I. J. Computer Network and Information Security, **2019, 3, 22-29**
Published Online March 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2019.03.04

Network Intrusion Detection System based PSO-SVM for Cloud Computing

Mahmoud M. Sakr
Department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt
E-mail: mahmoudsakr@ci.menofia.edu.eg

Medhat A. Tawfeeq
Department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt
E-mail: medhattaw@yahoo.com

Ashraf B. El-Sisi
Department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt
E-mail: ashraf.elsisi@ci.menofia.edu.eg

Received: 15 February 2019; Accepted: 01 March 2019; Published: 08 March 2019

Abstract—Cloud computing provides and delivers a pool of on-demand and configurable resources and services that are delivered across the usage of the internet. Providing privacy and security to protect cloud assets and resources still a very challenging issue, since the distributed architecture of the cloud makes it vulnerable to the intruders. To mitigate this issue, intrusion detection systems (IDSs) play an important role in detecting the attacks in the cloud environment. In this paper, an anomaly-based network intrusion detection system (NIDS) is proposed which can monitor and analyze the network traffics flow that targets a cloud environment. The network administrator should be notified about the nature of these traffics to drop and block any intrusive network connections. Support vector machine (SVM) is employed as the classifier of the network connections. The binary-based Particle Swarm Optimization (BPSO) is adopted for selecting the most relevant network features, while the standard-based Particle Swarm Optimization (SPSO) is adopted for tuning the SVM control parameters. The benchmark NSL-KDD dataset is used as the network data source to build and evaluate the proposed system. Acceptable evaluation results state that the proposed system is characterized by detecting the intrusive network connections with high detection accuracy and low false alarm rates (FARs).

Index Terms—Cloud Computing, Intrusion Detection System, Anomaly Detection, Feature Selection, Particle Swarm Optimization.

I. Introduction

Today, most organizations are relocating their computing services to the Cloud, since this makes their services available more conveniently to their users [1]. Cloud computing is a distributed model which supplies computing resources and services availability, quick accessibility and scalability. According to specific fundamental service models, cloud providers offer their services based on three models. Infrastructure as a Service (IaaS), Platform as Service (PaaS) and Software as a Service (SaaS). IaaS considered as the most used cloud service model in which they provide infrastructure resources such as physical or virtual machines such as Microsoft's Azure [2]. For the PaaS model, computing platforms are provided including development environments, operating systems (OSs), frameworks, libraries and tools such as Google App Engine [2]. For the SaaS model, considered as a complete software solution in which the cloud providers hosting licensed software and making them available for users such as Google apps [2]. For cloud deployment models, there are three popular different models. The public, private and hybrid cloud models. For the public cloud, cloud providers own and manage physical infrastructure and provide their resources to the public users by using the virtualization technique which allows multiple users sharing the same infrastructure and pay for their usage. For the private cloud, it's considered the best option for the companies which own a physical infrastructure as it provides higher security levels than public cloud. For the hybrid cloud, it combines both the public and private cloud models' characteristics where the data and applications can be interchanged between them which is considered as greater flexibility for the using company [1]. Since cloud computing is distributed in its nature, the hackers intensively target the cloud systems for exploiting its involved vulnerabilities [2]. Any unauthorized access by an intruder to the cloud is commonly known as an intrusion, whereas the intrusion detection is the process of monitoring and auditing the events that occur in the systems of the computers or the networks. Detecting and preventing network intrusions in the cloud environment are still from the major security

interests among researchers. The intrusion detection system (IDS) aims to preserve the confidentiality, integrity, and availability (CIA) of the networks and information systems in the cloud environment, since it plays vital roles in the security provisioning against the intruders. Depending on where the IDS is deployed, it can be categorized as four different types. The Host-based IDS (HIDS), Network-based IDS (NIDS), Virtual machine monitor/Hypervisor-based IDS (VMM-IDS) and Collaborative based IDS [2]. HIDS is placed inside the Host or VM to detect attacks inside them [3]. NIDS is placed at the entry network points e.g. routers or switches to be able to identify and detect abnormal behaviours of network events [3]. VMM-IDS can monitor the internal states of VMs from outside them through the VMM/hypervisors [3]. The collaborative-based IDS which consists of several IDSs like the HIDS, NIDS, or a hybrid of them, where they are deployed over a large network and cooperatively working in monitoring its events [3]. Depending on how the intrusion detection takes place, there are three detection methodologies, the anomaly-based, signature-based, and hybrid-based [3]. For the anomaly-based, IDS detect intrusions by looking for the abnormal behaviours patterns of the system activities or network events [4]. For the signature-based, IDS searches for any known attacks patterns by matching the known attacks signatures with the incoming events patterns [4]. For the hybrid-based, it combines both advantages of the signature-based and anomaly-based to provide more extensive intrusion detection capabilities [3]. The machine learning techniques are widely adopted for building the IDSs. In this paper, an anomaly-based NIDS is proposed for the cloud environment. Two types of the Particle swarm optimization (PSO) technique (the standard-based PSO and the binary-based PSO) are combined with the support vector machine (SVM) algorithm to develop the proposed system. NIDS can detect possible network attacks of the incoming network traffics flow which affecting multiple hosts or VMs in the cloud environment. Once detecting any intrusive network events, the network administrator is alerted to take defensive actions against such these suspected traffics. The rest of this paper is systemized as follows: Section II reviews some IDSs related works. Section III presents the proposed NIDS in a detailed manner. Section IV discusses the implementation and experimental results. The conclusion of the proposed work and future directions are presented in section V.

II. Related Works

In this section, some literature related to the proposed work is reviewed. Generally, classification algorithms of the machine learning are used widely to develop the anomaly-based IDSs. The classification algorithm learns the network data which are collected from either the network flow or the network logs to develop the intrusion detection/classification model. The learned detection model is used to detect the anomalies packets of the network flow that affecting VMs or hosts in the cloud environment. Network data preprocessing techniques along with optimizing the detection model are very vital and necessary for improving the efficiency of the IDS. Mostly researchers employing these techniques in developing their IDSs to obtain the best performance.

In [5], P. Ghosh and A. Kumar proposed an efficient IDS with the cooperation of both the HIDS and NIDS for the cloud environment. They used the information gain technique for selecting the most relevant features to enhance detection accuracy. K-nearest neighbour technique used as a network connections classifier, while the neural network used for detecting the types of attacks and the NSL-KDD dataset used to evaluate the detection system. Cloud administrator was notified with a report about the classification results. In [6], Popoola, Ebenezer and A.O. Adewumi proposed an efficient feature selection approach for detecting the intrusions in the network based on the Discrete Deferential Evolution (DDE) and Decision Tree (C4.5). Weka tool was used to apply and validate seven machine learning algorithms on the NSL-KDD dataset. DDE was used for reducing the dataset dimensionality and the classification results obtained demonstrated a significant enhancement on the detection accuracy and a reduction in both training and testing time. In [7], Kevric, Jasmin and S. Jukic proposed a combination of classifiers models based on the tree-based algorithms to detect the network intrusions. NSL-KDD dataset used for the system evaluation. Results state that combining classifiers based on the sum rule schema yield results much better than using an individual classifier. In [8], S. Mehibs and S. Hashim proposed a network-based IDS for the cloud environment. They used the fuzzy c mean algorithm to detect the network intrusive behaviours. The main advantage of their clustering technique was its ability to detect the new attacks. KDD99 dataset used to evaluate the proposed system which was characterized by a low false positive alarm and a high detection rate. In [9], Ngoc and T. Pham et al proposed an IDS by using boosting and bagging ensemble techniques along with the tree algorithm as a base classifier. NSL-KDD dataset was used for evaluating their proposed system. Results demonstrated that using bagging ensemble model with the J48 classifier yield the best performance when working with a subset of 35 selected features in terms of classification accuracy and false alarm rate. In [15], A. Antony and R. Priyadharshini proposed an IDS for the cloud environment. They used the cuckoo optimization-based method for preprocessing the network data due to optimize the detection accuracy. Results stated that their IDS performed better than other approaches compared.

III. The Proposed Anomaly-Based Network Intrusion Detection System (NIDS)

The objective of this research is presenting an anomaly-based NIDS for the cloud environment as shown in Fig. 1.

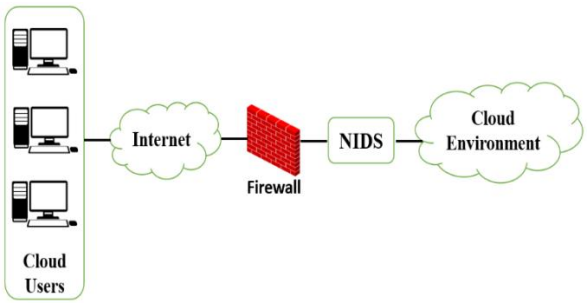

Fig.1. Network Intrusion Detection System in Cloud Environment

NIDS is deployed at the entry point of the cloud network and attached with the switches that connecting the networks of the cloud servers or VMs. NIDS audit and monitor the incoming network traffics flow to the cloud networks. Firstly, the network traffics (raw data) are captured using a packet sniffer. Then, the features of these network data flow are extracted and well preprocessed before the classification process. To reinforce the efficiency of intrusion detection process and decrease its time, the most significant features are selected from the incoming network data flow using the BPSO. These selected features are passed to the SVM anomaly-based detection model to be checked and determine its nature. Once the suspected traffics are detected, the cloud network administrator is notified to drop these suspected traffics and block their source IPs, while allowing other normal traffics to be passed. The benchmark NSL-KDD dataset [14] is used as the network data source for the proposed system. Fig. 2 illustrates the flow of the intrusion detection process. The block diagram of the development processes of the proposed system is shown in Fig. 3, while a description of the development steps is detailed below.

Fig.2. The flow of the Intrusion Detection Process

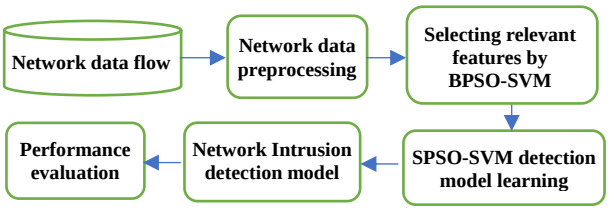

Fig.3. Development Processes of the Network Intrusion Detection System

A. Network Data Pre-processing

Network traffics data are very enormous and containing many features with various types and varied value ranges. Therefore, processing these data directly represents long time-consuming processes that result in inaccurate classifications which are not acceptable in the IDSs. The network data preprocessing methods are detailed as follows:

1. Data Mapping

Many features of the network connections are represented in a nominal form. Majority of classifiers cannot directly deal with such these symbolic data. Therefore, these features needed to be transformed into a numeric form. The class label feature value is converted to be either 0 for the normal classes or 1 for other abnormal classes.

2. Data Normalization

Network connections include many features with a varied range of values which causing a bias toward some network features over the others. Normalizing these features helps in avoiding the bias problem. This leads to improve the IDS effectiveness by converting the connection instances to a standard form by scaling its features values to a determined range based on a specific normalization method. Z-Score is a normalization method which is used to scale features values of each network connection X by the following formula [16]

$$Normalized\ (X_{id}) = \frac{X_{id} - mean\ (\text{feature}_d)}{std\ (\text{feature}_d)} \quad (1)$$

where i represents the instance number of network connection X, while d represents the network feature dimension.

B. Network Feature Selection

Irrelevant features clearly lead to slow the training and testing processes, decrease the classification accuracy and increase the detection time of the used detection model which are not suitable in the IDSs [6,18]. Eliminating the irrelevant features actually reducing the detection model complexity, speeding up the computations, overcoming model overfitting and therefore building a lightweight IDS. Therefore, improving the overall performance of the detection model. BPSO [12] is adopted to reduce the dimensionality of the network connection features by selecting the more optimal/relevant features subset from the incoming network data flow. Each candidate solution is presented as a binary vector of N length, where N is the network connection features number. If the feature value is set to 1 is to be kept and to 0 is to be discarded from the feature subset. Each features subset is evaluated by the SVM detection accuracy which is used as the BPSO fitness function. The pseudo-code description of the BPSO is shown in Table 1.

Table 1. Pseudo-code of the BPSO Algorithm

Algorithm 1: Binary-based Particle Swarm Optimization (BPSO)
Input: *Training data (full features network connection instances), N, M, D, c_1, c_2, r_1, r_2, w, C and σ /* N=Swarm size (no. of particles), M=(max_iterations), D=particle dimension (no. of features), c_1=cognitive learning factor, c_2=social learning factor, r_1 & r_2=random value [0:1], w=inertial coefficient, C=SVM penalty parameter, σ=SVM RBF kernel parameter*/*
Output: *Subset of the significant network features (Gbest)*

For particle i=0 to *N* **do**
  Randomly initialize position vector X_{id} with binary values for the network features
  Randomly initialize velocity vector V_{id}
End For
Initialize t=1
While (t ≠ M) or (fitness_value ≠ 100) **do**
  For each particle i **do**
    Calculate its fitness using SVM classifier
    If its fitness_value is better than $Pbest_{id}$
    Set $Pbest_{id}$ = its current fitness_value
    End If
  End For
  Set *Gbest* = Best previous particle fitness_value
  /* Updating each particle's velocity and position */
  For particle *i=0 to N* **do**
    For dimension d=0 to *D* **do**
      $V_{id}^{t+1} = wV_{id}^{t} + c_1 r_1 (Pbest_i^t - X_{id}^t) + c_2 r_2 (Gbest - X_{id}^t)$ (2)
      $Sigmoid\ (V_{id}^{t+1}) = \frac{1}{1+e^{-V_{id}^{t+1}}}$ (3)
      X_{id}^{t+1} = *1 if the* $Sigmoid\ (V_{id}^{t+1})$ > *random value [0:1], else = 0*
    End For
  End For
  Set t=t+1
End While

Table 2. Pseudo-code of the SPSO Algorithm

Algorithm 2: Standard-based Particle Swarm Optimization (SPSO)

Input: *Training data (optimal features network connection instances), N, D, M, c_1, c_2, r1, r2, w*
/ N= Swarm size (particles number), D=particle dimension (SVM control parameters number), M=(max_iterations), c_1=cognitive learning factor, c_2= social learning factor, r_1 & r_2=random value [0:1], w= inertial coefficient */*

Output: *Optimal values of SVM parameters [C, σ] (Gbest)*

For particle i=0 to *N* **do**
  Randomly initialize its position vector X_{id} with two values according to the [c, σ] ranges
  Randomly initialize its velocity vector V_{id}
End For
Initialize t=1
While (t ≠ *M*) or (fitness_value ≠ 100) **do**
  For each particle i **do**
    Calculate its fitness using SVM classifier
    If its fitness_value is better than $Pbest_{id}$
      Set $Pbest_{id}$ = its current fitness
    End If
  End For
  Set *Gbest* = Best previous particle fitness_value
  /* Updating each particle's velocity and position */
  For particle i=0 to *N* **do**
    For dimension d=0 to *D* **do**
      $V_{id}^{t+1} = wV_{id}^{t} + c_1 r_1 (Pbest_i^t - X_{id}^t) + c_2 r_2 (Gbest - X_{id}^t)$ (4)
      $X_{id}^{t+1} = X_{id}^{t} + V_{id}^{t+1}$ (5)
    End For
  End For
  Set t=t+1
End While

C. Intrusion Detection Model using SVM

SVM with its kernel function RBF is used as the detection model in the proposed NIDS. This detection model is well learned on the most significant network features obtained from the previous feature selection stage. The control parameters values of the SVM have a significant impact on its performance. So, it's much necessary selecting its parameters values which include the penalty parameter (C) that controls the flexibility of the separable hyperplane and the RBF kernel parameter (σ) which controls the correlation among the support vectors and determines the nonlinear mapping to a higher dimensional space. More detail about the SVM algorithm and its kernel functions is presented in [10,11]. SPSO [13] is a searching technique which is adopted to optimize the SVM performance by tuning its control parameters [C, σ]. Searching for these parameters best values is performed using a population of individuals/particles which are updated each generation until obtaining their best values. Each particle represents a candidate solution as a vector of two continuous values for [C, σ]. The detection accuracy of SVM is used as the fitness function to evaluate each candidate solution fitness. The SPSO pseudo-code description is shown in Table 2.

D. Evaluation Metrics of the Proposed NIDS

The study of the proposed NIDS performance was carried out using several metrics which are discussed below and all of them are concluded from the following measures.

True Positive (TP) denotes the number of intrusive connections that are correctly classified as intrusive. *True Negative (TN)* denotes the number of normal connections that are correctly classified as normal. *False Positive (FP)* denotes the number of normal connections that are incorrectly classified as intrusive. *False Negative (FN)* denotes the number of intrusive connections that are incorrectly classified as normal.

1. *Classification Accuracy (CA)* is the ratio between the whole correctly classified instances number to the whole number of existed instances. CA shows whether the proposed NIDS has the ability to raise proper alarms when detecting the attacks and not to generate false alarms when the connection network instance is a normal [17].

$$Classification\ Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

2. *False Positive Rate (FPR)* is another metric that indicates the proportion of the whole misclassified normal instances number to the whole normal instances number. FPR value indicates whether the proposed NIDS would generate many FP alarms. FPR value has to be lower as possible, otherwise, too many FP alarms might confuse the system admin [17].

$$False\ Positive\ Rate = \frac{FP}{FP+TN} \quad (7)$$

3. *False Negative Rate (FNR)* is another metric that can indicate the proportion between the whole misclassified attack instances number to the whole number of attack instances. FNR value shows whether the proposed NIDS would generate many false negative alarms. The value of FNR has to be lower as possible otherwise, too many attacks might not be detected [17].

$$False\ Negative\ Rate = \frac{FN}{FN+TP} \quad (8)$$

4. *Sensitivity or True Positive Rate (TPR)* is another metrics which indicates the ratio of the attacks instances correctly detected by NIDS to the total numbers of attacks class [6].

$$Sensitivity = \frac{TP}{TP+FN} \quad (9)$$

5. *Specificity or True Negative Rate (TNR)* is another metrics which indicates the ratio of the normal instances correctly detected by NIDS to the total numbers of normal class [6].

$$Specificity = \frac{TN}{TN+FP} \quad (10)$$

6. *Precision Rate (PR)* is another metric that can indicate the ratio of positives predicted values which are actually being positive. The proposed NIDS performance is directly affected by PR since a higher PR value indicates a lower value of the false positive rate and vice versa [6].

$$Precision = \frac{TP}{TP+FP} \quad (11)$$

7. *Attack Detection Rate (ADR) or Recall* is another metric that indicates the proportion of the whole detected attacks number by the NIDS to the whole actually existed number of attacks. ADR shows the capabilities of the proposed NIDS in detecting the various attacks [6].

$$Attack\ Detection\ rate = \frac{TP}{TP+FN} \quad (12)$$

8. *F-Measure* is another metric that used for evaluating the accuracy of the proposed NIDS with regarding to both the precision and the recall rate values [18].

$$F\text{-}Measure = \frac{2TP}{2TP+FP+FN} \quad (13)$$

IV. Implementation and Experimental Results

The experiments are conducted on a laptop with an Intel core i7 CPU, 4 GB RAM, and 64-bit Windows 10 as an operating system. NSL-KDD is the dataset used for conducting these experiments and considered as the most used benchmark dataset for developing and evaluating the IDSs [14]. This dataset has 41 network features which are categorized into basic, content-based and time-based features. Moreover, it contains 22 attacks types in the training dataset, while an extra 16 attacks types are included in the testing dataset. All these 38 attacks are categorized as follows [19]:

1. Denial of Service (DOS) attacks: Attackers use bots which are known as zombies for flooding the victim system with a huge packets number to exhaust the target system resources such as network bandwidth and computing power and rendering the victim system resources unreachable e.g. Smurf flooding attacks. These attacks target cloud availability.
2. Probe attacks: Attacker use these attacks to obtain information about the cloud network e.g. closed, opened and filtered ports to launch attacks on the services which are running on these ports e.g. Port scanning attacks. These attacks target both cloud integrity and confidentiality.
3. Remote to Local (R2L) attacks: Attackers sent packets to the target machine for searching any exploit to get unauthorized access to that machine e.g. password guessing attacks. These attacks target cloud integrity.
4. User to Root (U2R) attacks: Attackers try to get access to an authentic cloud instance then exploiting its vulnerabilities to gain the root privileges of a virtual machine or host. These attacks target cloud integrity. All these attacks classes are tabulated in Table 3.

Table 3. Attacks Categories with its types of the NSL-KDD Dataset

Attacks Class	Attacks Types
DoS	Back, Neptune, Land, Pod, Smurf, Udpstorm, Teardrop, Apache2, Worm, Processtable
Probe	Nmap, Ipsweep, Portsweep, Saint, Mscan, Satan
R2L	Ftp_write, Warezmaster, Httptunnel, Guess_Password, Phf, Warezclient, Snmpguess, Multihop, Xlock, Spy, Xsnoop, Sendmail, Snmpgetattack, Imap, Named
U2R	Rootkit, Loadmodule, Xterm, Sqlattack, Perl, PS, Buffer_overflow

Table 4. Parameters value used for the BPSO and SPSO

Parameter name	BSPO	SPSO
Particles number (s)	40	30
Iterations number (t)	30	40
Particle dimensions (d)	41	2
Inertia weight (w)	0.7	0.7
Cognitive learning factor (c_1)	0.8	0.8
Social learning factor (c_2)	0.9	0.9

The Classification model SVM, the BPSO network feature selection algorithm and the SPSO searching algorithm for the SVM control parameters are implemented in Python language. The used parameters values in conducting these algorithms are tabulated in Table 4.

For the used dataset, 20% is used for training the proposed NIDS, while the other 80% is used for evaluating and testing system performance. For the BPSO, the most relevant network features are selected where the dataset dimensionality is reduced from 41 to 23 network features which are presented in Table 5. For the SPSO, the values range determined for the SVM control parameters were [1:4000] for the C parameter and [1:30] for the σ parameter. Consequently, the best value obtained for these parameters [C, σ] were [33, 0.39].

Table 5. The Most Significant Network Features Selected by the BPSO

The most significant network features
duration, service, protocol_type, src_bytes, flag, land, dst_bytes, hot, wrong_fragment, num_failed_logins, urgent, logged_in, root_shell, su_attempted, num_root, num_compromised, num_file_creations, num_access_files, num_shells, num_outbound_cmds, is_guest_login, is_host_login, count

Developing the IDS is carried through conducting three different experiments during the detection model learning stage. Firstly, building its detection model by using the whole standard network features and without any control parameters tuning. Secondly, building its detection model by using the most significant network features and without any control parameters tuning. Thirdly, building its detection model by using the most significant network features then by tuning its control parameters. The IDS training and testing time of the three experiments are shown in Table 6.

Table 6. IDS Training and Testing Time of the Three Experiments

Measures	Intrusion detection system		
	SVM	BPSO + SVM	BPSO + SPSO + SVM
Features number	42	23	23
Training time (Sec)	403.4	6.96	5.3
Testing time (Sec)	263.9	16.2	7.8

The IDS performance evaluations of the last three experiments are tabulated in Table 7. From these experiments, it can be inferred the following. Firstly, the whole network connection features don't contribute equally in the classification process, where some could contribute less, some could contribute much and some could not contribute at all. Therefore, building the intrusion detection model on the basis of the whole standard network connection features results in decreasing the detection accuracy while increasing the FARs, the computation time and the complexity of the used IDS. All these previous cons are not suitable totally in the IDSs. Secondly, optimizing the intrusion detection model through tuning its control parameters results in improving the IDS's efficiency by increasing the classification accuracy, detection rate, TPR, TNR while decreasing the detection time and FPR along with FNR. Regarding other related IDSs, our proposed system is compared with some IDSs included in Refs. [5,6,7,9] as shown in Table 8. Results proved that our proposed system outperformed other related IDSs with a higher classification accuracy, TPR, TNR and a lower FPR.

Table 7. IDS Detailed Performance of the Three Experiments

Metrics	Intrusion detection system		
	SVM	BPSO + SVM	BPSO + SPSO + SVM
Classification accuracy (%)	91.69	97.75	99.10
False positive rate (FPR)	0.0039	0.0148	0.0087
False negative rate (FNR)	0.1672	0.0305	0.0091
True positive rate (TRP)	0.8327	0.9694	0.9908
True negative rate (TNR)	0.9960	0.9851	0.9912
Precession rate (%)	99.50	98.37	99.05
Detection rate / Recall (%)	83.27	96.94	99.08
F- Measure (%)	90.66	97.65	99.07

Table 8. Evaluation Comparison between the Proposed IDS and other Related IDSs.

Metrics	Intrusion detection systems				
	Proposed	[6]	[9]	[7]	[5]
Accuracy (%)	99.10	88.73	84.25	89.24	76.54
TPR (%)	99.08	89	-	83.9	-
TNR (%)	99.12	99.07	97.21	-	97.21
FPR (%)	0.87	0.93	2.79	-	2.79

V. Conclusions

In this paper, an efficient anomaly-based NIDS is proposed to detect different types of attacks (Dos, Probe, R2L, U2R) in the cloud environment. PSO optimization techniques (Binary-based and Standard-based) are integrated with the SVM classification algorithm to develop a robust intrusion detection model. The proposed NIDS is trained and tested on the benchmark NSL-KDD dataset and the evaluation results stated its efficacy in recognizing the normal behaviours and detecting the attacks with high detection accuracy and low rates of false alarms. Moreover, a comparison with other related IDSs is conducted and our proposed system outperformed

these systems. The future work is directed toward employing other evolutionary techniques in optimizing the control parameters of the classification algorithm along with using other optimal network features selection strategies.

REFERENCES

[1] P. Mell and T. Grance, The NIST Definition of Cloud Computing. NIST Special Publication, 2001. pp. 800-145.

[2] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel and M. Radaradan, A Survey of Intrusion Detection Techniques in Cloud. Journal of Network and Computer Applications, 2013. vol. 36, pp. 42–57.

[3] P. Mishra, E.S. Pilli, V. Varadharadan and U. Tupakula, Intrusion detection techniques in cloud environment. Journal of Network and Computer Applications, 2017. vol. 77, pp. 18-47.

[4] Amirreza Zarrabi and Alireza Zarrabi, Internet Intrusion Detection System Service in a Cloud. International Journal of Computer Science Issues, 2012. vol. 9, pp. 308-315.

[5] Partha Ghosh, A.K. Mandal and Rupesh Kumar, An Efficient Cloud Network Intrusion Detection System. Information Systems Design and Intelligent Applications, Advances in Intelligent Systems and Computing, Springer, 2015. vol. 339, pp. 91-99.

[6] Popoola, Ebenezer, A.O Adewumi, Efficient Feature Selection Technique for Network Intrusion Detection System Using Discrete Differential Evolution and Decision. International Journal of Network Security, 2017. vol. 19, pp. 660-669.

[7] Kevric, Dasmin, S. Dukic, A. Subasi, An effective combining classifier approaches using tree algorithms for network intrusion detection. Neural Computing and Applications, 2017. vol. 28, pp. 1051-1058.

[8] S. Mehibs and S. Hashim, Proposed Network Intrusion Detection System Based on Fuzzy C Mean Algorithm in Cloud Computing Environment. Journal of Babylon University/Pure and Applied Sciences, 2017. vol. 26, pp. 27-35.

[9] T. Pham, E. Foo, S. Suriadi and H. Jeffrey, Improving performance of intrusion detection system using ensemble methods and feature selection. Australasian Computer Science Week Multi conference, ACM, 2018. pp. 1-6.

[10] Vojislav Kecman, Support Vector Machines, Neural Networks, and Fuzzy Logic Models. Learning and Soft Computing. MIT Press, Cambridge, MA, 2001. pp. 122-184.

[11] Awad M and Khanna R, Support Vector Machines for Classification. Efficient Learning Machines, Apress, Berkeley, California, 2015. pp. 39-66.

[12] D. Kennedy and R. Eberhart, A discrete binary version of the particle swarm algorithm. IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation, 1997. vol. 5, pp. 4104–4108.

[13] D. Kennedy and R. Eberhart, Particle swarm optimization. IEEE International Conference on Neural Networks, 1995. vol. 4, pp. 1942–1948.

[14] M. Tavallaee, E. Bagheri, W. Lu and A.A. Ghorbani, A detailed analysis of the KDD CUP 99 data set. IEEE International Conference on Computational Intelligence for Security and Defense Applications, 2009. pp. 53-58.

[15] A. Antony, R. Priyadharshini and D.L. Epiphany, Cuckoo Optimization based Intrusion Detection System for Cloud Computing. International Journal of Computer Network and Information Security, 2018. vol. 10, pp. 42-49.

[16] I.S. Thaseen and C.A. Kumar, Intrusion detection model using a fusion of chi-square feature selection and multiclass SVM. Journal of King Saud University - Computer and Information Sciences, 2017. vol. 29, pp. 462-472.

[17] M.E. Elhamahmy and I.A. Saroit, A new approach for evaluating intrusion detection system. Artificial Intelligent Systems and Machine Learning, 2010. vol. 2, pp. 290-298.

[18] J.R. Beulah and D.S. Punithavathani, Simple hybrid feature selection for enhancing network intrusion detection with NSL-KDD dataset. International Journal of Applied Engineering Research, 2015. vol. 10, pp. 40498–40505.

[19] L. Dhanabal and S.P. Shantharadah, A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. International Journal of Advanced Research in Computer and Communication Engineering, 2015. vol. 4, pp. 446–452.

[20] L. Dhanabal and S.P. Shantharadah, A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. International Journal of Advanced Research in Computer and Communication Engineering, 2015. vol. 4, pp. 446–452.

Authors' Profiles

Mahmoud M. Sakr received the B.Sc. degree from Menoufia University, Faculty of Computers and Information, Computer Science department in 2014. Currently doing his post-graduation master degree studies in the Faculty of Computers and Information, Menoufia University. His main research interest includes intrusion detection systems, revolutionary optimization techniques and Machine learning.

Dr. Medhat A. Tawfeeq received the B.Sc. and M.Sc. degrees from Menofia University, Faculty of Computers and Information in 2005 and 2010, respectively and received his PhD degree in Computer Science in 2015. His research interest includes cloud computing, smart card security, intelligent systems, distributed system, fault tolerance.

Dr. Ashraf B. El-Sisi received the B.Sc. and M.Sc. degrees in Electronic Engineering and Computer Science Engineering from Menoufia University, Faculty of Electronic Engineering in 1989 and 1995, respectively and received his PhD degree in Computer Engineering & Control from Zagazig University, Faculty of Engineering in 2001. His current research interest includes cloud computing, privacy-preserving data mining, and intelligent systems.

How to cite this paper: Mahmoud M. Sakr, Medhat A. Tawfeeq, Ashraf B. El-Sisi,"Network Intrusion Detection System based PSO-SVM for Cloud Computing", International Journal of Computer Network and Information Security(IJCNIS), Vol.11, No.3, pp.22-29, 2019.DOI: 10.5815/ijcnis.2019.03.04

