I J C T A, 9(41) 2016, pp. 1012-1021 ISSN: 0974-5572
© International Science Press

FEATURE SELECTION USING MODIFIED CHARGED SYSTEM SEARCH FOR INTRUSION DETECTION SYSTEMS IN CLOUD ENVIRONMENT

Shivam Shakti[1], Partha Ghosh[1], Rupesh Kumar[2] and Santanu Phadikar[3]

Abstract: Cloud Computing is one of the most disruptive technology of modern age. It has revolutionized the concept of scalable services offered to the users over Internet. As a result of its extensive use, the vulnerabilities present in Cloud owes a threat to its security. Thus an Intrusion Detection System (IDS) is necessary in order to efficiently detect intrusions and protect the Cloud. The amount of incoming traffic is generally huge and it becomes necessary to reduce its volume by selecting only the relevant features for detecting intrusions. In this paper the authors have proposed a Modified-Charged-System-Search (MCSS) algorithm for feature selection. The proposed algorithm effectively manages the exploration-exploitation tradeoff and is fast in convergence. The selected features improve accuracy of the IDS and makes it more reliable. The proposed model is evaluated on KDD Cup 99 and NSL-KDD dataset. The results prove that MCSS algorithm is successful in selecting optimal feature subset for proposed IDS.

Key Words: *Cloud Computing, Intrusion Detection System, Charged System Search (CSS), Feature Selection, KDD Cup 99, NSL-KDD*

I. INTRODUCTION

Cloud computing is the most eminent way to increase the capacity or add capabilities to any existing infrastructure dynamically without training any new personnel or licensing new software. Cloud computing serves scalable, virtualized on-demand services to the end users with higher flexibility and lesser infrastructural investment. In recent few years, Cloud computing has grown tremendously from being a promising business concept to one of the fastest flourishing segments of the IT industry. It includes Cloud service models i.e. Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [1]. IaaS is the most commonly known Cloud service model in which resources like physical or virtual machines are provided to the users. The PaaS model provides a Computing platform which typically includes Operating System (OS), execution environment, database and web server. The SaaS model provides installation and operation of application software in the Cloud. Along with the several benefits of the Internet, it also comes with some serious issues regarding systems security, data security and user privacy. In order to make the networks secured, various methods have been developed such as Intrusion Detection Systems (IDS) and Firewalls. An intrusion is an unauthorized access to resources in a network, made by exploiting the vulnerable resources of the network. Intrusion detection and prevention in Cloud infrastructure is the major concern after data security. Intrusion Detection System (IDS) plays a vital role to maintain security and minimize the damage of network and computer security system [2].
An Intrusion Detection System (IDS) is defined as an effective security technology, which can detect and even react to computer related malicious activities [3], [4]. An IDS keeps an eye on the statistics of network activities for potential intrusion and security attacks [5]. IDS can be used to detect different types

of malicious network communications and computer system usage whereas conventional techniques such as firewalls are easily vulnerable to attack and often prone to errors in case of wrong configuration or ambiguous security policies [6]. IDS can be deployed at two different parts of the network i.e. at the network level and at the system level. On this basis, there are two different models of IDS [7]. Host Intrusion Detection Systems (HIDS) monitor system activities of the host computers on which they are installed. Network Intrusion Detection Systems (NIDS) are deployed at the choke points of the network [8]. They monitor and analyze network activities and track abnormal activities in the network. IDS mainly uses two methods to detect attack namely: Misuse detection and Anomaly detection [9]. In Anomaly detection, the IDS has a verified copy of the normal network or/and system behavior. If any incoming or outgoing packet deviates from this copy, then it is considered to be an abnormal behavior. Apart from this, in case of misuse detection the system is facilitated with a number of attack descriptions known as signature. This signature is used to match against the tracked data to detect threats. Using misuse detection we cannot find any new threat. Anomaly detection technique allows IDSs to detect unknown or new threats. These Anomaly detection models make use of Data Mining algorithms to extract meaningful patterns from the incoming traffic data [10]. These models use a number of attributes of an incoming traffic to do the anomaly detection. However, most of the selected features are deemed to be redundant which generally made the model suffer. Thus, feature selection algorithms are used to make Intrusion Detection Systems faster, efficient and robust [11], [12]. In this paper the authors have proposed Modified Charged System Search (MCSS) algorithm for feature selection in order to extract meaningful features and remove redundant features. The authors show that relevant feature subset can be selected efficiently by their metaheuristic search algorithm, which leads to an improved performance of Intrusion Detection Systems with increase in its detection rate .

The rest of the paper is organized as follows. Related work and preliminary study detailing the necessary background are discussed in section II and III. A brief description of the dataset used for evaluation is given in section IV. Section V explains the proposed model. The result and conclusion are discussed in section VI and VII respectively.

II. RELATED WORK

The Cloud Computing is a very vast and extensive paradigm thus involving several threats. A comprehensive analysis on these threats to the Cloud is done by Singh et.al [13]. Their work clearly suggests that these threats render Cloud vulnerable to a huge number of attacks and intrusions. This makes the use of an Intrusion Detection System inevitable for the Cloud environment. Patel et.al outlined several methods to implement an IDS for the Cloud which can either be misuse based or anomaly based [14]. They discussed some existing IDS and the challenges involved in development of a Cloud IDS. The use of data mining algorithms for IDS was done by Lee et.al [15]. They proposed to use frequent pattern matching algorithms to create a framework for their model of IDS. One of the significant task before applying an IDS model is to select necessary traffic attributes before evaluation. Nguyen et.al [16] compared the performance of several feature selection approach in their work and concluded that redundant features can be removed without affecting the classification accuracy, rather better selection of feature subset helped in improving the accuracy of IDS. In the subsequent work, use of optimization algorithms as wrapper methods were proposed for feature selection. In wrapper method, set of features is assigned a score based on accuracy and its different combinations are used to search for optimal feature subset. Anusha and Sathiyamoorthy did a comparative study of Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Cuttle Fish Optimization (CFA) in such an approach to show their effectiveness in feature selection [17]. Several metaheuristic optimization such as Ant Colony Optimization (ACO) proposed by Dorigo et.al [18] and Gravitational Search Algorithm (GSA) proposed by Rashedi et.al [19] proved to be good in case of optimization problems. The ACO algorithm uses a colony of ants which travel through the search space in order to find the optimal solution to a given problem. Al-Ani applied the ACO algorithm for feature selection and proved it to be better than GA in some of the standard feature

selection task [20]. The ACO algorithm was further employed in feature selection task for IDS to select optimal feature subset in the works of Aghdam and Kabiri [21]. It was found in their work that the detection error was significantly reduced by reducing the number of selected features. The Gravitational Search Algorithm is another metaheuristic algorithm in which the agents in search space are masses, which are attracted towards each other governed by gravitational force. As a result of this attraction the agents search the solution space reaching towards an optima. Papa et.al [22] proposed the GSA for feature selection using Optimal-Path Forest Classifier. They performed experiments on standard datasets to show that selected features through GSA gave higher accuracy than other feature selection methods. The Binary GSA algorithm was used for feature selection in IDS by Bhejat et.al [23]. They used a Multi-Layer Perceptron classifier with Binary GSA to show that feature reduction resulted in high classification accuracy and low false positive rate.
The aforementioned use of metaheuristics for feature selection prove that it may result in efficient Intrusion Detection System models. In this paper the authors have proposed a new metaheuristic algorithm using the concept of Charged System Search for relevant feature subset selection aiming to improve performance of the proposed IDS. The following section paves a path by discussing brief background for the proposed model.

III. PRELIMINARY STUDY

In this paper the authors have proposed Modified-Charged-System-Search (MCSS) algorithm for feature selection. The algorithm works on the shortcomings of Charged System Search (CSS) algorithm and improves the exploration-exploitation capability of the CSS algorithm. In this section the authors have described the theories related to the CSS algorithm for any optimization objective.

3.1 Charged System Search

Charged System Search (CSS) is a metaheuristic optimization algorithm proposed by Kaveh and Talatahari [24], which makes use of multiple agents. A number of agents, called Charged Particles (CPs), interact with each other to search for optimal solution in the search space. The algorithm makes use of Coulomb's and Gauss's law to calculate the electrostatic force acting on an agent. The movement of these agent is governed by Newton's laws of motion. The force between two charge particles with charges q_i and q_j is given by F_{ij} and the magnitude of the force is governed by Coulomb's law in equation (1).

$$F_{ij} = K_e \frac{q_i q_j}{r_{ij}^2} \qquad (1)$$

Where K_e is the Coulomb's constant and r_{ij} is the distance of separation between the two charges. A CP is considered a solid spherical charge and electric field due to it at a point outside this sphere is given by equation number (2).

$$E_{ij} = K_e \frac{q_i}{r_{ij}^2} \qquad (2)$$

Whereas, the electric field at a point inside the sphere is governed by Gauss's law and is given by equation (3).

$$E_{ij} = K_e \frac{q_i}{a^3} r_{ij} \qquad (3)$$

Where a is the radius of charged sphere and r_{ij} is the distance between the center of particle and the point at which electric field is calculated. According to the CSS algorithm all the good CPs attract the bad CPs towards itself, thus a force is exerted on all the bad CPs from the good ones. The charge q_i on each CP is given by their normalized fitness value. The total force on a CP is the sum of all the forces acting on it and thus is given by equation (4).

$$F_j = K_e q_j \sum_{i,i\neq j}\left(\frac{q_i}{a^3} r_{ij}.i_1 + \frac{q_i}{r_{ij}^2}.i_2\right) p_{ij}(X_i - X_j) \quad \begin{cases} i_1 = 1, i_2 = 0;\ r_{ij} < a \\ i_1 = 0, i_2 = 1;\ r_{ij} \geq a \end{cases} \tag{4}$$

Where p_{ij} is the probability function whose value determines whether the particle may or may not exert force. The initial positions $X_{i,0}$ are randomly generated and the initial velocity $V_{i,0}$ is taken to be zero for all the particles. The force F_j acting on a particle results in its new position and velocity, which is determined using the Newton's laws of motion in equation (5) and (6) respectively.

$$X_{j,new} = rand_{j1}.k_a.\frac{F_j}{m_j}.\Delta t^2 + rand_{j2}.k_v.V_{j,old}.\Delta t + X_{j,old} \tag{5}$$

$$V_{j,new} = \frac{X_{j,new} - X_{j,old}}{\Delta t} \tag{6}$$

Here k_a and k_v are the coefficients of acceleration and velocity, Δt is the change in time instance. The whole process is iterated until a convergence criteria is satisfied. The best CPs in every consecutive iteration is stored in Charged Memory (CM). This CM guides the CPs in current iteration towards the optimization objective. The CSS algorithm contains many parameters which control the exploration and exploitation but it may still suffer from the lack of randomness in exploration. The authors have proposed a Modified-Charged-System-Search algorithm in order to overcome this limitation and use it for feature selection to develop an efficient Intrusion Detection System.

IV. DATASET DESCRIPTION

In order to evaluate the proposed model the authors have made use of KDD and NSL-KDD dataset. The KDD dataset was prepared as part of 1999 KDD Intrusion Detection Contest. MIT Lincoln Labs set up a simulation representing a US Air force LAN and collected the raw TCP dump data over nine weeks. The original KDD Cup 99 dataset contains 4,898,431 and 311,029 records in train and test set respectively. In order to conduct the experiments the authors have made use of 10% of the KDD Cup 99 dataset. The KDD Cup 99 dataset suffered from many inherent problems as discussed by Tavallaee et.al [25]. The major one among them, being presence of many redundant records which makes the learning biased towards the frequent records. Many of these issues were solved in the NSL-KDD dataset and was made publicly available. The NSL-KDD dataset contains 125,973 and 22,544 records in the train and test set respectively. Although the NSL-KDD dataset still suffers from many problems as discussed by McHugh [26], it can be used as a benchmark to compare research works due to a lack of available data set. The test set has different probability distribution than the train set and this makes the task more realistic and challenging. In both the dataset each record has 41 features and are labelled as attack or normal activity. The 41 features are divided into following categories: basic features, content features, time-based traffic features and host-based traffic features. The dataset contains four types of attack: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L) and Probe. In the above experiments the task is to find the anomalies and thus all the four types of attacks are considered as anomaly to the normal user behavior. The raw dataset contains numeric as well as non-numeric values and thus label encoding is used in order to preprocess the dataset. The presence of continuous and discrete attributes make the task of applying the learning algorithms difficult, thus the dataset needs to be normalized. The min-max normalization method is used in order to bring all the values in a defined range. The final preprocessed and normalized 10% KDD Cup 99 and NSL-KDD dataset is used to evaluate the results of the experiment.

V. PROPOSED MODEL

The presence of huge number of redundant features in the incoming records effect the accuracy of Intrusion Detection Systems in Cloud. The authors have proposed a MCSS algorithm to select important feature subset in order to increase the classification accuracy. The proposed feature selection algorithm is first applied on the train dataset to extract the best possible feature subset and then several classifiers are used to evaluate the performance.

Using the concept of Charged System Search, Charged Particles (CP) in MCSS algorithm are initialized as randomly selected feature subset. The values in the CP vector is either 0 or 1 depending on whether the given feature is selected or not in the subset. The CP vector represents position of the particle in the search space. The set of CPs selected as the initial agents in the search space are represented by their positions $X = \{X_1, X_2, X_3 \dots X_N\}$ and velocity $V = \{V_1, V_2, V_3 \dots V_N\}$, here N is the total number of CPs. The position for each CP is randomly initialized and the initial velocity is assumed to be zero. The charge q_i on each CP is given by the normalized value of its fitness function in equation (7).

$$q_i = \frac{fit(X_i) - fitworst(X)}{fitbest(X) - fitworst(X)} \tag{7}$$

The proposed method uses cross validation accuracy for a classifier as the fitness function for each CP. The CPs are ranked according to their fitness value and best k CPs are stored in the Charge Memory (CM). The particles in CM now attract every other particle in the search space towards itself. The force on each CP due to particles in CM is calculated using equation (4). The radius a and mass m for each particle is assumed to be unity. Since the position vectors of each CP are binary values, hamming distance $h(.)$ is used to calculate the distance of separation between the particles and is given as in equation (8). Here X_{best} is the position of best CP and δ is a small value which prevents singularity.

$$r_{ij} = \frac{h(X_i, X_j)}{h\left((X_i \wedge X_j), X_{best}\right) + \delta} \tag{8}$$

There are two major issues related with many of the metaheuristic algorithm. Firstly, the tradeoff between exploration and exploitation during different stages of optimization. Secondly, to deal with agents which violate the search space and treads off the boundary of the space. The authors have proposed to solve both the problems in the following ways. The MCSS algorithm strikes good notches when considering the exploration-exploitation problem. However due to the random initialization of particles only at the beginning of process, the search may be influenced by it. This results in less amount of exploration necessary specially in a feature selection task where the position of CP are binary vectors and are actually represented as corners in a hyper dimensional cube. As a result a small change in position of CP vector can have huge impacts on direction of the search for optimal solution.

Gauss's and Coulomb's are the two laws which govern force acting on each CP. These are also responsible for controlling the exploration during the initial phase and exploitation during the end of search. As the CPs are far apart in the search space they are attracted by the Coulomb's law which increases as the particle come closer, this controls the exploration in early stages. As the CPs come closer during search, the forces are calculated by the Gauss's law which reduces the magnitude of force as they come closer and thus controlling the exploitation at the end. In order to overcome the problem of randomly selecting CP only in the initial step, the authors have proposed to add a few random particles in the search space with every iteration. The use of random particles in every iteration makes the MCSS algorithm more dynamic and increases its ability to explore the search space. As a result of this addition of particles, the direction of search is not restricted to only original CPs, rather the search can move towards new particles.

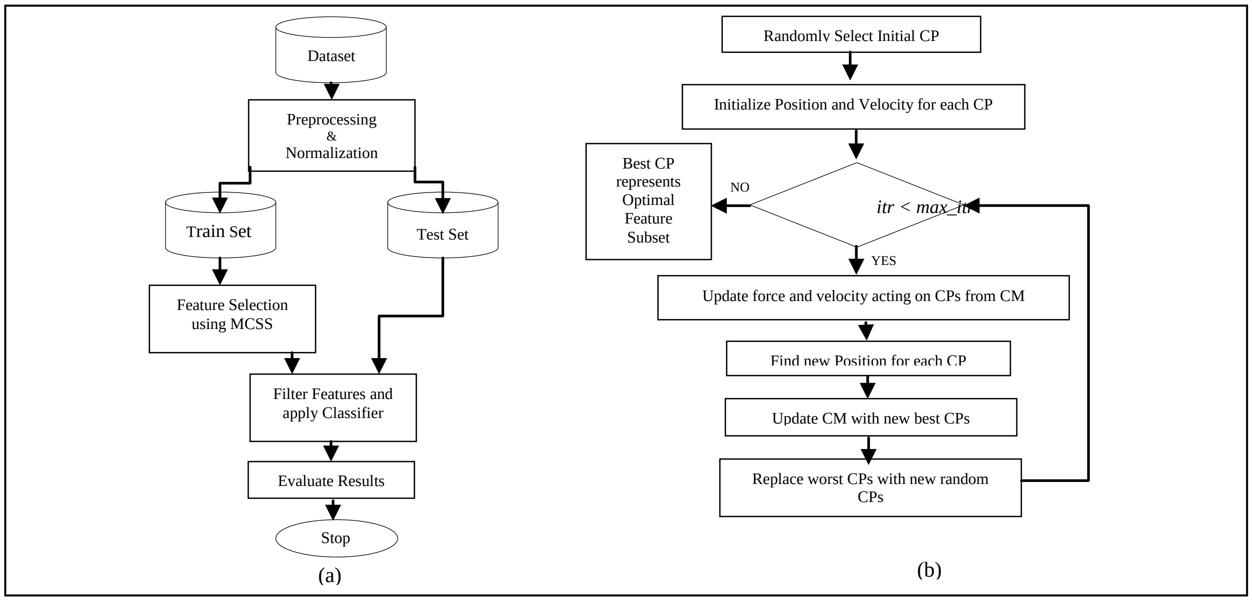

Dataset
Preprocessing & Normalization
Train Set
Test Set
Feature Selection using MCSS
Filter Features and apply Classifier
Evaluate Results
Stop
(a)
Randomly Select Initial CP
Initialize Position and Velocity for each CP
Best CP represents Optimal Feature Subset
NO
itr < max_itr
YES
Update force and velocity acting on CPs from CM
Find new Position for each CP
Update CM with new best CPs
Replace worst CPs with new random CPs
(b)

Figure 1. Flowchart of (a) proposed IDS (b) Feature Selection using MCSS

One of the other advantages of using such an approach is to remove the bad particles from the search and replace them with better ones which can lead to faster convergence. These random CPs do the necessary exploration in early stages and fades away as the search approaches a convergence. In addition to this the authors have also used variable size of CM which reduces with time in order to get a single best solution among the set of possible best solutions. The idea behind this being that as the search moves towards optimal solution, the memory for best solutions should be reduced in order to choose a single solution at the end. The use of equation (5) in the feature selection task may result into value of CP to escape the search space since the values can only be 1 or 0. The authors have proposed to use a new method to update the position of particles. The new velocity for each particle is updated using the following formula in (9).

$$V_{j,new} = rand_{j1}.V_{j,old} + rand_{j2}.k_a.\frac{F_j}{m_j}.\Delta t \tag{9}$$

Where $k_a = 1 - (iteration\_current/iteration\_total)$ which controls the exploitation term. The value of the new velocity determines the new position of each CP. A hyperbolic tangent function is employed to limit the values of velocity and the dimensions which exceed a value are changed by the given equation in (10).

$$X_{ij} = f(x) = \begin{cases} 1 - X_{ij}, & |\tanh(X_{ij})| > 0.5 \\ 0, & otherwise \end{cases} \tag{10}$$

Algorithm: MCSS for feature selection.

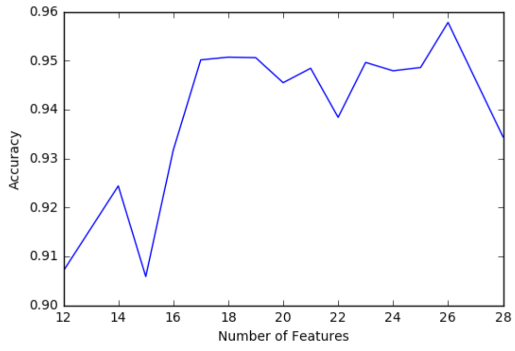
```
1:      initialize positions for N random CPs
2:      initialize velocity for each CP
3:      repeat:         calculate fitness value for each CP
4:              select k best CPs and assign them to CM
5:              calculate magnitude and direction of force on each CP due to particles in CM
6:              find new velocity and position for each CP
7:              replace w worst CPs with new random CPs
8:              update w
9:              update k
10:     while (convergence)
```

The MCSS algorithm is effective towards solving the tradeoff between exploration and exploitation. It also solves the problem of Charged Particles which tend to violate the boundaries of search space. These advantages add to efficient feature subset selection for the Intrusion Detection System. The results for the selected features from this algorithm are discussed in the following section.

VI. Result and Analysis

The security of a Cloud Computing environment is of utmost importance. The Cloud should be up and running at all time to provide on-demand services to its users. This requires an efficient and accurate Intrusion Detection System in order to protect it from different kinds of attack. The proposed anomaly based IDS proved to be reliable and successful in distinguishing anomalous user behaviors from the normal ones. The model's efficiency has been evaluated on the 10% KDD Cup 99 dataset and NSL-KDD dataset. The results for the proposed IDS using Modified-Charged-System-Search (MCSS) algorithm for feature selection is done in two parts.

In the first part of the proposed model, the MCSS algorithm uses cross validation accuracy for fitness value calculation for each Charged Particle. A 10 fold cross validation is performed on the train set for each feature subset and the feature subset with highest accuracy at the end of the search is selected as best feature subset. As the search proceeds towards optimal solution, CPs in search space with different number of selected features attain different accuracy. Figure 2 gives the snapshot of an instance in search where CPs have varying accuracy according to their selected features. This also shows that the redundant features reduce classification accuracy, while extracting relevant features lead to a better classification accuracy. The MCSS algorithm finds the optimal feature subset by initializing random CPs and then guiding bad CPs to the good CPs, searching for optima through the solution space. The cross validation accuracy of fittest CP in each iteration of the search algorithm is plotted in Figure 3. This shows that the MCSS algorithm achieves an early convergence towards an optimal solution.



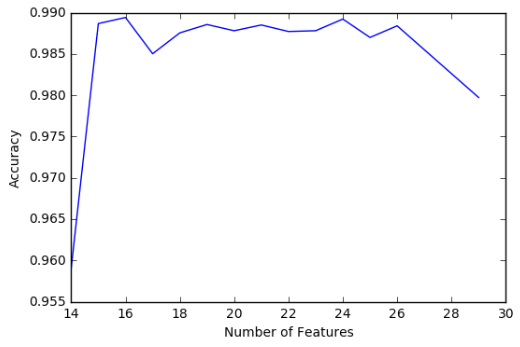
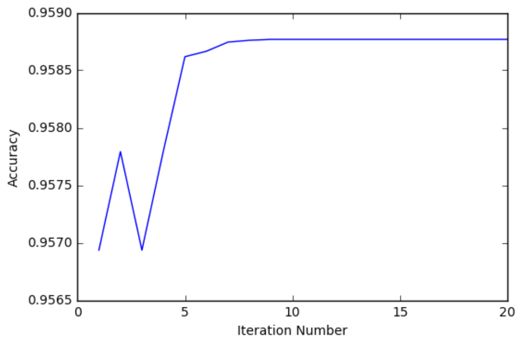


Figure 2. (a) Accuracy vs Features for NSL-KDD (b) Accuracy vs Features for KDD Cup 99



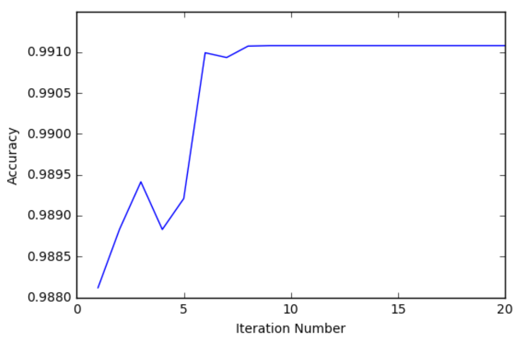


Figure 3. (a) Accuracy vs Iteration for NSL-KDD (b) Accuracy vs Iteration for KDD Cup 99

In the second part, three different classifiers are used to compare the performance of selected features on the train and test set for both the dataset. The authors have done a comparative study of several evaluation metrics for the three classifiers on selected features, to the value of these metrics for all 41 features of the dataset. Table 1 gives the value of these metrics for train and test set on Logistic Regression, Ada Boost and Random Forest classifiers using 41 features. These values are compared to the values of metrics by the three classifiers on the selected features given in Table 2. After feature selection, the results are produced using 21 features for NSL-KDD dataset and 17 features for KDD Cup 99 dataset.

Table-1 Evaluation metrics for 41 features

Dataset	Classifier	Train / Test	Accuracy	Precision	Recall	F1 – Score
NSL-KDD Dataset	Logistic Regression	Train	0.954236	0.9624	0.9382	0.9501
		Test	0.753459	0.9257	0.6163	0.7399
	Ada Boost	Train	0.987950	0.9906	0.9834	0.9870
		Test	0.747782	0.9146	0.6143	0.7350
	Random Forest	Train	0.9868463	0.9987	0.9730	0.9857
		Test	0.756919	0.9660	0.5939	0.7356
KDD Cup 99 Datset	Logistic Regression	Train	0.9923525	0.9972	0.9933	0.9952
		Test	0.9193063	0.9957	0.9037	0.9475
	Ada Boost	Train	0.999002	0.9995	0.9992	0.9993
		Test	0.921708	0.9958	0.9066	0.9491
	Random Forest	Train	0.999277	0.99997	0.9991	0.9995
		Test	0.923026	0.9989	0.9054	0.9499

Table-2 Evaluation metrics for selected features

Dataset	Classifier	Train / Test	Accuracy	Precision	Recall	F1 – Score
NSL-KDD Dataset (21 features)	Logistic Regression	Train	0.9577449	0.9695	0.9386	0.9537
		Test	0.7589602	0.9599	0.6016	0.7396
	Ada Boost	Train	0.970835	0.9742	0.9629	0.9685
		Test	0.751242	0.9605	0.5872	0.7288
	Random Forest	Train	0.9842109	0.9982	0.9678	0.9828
		Test	0.7624201	0.9647	0.6048	0.7435
KDD Cup 99 Dataset (17 features)	Logistic Regression	Train	0.99210153	0.9991	0.9911	0.9951
		Test	0.918271	0.9959	0.9022	0.9467
	Ada Boost	Train	0.997224	0.9992	0.9974	0.9983
		Test	0.9242286	0.9959	0.9096	0.9508
	Random Forest	Train	0.9975649	0.99992	0.9970	0.9984
		Test	0.9237207	0.9988	0.9064	0.9504

The results show that the classifiers used in the experiment perform better for the selected features. The Charged Particles in MCSS algorithm strikes balance between the exploration and exploitation during the search as well as it is successful in finding the optimal feature subset which is responsible for increase in accuracy of the classifiers. The proposed IDS model achieve high values for precision and true positive rate. The MCSS algorithm is an efficient metaheuristic optimization technique for feature selection. Thus, this feature selection technique can be applied with better anomaly detection models in order to achieve higher accuracy and lower false alarm rate. The proposed Intrusion Detection System proves to be reliable and effective in order to detection intrusions in the Cloud environment.

VII.CONCLUSION

The ever increasing use of Cloud services has made it more challenging to deal with its security. Use of Intrusion Detection System is an effective way to secure the Cloud from incoming attacks. Anomaly based approaches tend to be better in order to handle novel attacks. Although use of data mining and machine learning algorithms in anomaly detection based approaches have proved to be useful in effective Intrusion Detection Systems, the accuracy of models tend to be limited due to redundant and irrelevant traffic features. The proposed Modified-Charged-System-Search algorithm for feature selection is an effective method to select the best feature subset. The unnecessary attributes of incoming traffic can be removed to make the data more robust for learning algorithms. The results show that proposed model can achieve higher accuracy with less number of features than the total feature set. The proposed model suggested effective ways to handle exploration-exploitation tradeoff. It has also suggested a new approach to restrict the agents escaping the search space. The convergence of the MCSS algorithm is fast and is attributed to better exploration and exploitation. The use of less number of features in IDS improves the accuracy as well as makes it faster and efficient in detecting anomalous user behavior. Though there are several improvements which can be made to the model by incorporating variable weights and radius for each Charge Particle. In their future work the authors look forward to incorporate these features and make an Intrusion Detection and Prevention System (IDPS) which can not only detect attacks but also take necessary steps towards preventing those attacks in order to make the Cloud more secure.

REFERENCES

[1] R. Mittal and K. Soni, “Analysis of Cloud Computing Architectures,” *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 2, no. 5, pp. 2087–2091, 2013.

[2] J. D. Araújo and Z. Abdelouahab, “Virtualization in Intrusion Detection Systems A Study on Different Approaches for Cloud Computing Environments,” *Int. J. Comput. Sci. Netw. Secur.*, vol. 12, no. 11, pp. 9–16, 2012.

[3] C. C. Lin and M. S. Wang, “Genetic-clustering algorithm for intrusion detection system,” *Int. J. Inf. Comput. Secur. Inderscience*, vol. 2, no. 2, pp. 218–234, 2008.

[4] S. Mukherjee and N. Sharma, “Intrusion Detection using Naive Bayes Classifier with Feature Reduction,” *Procedia Technol. Elsevier*, vol. 4, pp. 119–128, 2012.

[5] O. Y. Al-Jarrah, A. Siddiqui, M. Elsalamouny, P. D. Yoo, S. Muhaidat, and K. Kim, “Machine-Learning-Based Feature Selection Techniques for Large- Scale Network Intrusion Detection,” in *Distributed Computing Systems Workshops (ICDCSW), 2014 IEEE 34th International Conference on*, 2014, pp. 177–181.

[6] U. Kanika, “Security of Network Using Ids and Firewall,” *Int. J. Sci. Res. Publ.*, vol. 3, no. 6, pp. 1–4, 2013.

[7] P. Ghosh, A. Saha, and S. Phadikar, “Penalty-Reward Based Instance Selection Method in Cloud Environment Using the Concept of Nearest Neighbor,” in *Procedia Computer Science Elsevier*, 2016, vol. 89, pp. 82–89.

[8] P. Chauhan and N. Chandra, “A Review on Hybrid Intrusion Detection System using Artificial Immune System Approaches,” *Int. J. Comput. Appl.*, vol. 68, no. 20, pp. 22–27, 2013.

[9] P. Ghosh, C. Debnath, D. Metia, and R. Dutta, “An Efficient Hybrid Multilevel Intrusion Detection System in Cloud Environment,” *IOSR J. Comput. Eng.*, vol. 16, no. 4, pp. 16–26, 2014.

[10] H. Han, X.-L. Lu, and L.-Y. Ren, “Using data mining to discover signatures in network-based intrusion detection,” in *IEEE Proceedings of the First International Conference on Machine Learning and Cybernetics*, 2002, pp. 13–17.

[11] S. Zaman and F. Karray, “Features Selection for Intrusion Detection Systems Based on Support Vector Machines,” in *6th IEEE Consumer Communications and Networking Conference. CCNC.*, 2009, pp. 1–8.

[12] Y. Hamid, M. Sugumaran, and L. Journaux, “A Fusion of Feature Extraction and Feature Selection Technique for Network Intrusion Detection,” *Int. J. Secur. Its Appl.*, vol. 10, no. 8, pp. 151–158, 2016.

[13] S. Singh, Y.-S. Jeong, and J. H. Park, “A Survey on Cloud Computing Security: Issues, Threats, and Solutions,” *J. Netw. Comput. Appl. Elsevier*, vol. 75, pp. 200–222, 2016.

[14] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, “An intrusion detection and prevention system in cloud computing: A systematic review,” *J. Netw. Comput. Appl. Elsevier*, vol. 36, pp. 25–41, 2013.

[15] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Security and Privacy, Proceedings of the 1999 IEEE Symposium on*, 1999, pp. 1–12.

[16] H. T. Nguyen, S. Petrović, and K. Franke, "A comparison of feature-selection methods for intrusion detection," in *Computer Network Security Springer*, vol. 6258, 2010, pp. 242–255.

[17] K. Anusha and E. Sathiyamoorthy, "Comparative Study for Feature Selection Algorithms in Intrusion Detection System," *Autom. Control Comput. Sci.*, vol. 50, no. 1, pp. 1–9, 2016.

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Trans. Syst. MAN, Cybern. B Cybern.*, vol. 26, no. 1, pp. 29–41, 1996.

[19] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Inf. Sci. Elsevier*, vol. 179, pp. 2232–2248, 2009.

[20] A. Al-Ani, "Ant Colony Optimization for Feature Subset Selection," *Int. J. Comput. Electr. Autom. Control. Inf. Eng.*, vol. 1, no. 4, pp. 999–1002, 2007.

[21] M. H. Aghdam and P. Kabiri, "Feature Selection for Intrusion Detection System Using Ant Colony Optimization," *Int. J. Netw. Secur.*, vol. 18, no. 3, pp. 420–432, 2016.

[22] J. P. Papa, A. Pagnin, S. A. Schellini, A. Spadotto, R. C. Guido, M. Ponti, G. Chiachia, and A. X. Falcão, "Feature selection through gravitational search algorithm," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2011, no. 1, pp. 2052–2055.

[23] A. R. Behjat, A. Mustapha, H. Nezamabadi–pour, M. N. Sulaiman, and N. Mustapha, "Feature subset selection using binary gravitational search algorithm for intrusion detection system," in *Intelligent Information and Database Systems Springer*, 2013, pp. 377–386.

[24] A. Kaveh and S. Talatahari, "A novel heuristic optimization method : charged system," *Acta Mech.*, vol. 213, no. 3, pp. 267–289, 2010.

[25] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications*, 2009, pp. 1–6.

[26] J. McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 4, pp. 262–294, 2000.