Received: April 11, 2017

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Suspicious Score Based Mechanism to Protect Web Servers against Application Layer Distributed Denial of Service Attacks

Renuka Devi Saravanan[1]*, Shyamala Loganathan[1], Saraswathi Shunmuganathan[2], Yogesh Palanichamy[3]

[1]*Vellore Institute of Technology University Chennai, India*
[2]*SSN College of Engineering, India*
[3]*Anna University, India*
* Corresponding author's Email: renukadevi.s@vit.ac.in

Abstract: Distributed Denial of Service attacks are becoming a serious issue for the developers and the users of the Internet. In recent times, the attackers are targeting the online applications and web services. Detecting such application level attacks are much challenging because the attack traffic mimics the legitimate behaviour. A more sophisticated mechanism is required to detect and mitigate such attacks. In this paper, a novel method for detecting application layer Distributed Denial of Service attack is proposed. Initially, web user behaviour on different perspectives is analyzed using the system log and key dimensions that are highly responsive to attacks are identified using Principal Component Analysis. The extracted key features are analyzed to fix up the appropriate thresholds for differentiating legitimate and illegitimate access. Each incoming session is examined and if found suspicious, the detection mechanism is invoked. The detection mechanism includes a score assignment mechanism which assigns the threat score based on the history and statistical analysis of the current characteristics. The sessions having acceptable score are then scheduled to get service from the server. Remaining sessions are considered malicious and dropped. The real data sets are taken for the simulation and the results are exhibited to show the efficiency of the proposed detection method. The results show that the proposed technique performs effective detection of constant flooding and repeated shot attacks with low false positives and low false negatives.

Keywords: Denial of service, Attack, Flooding, Application layer, Security, Botnet.

1. Introduction

The Internet plays a vital role in our day-to-day life. Due to its open access nature, Internet is prone to various kinds of attacks. One such threatening attack is Denial of Service attack (DoS). A DoS attack is an attempt to prevent a system offering resources or services to its intended users. The attackers especially target the victim's resources, such as network bandwidth, sockets, computing power, disk bandwidth, and I/O bandwidth. They launch the attack by flooding the victim with enormous illegitimate flows. When these attacks arise from a large number of distributed hosts,it is said to be Distributed Denial of Service (DDoS) attack. DDoS attacks have substantially more impact than the normal DoS attack because they increase the attack intensity with large of number of attack sources simultaneously. Kaspersky DDoS Intelligence Report [1] reveals that 67 countries were targeted by DDoS attacks in the third quarter of 2016. The longest DDoS attack lasted for about 184 hours.

To target a server, the DDoS master attacker requires a large number of distributed compromised hosts so that a huge volume of traffic can be generated to deny the service of the victim server to its intended clients. As a first step, the attacker discovers vulnerable systems on the network. The end systems that are running without antivirus or with expired antivirus software are considered to be the vulnerable systems. The attackers make use of

the vulnerability to gain access to the hosts. Then, the programs containing attack tools are installed on the compromised hosts. The compromised hosts are now called as bots, and large number of bots together forms a botnet. This botnet are then used to flood the victim under the control of the master attacker [2-3].

Traditionally, the attackers were targeting the network layer to exhaust the network bandwidth of the victim. SYN flooding, UDP flooding, and ICMP flooding are few among the traditional network based DDoS attacks. Recently, the attackers are focussing on the application layer targeting the online services and the applications running on the web servers. The defense mechanisms developed for network based attacks do not suit for defending the application layer based DDoS attacks because of the following reasons [4,5]: 1) The requests originating from the bots imitate the legitimate clients' request, thereby making the detection much harder, 2) The attackers impersonate as legitimate clients during the flash crowd.

In this paper, a novel approach for detecting and mitigating the high rate application layer DDoS attacks is proposed. Initially, the web user behaviour with respect to the applications running on the web server is analyzed with the help of system log and the thresholds for various key parameters are fixed in order to differentiate the legitimate and malicious users. These thresholds are then used for early detection of the DDoS attack. Then the defense mechanism is invoked to check whether the server is really targeted. The defense mechanism includes a score assignment mechanism which assigns the threat score to the suspected in-coming sessions based on the current characteristics of the session and the previous behaviour of the user. The sessions with the acceptable score are then scheduled by the scheduler based on the earned score and the rest of the sessions are considered malicious and dropped immediately.

The rest of the paper is organized as follows. Section 2 reviews the related work which has been done so far in detecting Application layer DDoS attacks. Section 3 explains the functionality of all the components, sample network topology and assumptions that are made in the proposed work. Section 4 presents the simulation results of the proposed technique and also focuses on the performance evaluation of the proposed technique and Section 5 provides the summary and possible extension of the proposed work.

2. Related works

Most of the research works have focused on detecting and mitigating network layer based DDoS attack. However, these methods will not suit well for application layer based DDoS attacks as mentioned in the previous section. Related to the application layer, Botz-4-sale [6] provides a kernel based solution to protect the server from DDoS attacks. This method identifies the malicious user by assigning a graphical test (Completely Automated Public Turing test to tell Computers and Humans Apart). This scheme assumes that humans can easily pass through the test but the bots cannot. The major drawback is the delay associated with the test and it may annoy the user. Yi Xie and Shun Zheng Yu [4] proposed a technique based on document popularity. The proposed scheme monitors the web traffic in order to find the dynamic variation in normal traffic. Based on the variation in the entropy on observed values, a hidden semi markov model based anomaly detector is used to detect the attacks. The main drawback of this approach is the high complexity.

DDoS Shield [7] makes use of a counter mechanism in which each session is assigned a continuous value based on the deviation from legitimate profile. The sessions with acceptable scores are then forwarded to the server. Yu et al. [8] proposed a Trust Management Helmet scheme where each user is assigned a license and trust. The attack is detected based on these two parameters. The overhead associated with processing and maintaining the license and trust value for each user is quite high in this scheme.

Liu et al. [9] proposed a mechanism against Tilt DDoS attacks in which behaviour analyser is used to extract the web user's behaviour and classifies the user as malicious or legitimate user. Based on the classification, it provides differentiated services to the end users. Yadav and Subramanian [10] proposed Deep learning, a neural network based approach to identify the significant features of Application layer DDoS attack and to detect the attack based on the extracted features. Yadav and Selvakumar [11] proposed a method based on feature construction and logistic regression to differentiate the legitimate and illegitimate user behavior. The above mentioned works [9-11] depend on the signature based detection which may not be suitable for detecting the newly emerging attacks.

Entropy-based clustering and likelihood analysis are used to distinguish legitimate and illegitimate sequences [12]. Yu et al. [13] proposed a Defense and Offense Wall (DOW) mechanism which relies

on the detection and currency technology. Detection technique drops the suspicious session whereas currency technique favours legitimate sessions thereby providing a level of guaranteed service to legitimate users. These mechanisms [12,13] mainly focus on the incoming traffic characteristics.

Srivatsa et al. [14] proposed a server-side middleware solution in which each client is assigned a priority by the server based on their past behaviour. Based on the priority level, differentiated services are provided to the users. Each request is evaluated through application specific knowledge and the client's priority is recomputed by the server for further reference. Lee et al. [15] proposed a detection mechanism based on the sequence of the requested web page. The web user behaviour is summarized by applying multiple principal component analysis and its reconstruction error is used as a key factor for detecting DDoS attacks. Beitollahi and Deconinck [16] proposed a mitigation scheme in which each connection is examined and scored based on their past access behaviour. The resources consumed by the connections having least scores are taken back by the server. Zolotukhin et al [17] proposed an anomaly-based approach for detecting application-layer DoS attacks that utilize cryptographic protocols. In the above mentioned works [15-17], the attack detection is purely based on the web access history.

From all the above related works, it is observed that, in majority of works, web user behavior is analyzed to differentiate the attack traffic from the legitimate traffic and accordingly filtering is applied for protecting the victim. On the other hand, the detection principle is based on the incoming traffic characteristics and the current system workload. Based on the above observations, a novel method incorporating the essence of both approaches (previous history and the current characteristics) is proposed in this paper to protect the web server from application layer DDoS Attacks.

3. Proposed work

In this paper, an efficient mechanism is proposed to defend against the application layer DDoS attacks. The overall system architecture is shown in Fig. 1. The proposed mechanism has two phases: offline phase and detection phase. In offline phase, the user behaviour and the application-related characteristics are extracted from the system log. The key features differentiating legitimate and illegitimate users are identified and the respective thresholds are fixed. In the detection phase, the incoming sessions are

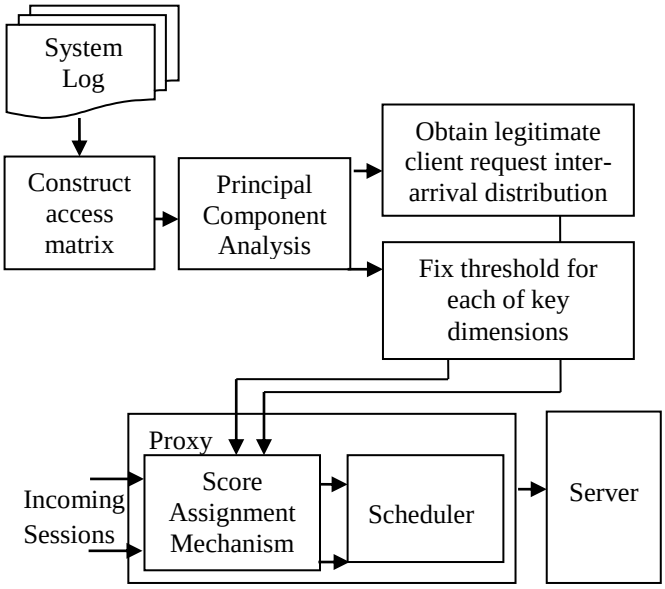


Figure. 1 System architecture

examined to identify the suspicious sessions. The suspected sessions are then evaluated based on the information collected during the offline phase and the threat scores are assigned. Based on the threat score, the requests are either dropped or forwarded to the scheduler. The scheduler then forwards the request to the server based on the scheduling policy. The detection mechanism is executed on the proxy system which is placed just before the server so that the malicious requests can be intercepted before reaching the server. The proposed mechanism is simulated and a comparative study is made to show the performance of the proposed detection technique.

3.1 Access matrix

Initially, web user behaviour and other characteristics with respect to each and every application running on a web server are analyzed to identify the key parameters for detecting the attack. This offline process requires a data structure for organising the data retrieved from the system log. One such data structure is the access matrix. The access matrix is constructed by extracting data from the system log. The access matrix is of size N x T, where N denotes the number of applications that are available in the server and T denotes the number of attributes considered. This Multidimensional matrix can be compressed to have linearly uncorrelated values by applying the dimensionality reduction technique.

3.2 Principal component analysis

Principal Component Analysis (PCA) [18, 19] is one of the common dimensionality reduction

techniques for identifying key features from the high dimensional data set. It transforms a very large number of correlated variables into a small number of uncorrelated variables. The uncorrelated variables are called principal components.

In this paper, PCA is used to extract the key features such that the combination of these features must be able to highlight the attack sessions from the legitimate sessions. Once the key features are identified, then the reasonable threshold for each of the key features is fixed in order to differentiate the legitimate and the illegitimate access behaviour. The PCA works as follows:

Let the sample (row) vector of the access matrix be

$$X_i = [x_{i1}, x_{i2},.. ,x_{iT}] \quad (1)$$

Let μ be the mean vector,

$$\mu = [y_1, y_2,\ldots y_T] \quad (2)$$

$$\text{where} \quad y_j = \left(\sum_{i=1}^{N} [x_{ij}] \right) / N \qquad 1 \le j \le N \quad (3)$$

where N is the total number of samples in the access matrix and T is the total number of attributes.
The mean subtracted matrix is given by

$$X_S = \sum_{i=1}^{N} (X_i - \mu) \quad (4)$$

The covariance matrix of the sample data set is

$$C = X_S X_S^T / (N-1) \quad (5)$$

The covariance matrix is always symmetric. An orthogonal basis can be calculated by finding its eigenvalues and eigenvectors which inturn can be obtained by finding the solutions of the characteristic equation

$$|C - \lambda I| = 0 \quad (6)$$

where I is the identity matrix and | | denotes the determinant of the matrix.

Once eigenvectors are found, arrange them in the order of descending eigen values to indicate the order of significance. The eigenvector with the highest eigen value is the first principal component of the data set. The eigenvector with the next highest eigen value forms the second principal component of the data set and so on. With this order, the components with lowest eigen values can be ignored. The key dimensions are extracted by choosing the first 'f' feature (eigen) vectors. If the row vector having the first 'f' eigenvectors is denoted by F, then the resultant uncorrelated data set (Y) is given by

$$Y = F \; X_S \quad (7)$$

The request rate, session rate and revisitation (reference) count are the core key features identified as a result of applying PCA. The reason for choosing PCA in this paper are 1) it is completely non-parametric, 2) it is an optimal technique for compressing a multi-dimensional data, 3) compression and decompression can be easily performed 4) No s pecial assumptions are required.

3.3 Detection mechanism

The detection mechanism is executed on the proxy system which exists just before the server so that the suspicious requests can be identified and dropped before it floods the server. The initial detection is made based on the information that was extracted during the offline phase. The final decision is based on the threat score earned by each of the suspicious session.

For every specific interval of time, the key features of the incoming traffic are extracted and examined against their respective threshold values which are predefined in the offline phase. If they exceed the threshold value then the incoming flow is considered as suspicious. This is the initial stage of identifying the attack. On detection of the suspicious session, score assignment mechanism is invoked to make sure whether the incoming request is really a suspicious one. It assigns threat score for each session based on the previous behaviour and the current characteristics of the incoming session. Score assignment mechanism is explained in section 3.3.1. The threat score indicates the level of suspiciousness of the given session. Based on the threat score and the system workload, the scheduler may either forward or drop the session.

Sample Network Topology

The sample network with DDoS attacks to demonstrate the proposed detection mechanism is shown in Fig. 2. The sample network consists of three distributed LAN sites that are connected together through Internet. In the DDoS attack scenario as shown in Fig. 2, the nodes 2, 6 and 11 are considered as DDoS attackers and the rest of the

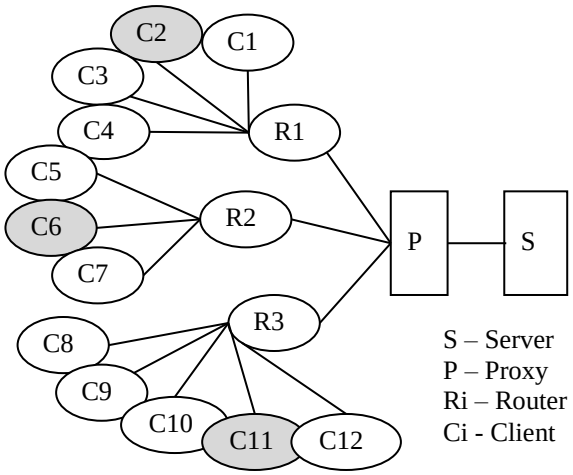


Figure. 2 Sample network model

nodes are considered as legitimate users. As mentioned before, the detection mechanism is implemented at the proxy P.

Assumptions

The following are the assumptions made to make the analysis simple and clear. It is assumed that:

- Only one DDoS attack targets the victim at any point in time.
- The network system is stable.
- The system log is stationary.
- The system logs used in the offline phase are not influenced by attackers.

3.3.1 Score assignment mechanism

The incoming sessions are monitored for every specific time interval. The system is said to be under attack if most of the key parameters exceed the respective thresholds. The sessions having request rate greater than the threshold (Tr) is considered malicious. The threshold (Tr) denotes the average request rate per session with respect to a particular application. The detection mechanism is then invoked to examine and rate the level of suspiciousness of each of the suspected sessions by computing the threat score. The score is assigned based on two parameters: request inter-arrival time and access history of that particular application by the user of the suspected session. The computation of the threat score based on the above mentioned parameters are described below.

Computing score with respect to the request inter-arrival time

The high rate DDoS attack succeeds by sending a huge number of requests then the usual scenario. The threat score is computed based on the deviation measured between the incoming request inter-arrival time and the ideal request inter-arrival time extracted during offline phase. In this paper, the Hellinger distance is used as the distance metric to find the deviation between the two probability distributions and is represented as $D_H(p,q)$ where p and q are the two probability distributions. The Hellinger distance is used in this paper in order to overcome the asymmetric properties of other popular distance metrics like Kullback–Leibler and information divergences [20].

The aggregate request inter-arrival times are extracted from the system log during non-attack case to obtain the sample ideal distribution. On detection of the flooding, the request inter-arrival time from each and every suspicious session is sampled for every Δt time within the sampling period of time, say T. Let $x_i[1], x_i[2],..., x_i[n]$ be the sequence of samples from a suspicious session flow where i $(i \geq 1)$ is the index of suspicious session, and $n = T / \Delta t$ (denotes the number of samples per session).

Let p(x) be the ideal distribution of the request inter-arrival time which is derived from the system log during the offline phase and q(x) be the probability distribution of the request inter-arrival time of one of the suspected session with the following condition

$$\sum_{x=1}^{n} p(x) = \sum_{x=1}^{n} q(x) = 1 \quad (8)$$

Where $1 \geq p(x) \geq 0, 1 \geq q(x) \geq 0$.

Then the Hellinger distance between the given two distributions is computed as in Eq. (9) [21]:

$$D_H(p, q) = [\sum_{x} (\sqrt{p(x)} - \sqrt{q(x)})^2]^{1/2} \quad (9)$$

In general, the values of the Hellinger distance metric range from 0 to 1.414. The values are then normalized to have the scores between 0 and 1 as given in Eq. (10).

$$score_{dis} = D_H(p, q) / \sqrt{2} \quad (10)$$

In general, the request inter-arrival time distribution observed for the incoming legitimate flow will not vary much from the ideal distribution and the score, if computed, will be 0 or close to 0. But, for the attack scenario, the deviation will be more (towards 1) which in turn indicates the suspicious level of the examined session.

Computing score based on the reference count

This score represents the legitimacy of the user on a given application. The score is based on the frequency of referring a particular application by the user. For each and every access to an application, the user gains a fraction of trust score. The score is computed as follows:

Let n be the total number of users for an applications running on a server for a specific period of time, $r_{i,j}$ be the number of references made by the user 'i' on an application 'j' within the given time period.

The probability that a user 'i' accessed a particular application 'j' is given by

$$P(u_{i,j}) = r_{i,j} / \sum_{i=1}^{n} r_{i,j} \quad (11)$$

This value indicates the legitimacy (trust) level of the user. Therefore, the threat score can be calculated from the trust score as given in Eq. (12).

$$score_{ref} = \alpha (1 - P(u_{i,j})) \quad (12)$$

The tuning parameter 'α' is used to normalize the threat score to be within the range of 0 to 1.

Likewise, each user is assigned a score. On the other hand, a new user is assigned with a default score in such a way that the score of the new user lies in between the scores of the legitimate and malicious users. i.e.,

$$score_{ref\ (legi)} < score_{ref(new)} < score_{ref(attacker)} \quad (13)$$

Computing the overall score

The overall score for a session can be calculated by aggregating the threat score computed across different dimensions. The score is always within the value 0 and 1and the score is computed as in Eq. (14).

$$score = \beta (score_{dis}) + (1- \beta) score_{ref} ; 0 \leq \beta \leq 1 \quad (14)$$

The weighting parameter β can be set according to which of the two parameters has greater potential for attack. In this paper, the parameter based on the history (ie., reference count) is given 0.4 weightage because the users' revisitation history cannot be given equal weightage with that of current characteristics especially during the flash crowds and the remaining weightage 0.6 is assigned to the parameter associated with the current characteristics (ie., request inter-arrival time). Hence β is set to the value '0.6'.

3.3.2 Scheduling policy

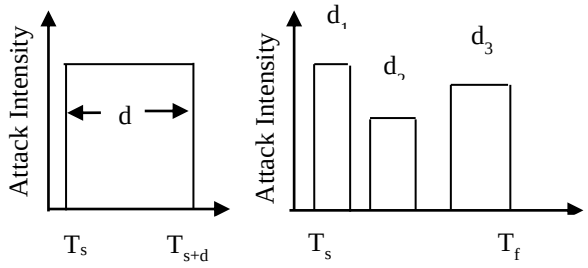
The legitimate sessions are directly sent to the scheduler which in turn forwards it to the server with the default scheduling policy. On the other hand, the suspected sessions are assigned the threat scores and the sessions with high threat score are dropped immediately. The sessions having the scores within acceptable level are forwarded to the server based on the lowest threat score first scheduling policy. By default, all the legitimate users are assigned a value of 0 as a threat score.

4. Experimental results and performance evaluation

In this section, the efficiency of the proposed mechanism is evaluated. NS2 simulator is used to implement the proposed work. The simulation includes 100 nodes, out of which attack nodes are randomly chosen and the percentage of attack nodes to the total nodes is approximately 20%. The real data sets are used for simulating the legitimate and attack traffic. For generating the legitimate traffic, traces of FIFA World CUP data set [22] is used. In this paper, user's trace collected from day 90 of the FIFA World Cup 98 [22] is used. For generating the attack traffic, we used the traces collected from the CAIDA "DDoS Attack 2007" Dataset [23]. The simulation is carried out by varying the attack intensity from 25% to 150%.

In this paper, two types of attack are considered: constant flooding attack and repeated shot attack. The constant flooding attack refers to a situation where the attackers flood the victim with a constant request rate for a period of time as shown in Fig. 3(a). The attack intensity and the duration of the attacks are set randomly by the attack sources. The repeated shot attack sends a burst of attack traffic with a varying time period as shown in Fig. 3(b). The duration of the attack is short and variable but the attack intensity is very high. The interval between the attacks is randomly set for each of the attack nodes.

In normal case, the legitimate requests are directly scheduled by the scheduler without invoking the detection mechanism. Once the flooding attack is suspected, then 80% of the scheduler's backlog queue is allocated for the identified genuine sessions and the remaining 20% is allocated for the suspicious session so that no session is dropped immediately without examining



T_s – Starting time of the attack
T_f – Finishing time of the attack
d – Duration of the attack
d_1, d_2, d_3 – Duration of each shot

(a) (b)

Figure. 3 DDoS Attacks: (a) Constant flooding attack and (b) Repeated shot attack

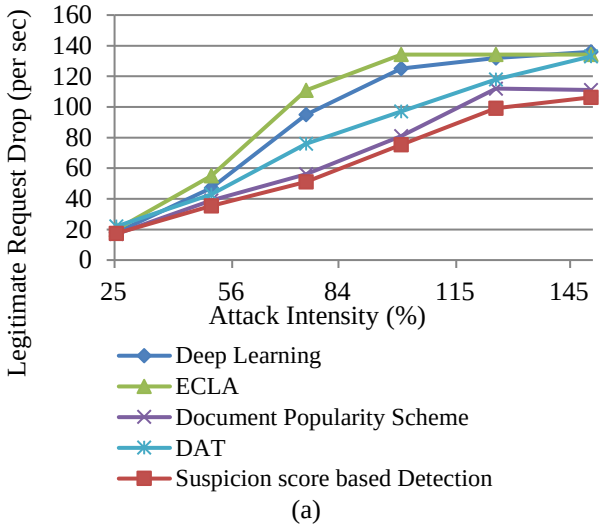
its level of suspiciousness. This allocation is made to optimize the detection. If the legitimate allocation is less than 80% then the drop rate of the legitimate requests is more which will not yield a good result. If the allocation is more than 80%, then there is no much variation in the legitimate acceptance rate but the drop rate of the suspicious requests is more. Hence the 80-20 allocation is made to improve the utilization of the available space by the suspected sessions without affecting the legitimate sessions. By this way, the genuine user gets guaranteed service at all times and the attacker gets limited response from the server.

The 20% of the scheduler's backlog queue can be utilized by the suspected sessions. The level of utilization by the suspected sessions is based on their individual score. Let the total number of suspected session in a specific period of time is N_s and the score earned by the suspected session i is $score_i$. Then each session gains its share of the available space as given in Eq. (15).

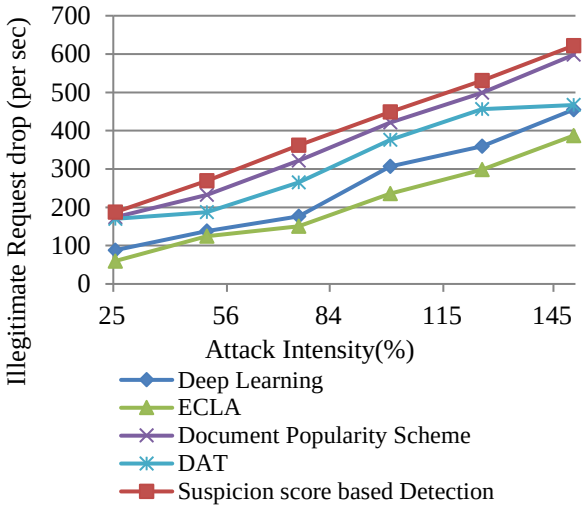
$$share_i = (20\% \text{ of scheduler backlog queue}) \, score_i \Big/ \sum_{i=1}^{Ns} score_i \quad (15)$$

In this way, even the suspected sessions get certain level of access to the server. In addition, the legitimate users get the guaranteed service even during attack.

The proposed mechanism is compared with the Deep Learning Mechanism [10], Entropy based Clustering and Likehood Analyasis (ECLA))[12], Defence against Tilt DoS Attack (DAT) [9] and Document Popularity Scheme [4] and the results are presented in this section to show the performance of the proposed work.



(a)



(b)

Figure. 4 Drop Rate for the Constant flooding attack: (a) Legitimate drop and (b) Illegitimate drop

Fig. 4(a) and Fig. 4(b) show the legitimate and illegitimate request drop rate respectively for the constant flooding attack with the existing and the proposed techniques. Fig. 5(a) and Fig. 5(b) show the legitimate and illegitimate request drop rate respectively for the repeated shot attack with the existing and the proposed techniques.

In addition to the above mentioned types of attack, the performance of the proposed detection mechanism is analyzed by considering both DoS attack (by single attacker) and DDoS attack (by distributed attackers) with the same attack intensity. Fig. 6(a) and Fig. 6(b) shows the result of the legitimate and illegitimate request drop for both cases considering the constant flooding attack. The drop rate of the legitimate and the illegitimate request are less in case of distributed attacker because the distributed attacks are highly difficult to

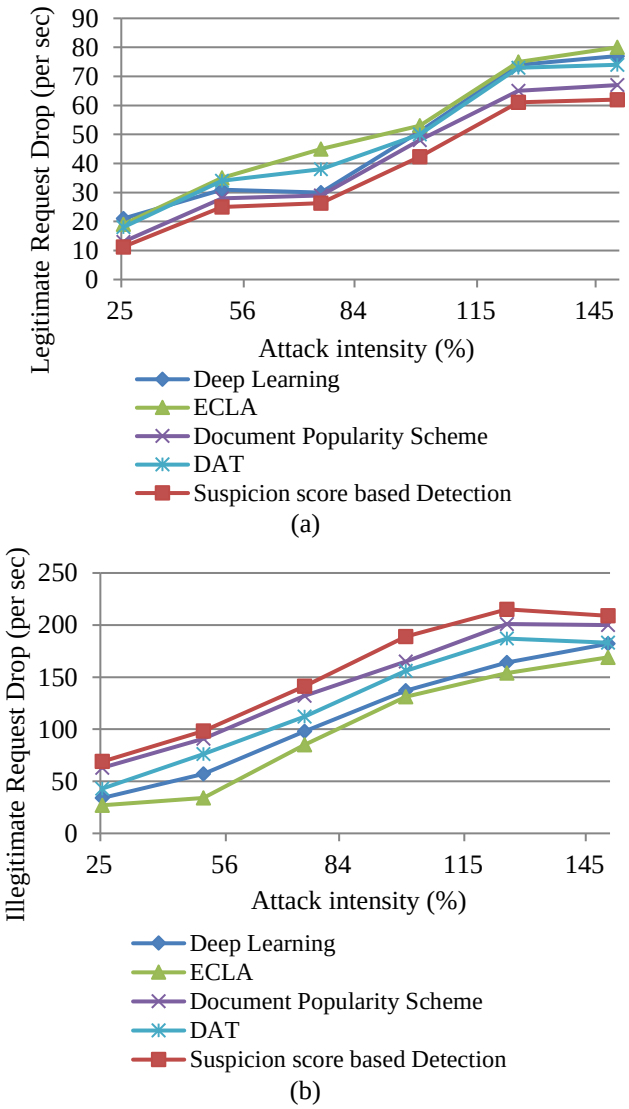


Figure. 5 Drop Rate for the Repeated Shot attack: (a) Legitimate drop and (b) Illegitimate drop

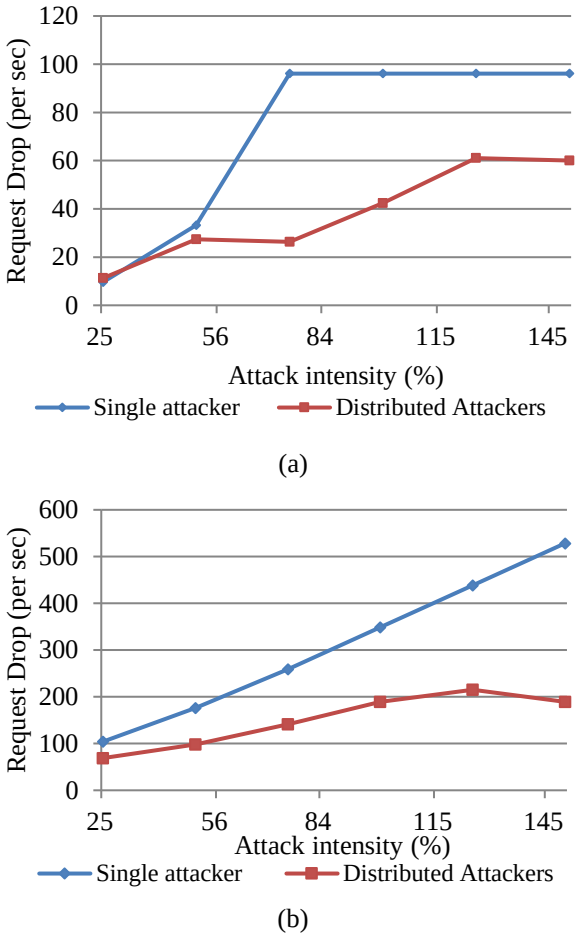


Figure. 6 Drop Rate with respect to constant flooding single attacker(DoS) and Distributed attackers(DDoS): (a) Legitimate drop and (b) Illegitimate drop

be detected when compared to a single source due to the wide distribution of attack traffic. In addition, all suspected sessions cannot be dropped immediately. Fig. 7(a) and Fig. 7(b) shows that the proposed system performs well by accepting large number of the legitimate request and rejecting the illegitimate request to the maximum. The false positives and the false negatives are very low in the proposed technique as shown in Fig. 7(c) and Fig. 7(d). From the above results, it is clear that the proposed technique performs effective detection of constant flooding and repeated shot attacks with low false positives. It also provides guaranteed service to the legitimate users.

5. Conclusion and future work

Protecting web servers from the DDoS attacks is an essential task. A good DDoS defense mechanism should be simple to implement and easy to deploy. Detecting attacks based on either the past history or the current characteristics of the incoming traffic may not be a complete solution. Instead, the essence of both the methods can be combined together to form an efficient detection mechanism. One such novel mechanism is proposed in this paper. Initially, web user characteristics are analyzed from the system log and the key features are extracted using the principal component analysis technique. The key features are used for early detection of the DDoS attack. Further, the incoming session characteristics are examined and their level of suspiciousness is computed to clearly differentiate the illegitimate users from the legitimate users. Based on this measure, the session is either served or dropped. This mechanism is implemented on the proxy system in order to avoid the unnecessary flooding at the server. The resilient schedulers can be employed for improving the reliability of the overall mechanism. The simulation result shows that the

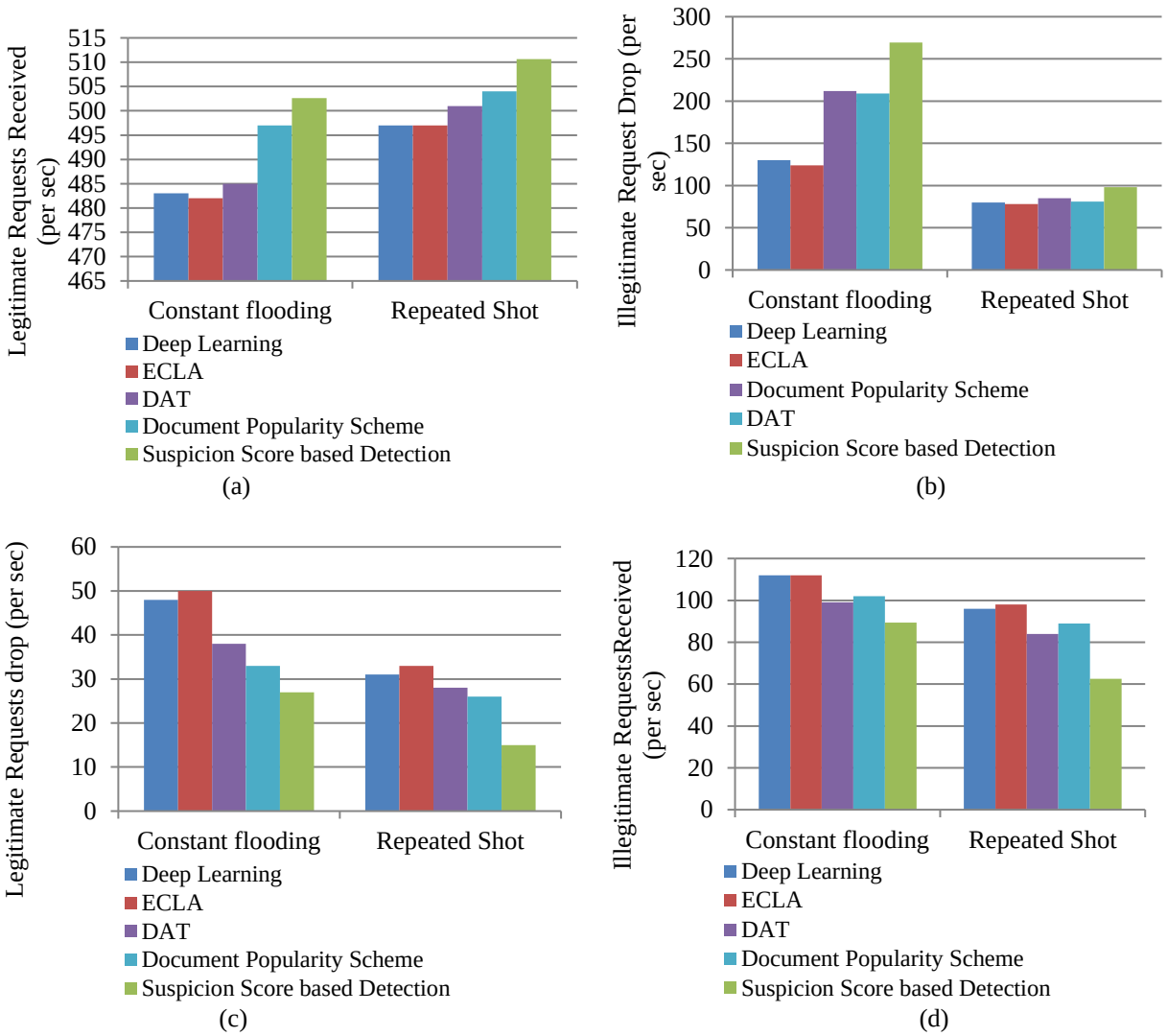


Figure. 7 Comparison between Constant flooding attack and repeated shot attack: (a) Legitimate Accepted, (b) Illegitimate Drop, (c) False Positives, and (d) False Negatives

proposed technique helps in effective detection of DDoS attacks and provides guaranteed service to legitimate users.

The work can further be extended by deploying resilient proxies to improve the reliability of the overall mechanism. Also, we are interested in protecting the web server from super botnet attacks where multiple botnets target a server at the same time.

References

[1] Kaspersky DDoS Intelligence Report for Q3, 2016, https://securelist.com/analysis/quarterly-malware-reports/76464/kaspersky-ddos-intelligence-report-for-q3-2016/

[2] D. Moore, C. Shannon, and D.J. Brown, "Inferring Internet Denial-of-Service Activity", *ACM Transactions on Computer Systems*, Vol. 24, No. 2, pp. 115-139, 2006.

[3] E. Alomari, S. Manickam, B. Gupta, S. Karuppayah, and R. Alfaris, "Botnet- based Distributed Denial of Service (DDoS) Attacks on Web Servers: Classification and Art", *International Journal of Computer Applications*, Vol. 49, No. 7, 2012.

[4] Y. Xie, and S. Z. Yu, "Monitoring the Application-Layer DDoS Attacks for Popular Websites", *IEEE Transactions on Networking*, Vol. 17, No.1, pp. 15-25, 2009.

[5] C. Xu, C. Du, and X. Kong, "An Application Layer DDoS Real-Time Detection Method in Flash Crowd", In: *Proc. of School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications*, , Vol. 30, pp. 68-73, 2012.

[6] S. Kandula, D. Katabi, M. Jacob, and A.W. Berger, "Botz-4-sale: surviving organized DDoS attacks that mimic flash crowds", In: *Proc. of 2nd conf. On Symposium on Networked Systems*

Design & Implementation, USENIX Association, Vol. 2, pp. 287-300, 2005.

[7] S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci, and E. Knightly, "DDoS-Shield: DDoS-Resilient Scheduling to Counter Application Layer attacks", *IEEE Transactions on Networking*, Vol. 17, No. 1, pp. 26-39, 2009.

[8] J. Yu, C. Fang, L. Lu and Z. Li, "A Lightweight Mechanism to Mitigate Application Layer DDoS Attacks", In: *Proc.* of *International Conf. On Scalable Information Systems*, *Springer Berlin Heidelberg of Infoscale,* pp. 175-191, 2009.

[9] H. Liu, and K. Chang, "Defending systems Against Tilt DDoS attacks", In: *Proc. of International Conf. On Telecommunication Systems, Services, and Applications*, *IEEE*, pp.22-27, 2011.

[10] S. Yadav, and S. Subramanian, "Detection of Application Layer DDoS attack by feature learning using Stacked AutoEncoder", In: *Proc. of International Conf. On Computational Techniques in Information and Communication Technologies,* pp: 361 – 366, 2016.

[11] S. Yadav, and S. Selvakumar, "Detection of application layer DDoS attack by modeling user behavior using logistic regression", In: *Proc. Of International Conf. On Reliability, Infocom Technologies and Optimization*, pp. 1 – 6, 2015.

[12] P. Chwalinski, R. Belavkin, and X. Cheng, "Detection of application layer DDoS attack with clustering and likelihood analysis", In: *Proc. of IEEE Globecom Workshops*, pp: 217 – 222, 2013.

[13] J. Yu, Z. Li, H. Chen, and X. Chen, "A Detection and Offense Mechanism to Defend Against Application Layer DDoS Attacks", In. *Proc.of International Conf. on Networking and Services*, *IEEE*, pp. 54-54, 2007.

[14] M. Srivatsa, A. Iyengar, J. Yin, and L. Liu, "A Middleware System for Protecting Against Application Level Denial of Service Attacks", *International Federation for Information Processing, LNCS 4290*, pp. 260–280, 2006.

[15] 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*, No. 1, pp. 50, 2011.

[16] H. Beitollahi and G. Deconinck, "Tackling Application-layer DDoS Attacks", *Procedia Computer Science*, Vol. 10, pp.432-441, 2012.

[17] M. Zolotukhin, T. Hämäläinen; T. Kokkonen, and J. Siltanen, " Increasing web service availability by detecting application-layer DDoS attacks in encrypted traffic", In: *Proc. Of International Conf. On Telecommunications, IEEE*, pp.1-6, 2016.

[18] L. I. Smith, "A tutorial on Principal Components Analysis", University of Otago, New Zealand, 2002.

[19] I. K. Fodor., A survey of dimension reduction techniques, *LLNL technical report*, 2002.

[20] A. Garrido, and Facultad de Ciencias de la UNED, "About some properties of the Kullback-Leibler divergence", *Advanced Modeling and Optimization*, Vol. 11, No. 4, 2009.

[21] K. Li, W. Zhou, P. Li, J. Hai, and J. Liu, "Distinguishing DDoS Attacks from Flash Crowds Using Probability Metrics", In: *Proc. of International Conf. on Network and System Security*, *IEEE*, pp. 9-17, 2009.

[22] "FIFA World Cup 1998 dataset", http://ita.ee.lbl.gov/html/contrib/WorldCup.html

[23] "The CAIDA UCSD "DDoS Attack 2007" dataset, http://www.caida.org/data/passive/ddos-20070804_dataset.xml