# A HYBRID APPROACH TO COUNTER APPLICATION LAYER DDOS ATTACKS

S. Renuka Devi and P. Yogesh

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

***ABSTRACT***

*Distributed Denial-of-Service (DDoS) attacks are a growing threat across Internet, disrupting access toInformation and services. Now a days, these attacks are targeting the application layer. Attackers are employing techniques that are very difficult to detect and mitigate. This paper proposes a hybrid detection scheme based on the trust information and information theory based metrics. Initial filtering is based on the trust value scored by the client. Then the information based metric, entropy, is applied for final filltering of suspicious flow. Trust value for a client is assigned by the server based on the access pattern of the client and updated everytime when the client contacts the server. The request from the client always includes this trust value to identify itself to the server. The Web user browsing behaviour (HTTP request rate, page viewing time and sequence of the requested objects) of the client is captured from the system log during non-attack cases. Based on the observation, Entropy of requests per session is calculated and used for rate limiting the flow further. A scheduler is included to schedule the session based on the trust value of the user and the system workload.*

***KEYWORDS***

*DDoS, Application Layer, Trust value &Entropy*

## 1. INTRODUCTION

A denial-of-service (DoS) attack is an attempt by attackers to prevent the legitimate users from using the information service. In a DDoS attack, these attempts come from a large number of distributed hosts that coordinate to flood the victim with an abundance of attack packets simultaneously. Distributed denial-of-service (DDoS) attacks present serious threats to servers in the Internet. DDoS attacks involve in saturating the target machine with requests, such that it cannot respond to legitimate traffic. Such attacks usually lead to a server overload.

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.

DOI:10.5121/ijcis.2012.2204 

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 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 attacksare 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). For each session request, the client attaches its trust value assigned by the server in the past session. New clients can directly send the request and they are assumed to be assigned with the lowest trust value by default. Based on the validity of the trust value, initial filtering is applied. Then 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 and dropped. Otherwise, the scheduler schedules the session. The key features of the proposed work are

- Two level of filtering is applied 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 DDoSdefense mechanism using trust value and 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

One way to protect from DoS attacks is to allow only authorized clients to access the web server. Compared with non-attack cases, the number of requests in a session increases significantly in a very short time period in DDoS attack cases. Considering the above two issues, a hybrid approach for countering application layer DDoSatttacks is proposed. This approach gives priority to the good (legitimate) clients, while severely limiting the access to the attackers.

Each client is assigned with a trust value by the server based on the access behavior. A client's trust value is embedded in a HTTP cookiethat is included in all server responses to the client. Using the cookie, a legitimate client can include the trust value in all its future requests to identify itself to the server. A client presenting a valid trust value to the server will be given the priority over other requests. New clients are assumed to be assigned with the lowest trust value by default by the server and updated in the response. The trust value varies according to the access pattern of the client. The trust values are assigned in such a way that

$$trust_{attacker} < trust_{new\ user} < trust_{legitimate\ user.}$$

In addition, 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. Entropy is applied as a second layer of filtering the suspicious flow. The second filtering mechanism is required to identify an attacker who acts like a legitimate client because, an attacker may behave benignly until it attains a highest trust value and then begin to misbehave.

The detection of DDoS attack is carried out as follows: Initially, the client embeds its trust value on the session request and sends it to the server. The server, on receiving the session request, validates the trust value. If valid, it forwards the request. Otherwise, the session is considered suspicious and dropped. Then the entropy for the incoming requests in a session is calculated and the degree of deviation with the predefined value is estimated. The greater the deviation, the more suspicious the session is. If the session is found suspicious, then it is assigned with the lowest trust value and dropped immediately. Otherwise, the requests are scheduled to get the service from the web server. The trust value is updated and embedded in the response message of the server for future use.

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 either drops or forwards the requests based on the trust value obtained in the past session, calculates the entropy deviation of request rate. If the deviation is more (exceeds threshold), then drop the session immediately. Otherwise, schedule the session based on the system workload and the trust value of the user. The client who behaves better in past session will obtain higher degree of trust. The highest trust value first policy is used to schedule the requests for the server.

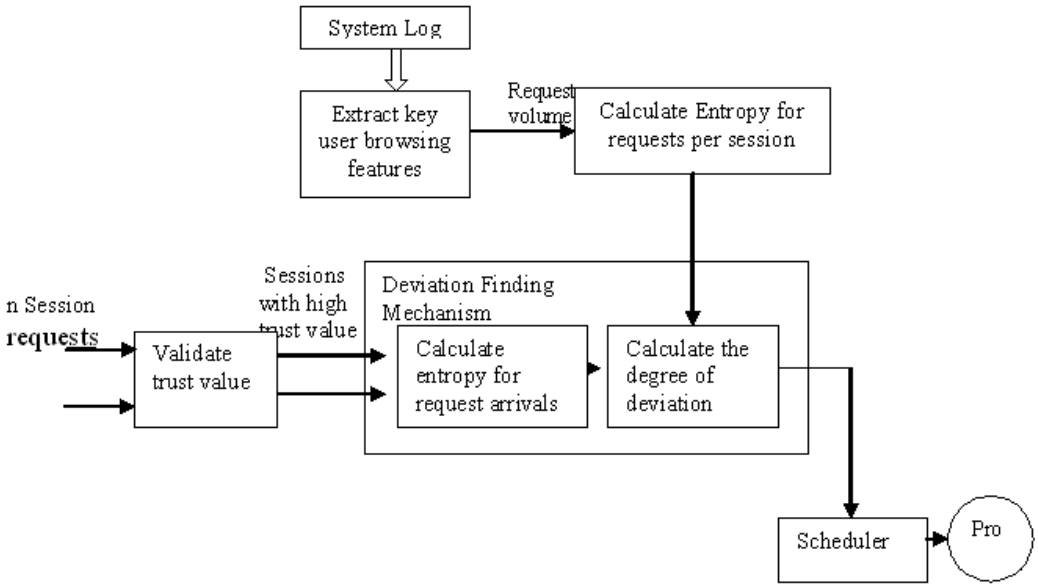


Figure 1. System Architecture

### 3.1. Trust value computation

Once the request is accepted, the request is forwarded to the application. When the server sends a response to the client, it updates the trust value as follows:

Let req be the client's request and res be the corresponding response generated by the server. Let $t_{rs}$be the time taken by the server to respond for the request *req* and *ut*denotes the utility of the request, *req*.

In this approach, a simple benefit function is used.

$$B(req) = ut - \gamma * t_{rs} \quad (1)$$

where $\gamma$ is a tunable parameter.

Here, additive increase multiplicative decrease strategy is used to calculate the new trust value.

If $B(req) > 0$, then the new trust value is computed as follows:

$$Trust_{new} = trust_{old} + \alpha * B(req) \quad (2)$$

Otherwise,

$$Trust_{new} = trust_{old} /(\beta(1-B(req))) \quad (3)$$

The additive increase ensures that the trust value slowly increases as the client behaves benignly; while the multiplicative decrease ensures that the trust value drops very quickly upon detecting a DoS attack from the client.

### 3.2. 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,

$$|(r_j,t)| = \sum_{i=1}^{\infty} r_{ij} \quad (4)$$

For a given interval $\Delta t$, the variation in the number of requests per session j is given as follows;

$$N_j(r_j, t+\Delta t) = |(r_j, t+\Delta t)| - |(r_j, t)| \quad (5)$$

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

$$P_j(r_j) = N_j(r_j, t+\Delta t) / \sum_{i=1}^{\infty}\sum_{j=1}^{\infty} N_j(r_j, t+\Delta t) \quad (6)$$

Let R be the random variable of the number of requests per session during the interval $\Delta t$, therefore, the entropy of requests per session is given as

$$H(R) = -\sum_{j} P_j(r_j) \log P_j(r_j) \quad (7)$$

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

$$0 \le H(R) \le \log N \quad (8)$$

where N is the number of the requests.

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

$$|H(R) - C| > \text{threshold, } t \quad (9)$$

Where C is the maximum capacity of the session.

### 3.3 Scheduler

If the user is legitimate, then the scheduler schedules the session based on the highest trust value first (user with highest trust value) 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. Algorithm to compute the entropy from system log

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:

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

### 3.5. Detection Algorithm

Input the predefined entropy of requests per session
Define the threshold for allowable deviation ($T_d$)

For each session waiting for detection
    Extract the trust value from the request
    Validate the trust value
    If the trust value issued is valid
        Extract the requests arrivals
            Compute the entropy for each session using (7)

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

            Compute the degree of deviation:

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

            If the degree of deviation is less than the allowable threshold ($T_d$), then
                Allow the session to get service from the web server
            Update the trust value
            Embed the trust value in the response message and send it to client
        Else
            The session is malicious; drop it
    Else
        Assign the lowest trust value to the client
        Drop the session

# 4. CONCLUSION

In this paper, an effective and efficient hybrid scheme against DDoS attacks based on trust value and information metric (entropy) is proposed. The proposed scheme provides double check point to detect the malicious flow from the normal flow. This approach not only counters the illegitimate flows but also avoids the flooding of the legitimate flows. Trust value is used to detect the legitimate user from the attackers at the first level. Then, based on the information metric of the current session, the sessions that are assumed to be suspicious are dropped. The legitimate flows are then scheduled by the scheduler based on the system workload the trust value of the client. Thus the legitimate clients gets more priority in accessing the information and services.

## REFERENCES


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

[2] SupranamayaRanjan, 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 DDoSattack," 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.