DETECTION OF APPLICATION LAYER DDOS ATTACKS USING INFORMATION THEORY BASED METRICS

S. Renuka Devi and P. Yogesh

Department of Information Science and Technology,
College of Engg. Guindy, Anna University, Chennai. India.
renusaravanan@yahoo.co.in, yogesh@annauniv.edu

ABSTRACT

Distributed Denial-of-Service (DDoS) attacks are a critical threat to the Internet. Recently, there are an increasing number of DDoS attacks against online services and Web applications. These attacks are targeting the application level. Detecting application layer DDOS attack is not an easy task. A more sophisticated mechanism is required to distinguish the malicious flow from the legitimate ones. This paper proposes a detection scheme based on the information theory based metrics. The proposed scheme has two phases: Behaviour monitoring and Detection. In the first phase, the Web user browsing behaviour (HTTP request rate, page viewing time and sequence of the requested objects) is captured from the system log during non-attack cases. Based on the observation, Entropy of requests per session and the trust score for each user is calculated. In the detection phase, the suspicious requests are identified based on the variation in entropy and a rate limiter is introduced to downgrade services to malicious users. In addition, a scheduler is included to schedule the session based on the trust score of the user and the system workload.

KEYWORDS

DDoS, Application Layer & Entropy

1. INTRODUCTION

DDoS attacks involve in saturating the target machine with external communications requests, such that it cannot respond to legitimate traffic. Such attacks usually lead to a server overload. DDoS attacks are implemented purposefully to force the targeted computer(s) to reset, or to consume its resources such as network bandwidth, computing power, and operating system data structures so that it can no longer provide its intended service.

To launch a DDoS attack, the attackers first establishes a network of compromised computers that are used to generate the huge volume of traffic needed to deny services to legitimate users of the victim. Then the attacker installs attack tools on the compromised hosts of the attack network. The hosts running these attack tools are known as zombies, and they can be used to carry out any attack under the control of the attacker. In addition, the attacker will mimic the network traffic pattern of flash event to make the detection tougher. Most of the existing techniques cannot discriminate the DDoS attacks from the surge of legitimate accessing.

Traditionally, DDoS attacks are carried out at the network layer. Recently, there are an increasing number of DDoS attacks against online services and Web applications. These attacks are targeting

David C. Wyld, et al. (Eds): CCSEA, SEA, CLOUD, DKMP, CS & IT 05, pp. 217–223, 2012.
 DOI : 10.5121/csit.2012.2223

the application level. Application layer DDoS attacks may focus on exhausting the server resources such as Sockets, CPU, memory, disk/database bandwidth, and I/O bandwidth. These attacks are typically more efficient than TCP or UDP-based attacks, requiring fewer network connections to achieve their malicious purposes. They are also harder to detect, both because they don't involve large amounts of traffic and because they look similar to normal benign traffic.
An application layer DDoS attacks are classified into the following types [2]: (1) session flooding attack in which session connection request rate is higher than the requests from the legitimate users; (2) request flooding attack in which sessions containing more number of requests than usual is sent; and (3) asymmetric attack in which sessions containing high-workload requests are sent. This paper focuses on how to mitigate the request flooding attack.

The proposed scheme monitors the user's browsing behaviour (e.g. HTTP request rate, page viewing time and requested sequence of objects and their order) in the past and calculates the allowable request rate (entropy) and the user's trust level based on their visiting history. Based on the observed values, the detection is made. Initially the entropy of incoming requests is calculated and compared with the allowable rate. If the deviation exceeds a threshold then that session is considered to be malicious. Otherwise, based on trust score of the user, rate limiter filters the session. Then the scheduler schedules the session. The key features of the proposed work are

- Double check is made before servicing a client
- Easily deployable
- Low False rejection rate (The fraction of the rejection of requests from legitimate users over the total number of requests from legitimate users is called the False Rejection Rate).

The rest of this paper is organized as follows: Section 2 presents the related work to defend against the DDoS attacks. Section 3 describes the proposed DDoS defense mechanism using Information theory-based metric. The key components of the proposed scheme and its functionality are discussed in detail. Finally, the paper concludes with Section 4.

2. RELATED WORKS

Most of the available schemes attempt to detect attacks by analyzing the packet header information, packet arrival rate and so on. They treat anomalies as deviations in the IP attributes, e.g., source IP address, TTL, and the combination of multiple attributes. Wang, Jin, and Shin [12], proposed a victim based solution where a received IP packet is discarded if major discrepancies exist between its hop count and the value stored in the previously built table. In StackPi [13], a packet is marked deterministically by routers along its path towards the destination. The victim can associate Stackpi marks with source IP addresses to detect source IP address spoofing. In Differential Packet Filtering against DDoS Flood Attacks [14], author relies on probabilistic means to determine risky packets. This scheme is adaptive to traffic change and attempts to sustain quality of service.

Cabrera et al. [16] used the management information base (MIB) data which include parameters that indicate different packet and routing statistics from routers to achieve the early detection. Yuan and Mills [17] used the cross-correlation analysis to capture the traffic patterns and then to decide where and when a DDoS attack possibly arises. Keromytis, Misra and Rubenstein [15] present the conception of Secure Overlay Services (SOS) overlay network through which the legitimate traffic is sent. SOS network is able to change overlay topology dynamically to avoid DDoS and can survive in case that some key nodes are attacked. 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.

With respect to the Application layer DDoS attacks, Ranjan et al. [2] proposed a counter mechanism that used statistical methods to detect characteristics of HTTP sessions and employed rate-limiting as the primary defense mechanism. Yi Xie and Shun-Zheng Yu [10] proposed a scheme based on document popularity in which anomaly detector based on hidden semi-Markov model is used to detect the attacks. The biggest problem of Hidden Semi-Markov method was the algorithm complexity. Yen and Lee [18] defended the application DDoS attacks with constraint random request attacks by the statistical methods.

Wang et al. [4] proposed a relative entropy based application layer-DDoS detection method in which the click ratio of the web object is defined and the cluster method is used to extract the click ratio features. With the extracted features, the relative entropy is calculated to detect the suspicious sessions. Yu et al. [1] proposed an information theory based detection mechanism in which the distance of the package distribution behaviour among the suspicious flows is used to discriminate mimicking flooding attacks from legitimate accessing. Oikonomou and Mirkovic [7] analyzed many websites' log, and proposed defenses against application layer DDoS attacks via human behaviour modelling which differentiate DDoS bots from human users. Kandula et al. [8] proposed a system to protect a web cluster from DDoS attacks by CAPTCHAs in which the users who solve the puzzles can only get access to the services. This method assumed that human users can identify the distorted images, but the machine can not.

Liu and Chang [3] proposed a DAT (Defense against Tilt DoS attack) scheme. DAT monitors a user's features throughout a connection session to determine whether he is malicious user or not. For different behaved users, DAT provide differentiated services to them. Jie Yu [6] proposed a Trust Management Helmet scheme in which a user is assigned a license and a trust based on which detection is made.

3. PROPOSED WORK

Compared with non-attack cases, the number of requests in a session increases significantly in a very short time period in DDoS attack cases. The proposed scheme includes two phases. In the first phase, the system analyzes the web user characteristics and assigns a score to each user. Initially, the user's browsing behaviour in multiple aspects is extracted from the system log during non-attack cases. Then the entropy of requests per session is calculated. Entropy is an information theoretical concept, which is a measure of randomness. The entropy is employed in this paper to measure changes of randomness of requests in a session for a given time interval. Then, based on the visiting history [e.g., page viewing time, sequence of requested objects] of the user, a trust score is evaluated for each user.

In the second phase, detection of DDoS attack is carried out as follows: the entropy for the incoming requests in a session is calculated and the degree of deviation with the predefined value (computed in the first phase) is estimated. The greater the deviation, the more suspicious the session is. In addition, this paper also includes counter-attack mechanisms such as rate-limiter and scheduler to downgrade services to malicious users. If the session is found suspicious, then based on the trust score of the user, the session may be dropped or scheduled to get the service from the web server. Rate limiter sets proper thresholds and limits based on which filtering is applied. Scheduler then decides when to schedule the requests based on the system workload.

Fig.1 shows the system architecture. The detection mechanism is deployed at the server. A session connection request first reaches the system, and then the proposed scheme calculates the entropy deviation of request rate. If the deviation is more (exceeds threshold), then drop the session immediately. If the deviation is within limit, then the rate limiter filters the session based

on the trust score of the user. The client who behaves better in history will obtain higher degree of trust. If the user is considered legitimate, then the scheduler schedules the request based on the workload of the system. The highest trust score first policy is used to schedule the requests for the server.

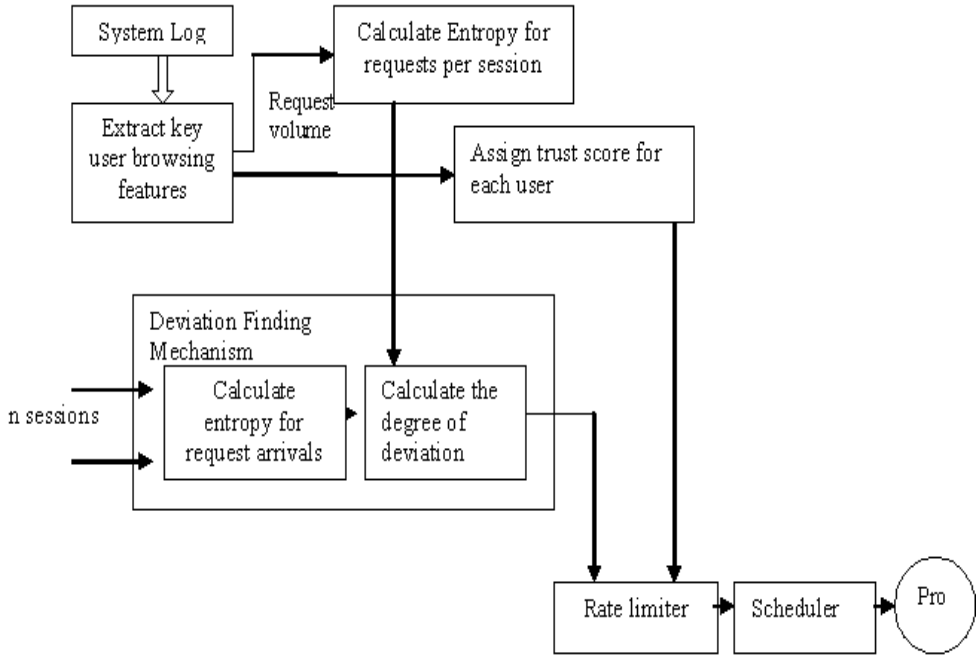

Figure 1. System Architecture

3.1. Entropy calculation

Let the request in a session be denoted as r_{ij}, where i, j $\in$ I, a set of positive integers. 'i' denotes the request number in session 'j'. Let $|(r_j,t)|$ denote the number of requests per session j, at a given time t. Then,

$$|(r_j,t)| = \sum_{i=1}^{\infty} r_{ij} \tag{1}$$

For a given interval Δt, the variation in the number of requests per session j is given as follows;

$$N_j (r_j, t+\Delta t) = |(r_j, t+\Delta t)| - |(r_j, t)| \tag{2}$$

The probability of the requests per session j, is given by

$$P_j(r_j) = N_j (r_j, t+\Delta t) / \sum_{i=1}^{\infty}\sum_{j=1}^{\infty} N_j (r_j, t+\Delta t) \tag{3}$$

Let R be the random variable of the number of requests per session during the interval Δt, therefore, the entropy of requests per session is given as

$$H(R) = -\sum_{j} P_j(r_j) \log P_j(r_j) \quad (4)$$

Based on the characteristics of entropy function, the upper and lower bound of the entropy H(R) is defined as

$$0 \leq H(R) \leq \log N \quad (5)$$

where N is the number of the requests.

Under DoS attack, the number of request increases significantly and the following equation holds

$$|H(R) - C| > \text{threshold, } t \quad (6)$$

Where C is the maximum capacity of the session.

3.2. Rate Limiter

To avoid falsely detection, rate-limiter is introduced. Once the entropy is calculated, compute the degree of deviation from the predefined entropy. The system first sets a threshold for acceptable deviation. If the computed deviation exceeds the threshold, then the session is forced to terminate immediately. Otherwise, second level filter is applied by the rate limiter. The system also defines a threshold for validating a user based on the trust score. A user is considered to be legitimate only if the trust score exceeds the threshold. Otherwise, the user is considered malicious and the session is dropped immediately. The legitimate sessions are then passed to the scheduler for getting service from the server.

3.3. Scheduler

If the user is legitimate, then the scheduler schedules the session based on the lowest suspicion first (user with highest trust score) policy. The well-behaved users will have a little or no deviation. In such case, the legitimate user gets a quicker service. In addition to the scheduling policy, system workload is also considered before scheduling the request for getting service.

3.4. Monitoring Algorithm

Input: system log

1. Extract the request arrivals for all sessions, page viewing time and the sequence of requested objects for each user from the system log.

2. Compute the entropy of the requests per session using the formula:

$$H(R) = -\sum_{j} P_j(r_j) \log P_j(r_j)$$

3. Compute the trust score for each and every user based on their viewing time and accessing behaviour.

3.5. Detection Algorithm

Input the predefined entropy of requests per session and the trust score for each user.

Define the threshold related with the trust score (T_{ts})
Define the threshold for allowable deviation (T_d)
For each session waiting for detection
Extract the requests arrivals
Compute the entropy for each session using (4)

$$H_{new}(R) = -\sum_{j} P_j(r_j) \log P_j(r_j)$$

Compute the degree of deviation:

$$D = |H_{new}(R)| - |H(R)|$$

If the degree of deviation is less than the allowable threshold ($T_{d)}$, and user's trust score is greater than the threshold (T_{ts}), then
Allow the session to get service from the web server
Else
The session is malicious; drop it

4. CONCLUSION

In this paper, an effective and efficient defense scheme against DDoS attacks based on information metric (entropy) is proposed. The proposed scheme provides double check point to detect the malicious flow from the normal flow. It validates the legitimate user based on the previous history. Based on the information metric of the current session and the user's browsing history, it detects the suspicious session. Once detected, a rate limiter and a scheduler are used to downgrade service to the malicious users and to schedule the less suspicious session based on the system workload and the user's trust level.

REFERENCES

[1] Shui Yu, Wanlei Zhou, Robin Doss, & Weijia Jia, (2011) "Traceback of DDoS Attacks using Entropy Variations", IEEE Transactions on Parallel and Distributed Systems.

[2] Supranamaya Ranjan, Ram Swaminathan, Mustafa Uysal, Antonio Nucci, & Edward Knightly, (2009) "DDoS-Shield: DDoS-Resilient Scheduling to Counter Application Layer attacks", IEEE/ACM Transactions on Networking, Vol. 17, No. 1.

[3] Huey-Ing Liu & Kuo-Chao Chang, (2011) "Defending systems Against Tilt DDoS attacks", 6th International Conference on Telecommunication Systems, Services, and Applications.

[4] Jin Wang, Xiaolong Yang & Keping Long, (2010) "A New Relative Entropy Based App-DDoS Detection Method", IEEE Symposium On Computers And Communications (Iscc).

[5] S. Yu, W. Zhou & R. Doss, (2008) "Information theory based detection against network behavior mimicking DDoS attack," IEEE Communications Letters, vol. 12, no. 4, pp. 319–321.

[6] Jie Yu, Chengfang Fang, Liming Lu & Zhoujun Li, (2009) "A Lightweight Mechanism to Mitigate Application Layer DDoS Attacks", in Proceedings of Infoscale'2009.

[7] G.Oikonomou & J.Mirkovic, (2009) "Modeling human behavior for defense against flash-crowd attacks", ICC2009.

[8] S.Kandula, D.Katabi, MJacob & A.W.Berger, (2005) "Botz-4-sale: surviving organized DDoS attacks that mimic flash crowds", in Proc. Second Symp. Networked Systems Design and Implementation (NSDI).

[9] J. Yu, Z. Li, H. Chen & X. Chen, (2007) "A Detection and Defense Mechanism to Defend Against Application Layer DDoS Attacks", in Proceedings of ICNS'07.

[10] Yi Xie & Shun-Zheng Yu, (2009) "Monitoring the Application-Layer DDoS Attacks for Popular Websites", IEEE/ACM Transactions on Networking, Vol. 17, No. 1.

[11] L. Limwiwatkul & A. Rungsawangr, (2004) "Distributed denial of service detection using TCP/IP header and traffi measurement analysis," in Proc. Int. Symp. Commun. Inf. Technol., Sappoo, Japan, Oct. 26–29, pp. 605–610.

[12] Haining Wang, Cheng Jin & Kang G. Shin, (2007) "Defense Against Spoofed IP Traffic Using Hop-Count Filtering", IEEE Transactions on Networking,vol.15.No.1, pp.40-53.

[13] Perrig A., Song D, & Yaar A., (2003) "StackPi: a new defense mechanism against IP spoofing and DDoS attacks", CMU technical report.

[14] Tanachaiwiwat, S. & Hwang, K., (2003) "Differential packet filtering against DDoS flood attacks." ACM Conference on Computer and Communications Security (CCS).

[15] Keromytis, A.D., Misra, V., & Rubenstein, D., (2004) "SOS: an architecture for mitigating DDoS attacks", Selected Areas in Communications, IEEE Journal vol. 22, no. 1.

[16] J. B. D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran & R. K. Mehra, (2001) "Proactive detection of distributed denial of service attacks using MIB traffic variables a feasibility study", in Proc. IEEE/IFIP Int. Symp. Integr. Netw. Manag., pp. 609–622.

[17] J. Yuan & K. Mills, (2005) "Monitoring the macroscopic effect of DDoS flooding attacks," IEEE Trans. Dependable and Secure Computing, vol. 2, no. 4, pp. 324–335.

[18] W. Yen & M.-F. Lee, (2005) "Defending application DDoS with constraint random request attacks," in Proc. Asia-Pacific Conf. Commun., Perth, Western Australia, pp. 620–624.