and transits to another state at time $t + d$, for $d \in \{1, \ldots, D\}$. It is generally not observable for these states. The observable variables are a series of observations $O = (\overline{o_1}, \ldots, \overline{o_T})$ where o_t denotes the observable output at time t and T is the number of samples in the observed sequences. $b_m(o_{a|b})$ represents the observation sequence from time a to time b $\left(\text{i.e.,} \left\{\overline{o_t} : a \le t \le b\right\}\right)$. When the "conditional independence" of outputs is assumed, $b_m(o_{a|b}) = \prod_{t=a}^{b} b_m(o_t)$.

B. Problem Formulation

We use the hidden semi-Markov model to describe the dynamic changing process of web user behaviors. The hidden state q_t is used to present the distribution of web objects' popularity, namely the user behavior, at time t. In general, the distribution of web objects' popularity is unobservable. The observable output is the web objects' click rates:

$$Click_{it} = \frac{c_{it}}{\sum_{i=1}^{N} c_{it}} \tag{1}$$

where c_{it} is the click number of the ith time unit, and N is the number of the web server's objects. $p_m(d)$ is the user behavior retention time distribution. The change of web user behaviors can be considered as a transition of the hidden state (i.e., from q_{t-1} to q_t). We can estimate the parameter with the following forward and back algorithm [14].

The forward variable is defined as follows:

$$\alpha_t(m, d) = \Pr\left[o_1^t, (q_t, \tau_t) = (m, d)\right]. \tag{2}$$

A transition into state $(q_t, \tau_t) = (m, d)$ takes place either from $(q_{t-1}, \tau_{t-1}) = (m, d+1)$ or from $(q_{t-1}, \tau_{t-1}) = (n, 1)$ for some $n \ne m$. Therefore, we could obtain the following forward recursion formulas:

$$\alpha_t(m, d) = \alpha_{t-1}(m, d+1) b_m(o_t) + \left(\sum \alpha_{t-1}(n, 1) a_{nm}\right) \cdot b_m(o_t) p_m(d), \quad d \ge 1 \tag{3}$$

$$\alpha_1(m, d) = \pi_m b_m(o_1) p_m(d). \tag{4}$$

The backward variable is defined as follows:

$$\beta_t(m, d) = \Pr\left[o_{t+1}^T, (q_t, \tau_t) = (m, d)\right]. \tag{5}$$

It can be seen that when $d > 1$ the next state must be $(q_{t+1}, \tau_{t+1}) = (m, d-1)$, and when $d = 1$ it must be $(q_{t+1}, \tau_{t+1}) = (n, d')$ for some $n \ne m$ and $d' \ge 1$. We thus obtain the following backward recursion formula:

$$\beta_t(m, d) = b_m(o_{t+1}) \beta_{t+1}(m, d-1), \quad \text{for } d > 1 \tag{6}$$

and

$$\beta_t(m, 1) = \sum_{n \ne m} a_{mn} b_n(o_{t+1}) \left(\sum_{d \ge 1} p_n(d) \beta_{t+1}(n, d)\right) \tag{7}$$

Especially when $t = T$:

$$\beta_T(m, d) = 1, \quad d \ge 1. \tag{8}$$

Three joint probability functions can be expressed in terms of the assumed model parameters and the forward and backward variables defined above:

$$\xi_t(m, n) = \Pr\left[o_1^T, q_{t-1} = m, q_t = n\right] = \alpha_{t-1}(m, 1) a_{mn} b_n(o_t) \cdot \left(\sum_{d \ge 1} p_n(d) \beta_t(n, d)\right) \tag{9}$$

$$\eta_t(m, d) = \Pr\left[o_1^T, q_{t-1} \ne m, q_t = m, \tau_t = d\right] = \left(\sum_{n \ne m} \alpha_{t-1}(n, 1) a_{nm}\right) b_m(o_t) p_m(d) \beta_t(m, d). \tag{10}$$

$$\gamma_t(m) = \Pr\left[o_1^T, q_t = m\right]. \tag{11}$$

Then, the model parameters can be re-estimated by the following formulas:

$$\hat{q}_t = \arg \max_{1 \le m \le M} \Pr\left[q_t = m \mid o_1^T\right] = \arg \max_{1 \le m \le M} \gamma_t(m), \quad \text{for } t = T, T-1, \ldots, 1. \tag{12}$$

$$\hat{\pi}_m = \frac{\gamma_1(m)}{\sum_{n=1}^{N} \gamma_1(n)} \tag{13}$$

$$\hat{a}_{mn} = \frac{\sum_{t=1}^{T} \xi_t(m, n)}{\sum_{t=1}^{T} \sum_{n=1}^{N} \gamma_t(n)} \quad (14)$$

$$\hat{p}_m(d) = \frac{\sum_{t=1}^{T} \eta_t(m, d)}{\sum_{d=1}^{D} \sum_{t=1}^{T} \eta_t(m, d)} \quad (15)$$

$$\hat{b}_m(v_k) = \frac{\sum_{t=1}^{T} \gamma_t(m) \delta(o_t - v_k)}{\sum_{k} \sum_{t=1}^{T} \gamma_t(m) \delta(o_t - v_k)} \quad (16)$$

Where $\{v_k\}$ is the set of observable values, and if $o_t = v_k$, $\delta(o_t - v_k) = 1$, otherwise $\delta(o_t - v_k) = 0$.

The average entropies AE of observed sequences fitting to the HsMM model are used to detect app-DDoS attacks:

$$Entropy = \Pr[o_1^T \mid \lambda] = \sum_{m,d} \Pr[o_1^T, (q_T, \tau_T) = (m, d) \mid \lambda] \quad (17)$$

$$ALE = \frac{\ln(\Pr[o_1^T \mid \lambda])}{T}. \quad (18)$$

C. Clustering Method

Clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups. It is a main task of exploratory data mining, and a common technique for statistical data analysis used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.

Since the number of web objects is enormous, it is difficult to deal with the multidimensional observed data without mass computation when training the hidden semi-Markov model. Thus, we apply clustering method to reduce the dimension of observed data. The specific clustering algorithm is shown in Fig.1.

We assume the cluster results as:

$$C = \{c_1, c_2, \ldots, c_M\}. \quad (19)$$

$$Num = \{n_1, n_2, \ldots, n_M\}. \quad (20)$$

Where C denotes the M classes, and n_i denotes the number of the web objects in c_i. Then we calculate the click rates of each class according (1).

VI. EXPERIMENTS

We simulate 1200 client nodes which play as normal users from the semifinals of FIFA WorldCup98 [18]. We randomly select 15% of these nodes as compromised nodes. Furthermore, we assume the attackers can intercept a small portion of requests from normal web users and make the same request or "hot" pages to launch the app-DDoS attack to the victim server. But we also assume that the app-DDoS attacks cannot simulate the dynamic process of the web user behavior or the request sequence, because it is a pre-designed routine and cannot capture the dynamics of the user and the networks. Thus, when the attack begins, each compromised node replays a snippet of another historical flash crowd trace. The interval between two consecutive attack requests is decided by the attack rate. In our experiments, we simulate constant rate attack and increasing rate attacks. We set the time unit to 5s and group 12 consecutive observations into one sequence. The "moving" step is one time unit and a new sequence is formed using the current observation and the preceding 11 observations.

Algorithm 1 The K-means Clustering

Input: Web objects' click rates dataset D
Number clusters K
Output: Set of cluster representatives C
Cluster membership vector **m**

/* Initialize cluster representatives C */
Randomly choose K data points from D
Use these K data points as initial set of C
repeat
/* Data Assignment */
Reassign points in D to closet cluster mean
Update **m** such that m_i is cluster ID of ith point in D
/* Relocation of means */
Update C such that c_j is means of points in jth cluster
Until convergence of objective function
$\sum_{i=1}^{N} (\arg\min_j \|\mathbf{x}_i - \mathbf{c}_j\|_2^2)$

Figure 1. The Clustering Algorithm

A. Attacks during Flash Crowd

The emulation process lasts about 6 h. The first 2 h data are used to train the model, and the remaining 4 h of data including a flash crowd event are used for test. The emulated app-DDoS attacks are mixed with the trace chose from the period of [3.5h, 5.5h].

Fig.2 shows the average entropies of observations varying with the time, when constant rate attacks are emulated. Curve a represents the dynamic entropy varying process of normal flash crowd and curve b represents the dynamic entropy varying process of flash crowd mixed with constant rate app-DDoS attacks. We can see that the average entropy of observations does not change much during the flash crowd period of [2h, 3.5h], which implies that the main web user behaviors do not have obvious varieties during the flash crowd event.

However, during the period of [3.5h, 5.5h], constant rate attacks appear and the average entropy of observations decreases sharply. In the duration of constant rate attack, there is significantly deviation from the average entropy of normal observations. Therefore, we could make use of this phenomenon to detect app-DDoS attack.

Fig.3 shows the average entropy of observation varying with the time, when increasing rate attacks are emulated. Curve *a* represents the dynamic entropy varying process of normal flash crowd and curve *b* represents the dynamic entropy varying process of flash crowd mixed with increasing rate app-DDoS attacks. As shows in Fig.3, during the period of [3.5h, 5.5h], increasing rate attacks appear and the average entropy of observations decreases gradually. At the end of increasing rate attacks, the average entropy increases gradually to the value range of the normal case. Similar to the situation described above, the average entropy in the duration increasing rate attacks is significantly smaller than the one in normal flash crowd period.

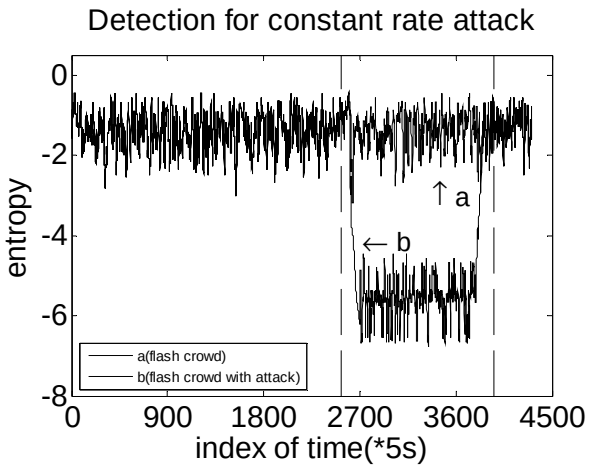

Figure 2. Entropy versus time of constant rate attack

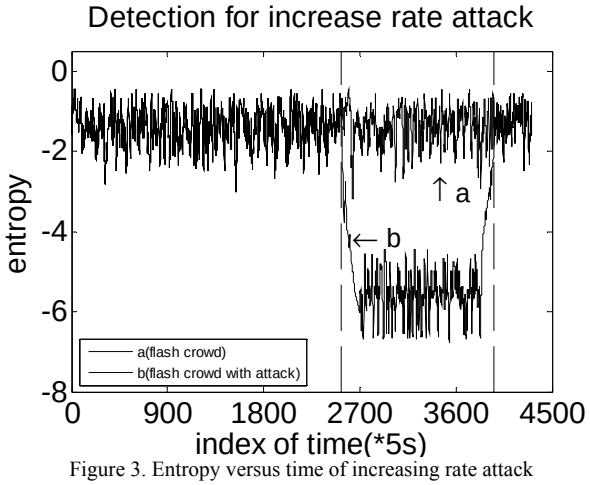

Figure 3. Entropy versus time of increasing rate attack

B. Performance

In the above scenarios, based on the average entropy of observations fitting to the HsMM model, we can detect the abnormity caused by the DDoS attack. Fig.4 shows the distributions of average entropy. It is easy to see that there exist significant differences in entropy distributions between two groups: the entropies of normal web traffic are larger than -3, but most entropies of the traffic containing attacks are less than -4. Therefore, we could make use of this result to identify the app-DDoS attacks from the normal web traffic when conducting DDoS detection. As shows in Fig.5, if we take -2.7 for the threshold value of normal web traffic's average entropy, the false negative rate (FNR) is about 2%, and the detection rate (DR) is about 97%. We can see that the algorithm can correctly detect the app-DDoS attacks which happened with a flash crowd event by making use of the dynamic HsMM models of web user behaviors.

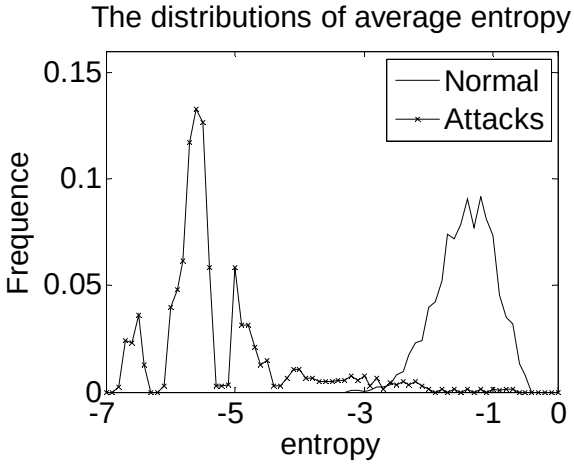

Figure 4. Distributions of average entropy

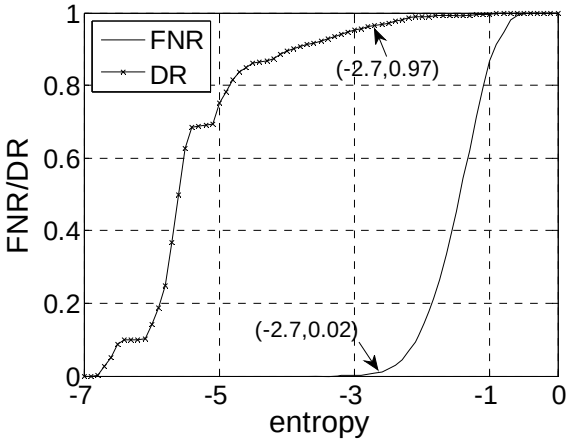

Figure 5. Cumulate distribution of average entropies

VII. CONCLUSION

In this paper, we applied the hidden semi-Markov model to describe the web user behaviors which can be represented by the click rates of web objects. By training the observed data of normal web traffic with forward and backward algorithm, we obtain the hidden semi-Markov model parameters. The average entropies of observed sequences fitting to the HsMM model are used to detect app-DDoS attacks. The experiments show that there is obvious obviation from the average entropies of normal

observations in the duration of app-DDoS attacks. In order to reduce the amount of calculation in training HsMM model, we apply clustering method to reduce the dimension of observed data. In addition, in future we can consider how to cope with the app-DDoS attacks launched by requesting dynamic webpage.

ACKNOWLEDGMENT

This work and related experiment environment is supported by the National High Technology Research and Development Program of China under Grant No. 2011AA01A102, the National Key Technology R&D Program under Grant No. 2012BAH18B04 and the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA06010302. We are sincerely grateful to their support.

REFERENCES

[1] C. Douligeris, A. Mitrokotsa, "DDoS attacks and denfense mechanisms: Classification and state-of-the-art," Computer Networks: The Int. J. Computers and Telecommunications Networking, v 44, n 5, p 643-666, Apr 2004.

[2] "Incident Note IN-2004-01 W32/Novarg. A virus," CERT. [Online]. Available: http://www.cert.org/incident_notes/IN-2004-01.html

[3] S. Kandula, D. Katabi, M. Jacob, A. W. Berger, "Botz-4-Sale:Surviving Organized DDoS Attacks that Mimic Flash Crowds," MIT, Tech. Rep. TR-969, 2004[Online]. Available: http://www.usenix.org/events/nsdi05/tech/kandula/kandula.pdf.

[4] Yi. Xie, Shunzheng. Yu, "Monitoring the application-layer DDoS attacks for popular websites," IEEE/ACM Transactions on Networking, v 17, n 1, February 2009.

[5] J. Mirkovic, G. Prier, P. Reiher, "Attacking DDoS at the source," In Proc. Int. Conf. Network Protocols, p 312-321, 2002.

[6] Shui. Yu, Wanlei. Zhou, Weijia. Jia, Song. Guo, "Discriminating DDoS attacks from flash crowds using flow correlation coefficient," IEEE Transactions on Parallel and Distributed System, v 23, n 6, June 2012.

[7] Changwang. Zhang, Zhiping. Cai, Weifeng Chen, "Flow level detection and filtering of low-rate DDoS," Computer Networks, v 56, n 15, p 3417-3431, October 2012.

[8] J. B. D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran, R. K. Mehra, "Proactive detection of distributed denial of service attacks using MIB traffic variables a feasibility study," In Proc. IEEE/IFIP Int. Symp. Integr. Netw. Manag., p 609-622, May 2001.

[9] J. Yuan, K. Mills, "Monitoring the macroscopic effect of DDoS flooding attacks," IEEE Trans. Dependable and Secure Computing, v 2, n 4, p 324-335, October 2002.

[10] A. Soule, K. Salamatian, N. Taft, "Combining filtering and statistical method for anomaly detection," In Proceedings of Internet Measurement Conference, p 331-344, 2005.

[11] L. Limwiwatkul, A. Rungsawangr, "Distrubited denial of service detection using TCP/IP header and traffic measurement analysis," In Proc. Int. Symp. Commun. Inf. Technol., Sappoo, Japan, p 605-610, October 2004.

[12] S. Noh, C. Lee, K. Choi, G. Jung, "Detecting Distributed Denial of Service (DDoS) attacks through inductive learning," Lecture Notes in Computer Science, v 2690, p 286-295, 2003.

[13] Jun. Cai, Shunzheng. Yu, Yu. Wang, "The community analysis of user behaviors network for web traffic", Journal of Software, v 6, n 11, p 2217-2224, 2011.

[14] S. Z. Yu, H. Kobayashi, "An efficient forward-back algorithm for an explicit duration hidden Markov model." IEEE Signal Process. Lett., v 10, n 1, p 11-14, Jan 2003.

[15] Hui. Chen, "Relationship between motivation and behavior of SNS User", Journal of Software, v 7, n 6, p 1265-1272, 2012.

[16] Y. Xie, S. Yu, "A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors", IEEE Transaction on Networking, v 17, n 1, February 2009.

[17] Wei. Yu, Shijun. Li, Yunlu. Zhang, Zhuo Zhang, "Mining users similarity of interest in web community", Journal of Computers, v 6, n 11, p 2357-2364, 2011.

[18] [Online]. Available: http://ita.ee.lbl.gov/html/traces.html.

[19] S. Ranjan, R. Swaminathan, M. Uysal, E. Knightly, "DDoS-resilient scheduling to counter application layer attacks under imperfect detection," In Proceedings of IEEE INFOCOM, April 2006.

[20] Qingzhang. Chen, Yanqing. Ou, Hang. Sun, "Design and implement of customer communication behavior analysis system", Journal of Software, v 6, n 8, p 1484-1491, 2011.

[21] W. Yen, M.-F. Lee, "Defending application DDoS with constraint random request attacks," In Proc. Asia-Pacific Conf. Commun., Perth, Western Australia, p 620-624, October 2005.

[22] Xiaohua. Hu, Tao. Mu, Weihui. Dai, Hongzhi. Hu, Genghui. Dai, "Analysis of browsing behaviors with ant colony clustering algorithm", Journal of Computers, v 7, n 12, p 3096-3102, 2012.

[23] L. R. Rabiner, "A Tutorial on hidden markov models and selected applications in speech recongnition," In Proc. of IEEE, v 77, n 2, p 257-286, February 1989.