BIOSEC - Predicting Distributed Denial of Service Attack in Biological Sciences Applications

Umarani S.[1*] and Sharmila D.[2]

1. Department of Information Technology, Maharaja Engineering College, Avinashi, Tamil Nadu, INDIA
2. Department of Electronics and Instrumentation Engineering, Bannari Amman Institute of Technology, Sathyamangalam, INDIA
*umashenna@gmail.com

Abstract

In Distributed Denial of Service (DDoS) attack is a continuous critical threat to the Internet. Derived from the low layers, new Application layer- based DDoS (App-DDoS) attacks utilizing legitimate Hyper Text Transfer Protocol (HTTP) requests to overwhelm victim resources are more undetectable. This work proposes a method to detect DDoS attacks on a genome website used for biological application from traffic flow traces from which an Access Matrix (AM) is created. As, it is multi-dimensional, Principle Component Analysis (PCA) reduces attributes used in detection. PCA can be used for feature transformation into high dimension for finding the feature subsets. Further, their performance can be determined in training and testing process of Support Vector Machine (SVM) for classification in terms of detection rate and false alarms. SVM is well-liked for classification of patterns because it gives good results as compared to other methods because even with less information about the dataset provided, SVM outperforms in testing. Features are classified with SVM which is optimized using BAT and Cuckoo Search (CS) algorithm in parallel to improve classification rate. The results obtained from the proposed hybrid heuristics SVM-CS BAT shows better performance compared to standard classifiers.

Keywords: Distributed Denial of Service (DDoS) Attack, Support Vector Machine (SVM), Hybrid Heuristic.

Introduction

The phenomenal advance in the research of genomes has made it one of the most active area of biomedical research. Genome research aims in discovering gene variation, and its link with diseases. The genetic database also contains information and medical history of the subjects/patients. Thus, there is a vulnerability of exposure if the data is compromised. The database is generally accessible to various researchers, technicians, administrators. It is required to include security mechanism to maintain the confidentiality and privacy of the subjects. This work proposes a method to detect DDoS attacks on a genome website used for biological application from traffic flow traces.

Denial of Service (DoS) attacks are a typical kind of assault which is a huge threat to the Internet. In DoS assaults, the aim of the attackers is to tie up selected resources of the victims, typically through the transmission of extremely huge volumes of apparently legal traffic demanding some service from the victims. It exposes huge loopholes not only in particular applications, but also in the whole Transmission Control Protocol/Internet Protocol (TCP/IP) protocol setting. DoS attacks are regard to occur solely when access to computers or networks resources are blocked or deteriorated with malicious intent by an attacker.

The current attacks on trendy web sites like Amazon, Yahoo, e-Bay and Microsoft and their resultant disruption of services have uncovered the weakness of the Internet to DDoS attacks. It has been observed through reports that more than 85% of the DoS attacks use TCP. The TCP SYN flooding is the most commonly-used attack. It consists of a stream of spoofed TCP SYN packets directed to a listening TCP port of the victim. Not only the Web servers but also any systems connected to the Internet providing TCP-based network services, such as File Transfer Protocol (FTP) servers or Mail servers, are susceptible to the TCP SYN flooding attacks[1].

Since 2001, DDoS attacks are rising at an alarming rate. Due to the severity of DDoS attacks in several business domains, several defense methods were formulated utilizing statistical techniques for defending against these attacks. Previous techniques suggested usage of statistical techniques exploiting features of packet headers in packets like time to live, IP address and so on. The methods were extremely dependent on traffic features. Every statistical based method for DDoS attacks function well in TCP/IP layers however they are not adaptable or applicable for few generic as well as particular DDoS assaults which work on application or higher layers [2&3].

Since 2001, DDoS attack is growing rapidly till date. Because of the seriousness of DDoS attack in various business fields, many defense mechanisms were developed using statistical methods to defend against this attack. Earlier methods proposed to using statistical method make use of the attributes of the header in a packet such as time-to-live, IP address, etc. These approaches highly depend on the traffic characteristics. All the statistical based approaches for DDoS attack work well in TCP/IP layers whereas they are not adaptable and also not applicable for some typical as well as special DDoS attacks that are working on the application layer (higher layer)[4].

Detecting App-DDoS has the following challenges:

- App-DDoS uses higher layer protocols such as HTTP to pass through the detection system, which are designed for lower layer.
- Along with flooding, App-DDoS also consumes resources of the targeted victim server and either trace the average request rate of the legitimate user and uses the same rate for attacking the server or employs large-scale botnet to generate low rate attack flows. This cause the detection system to detect the DDoS attack more complex.

When the simple Net-DDoS attacks fail, attackers shift their offensive strategies to application-layer attacks and establish a more sophisticated type of DDoS attacks. To circumvent detection, they attack the victim Web servers by HTTP GET requests (e.g., HTTP Flooding) and pulling large image files from the victim server in overwhelming numbers. In another instance, attackers run a massive number of queries through the victim's search engine or database query to bring the server down. It call such attacks App-DDoS attacks. On the other hand, a new special phenomenon of network traffic called flash crowd, has been noticed by researchers during the past several years[5].

On the Web, "flash crowd" refers to the situation when a very large number of users simultaneously access a popular Website, which produces a surge in traffic to the Website and might cause the site to be virtually unreachable. Because burst traffic and high volume are the common characteristics of App-DDoS attacks and flash crowds, it is not easy for current techniques to distinguish them merely by statistical characteristics of traffic. Therefore, App-DDoS attacks may be stealthier and more dangerous for the popular Websites than the general Net- DDoS attacks when they mimic (or hide in) the normal flash crowd.

The design and implementation of a comprehensive solution which can defend Internet from variety of DDoS attacks is hindered by following challenges[6]:

- Large number of unwitting participants.
- No common characteristics of DDoS streams.
- Use of legitimate traffic models by attackers.
- No administrative domain cooperation.
- Automated DDoS attack tools.
- Hidden identity of participants because of source addresses spoofing.
- Persistent security holes on the Internet.
- Lack of attack information.
- Lack of standardized evaluation and testing approaches.

In order to build a comprehensive DDoS defense solution in light of these challenges, Robinson et al. recommended following DDoS defense principles[7]:

- As DDoS is a distributed attack and because of high volume and rate of attack packets distributed instead of centralized defense is the first principle of DDoS defence.
- High Normal Packet Survival Ratio (NPSR) (ratio of number of normal packets received to total number of packets reaching at the server), i.e., less collateral damage is the prime requirement for a DDoS defense.
- A DDoS defense method should provide secure communication for control messages in terms of confidentiality, authentication of sources, integrity and freshness of exchanged messages between defense nodes.
- A partially and incrementally deployable defense model is successful as there is no centralized control for Autonomous Systems (AS) in Internet.
- A defense system must take into account future compatibility issues such as interfacing with other systems and negotiating different defense policies.

Heuristic-based defense systems against DDoS attack confront the problem of threshold adjustment. These approached may need to calculate its own threshold to judge the current observing traffic. The similarity, distance, classification, clustering and/or prediction analysis may be applied in this research area. In contrast to statistic-based approach, heuristic-based approach does not require to learn/define the normal situation before comparing to anomalous situation. The drawback of heuristic detection approaches is their inability to consider legitimate traffic mixed with attacking traffic. Hence, packets from legitimate users may be blocked or eliminated during attack incidents occur[8]. In addition, the threshold of this approach needs to be optimisation when deploys to a DDoS defense system. In this work, BAT and CS algorithm are used.

SVM are an effective technique for solving classification and regression problems. SVM is originally an implementation of Vapnik's Structural Risk Minimization (SRM) principle, which is known to have low generalization error or equivalently does not suffer much from over fitting to the training data set. SVM is particularly effective on data sets that are linearly separable, i.e. where hyperplane H can be found that partitions the instances into two classes such that instances in one class entirely fall on one side of H. Since there is an infinite number of candidate hyperplanes that can be selected, SVM selects the hyperplane H so that it maximizes its distance to the nearest data points in either class. This is referred to as margin maximization. This transformation often comes in the form of mapping to a high-dimensional space. A function used to perform such a transformation is called a kernel function. Thus, kernel functions play a pivotal role both in the theory and application of SVM[9].

This work proposes a method to detect DDoS attacks from traffic flow traces. Section 2 reviews literature related to the proposed work. Section 3 explains methodology and Section 4 discusses the results of experiments conducted in proposed work. Section 5 concludes the work.

Related Works

Robinson & Thomas[10] analyzed the threats and consequences by DOS attacks. The nature of the attacks, and an attempt to explain what the true motives behind these attacks have been done. The different mitigation processes of DDoS attacks in different situations have been analyzed to explore the power of methods using Hop-Count Filtering and probabilistic approaches.

Durcekova et al[11] focused on application layer DoS and DDoS attacks detection, because these attacks present a continuous critical threat to the Internet services. DDoS attacks are typically carried out at the network layer. However, there was evidence to suggest that application layer DDoS attacks can be more effective than the traditional ones. Over some period of time, researchers proposed many solutions to prevent the DoS/DDoS attacks from different OSI layers, but there has been done only a very small research on application layer. In this work considered sophisticated attacks that utilize legitimate application layer requests from legitimately connected network machines to overwhelm Web server. Since the attack signature of each application layer DDoS was represented in abnormal user behavior, the author proposed several mechanisms, which can be used for application DoS/DDoS attack detection.

Tan et al[12] presented a DoS attack detection system that uses Multivariate Correlation Analysis (MCA) for accurate network traffic characterization by extracting the geometrical correlations between network traffic features. The MCA-based DoS attack detection system employs the principle of anomaly based detection in attack recognition. This makes the solution capable of detecting known and unknown DoS attacks effectively by learning the patterns of legitimate network traffic only. Furthermore, a triangle-area-based technique was proposed to enhance and to speed up the process of MCA. The effectiveness of the proposed detection system was evaluated using KDD Cup 99 data set, and the influences of both non-normalized data and normalized data on the performance of the proposed detection system are examined. The results show that the system outperforms two other previously developed state-of-the-art approaches in terms of detection accuracy.

Katkar and Kulkarni[13] presented the importance of Intrusion Detection System (IDS) for detection of DoS/DDoS network attacks has also grown. Different techniques such as data mining and pattern recognition are being used to design IDS. Naïve Bayesian was a widely-used classifier for design of IDS. The author evaluated variation in performance of Naïve Bayesian classifier for intrusion detection when used in combination with different data pre-processing and feature selection methods. Experimental results prove that accuracy of Naïve Bayesian classifier was improved and performs better than other classifiers when used in combination with feature selection and data pre-processing methods.

Katkar and Bhatia[14] evaluated the effect of various data preprocessing methods on the detection accuracy of DoS/DDoS attack detection IDS and proves that numeric to binary preprocessing method performs better compared to other methods. Experimental results obtained using KDD 99 dataset are provided to support the efficiency of proposed combination.

Gu et al[15] proposed a new method to detect application-layer DDoS attack based on IP Service Request Entropy (SRE) time series. By approximating the Adaptive Auto-Regressive (AAR) model, the SRE time series was transformed into a multidimensional vector series regarded as a description of current users' visiting patterns. Furthermore, a SVM classifier was applied to classify vector series and identify the attacks. The simulation results show that this approach not only can distinguish between normal traffic and DDoS attack traffic, but also was suitable to detect DDoS attack against the large scale network traffic, which does not arouse the sharp changes of the network traffic.

Barati et al[16] proposed an architecture of a detection system for DDoS attack. Genetic Algorithm (GA) and Artificial Neural Network (ANN) are deployed for feature selection and attack detection respectively in the hybrid method. Wrapper method using GA was deployed to select the most efficient features and then DDoS attack detection rate is improved by applying Multi-Layer Perceptron (MLP) of ANN. Results demonstrate that the proposed method was able to detect DDoS attack with high accuracy and deniable False Alarm.

Htun and Khaing[17] explored feature selection and classification methods for DoS attacks detection since they are the most threatening intrusions these days using with Random Forests (RF) and KNN for feature selection and classification respectively. The experimental results presented in this work show that by estimating the most important features of the data set using RF-KNN. The purpose of this work was to study the best features selection algorithm, RF in building an IDS that is computationally efficient and effective and the best classification algorithm KNN that have been widely used for IDS. Experimental results prove that the proposed method can get the high accuracy in detection those known and unknown attacks by using WEKA tool.

Subbulakshmi et al[18] presented the DDoS detection dataset and detect them using the Enhanced SVM (ESVM). The DDoS dataset with various direct and derived attributes is generated in an experimental testbed which has 14 attributes and 10 types of latest DDoS attack classes. Using the generated DDoS dataset the Enhanced Multi Class SVMs (EMCSVM) was used for detection of the attacks into various classes. The performance of the EMCSVM was evaluated over SVM with various parameter values and kernel functions. It was inferred that EMCSVM produces better classification rate for the DDoS dataset with ten types of latest DDoS attacks when compared with the kddcup 99 dataset which has six types of DoS attacks.

Zhou et al[19] proposed a new method to detect Application-Layer DDoS (AL-DDoS) attacks. The work distinguishes itself from previous methods by considering AL-DDoS attack detection in heavy backbone traffic. Besides, the detection of AL-DDoS attacks was easily misled by flash crowd traffic. In order to overcome this problem, the proposed method constructs a Real-time Frequency Vector (RFV) and real-timely characterizes the traffic as a set of models. By examining the entropy of AL-DDoS attacks and flash crowds, these models can be used to recognize the real AL-DDoS attacks. Compared with previous methods, the results show that the approach was very effective in defending AL-DDoS attacks at backbones.

Wen et al[20] presented the design and implementation of CALD, an architectural extension to protect Web servers against various DDoS attacks that masquerade as flash crowds. CALD provides real-time detection using mess tests but is different from other systems that use resembling methods. First, CALD uses a front-end sensor to monitor the traffic that may contain various DDoS attacks or flash crowds. Second, CALD dynamically records the average frequency of each source IP and check the total mess extent. Third, CALD may divide the security modules away from the Web servers. As a result, it keeps maximum performance on the kernel web services, regardless of the harassment from DDoS. In the experiments, the records from www.sina.com and www.taobao.com have proved the value of CALD.

Ramana et al[21] proposed a novel scheme based on document popularity and also AM was defined to capture the spatial-temporal patterns of a normal flash crowd. A novel attack detector based on hidden semi-Markov model was proposed to describe the dynamics of AM and to detect the attacks. The entropy of document popularity fitting to the model was used to detect the potential DDoS attacks in application-layer. This work analysis the attack detector with existing system drawback which presents proposed approach more efficient.

Xu et al[22] presented a new detection method based on Kernel Principle Component Analysis (KPCA) and Particle Swarm Optimization (PSO)-SVM. The KPCA was used to obtain the important characteristics of the intrusion data to eliminate the redundant features. Then the PSO was used to optimize the SVM parameters. Experimental results show the proposed approach can be enhanced the detection rate, and performs better than the PCA based methods.

Methodology

A novel technique to optimize the SVM parameters using Hybrid heuristic algorithm is proposed. BAT and CS are integrated in a novel architecture. A cloud datacentre was simulated and each VM was designed to handle a gene selection task. Attacks were introduced from multiple sources on the datacentre.

Fields used in request structure are given as follows:

Timestamp: Represents time of request and is stored as number of seconds since Epoch. Timestamp information is converted to GMT for portability. For calculating local time, each timestamp is adjusted.
Client ID: A unique integer identifier was given for each client and due to some privacy concerns the mappings were not released;
Object ID: A unique integer identifier for requested URL
Size - number of bytes in response
Method: Method in client's request
Status: This field has two pieces of information; the 2 highest order bits contain HTTP version indicated in client's request and remaining 6 bits indicate response status code
Type: Type of file requested
Server: Gives details of which server handled request.

For log collection, the log collection period (April 30th, 1998 to July 26th, 1998), 33 different World Cup HTTP servers were used at Paris, France; Plano, Texas; Herndon, Virginia; and Santa Clara, California is referred. A total of 1,352,804,107 requests were received at the World Cup site.[23]

Access Matrix (AM): The AM model is the policy for user authentication, and has several implementations such as Access Control Lists (ACLs) and capabilities. It is used to describe which users have access to what objects. The AM consists of four major parts a list of objects, a list of subjects, a function T which returns an object's type and the matrix itself, with the objects making the columns and the subjects making the rows. In the cells where a subject and object meet lie the rights the subject has on that object. Some example access rights are read, write, execute, list and delete[24].

An AM has several standard operations associated with it:

- Entry of a right into a specified cell
- Removal of a right from a specified cell
- Creation of a subject
- Creation of an object
- Removal of an subject
- Removal of an object

AM represents spatial and temporal patterns of access to a specific site or web document at a specific time. Website popularity depends on hit rate which is defined as equation (1):

$$P_{it} = \frac{b_{it}}{\sum_{i=1}^{N} b_{it}} \tag{1}$$

Here, b_{it} represents number of requests for a document i in a web server at time unit t, and N represents total number of documents in entire web server. If number of users requesting document i at time unit t is represented as c_{it}, then average revisit of user for document i is calculated by $\frac{b_{it}}{c_{it}}$ .Normalized user revisit is represented as r_{it} and number of observation time units is represented as T *in* equation (2 & 3).

$$r_{it} = \frac{\textit{average request number per user on the } i^{th} \textit{ document at } t^{th} \textit{ time unit}}{\textit{average request amount per user at } t^{th} \textit{ time unit}} \tag{2}$$

$$= \frac{b_{it} / c_{it}}{\sum_{i=1}^{N} \left(b_{it} / c_{it} \right)}, \; i \in [1, N], \; t \in [1, T] \tag{3}$$

Then the AM of the dimension N x T can be constructed[25] by in equation (4):

$$A_{N*T} = \left[\bar{a}_1, \bar{a}_2, ..., \bar{a}_T \right] = \left[a_1, a_2, ..., a_N \right]^T \tag{4}$$

Where, $\bar{a}_t = \left(a_{1t}, ..., a_{Nt} \right)^T$,

$a_i = (a_{it}, \ldots., a_{iT})^T$ and

$a_{it} = p_{it} \; or \; r_{it}$

AM represents spatio temporal pattern of access of a document *i*. $\bar{a}_t = \left(a_{1t}, ..., a_{Nt} \right)^T$ represents spatial distribution of popularity at time unit *t* and a_i represents use of document *i* during in varying time unit a_t and related to interest of user. Value of a_i depends on number of clicks and user's browsing time.

Principal Component Analysis (PCA)

PCA was invented by Karl Pearson in 1901 also known as Karhunen-Loeve transform. PCA is a commonly used technique for determining the patterns of high dimension in data. Primarily idea of this statistical method is that a dataset can be analysed in terms of relationship among the individual points present in that specific set. PCA explores the correlations between each of the feature and determine the important axis to express the scattering of data. It transforms original variables of large number into lesser number. This is done by few orthogonal linear combinations for original variables with greatest variance[26]. The goals of PCA are to (a) extract the most important information from the data table, (b) compress the size of the data set by keeping only this important information, (c) simplify the description of the data set, and (d) analyze the structure of the observations and the variables[27].

In order to achieve these goals, PCA computes new variables called principal components which are obtained as linear combinations of the original variables. The first principal component is required to have the largest possible variance (i.e., inertia and therefore this component will "explain" or "extract" the largest part of the inertia of the data table). The second component is computed under the constraint of being orthogonal to the first component and to have the largest possible inertia. The other components are computed likewise. The values of these new variables for the observations are called factor scores, these factors scores can be interpreted geometrically as the projections of the observations onto the principal components.

PCA defines independence by considering the variance of the data in the original basis. It seeks to de-correlate the original data by finding the directions in which variance is maximised and then use these directions to define the new basis[28&29]. Recall the definition for the variance of a random variable, Z with mean, μ in equation (5).

$$\sigma_Z^2 = E\left[\left(Z - \mu \right)^2 \right] \tag{5}$$

Suppose it have a vector of n discrete measurements, $\tilde{r} = (\tilde{r}_1, \tilde{r}_2, ..., \tilde{r}_n)$, with mean μ_r . If it subtracts the mean from each of the measurements, then it obtain a translated set of measurements $r = (r_1, r_2, ..., r_n)$, that has zero mean. Thus, the variance of these measurements is given by the relation in equation (6):

$$\sigma_r^2 = \frac{1}{n} r r^T \tag{6}$$

If it has a second vector of n measurements, $s = (s_1, s_2, ..., s_n)$, again with zero mean, then it can generalise this idea to obtain the covariance of r and s. Covariance can be thought of as a measure of how much two variables change together. Variance is thus a special case of covariance, when the two variables are identical. It is in fact correct to divide through by a factor of n − 1 rather than n, a fact which it shall not justify here in equation (7),

$$\sigma_{rs}^2 = \frac{1}{n-1} r s^T \tag{7}$$

It can now generalise this idea to considering the m × n data matrix, X. Recall that m was the number of variables, and n the number of samples. It can therefore think of this matrix, X in terms of m row vectors, each of length n in equation (8).

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ . & . & . & . \\ . & . & . & . \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_m \end{pmatrix} \in \Box^{m \times n}, \; x_i^T \in \Box^{n} \tag{8}$$

Since it have a row vector for each variable, each of these vectors contains all the samples for one particular variable. So for example, xi is a vector of the n samples for the ith variable. It therefore makes sense to consider the following matrix product in equation (9).

$$C_X = \frac{1}{n-1} XX^T = \frac{1}{n-1} \begin{pmatrix} x_1 x_1{}^T & x_1 x_2{}^T & \dots & x_1 x_m{}^T \\ x_2 x_1{}^T & x_2 x_2{}^T & \dots & x_2 x_m{}^T \\ . & . & . & . \\ . & . & . & . \\ x_m x_1{}^T & x_m x_2{}^T & \dots & x_m x_m{}^T \end{pmatrix} \in \Box^{m\times n} \qquad (9)$$

If it look closely at the entries of this matrix, it see that it have computed all the possible covariance pairs between the m variables. Indeed, on the diagonal entries, it have the variances and on the off-diagonal entries, it have the co-variances. This matrix is therefore known as the Covariance Matrix.

Support Vector Machine (SVM)

SVM method can get the optimal solution whether the sample size tends to be finite or infinite. By selecting the non-linear mapping function (kernel function) to map the data samples that cannot be linear separated to high dimensional feature space and structuring the optimal hyperplane in the high-dimensional feature space, a non-linear-separable problem can be converted into a linearly separable one in the high dimensional feature space. Besides, it solves the dimension problem and its complexity has nothing to do with the sample's dimension[30].

It consider data points of the form in equation (10)[31]:

$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)........., (x_n, y_n)\} \qquad (10)$$

Where $y_n = 1/-1$, a constant denoting the class to which that point x_n belongs. n = number of sample. Each x_n is p-dimensional real vector. The scaling is important to guard against variable (attributes) with larger variance. It can view this Training data, by means of the dividing (or separating) hyperplane, which takes in equation (11):

$$w \,.\, x \;+\; b \;=\; o \qquad (11)$$

Where b is scalar and w is p-dimensional Vector. The vector w points perpendicular to the separating hyperplane. Adding the offset parameter b allows us to increase the margin. Absent of b, the hyperplane is forced to pass through the origin, restricting the solution. As it was interesting in the maximum margin, are interested SVM and the parallel hyperplanes. Parallel hyperplanes can be described by equation (12):

$$w.x \;+\; b \;=\; 1$$

$$w.x \;+\; b \;=\; -1 \qquad (12)$$

If the training data are linearly separable, it can select these hyperplanes so that there are no points between them and then try to maximize their distance. By geometry, it finds the distance between the hyperplane is 2 / │w│. So, it wants to minimize │w│. To excite data points, it need to ensure that for all I either in equation (13):

$$w.\, x_i \;.\, b \;.\, 1 \; or \; w.\, x_i \;.\, b \;.\; -1 \qquad (13)$$

This can be written as equation (14):

$$y_i \;(\, w.\, x_i \;.\, b) \geq 1 \,, \quad 1 \leq i \leq n \qquad (14)$$

Samples along the hyperplanes are called Support Vectors (SVs). A separating hyperplane with the largest margin defined by M = 2 / │w│ that is specifies support vectors in equation (15) means training data points closets to it.

$$y_j [w^T .x_j + b] = 1 \qquad , i = 1 \qquad (15)$$

SVM classifier may be given by equation (16)[32]:

$$\eta = \sum_{i=1}^{M} \alpha_i y_i K(\varphi_i, \varphi) + b \qquad (16)$$

wherein η refers to the classification outcome for the sample, refers to the Lagrange multiplies, y_i represents the category, and $y_i \in \{-1, 1\}$. K (φ_i, φ) denotes the kernel function while b denotes the deviation factor. The optimum hyperplane which SVM classifier generated in the higher dimensional features space is equation (17):

$$f(\varphi) = \text{sgn}(\sum_{i \in SV} \alpha_i y_i (K(\varphi_r, \varphi_i) + K(\varphi_s, \varphi_i))) \qquad (17)$$

Wherein equation (18):

$$b = \frac{1}{2} \sum_{i \in SV} \alpha_i y_i (K(\varphi_r, \varphi_i) + K(\varphi_s, \varphi_i)) \qquad (18)$$

SV represents the Support Vector and φ_r implies positive support vector, φ_s implies negative support vector. The coefficient may be got through the quadratic programming below in equation (19 & 20):

$$\max \;\; w(a) = \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j K(x_i, x_j) \qquad (19)$$

$$s.t. \; \sum_{i=1}^{n} a_i y_i = 0$$

$$0 \leq \alpha_i \leq C \;\; (i = 1, 2, ..., M), \qquad (20)$$

wherein C represents the variable for pricing the misclassification. Prior to SVM's classification of traffics, it ought to undergo training procedure for developing classification model. It uses the LibSVM library for implementing SVM.

In the SVM, there are kernels, as listed below, and any of those can be chosen to achieve the boundary function. Their detailed usages and descriptions, including parameters definitions, can be found in equation (21 to 24)[33]:

Linear kernel: $k(x_i, x_j) = x_i . x_j$ (21)

Polynomial kernel: $k(x_i, x_j) = (\gamma x_i^T x_j + r^d)^2$ (22)

RBF kernel: $k(x_i, x_j) = \exp(-\gamma \,\|\, x_i - x_j \,\|^2)$ (23)

Sigmoid kernel: $k(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$ (24)

SVM's drawback limits use of SVM on academic and industrial platforms: there are free parameters (SVM hyper parameters/SVM kernel parameters) that are user defined. As SVM regression models quality depends on proper setting of parameters, the issue for practitioners applying SVM is how to set parameter values (to ensure good generalization performance) for a training dataset.

To ensure an efficient SVM model, two extra parameters: C and σ^2 (sigma squared) are carefully predetermined. First parameter C determines trade-offs between minimization of fitting error and minimization of model complexity. Second parameter, σ^2, is Radial Basis Function (RBF) kernel bandwidth. In this work, BAT and Cuckoo Search (CS) algorithm is used to optimize the parameter selection for SVM.

BAT Algorithm

Bat Algorithm, initially proposed by Yang, is inspired by echolocation behaviour of bats. Echolocation is one kind of sonar which is used by bats to forage prey and also to avoid obstacles in their path. Bats transmit very loud and high frequency sound continuously and listen for the echo that reflects back from the surrounding objects. Thus a bat can compute direction and distance of the object from the transmitting and receiving wave. Moreover bats can discriminate between a prey and an obstacle easily even in complete darkness[34].

The bat algorithm with the following three idealised rules[35]:

- All bats use echolocation to sense distance, and they also ‘know’ the difference between food/prey and background barriers in some magical way;
- Bats fly randomly with velocity v_i at position x_i with a frequency $f_{\min}$, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0,1]$, depending on the proximity of their target;
- Although the loudness can vary in many ways, it assume that the loudness varies from a large (positive) A_0 to a minimum constant value $A_{\min}$.

Bat is a swarm intelligence algorithm which performs searches using a population of agents. For SVM parameter selection, BAT will search for the best C and σ based on the accuracy of SVM. Each agent i has a current position $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,d})^t$ and a current flying velocity $v_i = (v_{i,1}, v_{i,2}, ..., v_{i,d})^t$, where d is the problem dimension. To find the optimal position, each agent (or bat) updates its position and velocity according to the following equations[36]:

Movement of Virtual Bats: After a random initialization, the new positions (solutions) and velocities at time step t are updated as equation (25):

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \times \beta$$
$$v_i^{t+1} = v_i^t + (x_i^t - x_*) \times f_i$$
$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{25}$$

where $\beta \in [0,1]$, is a random vector drawn from a uniform distribution. $x*$ is the current global best solution. Initially, each bat is randomly assigned with a frequency in $[f_{\min}, f_{\max}]$. The values of $f_{\min}$ and $f_{\max}$ depend on the domain size of the problem of interest.

For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally according to the equation (26):

$$x_{new} = x_{old} + \varepsilon * A^t \tag{26}$$

where ε is a random vector drawn from a uniform distribution in the interval [−1, 1]. A^t Is the average loudness of all the bats at time step t. The global best solution $x*$ can be updated when the best fitness value obtained by all the N bats is superior to the previous f ($x*$).

Loudness and Pulse Emission: The loudness A_i^t and emission rates r_i^t decrease and increase respectively, only if the new solutions are updated, which means that these bats are moving towards their prey[37&38]. These can be formulated as equation (27).

$$If\,(rand(0,1) < A_i^t \;\&\&\, f(x_i) < f(x))$$
$$f(x) = f(x_i)$$
$$A_i^{t+1} = \alpha A_i^t$$
$$r_i^{t+1} = r_i^0 (1 - e^{-\gamma t}) \tag{27}$$

where rand (0, 1) is a random vector drawn from a uniform distribution. αandγ are two constants, α is similar to the cooling factor of a cooling schedule in simulated annealing. The initial loudness A_i^0 and pulse rate r_i^0 are random numbers uniformly distributed in the interval [1, 2] and [0, 1], respectively.

Cuckoo Search (CS) Algorithm

CS optimization algorithm is one of evolutionary algorithms and it was introduced by Yang and Deb in the year in 2009. The lifestyle and behavior of a bird called the Cuckoo was inspired by the developers of this algorithm. The brooding nature of this bird is different from the other birds. Cuckoo bird does not use its nest for laying the eggs and use other bird’s nest for laying eggs. If the host bird finds that the eggs are not belongs to other bird, it will throw away or leave the nest. The grown cuckoo bird becomes a mature bird, and then it continues the mother's life instinctively[40]. The cuckoo is considered special bird because it has many of the characteristics that distinguish it from other birds. It is characterized by aggressive breeding strategy. Cuckoo lays their eggs in the nest of another species, sometimes the cuckoo's egg in the host nest is discovered may lead to the removal of other eggs or abandons the nest and builds their own brood somewhere else in.

In a simple form, each nest has an egg. The algorithm is capable of being extended to more complicated cases where each nest has many eggs representing a solution set. CS is based on three idealized rules[41]:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;
- The best nests with high quality of eggs (solutions) will carry over to the next generations;
- The number of available host nests is fixed, and a host can discover an alien egg with probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest to build a completely new nest in a new location.

Typically, foraging routes of creatures is technically an arbitrary walk as the next move is on the basis of current position and transition probability to the next locale. Selecting the direction relies on a certain probability that may be abstracted mathematically. Several researches have employed these activities in optimizations, optimal searches and initial outcomes reveal its promise[42].

Levy flight is the most popular technique used and handles: The generation of how each step should be. The random direction of flight which is given by equation (28):

$$L = \frac{u}{|v|^{1/\beta}} \tag{28}$$

Where β is the scaling value with a range of [1, 2]. $u$and v are generated from normal distribution and shown in equation (29):

$$u \square N(0, \sigma_u^2), \; \text{v} \square N(0, \sigma_v^2) \tag{29}$$

Where σu and σv are calculated using equation (30):

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma[(1+\beta)/2]\beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \; \sigma_v = 1 \tag{30}$$

where Γ is the gamma function.

A huge problem in metaheuristic design as well as calibration is not merely in building them for maximum performance, but in also making them resilient, in that, they offer a consistent excellent quality of performance over a huge set of problem settings and features[39]. Parallel metaheuristics focus on addressing both problems. In the proposed hybrid CS-BAT algorithm, the CS and BAT algorithms start from various initial solutions which help search various areas of the solution space and return various solutions. The various areas of the solution space searched become a source of parallelism for metaheuristic techniques. In the proposed hybrid heuristics SVM CS-BAT algorithm, the initial random solutions are split and both the algorithms are run in parallel. The best of the solutions from both the techniques become the initial solution for the next iteration.The local optima is done faced by CS and BAT is overcome by using greedy algorithm.

Results and Discussion

In this section, the SVM, PCA-SVM, SVM-BAT, PCA-SVM-BAT, SVM-CS BAT parallel, PCA SVM-CS BAT parallel, SVM-CS and PCA-SVM-CS technique are evaluated. The table 1 shows the summary of results. Figure 1 & 2 shows the detection rate and false positive rate. Table 2 & figure 3 shows best fitness of CS, BAT & hybrid.

Table 1
Summary of Results

Techniques Used	Detection Rate	False Positive Rate
SVM (10,0.01)	92.82	0.0484
PCA - SVM (10,0.01)	91.01	0.0482
SVM-BAT	93.59	0.0369
PCA SVM BAT	92.94	0.037
SVM - CS BAT Parallel	95.27	0.0211
PCA SVM - CS BAT Parallel	96.76	0.0156
SVM CS	93.87	0.0479
PCA - SVM CS	94.78	0.0473

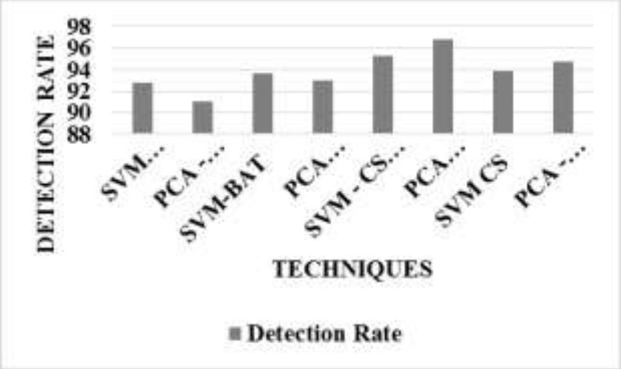

Figure 1: Detection Rate

From the figure 1, it can be observed that the PCA SVM - CS BAT Parallel has higher detection rate by 4.15% for SVM (10, 0.01), by 6.12% for PCA - SVM (10, 0.01), by 3.33% for SVM-BAT, by 4.02% for PCA SVM BAT, by 1.55% for SVM - CS BAT Parallel, by 3.03% for SVM CS and by 2.06% for PCA-SVM CS.

From the figure 2, it can be observed that the PCA SVM - CS BAT Parallel has lower false positive rate by 102.5% for SVM (10, 0.01), by 102.19% for PCA - SVM (10, 0.01), by 81.14% for SVM-BAT, by 81.36% for PCA SVM BAT, by 29.97% for SVM - CS BAT Parallel, by 101.73% for SVM CS and by 100.79% for PCA-SVM CS.

From the figure 3, it can be observed that the CS has averagely higher best fitness by 2.52% for BAT and by 21.32% for hybrid.

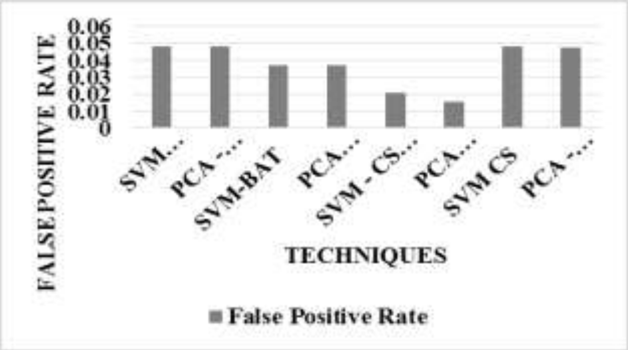

Figure 2: False Positive Rate

Table 2
Best Fitness

Number of iterations	CS	BAT	Hybrid
1	0.9569	0.9345	0.9238
20	0.9214	0.9135	0.9007
40	0.8966	0.8879	0.8513
60	0.85	0.8309	0.8302
80	0.8397	0.824	0.8126
100	0.8125	0.7873	0.7472
120	0.7943	0.7472	0.7293
140	0.6996	0.6894	0.6282
160	0.6564	0.6551	0.5657
180	0.6332	0.6229	0.5268
200	0.5988	0.5767	0.4106
220	0.5373	0.5286	0.3606
240	0.5155	0.489	0.3096
260	0.4736	0.4526	0.2828
280	0.4318	0.4104	0.2194
300	0.343	0.2957	0.2031
320	0.3308	0.2906	0.1763
340	0.2759	0.2508	0.1632
360	0.2616	0.2515	0.163
380	0.2531	0.2488	0.1645
400	0.2431	0.2506	0.1653
420	0.2431	0.2513	0.1636
440	0.2431	0.2471	0.1627
460	0.2406	0.2519	0.1658
480	0.2425	0.2471	0.1651
500	0.2406	0.2472	0.1653
520	0.241	0.2501	0.1645

Conclusion

This work proposed the optimization technique for the Support Vector Machine classifier to improve the classification rate of Application Level Distributed Denial of Service Attacks. In the feature selection stage Principal Component Analysis was used with improved results. To overcome the local optima problem faced by Cuckoo Search and BAT algorithm, a parallel metaheuristic architecture is proposed which showed improved prediction rates when tested on real time data. Further work can be carried out to investigate the impact of serialization of the proposed technique.

References

1. Gupta B.B., Joshi R.C. and Misra M., Distributed denial of service prevention techniques, arXiv preprint arXiv:1208.3557 **(2012)**

2. Mittal A., Shrivastava A.K. and Manoria M., A review of DDoS attack and its countermeasures in TCP based networks, *International Journal of Computer Science and Engineering Survey*, **2(4)**, 177 **(2011)**

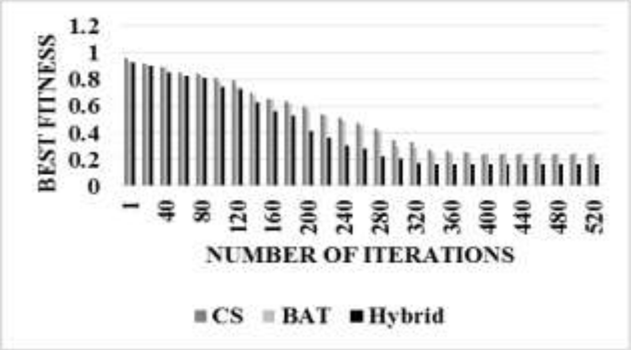

Figure 3: Best Fitness

3. Bhaya W. and Manaa M.E., Review Clustering Mechanisms of Distributed Denial of Service Attacks, *Journal of Computer Science*, **10(10)**, 2037 **(2014)**

4. Bharathi R., Sukanesh R., Xiang Y. and Hu J., A PCA based framework for detection of application layer ddos attacks, *Wseas transactions on information science and applications*, **9(12)**, 389-398 **(2012)**

5. Raghu D., Arani M. and Jacob C.R., Comparison of DDoS Attacks and Fast ICA Algorithms on The Basis of Time Complexity, *International Journal of Computer Applications in Engineering Sciences* **(2011)**

6. Bhuyan M.H., Kashyap H.J., Bhattacharyya D.K. and Kalita J.K., Detecting distributed denial of service attacks: methods, tools and future directions, *The Computer Journal* **(2013)**

7. Ankali S.B. and Ashoka D.V., Detection architecture of application layer DDoS attack for internet, *International Journal of Advanced Networking and Applications*, **3(1)**, 984 **(2011)**

8. Thapngam T., Yu S., Zhou W. and Makki S.K., Distributed Denial of Service (DDoS) detection by traffic pattern analysis, *Peer-to-peer Networking and Applications*, **7(4)**, 346-358 **(2014)**

9. Aburomman A.A. and Reaz M.B.I., A novel SVM-kNN-PSO ensemble method for intrusion detection system, *Applied Soft Computing*, **38**, 360-372 **(2016)**

10. Robinson R.R. and Thomas C., Evaluation of mitigation methods for distributed denial of service attacks, In 2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA), IEEE, 713-718 **(2012)**

11. Durcekova V., Schwartz L. and Shahmehri N., Sophisticated denial of service attacks aimed at application layer, In Elektro, IEEE, 55-60 **(2012)**

12. Tan Z., Jamdagni A., He X., Nanda P. and Liu R.P., A system for denial-of-service attack detection based on multivariate correlation analysis, *IEEE Transactions on Parallel and Distributed Systems*, **25(2)**, 447-456 **(2014)**

13. Katkar V.D. and Kulkarni S.V., Experiments on detection of Denial of Service attacks using Naive Bayesian classifier, In Green Computing, Communication and Conservation of Energy (ICGCE), 2013 International Conference, IEEE, 725-730 **(2013)**

14. Katkar V.D. and Bhatia D.S., Lightweight approach for detection of denial of service attacks using numeric to binary preprocessing, In Circuits, Systems, Communication and Information Technology Applications (CSCITA), 2014 International Conference, IEEE, 207-212 **(2014)**

15. Gu X., Wang H., Ni T. and Ding H., Detection of application-layer DDoS attack based on time series analysis, *Journal of Computer Applications*, **8**, 36 **(2013)**

16. Barati M., Abdullah A., Udzir N.I., Mahmod R. and Mustapha N., Distributed Denial of Service detection using hybrid machine learning technique, In Biometrics and Security Technologies (ISBAST), 2014 International Symposium, IEEE, 268-273 **(2014)**

17. Htun P.T. and Khaing K.T., Detection Model for Daniel-of-Service Attacks using Random Forest and k-Nearest Neighbors, *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, **2 (2013)**

18. Subbulakshmi T., Bala Krishnan K., Shalinie S.M., Anand Kumar D., Ganapathi Subramanian V. and Kannathal K., Detection of DDoS attacks using Enhanced Support Vector Machines with real time generated dataset, In 2011 Third International Conference on Advanced Computing, IEEE, 17-22 **(2011)**

19. Zhou W., Jia W., Wen S., Xiang Y. and Zhou W., Detection and defense of application-layer DDoS attacks in backbone web traffic, *Future Generation Computer Systems*, **38**, 36-46 **(2014)**

20. Wen S., Jia W., Zhou W., Zhou W. and Xu C., CALD: Surviving various application-layer DDoS attacks that mimic flash crowd, In network and system security (nss), 2010 4th international conference, IEEE, 247-254 **(2010)**

21. Ramana V.V., Choudary P.S. and Dhone M.B., Analysis & Study of Application Layer Distributed Denial of Service Attacks for Popular Websites, *International Journal of Computer Science and Telecommunications* **(2011)**

22. Xu X., Wei D. and Zhang Y., Improved detection approach for distributed denial of service attack based on SVM, In Circuits, Communications and System (PACCS), 2011 Third Pacific-Asia Conference, IEEE, 1-3 **(2011)**

23. Online, Available: http://ita.ee.lbl.gov/html/traces.html.

24. Prabha S. and Anitha R., Mitigation of Application Traffic DDoS Attacks with Trust and AM Based HMM Models, *International Journal of Computer Applications IJCA*, **6(9)**, 26-34 **(2010)**

25. Xie Y. and Yu S.Z., Monitoring the application-layer DDoS attacks for popular websites, *Networking, IEEE/AcM Transactions*, **17(1)**, 15-25 **(2009)**

26. Kausar N., Samir B.B., Sulaiman S.B., Ahmad I. and Hussain M., An approach towards intrusion detection using PCA feature subsets and SVM, In Computer & Information Science (ICCIS), 2012 International Conference, IEEE, 569-574 **(2012)**

27. Abdi H. and Williams L.J., Principal component analysis, *Wiley Interdisciplinary Reviews: Computational Statistics*, **2(4)**, 433-459 **(2010)**

28. Shlens J., A tutorial on principal component analysis, arXiv preprint arXiv:1404.1100 **(2014)**

29. Richardson M., Principal component analysis, URL: http://people. maths. ox. ac. uk/richardsonm/SignalProcPCA. pdf (last access: 3.5. 2013), Aleš Hladnik Dr., Ass. Prof., Chair of Information and Graphic Arts Technology, Faculty of Natural Sciences and Engineering, University of Ljubljana, Slovenia ales. hladnik@ ntf. uni-lj. Si **(2009)**

30. Liu Y., Yin J., Cheng J. and Zhang B., Detecting DDoS attacks using conditional entropy, In 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), IEEE, **13**, V13-278 **(2010)**

31. Durgesh K.S. and Lekha B., Data classification using support vector machine, *Journal of Theoretical and Applied Information Technology*, **12(1)**, 1-7 **(2010)**

32. Ni T., Gu X., Wang H. and Li Y., Real-time detection of application-layer DDoS attack using time series analysis, *Journal of Control Science and Engineering* **(2013)**

33. Horng S.J., Su M.Y., Chen Y.H., Kao T.W., Chen R.J., Lai J.L. and Perkasa C.D., A novel intrusion detection system based on hierarchical clustering and support vector machines, *Expert Systems with Applications*, **38(1)**, 306-313 **(2011)**

34. Mandal S., Saha G. and Pal R.K., Recurrent Neural Network Based Modeling of Gene Regulatory Network Using Bat Algorithm, arXiv preprint arXiv:1509.03221 **(2015)**

35. Yang X.S. and He X., Bat algorithm: literature review and applications, *International Journal of Bio-Inspired Computation*, **5(3)**, 141-149 **(2013)**

36. Enache A.C. and Sgarciu V., Enhanced intrusion detection system based on bat algorithm-support vector machine, In 2014 11th International Conference on Security and Cryptography (SECRYPT), IEEE, 1-6 **(2014)**

37. Meng X., Gao X.Z. and Liu Y., A Novel Hybrid Bat Algorithm with Differential Evolution Strategy for Constrained Optimization, *International Journal of Hybrid Information Technology*, **8(1)**, 383-396 **(2015)**

38. Yang X.S., A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization (NICSO 2010), Springer Berlin Heidelberg, 65-74 **(2010)**

39. Balasubramaniyan S. and Sivakumaran T., Optimal location of facts devices for power quality issues using pso and bat algorithm, *Journal of Theoretical & Applied Information Technology*, **64(1) (2014)**

40. Baskan O., Determining optimal link capacity expansions in road networks using Cuckoo Search algorithm with Lévy Flights, *Journal of Applied Mathematics* **(2013)**

41. Yang X.S. and Deb S., Engineering optimization by cuckoo search, *International Journal of Mathematical Modelling and Numerical Optimization*, **1**, 330–343 **(2009)**

42. Valian E., Mohanna S. and Tavakoli S., Improved cuckoo search algorithm for global optimization, *International Journal of Communications and Information Technology*, **1(1)**, 31-44 **(2011)**.