AN IMPROVED BIO-INSPIRED BAT ALGORITHM FOR DETECTION AND PREVENTION OF HTTP FLOOD DDOS ATTACK USING MACHINE LEARNING METRICS

P Krishna Kishore[1], S Ramamoorthy[2], V N Rajavarman[3]

[1]Research Scholar, Department of Computer Science and Engineering, Dr. M.G.R Educational and Research Institute, Chennai, India,

[2]Professor & Dean MCA, Dr. M.G.R Educational and Research Institute, Chennai, India,

[3]Professor & Deputy Dean, Dr. M.G.R Educational and Research Institute, Chennai, India,

Abstract

Now a day's most of the peoples using internet, which is commonly victimized to the Distributed Denial of Service (DDOS) attack, which is intentionally occupies the computing resources and bandwidth in order to deny that services to possible users. The attack situation is to flood the packets hugely. If the attack source is single, then the attack is referred as denial of service (DOS) and if attack is sourced from different servers, then it is referred as DDOS. Over a decade many of the researchers considered the detection and prevention of DDOS attack as research objective and succeeded to deliver few major DDOS detection and prevention strategies. How fast and early detection of DDOS attack is done in streaming network transactions is still a major research objective in present level of internet usage. Unfortunately the current bench-marking DDOS attack detection strategies are failing to justify the objective called ''fast and early detection of DDOS attack". In order to this, in this paper we devised a Bio-Inspired Anomaly based application layer DDoS attack (App-DDOS Attack) detection that is in the aim of achieving fast and early detection. The proposed model is a bio-inspired Bat Algorithm that used to achieve the fast and early detection of the App-DDOS by HTTP flood. The experiments were carried out on bench marking CAIDA dataset and the results delivered are boosting the significance of the proposed model to achieve the objective of the paper.

Keywords:- DENIAL OF SERVICE (DOS) ATTACKS, DISTRIBUTED DOS (DDOS) ATTACKS, APPLICATION LAYER DDOS (APP-DDOS), BIO INSPIRED APPROACHES.

I. INTRODUCTION

Global network of computers interconnected through different media using a standard protocol is called internet. Modern human beings rely on the Internet for their education, trade, socialization and entertainment, among many other important aspects of human life. Information sharing, E-commerce and entertainment have taken a new dimension. Evidently, the Internet is the biggest revolution in the computing and communications world. DoS attack is an intentional attempt by malicious users to completely disrupt or degrade the availability of services/resources to legitimate users. Distributed denial of service (DDoS) attack is a form of DoS attack which slowdowns the server in responding to the client/refuses the client request. The recent familiar

victims of DDOS attack are explored in [1,2] and strategies for successful attack mitigating are explored in [3].

The DDoS attacks are classified based on [4,5] into different factors. On the basis of network protocol stack, DDoS can be further classified as Network/transport level and Application level DDOS attacks.

Network/transport level DDoS attack: These attacks are launched at half opened connections by using TCP, UDP, ICMP and DNS protocols. Application level DDoS attack: These attacks typically consume less bandwidth and are stealthier in nature in comparison to volumetric attacks. However, once identified, these attacks will be stopped and back-traced to source more simply than the other varieties of DDoS attacks.

Application Layer DDoS attack is a DDoS attack that sends out requests following the communication protocol, thus these requests are indistinguishable from legitimate requests in the net- work layer .Consequently, traditional defense systems become less or even not applicable for application layer DDoS attacks which make use of the asymmetric computation between client and server, as they are proper-looking requests from the protocol and traffic.

Flooding attacks: Flooding attacks are launched in following ways [5, 6, 41, 42]:

In Reflection/Amplification based flooding attacks, the attacker initiates small DNS queries with forged source IP addresses which provoke a large extent of network traffic. And the DNS response messages are significantly larger than DNS query messages. As the result, this large extent of network traffic is directed towards the targeted system to incapacitate it.

HTTP based flooding attacks are classified into four types: In Session flooding attack, the Session connection request rates initiated from the attackers are higher the requests generated from legitimate users. Thus the server resources are exhausted and lead to flooding.

In Request flooding attack, the attacker send sessions that contains more number of requests than the normal users, which leads to flooding.

In Slow request/response attack, the attacker sends HTTP request in pieces slowly (one at a time) and the request is not complete initially. As the result, the server keeps the indulged resources in waiting stage until it receives the entire data.

The major focus of an HTTP flood DDoS attack is toward generating attack traffic that closely simulates legitimacy of a human user. inability to access any web site, dramatic increase in the number of spam emails received.

II. Related work

The recent escalation of application layer DoS attacks have attracted a significant interest of a research community. These research efforts can be broadly divided into several groups: application-based, puzzle-based approaches and network traffic characteristics based.

Application-based techniques are generally geared toward legitimate and thus expected characteristics of an application behavior. These approaches include detection of deviations from normal behavior of users browsing web pages [7–10], monitoring characteristics of HTTP sessions [11, 12], monitoring a number of client's requests [13], and analyzing popularity of certain websites [14]. In many of these approaches, rate-limiting serves as a primary defense mechanism.

One of these techniques is the detection of attacks using CAPTCHA puzzle [14]. Although this technique may offer a simple approach to attack detection and mitigation, a number of studies showed its ineffectiveness [15, 16].

Monitoring characteristics of network traffic for application-layer DoS detection is defined and has been employed for differentiation of flash crowd and true DoS attacks as per the suggestions made in [17]. The approach has also found its application in several studies in a form of IP address monitoring [18,19]. a number of research efforts were focused on various detection and mitigation techniques [20, 21, 30, 23]. Most of these techniques focus specifically on characteristics of incoming network traffic aiming to reveal/prevent patterns specific to low-rate DoS attacks. As such Tang [20] developed a CUSUM-based approach that monitors packet arrival rate. Macia-Fernandez et al. proposed to modify the implementation of application servers in terms of their processing of incoming requests [21].

Fadir Salmen et al. [22] created digital signature of network phase for flow analysis by using two meta-heuristic approaches. To investigate the behavior of planned approaches they injected abnormal traffic and showed improved accuracy in detection DDoS attacks however the primary model is incapable for detection the DoS attacks.

In [23,24] the authors proposed a model where the conversations between server and its client and The practical cyber surroundings that generated realistic traffic patterns of end users are used to check the proposed approach.

Vijayalakshmi et al. [25] proposed IP Trace back defense mechanism used to detect both network layer and application layer attacks. Attacker's source identification using entropy (Entropy variation is calculated). In mitigation component, when attack is detected an alert file generated.

Yu et al. [26] proposed TMH (Trust Management Helmet) which is light weight mechanism uses trust management to differentiate normal (legitimate) user and attackers. Wen et al. [27] proposed CALD is a defense mechanism to protect web servers against application layer DDoS attacks that pretend as flash crowds. The anomalous source IP (identified based on thresh- old of entropy) is sent to filter so that it can defend the attack.

Liu et al. [28] proposed DAT (Defending Systems Against Tilt DDoS Attacks) is built with two coordinated defenders namely In/egress filter (IF) and Behavior Analyzer (BA). The counter-attack mechanism offers different services to each user depending on their degree of deviation.

Yu et al. [29] proposed DOW (Defense and Offense Wall) defense mechanism is integration of detection technology and currency technology.

Yang et al. [30] proposed a generalized entropy metrics and information distance metrics to detect low rate application layer ddos attack. This is a router based solution and requires control of all routers in the network. Ying et al. [31] proposed a Group- Testing based approach. Prasad et al. [32] defined machine learning strategy called Anomaly based Real Time Prevention (ARTP) of under rated App-DDoS attacks. The complexity of the process reduced and attained maximum detection accuracy compared to other existing machine learning approaches. The results are good but still it can be improved further.

Senthilnath et al. [33,34] explored the use of firefly algorithm for clustering. Local Minima is obtained by using the k-means clustering and this drawback was overcome by the firefly algorithm. The authors used k-means and firefly algorithm for clustering purpose which increases the time complexity and this is not applicable for detecting application layer DDoS attacks in real time analysis.

In [35, 36], the model of Intelligent IDS proposed that based on ontology. The proposed system thwarts ontology based on encoding scheme, In [37] deployment of wireless sensor networks and mobile ad-hoc networks in applications poses the threat of various cyber hazards, intrusions and

attacks as a consequence of these networks' openness.

In [43] the authors investigated the effect of signaling attacks and storms in mobile networks, focusing on signaling anomalies that exploit the radio resource control (RRC) protocol in UMTS net- works. In [44] authors defined the detection and mitigation technique for storms that uses a software counter for each mobile user, within mobile devices or in signaling system.

The proposed solution assesses the similarities of HTTP transactions with fair and flood data chosen for training. The proposed model extracts the features from request stream observed during an absolute time interval rather than based on user sessions and packet patterns. Unique set of features proposed (see Section 3.1). The cosine similarities identify attribute set that imposes a discernibility relation. The signatures set that imposes the discernibility is critical since, the signatures having similar context in both records of normal and flood formats are obsolete to differentiate, hence such signatures minimizes the process complexity.

III. An Improved Bio-Inspired Bat algorithm for Detection and Prevention of HTTP flood DDOS attack using Machine Learning Metrics

3.1. The exploration of the metrics considered to train and test the model

The need of metrics should explore in contrast to packet pat- terns. The detailed exploration of the constraints observed in existing contemporary models, which are stated in related work (see Section 2), it is obvious to state that, in distributed environment, diversified packet flow is easy to achieve through minimal time frames and session time. The arrival rate based on human users, including a proxy server seems to constitute the non-pattern (random) cases. Hence, to challenge this constraint, this manuscript devised a novel set of metrics, which are derived from absolute time interval rather than the session time and packet patterns.

3.2 Discovering time frame length

Let CS be the cached user sessions $CS = \{S_1, S_2, ..., S_{|C|}\}$ and each session is set of transactions given for Training, such that each request is said to be transaction $\{t \exists t \in S_i \wedge S_i \in CS\}$ labeled as N (normal) or D (DDOS attack). The cached transactions CS is segregated into CS_N and CS_D, those contains requests labeled as N (nor- mal) and labeled as D (DDOS attack) respectively.

For each dataset CS (which is the aggregation of CSN and CSD), order the sessions in ascending order of their initiated time.

Let

$$SB = \{sb(s_1 \exists s_1 \in SC), sb(s_2 \exists s_2 \in SC), ..., sb(s_{|SC|} \exists s_{|SC|} \in SC)\}$$

as the set that represents the session begin time of all sessions belongs to SC.

Let

$$SE = \{se(s_1 \exists s_1 \in SC), se(s_2 \exists s_2 \in SC), ..., se(s_{|SC|} \exists s_{|SC|} \in SC)\}$$

as the set that represents the session end time of all sessions belongs to SC.

Let

$$SL = \{sl(s_1 \exists s_1 \in SC), sl(s_2 \exists s_2 \in SC), ..., sl(s_{|SC|} \exists s_{|SC|} \in SC)\}$$

as the set that represents the session life time of all sessions belongs to SC. Session life time of a session $\{s_i \exists s_i \in CS\}$ is calculated as follows:

$$sl(s_i \exists s_i \in CS) = se(s_i \exists s_i \in CS) - sb(s_i \exists s_i \in CS)$$

Find session begin time absolute deviation (Leys, 2013) of SB

$$sbtAD = \frac{\sqrt{\sum_{i=1}^{|SB|}(<SB> - sb(si \exists si \in SB))2}}{|SB|}$$

Here in the above equation, SB is the size of SB and SB is the average of SB. Find session end time absolute deviation of SE

$$setAD = \frac{\sqrt{\sum_{i=1}^{|SEN|}(<SE> - se(si \exists si \in si \in SE))2}}{|SE|}$$

Here in the above equation, |SE| is the size of SE and <SE> is the average of SE. Then the absolute time frame *atf* can be measured as follows

atf = (<SE> + seAD) – (<SB> + sbAD)

Cluster the sessions by *rmsd* (SB_N) and *rmsd* (SE_N) distance Finding K (count of centroids) value as follows:

Let K be the set of centroids and move session with least session begin time to the K.

For each session $\{s_i \exists s_i \in SB_N \wedge i=1,2,....,|SB_N|\}$ Begin
flag = true
For each session $\{ s_j \exists s_j \in K\}$*Begin*
If ($\sqrt{(sb(si) - sb(sj))2}$ < sbtAD ||$\sqrt{(se(si) - se(sj))2}$ < setAD)
then Begin
flag=false
End
End
If (flag) then Begin
K← s_i
End
End

Then apply K-Means to find number clusters with sessions in approximately similar time frames.
Let C = {c_1, c_2,…., c_K} be the set of clusters of size K.
Then for each cluster $\{c_i \exists c_i \in C \wedge i=1,2,3,...,K\}$, find the time frame as the elapsed time between least session begin time and max session end time as follows:
For each$\{c_i \exists c_i \in C \wedge i=1,2,3,...,K\}$ Begin
Let SB_N (c_i) = {sb_1,$sb_{2,....}$,$sb_{|ci|}$} be the ascending ordered set of session begin times of the sessions belongs to cluster c_i.
Let SE_N (c_i) = {se_1,$se_{2,....}$,$se_{|ci|}$} be the descending ordered set of session end times of the sessions belongs to cluster c_i.
Then the time frame *tf* (c_i) of the cluster c_i is measured as follows:

$$tf\ (c_i) = \sqrt{(se1 - sb1)2}$$

Then find the average of time frames length observed from all the clusters as follows:

$$<tf\ (C)> = \frac{\sum_{i=1}^{K} tf(Ci)}{K}$$

Further find Time Frame Absolute Deviation *tf*AD observed from all the clusters as follows

$$tfAD = \sqrt{\frac{\sum_{i=1}^{K}(<tf(C)> - tf(ci))2}{K}}$$

Then fix the time frame *tf* as the sum of average of time frames length and Time Frame Absolute Deviation as follows.

$$tf = <tf(C)> + tfAD$$

3.2.1. Maximum number of sessions (ms)

All transactions are formed into sessions that can be either random or variable timings. There exists different number of sessions for each time interval. Count of number of sessions observed in one time interval gives maximum number of sessions of that time interval which helps in observing the user sessions to detect application layer DDoS attacks.

3.2.2. Page access count (pac)

User will access multiple pages in different sessions of time interval. How many pages are accessed in one time interval helps in observing whether the environment in network is malicious or normal. Page access count of absolute time interval is the number of web pages accessed in that time interval.

3.2.3. Minimum time interval between two pages (mti)

This feature is calculated for two page requests which are in sequence of absolute time interval. How frequently the web pages are accessed by the user and the least amount of time gap that is required between two pages is measured that will help in observing the user behavior. Average of unique time gaps between two page requests which are in sequence of absolute time interval gives its minimum time interval. Let the unique time gap set of interval be tg = {tg_1, tg_2,…}

$$\text{Minimum Time Interval (mti)} = \frac{\sum_{i=1}^{|tg|} tgi}{|tg|}$$

3.2.4. Packets observed per each type of packet (PC)

Request can be sent through any of the packets like HTTP, FTP, SMTP etc.,. Each time interval contains different type of packets for which count of each packet is measured. The deviation in count of packets from one time interval to another time interval signifies the attack packet presence in the traffic. P = {p_1,p_2,p_3,…} the packets observed in that interval and pc = {p_1c,p_2c,p_3c,…} be the number of packets observed for each type of packet.

3.3. The dataset preparation

For given Flood and normal transaction sets CS_N and CS_D the record sets absolute time interval (ati) is formed as follows:

Each absolute time interval is considered as one record that contains the values of attributes in order of session time observed (see Section 3.1.2), Max number of sessions (Section 3.1.3), page access count (Section 3.1.4), Minimum Time interval (Section 3.1.5) and packet observed for each type (sec Section 3.1.6). These attributes will be referred as a set al in further draft of the article. The number of attributes in each record will be 5 which is the size of al that can be referred as $|a_l|$.

3.4. Bat approach

Many bio-inspired algorithms exist; bat algorithm belongs to this class which is based on swarm intelligence. The bat algorithm uses the echo based location determining behavior of bats to solve both single objective and multi-objective optimization problems. In proposed approach bat algorithm is used for classification (classify the attack traffic and normal traffic).

3.4.1. Nature of bats

Bats can find their prey and discriminate different types of insects even in complete darkness. The BAT algorithm is a population based evolutionary algorithm where each bat represents a solution. As the bat approaches near its prey, it reduces the loudness of their echo and increases the rate of the sound pulse.

The three generalized rules for bat algorithms:

(1) All bats use echolocation to sense distance, and they also guess the difference between food/prey and back-ground barriers in some magical way.
(2) Bats fly randomly with velocity v_i at position xi with a fixed frequency f_{min}, varying wavelength and loudness A_0 to search for prey. They can automatically adjust the wave-length (or frequency) of their emitted pulses and adjust the rate of pulse emission r depending on the proximity of their target.
(3) Although the loudness can vary in many ways, it is assumed that the loudness varies from a large (positive) A_0 to a minimum constant value A_{min}.

3.4.2. Classification using bat algorithm

An initial population of bats is generated. After initialization modify the parameters needed for fitness, and subsequently the fitness is evaluated for each bat in population.

Step1: for i=1: number of iteration
delwt=wt
Step 2: for j=1:n
read D_j
Step 3: compute each bat $_j$ frequency as f_j
f_j = c1* Mean(D_j)
Step 4: compute class (object) distance from bat_j as S object j
S $object_j$ = f_j * D_j * delwt
Step 5: compute for each class
E_j = S $object_j$ -1, update wt=wt-2*μ*E_j
Step 6: compute the new position P_j and change the pulse rate
Controller c1 of bat_j
if E_j < E_j - 1
P_j = P_j + E_j
c1 = f_j + c2 * P_j + E_j
end
Step 7: end
Step 8: calculate the error and update wt
err(i) = Mean(E), wt =wt - delwt
Step 9: end
Step 10: plot the error as sigmoid(err)
Step 11: use the above wt and f with testing data
Step 12: print confusion matrix from S object $_j$
Step 13: compute percentage of accuracy

Prepared dataset of both normal and attack is given as input for the Bat algorithm individually. For normal dataset, each record is considered as one bat and compare with the remaining bats to know how much distance it has to move towards the remaining bats. This has to be done for all the remaining bats. Updated records are carried over to next iteration or generation. Maximum number of iterations has to be performed for getting the accurate classifiers. Once all the iterations are completed, the normal classifier is extracted and marked as normal signature. The same process is carried for attack training records to get the attack signature.

3.4.3. Application layer DDoS attack detection

Testing dataset has to be preprocessed by using the dataset pre- processing process. Prepare the

dataset with five attributes as like in the dataset preparation. Calculate the total weight (light intensity) of the testing records individually. Calculate the cosine similarity of testing record with both normal and attack signatures and declare whether the testing record is attack or normal by using the rules defined in Section 4.2.

IV. Experimental Results

4.1. CAIDA dataset

The proposed technique is tested against CAIDA [38] (Center for Applied Internet Data Analysis) dataset 2007. Core Objectives of this dataset are collection and sharing of data for research or scientific analysis of internet traffic, topology, routing, performance and security related events. Dataset contains the parameters like server IP address, Timestamp, Time Zone, Object ID/URL of the web page, Response code/status, Number of bytes sent.

Table. 1. data set details

Number of transactions considered for both training and testing	213,066
Number of Normal transactions	94,164
Number of Attack transactions	118,902
Number of transactions for training	127,839
Number of transactions for testing	85,227

Table. 2. Rules defined for attack detection.

Rule 1	Weight of the testing time interval is less than the normal classifier weight and greater than the attack weight	$A(w) < T(w) \leq N(w)$	Normal
Rule 2	2.1 similarity of testing record with the normal classifier is more than 98 percent 2.2 similarity of testing record with the attack classifier is more than 98 percent	Similarity(test, normal)≥ 98% Similarity(test, attack)≥ 98%	Normal Attack
Rule 3	Similarity of testing record with normal classifier is more than the similarity of testing record with attack classifier	Similarity(test, normal) > Similarity(test, attack)	Normal
Rule 4	All the above conditions are failed	suspicious	

Table. 3. Performance parameter calculations.

Total Number of records consider for training and testing		213066
Total Number of intervals consider for training and testing		401
Number of intervals used for training (Normal + Attack)		243 (108 +135)
Number of intervals used for testing (Normal + Attack)		158 (69 +89)
True Positive (tp)	The number of transactions identified as intruded, which are actually intruded	91
False Positive (fp)	The number of transactions identified as normal, which are actually intruded	3
True Negative (tn)	The number of transactions identified as normal, which are actually normal	69
False Negative (fn)	The number of transactions identified as intruded, which are actually normal.	5
Precision	$\frac{tp}{tp + fp}$	0.945

Recall/sensitivity	$\frac{tp}{tp+fn}$	0.94
Specificity	$\frac{tn}{tn+fp}$	0.936
Accuracy	$\frac{tn+tp}{tp+tn+fp+fn}$	0.948
F-Measure	$2 * \left(\frac{recall*precision}{recall+precision}\right)$	0.9457

Table. 4. Comparison of Bat algorithm with ARTP and FCAAIS.

	Bat algorithm	**ARTP**	**FCAAIS**
Precision	0.945	0.895	0.869
Recall	0.94	0.985	0.942
Specificity	0.936	0.914	0.894
Accuracy	0.948	0.944	0.917
F-measure	0.9457	0.938	0.855

4.2. Training & testing records

The total number of transactions considered for experiments were 213,066 which includes N (normal-94164) and D (DDoS attack-118902). The total transactions are partitioned for training and testing into 60 %(127,839) and 40 %(85,227) respectively. Each metric is calculated on the dataset CS which includes N (normal) as CSN and D (DDoS attack) as CSD and its detection accuracy is assessed. Number of intervals are 401. The number of intervals in normal dataset DSN is 178 in which 60% of transactions i.e., 108 are considered for the training process and 40% of transactions i.e., 69 for the testing process. The total number of intervals in attack dataset DSD is 224 in which 60% of transactions i.e., 135 are considered for the training process and 40% of transactions i.e., 89 for the testing process. The details are given in Table 1.

Calculate all the attributes for each interval. Testing time interval Cosine similarity is calculated with both attack and normal signatures and at last classifies the testing time interval according to proposed rules in Table 2.

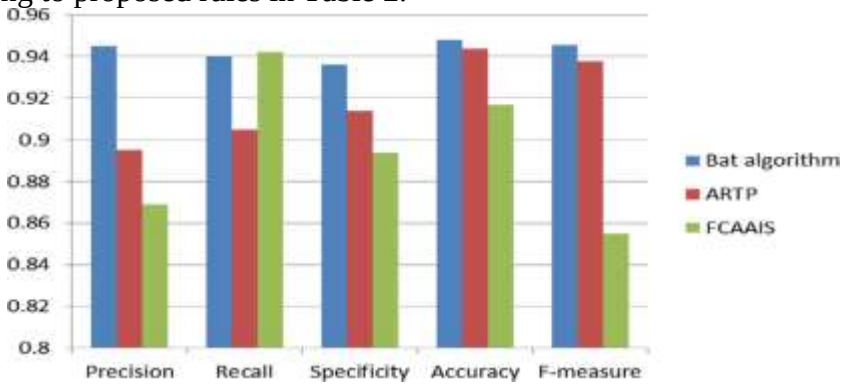

Fig. 1. Comparison of Bat algorithm with ARTP and FCAAIS.

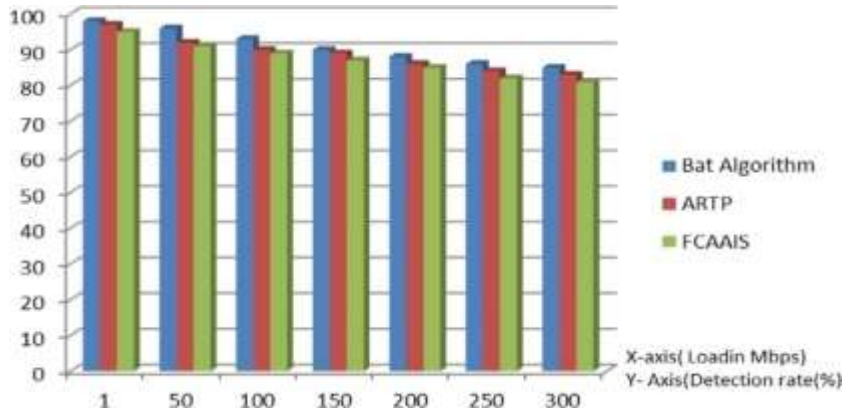

Fig. 2. Comparison of Detection rate observed in Bat algorithm with ARTP and FCAAIS.

4.3. Performance evaluation

The performance of proposed approach is evaluated based on the following parameters [22]. The calculations are shown in Table 3.

- Precision shows the class agreement of the data labels with the positive labels given by the classifier.
- Recall shows the effectiveness of a classifier to identify the positive labels.
- Specificity shows how effectively a classifier identifies the negative labels.
- Accuracy shows the overall effectiveness of a classifier.
- F-measure shows the relation between data's positive labels and those given by a classifier.

Munivara Prasad proposed ARTP [31] for Detecting Application layer DDoS attacks by using the Machine learning approach. Jyothsna and Prasad proposed FAIS [39] and FCAAIS [40] for detecting DDoS attacks. The experiments in above papers are conducted on the same dataset and results are indicating that these models are also scalable and robust towards forecasting the DDoS attacks scope of a network transaction (observed detection accuracy is approx. 91%), which influence the statistical metrics defined for measuring the performance. As per these results, the accuracy of our proposed model was improved when compared to FCAAIS, ARTP and also attained maximum prediction accuracy which is shown in Table 4 and the performance comparisons are given in Figs. 1–3.

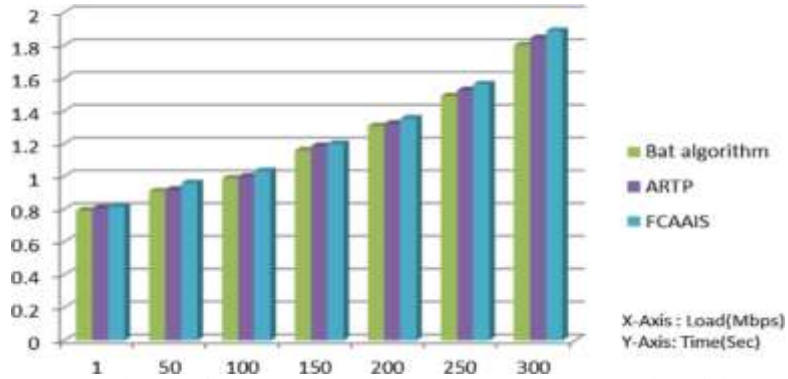

Fig. 3. Comparison of processing time observed in Bat algorithm with ARTP and FCAAIS.

V. CONCLUSION

Bio-Inspired Anomaly based HTTP-Flood Attack Detection (BIFAD) devised in this article. In this, we adopted the Bat algorithm, which is a bio-inspired approach with magnified speed in search. First we defined feature metrics to identify the request stream behavior is of attack or normal. Unlike traditional approaches, the assessment of feature metrics done on the stream of requests observed in an absolute time interval rather in a session. The Second contribution is to customize the Bat algorithm to train and test. The devised Bat algorithm amplified the detection accuracy with minimal process complexity. The experiments were conducted using CAIDA dataset. Hence the model devised here in this paper is significantly accurate and retains the maximal prediction accuracy.

References

1. J. Udhayan, R. Anitha, Demystifying and rate limiting ICMP hosted DoS/DDOS flooding attacks with attack productivity analysis, in: IEEE International Conference on Advance Computing, 2009, pp:558–564.
2. Xia Chun-Tao, D. Xue-Hui, C. Li-Feng, An algorithm of detecting and defending CC attack in real time, in: International Conference on Industrial Control and Electronics Engineering, 2012, pp. 1804–1806.
3. S.M. Lee, Distributed denial of service: taxonomies of attacks, tools, and countermeasures, in: Proceedings of the International Workshop on Security in Parallel and Distributed Systems, San Francisco, 2004, pp. 543–550.
4. Raj kumar, Manisha Jitendra Nene, A survey on latest DoS attacks: classification and defense mechanisms, Proc. Int. J. Innov. Res. Comput. Commun. Eng. 1 (8) (2013).
5. Erol Gelenbe, Michael Gellman, George Loukas, An autonomic approach to denial of service defence, in: Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005, June 2005, pp. 537–541.
6. Yadong Wang, Lianzhong Liu, et al., A survey of defense mechanisms against application layer distributed denial of service (DDoS) attacks, IEEE Commun. Surv. Tut. (2015).
7. Y. Xie, S. Zheng Yu. A novel model for detecting application layer DDoS attacks, in: First International Multi-Symposiums on Computer and Computational Sciences, 2006. IMSCCS ’06, vol. 2, 2006, pp. 56–63.
8. J. Yu, Z. Li, H. Chen, X. Chen. A detection and offense mechanism to defend against application layer DDoS attacks, in: Third International Conference on Networking and Services, 2007. ICNS, 2007, pp. 54–54.
9. C. Ye, K. Zheng, Detection of application layer distributed denial of service, in: 2011 International Conference on Computer Science and Network Technology (ICCSNT), vol. 1, 2011, pp. 310–314.
10. S. Ranjan, et al., DDoS-resilient scheduling to counter application layer attacks under imperfect detection, in: Proceedings of IEEE INFOCOM, 2006, pp. 23–29.
11. S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci, E. Knightly, DDoS-shield: DDoS- resilient scheduling to counter application layer attacks, IEEE/ACM Trans. Netw. 17 (1) (2009) 26–39.
12. Y. Xuan, I. Shin, M. Thai, T. Znati, Detecting application denial of-service attacks: A group-testing-based approach, IEEE Trans. Parallel Distrib. Syst. 21 (8) (2010) 1203–1216.
13. Y. Xie, S. Zheng Yu, Monitoring the application-layer DDoS attacks for popular websites, IEEE/ACM Trans. Netw. 17 (1) (2009) 15–25.

14. M. Mehra, M. Agarwal, R. Pawar, D. Shah, Mitigating denial of service attack using captcha mechanism, in: Proceedings of the International Conference & Workshop on Emerging Trends in Technology, ICWET '11, ACM, New York, NY, USA, 2011, pp. 284–287.
15. J. Jung, B. Krishnamurthy, M. Rabinovich, Flash crowds and denial of service attacks: characterization and implications for cdns and web sites, in: Proceedings of the 11th International Conference on World Wide Web, WWW '02, ACM, New York, NY, USA, 2002, pp. 293–304.
16. S.Y. Nam, T. Lee, Memory-efficient IP filtering for countering DDoS attacks, in: Proceedings of the 12th Asia-Pacific Network Operations and Management Conference on Management Enabling the Future Internet for Changing Business and New Computing Services, APNOMS'09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 301–310.
17. S. Bhatia, D. Schmidt, G. Mohay, Ensemble-based DDoS detection and mitigation model, in: Proceedings of the Fifth International Conference on Security of Information and Networks, ACM, 2012, pp. 79–86.
18. Y. Tang, Countermeasures on application level low-rate denialof-service attack, in: Proceedings of the 14th International Conference on Information and Communications Security, ICICS'12, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 70–80.
19. G. Maciá-Fernández, R.A. Rodríguez-Gómez, J.E. Díaz-Verdejo, Defense techniques for low-rate dos attacks against application servers, Comput. Netw. 54 (15) (2010) 2711–2727.
20. G. Maciá-Fernández, J.E. Díaz-Verdejo, P. García-Teodoro, Evaluation of a low- rate dos attack against iterative servers, Comput. Netw. 51 (4) (2007) 1013– 1030.
21. G. Macia-Fernandez, J. Diaz-Verdejo, P. Garcia-Teodoro, Mathematical model for low-rate dos attacks against application servers, IEEE Trans. Inf. Forensics Secur. 4 (3) (2009) 519–529.
22. Fadir Salmen, Paulo R. Galego Hernandes Jr, et al., Using firefly and genetic meta heuristics for anomaly detection based on network flows", in: AICT : The Eleventh Advanced International Conference on Telecommunications, 2015.
23. Mikhail Zolotukhin, Timo Hamalainen, et al., Increasing web service availability by detecting application-layer DDoS attacks in encrypted traffic, in: IEEE, 23rd International Conference on Telecommunications (ICT).
24. Georgios Kambouakis, Tassos Moschos, Dimitris Geneiatakis, Stefanos Gritzalis, A fair solution to DNS amplification attacks. Second International Workshop on Digital Forensics and Incident Analysis, 2007, WDFIA 2007.
25. M. Vijayalakshmi, S. Mercy Shalinie, IP traceback system for network and application layer attacks, 2012 International Conference on Recent Trends In Information Technology (ICRTIT), IEEE, 2012.
26. J. Yu, C. Fang, L. Lu, Z. Li, Mitigating application layer distributed denial of service attacks via effective trust management, IET Commun. 4 (16) (2010) 1952–1962.
27. S. Wen, W. Jia, W. Zhou, W. Zhou, C. Xu, Cald: Surviving various application- layer DDoS attacks that mimic flash crowd, 2010 4th International Conference on Network and System Security (NSS), IEEE, 2010.
28. Huey-Ing Liu, Kuo-Chao Chang, Defending systems against tilt DDoS attacks, 2011 6th International Conference on Telecommunication Systems, Services, and Applications (TSSA), IEEE, 2011.

29. Jie Yu, Zhoujun Li, Huowang Chen, Xiaoming Chen, A detection and offense mechanism to defend against application layer DDoS attacks, 2007. ICNS. Third International Conference on Networking and Services, IEEE, 2007.
30. Yang Xiang, Ke Li, Wanlei Zhou, Low-rate DDoS attacks detection and traceback by using new information metrics, IEEE Trans. Inf. Forensics Secur. 6 (2) (June 2011) 426–437.
31. Ying Xuan, Incheol Shin, My T. Thai, Taieb Znati, Detecting application denial- of-service attacks: a group-testing-based approach, IEEE Trans. Parallel Distrib. Syst. 21 (8) (2010) 1203–1216.
32. K. Munivara Prasad, A. Rama Mohan Reddy, K. Venugopal Rao, Anomaly based real time prevention of under rated App-DDoS attacks on web: an experiential metrics based machine learning metrics, Indian J. Sci. Technol. 9 (27) (2016), https://doi.org/10.17485/ijst/2016/v9i27/87872.
33. J. Senthilnath, S.N. Omkar, V. Mani, Clustering using firefly algorithm: performance study, ELSEVIER, Swarm Evol. Comput. 1 (2011) 164–171.
34. Sufian Hameed, Usman Ali, On the Efficacy of Live DDoS Detection with Hadoop, arXiv preprint arXiv:1506.08953, 2015.
35. A. Razzaq et al., Foundation of semantic rule engine to protect web application attacks, in: 2011 Tenth International Symposium on Autonomous Decentralized Systems, IEEE, 2011, p. 2011.
36. Saeed Javan Mardi, Mohammad Shojafar, Shahdad Shariat Madari, et al., PGSW-OS: a novel approach for resource management in a semantic web operating system based on a P2P grid architecture, J. Supercomput. 69 (2) (2014) 955–975.
37. Shahaboddin Shamshirband, Nor Badrul Anuar, Miss Laiha Mat Kiah, Ahmed Patel, An appraisal and design of a multi-agent system based cooperative wireless intrusion detection computational intelligence technique, Eng. Appl. Artif. Intell. 26 (9) (2013).
38. MIT, M. I. . Darpa Intrusion Detection Evaluation. Retrieved from Lincoln Laboratory: <https://www.ll.mit.edu/ideval/data/1998data.html>.
39. V. Jyothsna, V.V. Rama Prasad, Anomaly based network intrusion detection through assessing Feature Association Impact Scale (FAIS); Inderscience, Int. J. Inform. Comput. Secur. (IJICS), 2016, in press.
40. V. Jyothsna, V.V. Rama Prasad, FCAAIS: Anomaly based network intrusion detection through feature correlation analysis and association impact scale, in: ICT Express, The Korean Institute of Communications Information Sciences, Elsevier, August 2016, in press.
41. E. Gelenbe, G. Loukas, A self-aware approach to denial of service defence, Comput. Netw. 51 (5) (2007) 1299–1314.
42. Daniele Gianni, Georgios Loukas, Erol Gelenbe, A simulation framework for the investigation of adaptive behaviours in largely populated building evacuation scenarios, in: Organised Adaptation in Multi-Agent Systems workshop at the 7th International Conference on Autonomous Agents and Multiagent Systems, May 2008.
43. Gokce Gerbil, Omer H. Abdelrahman, Mihajlo Pavloski, Erol Gelenbe, Modeling and analysis of RRC-based signalling storms in 3G networks, IEEE Trans. Emerging Topics Comput. 4 (1) (2016) 113–127.
44. Erol Gelenbe, Omer H. Abdelrahman, Gokce Gorbil, Detection and mitigation of signaling storms in mobile networks, in: IEEE ICNC, 2016, pp. 1–5.