Antlion Optimization-Based Feature Selection Scheme for Cloud Intrusion Detection Using Naïve Bayes Algorithm

Haruna Atabo Christopher[1], Shafi'i Muhammad Abdulhamid[1], Sanjay Misra[2(✉)], Isaac Odun-Ayo[2], and Mayank Mohan Sharma[3]

[1] Federal University of Technology, Minna, Nigeria
shafii.abdulhamid@futminna.edu.ng
[2] Covenant University, Ota, Nigeria
{sanjay.misra,isaac.odun-ayo}@covenantuniversity.edu.ng
[3] Zillow Inc., San Francisco, USA

Abstract. The popularity of cloud computing is due to its countless benefits which include flexibility, scalability, and cost effectiveness. This refers to the availability of services and computing resources on demand to users with little management drive via internet technology. One of the major challenges faced by this technology is the issue of security which is making both service providers and users to worry about the safety of cloud resources. It is on this note that Cloud Intrusion Detection System (CIDS) is mostly deployed into cloud environment to identify and also prevent attacks in some instance. In this research work, a cloud intrusion detection system that identifies malicious activities inside cloud, utilizing Ant Lion Optimization (ALO) algorithm for feature selection and Bayesian Classifier was developed. Experimental result shows 96.22% accuracy, 0.379% FPR, 96.16% (Recall, Precision and F-Measure), and 92.36% Kappa Statistics.

Keywords: Ant Lion Optimization · Bayesian classifier · CIDS · Feature selection · Cloud computing

1 Introduction

Cloud computing refers to the availability of services and computing resources on demand by service providers to users with little management drive via internet technology. This technology has become more common among people due to its many benefits which include scalability, flexibility and cost effectiveness. Three types of cloud services are offered to cloud users [1, 2]. They are; Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). In SaaS, different applications are accessed by users via the internet. Examples include Dropbox, and ZenDesk. While in PaaS, users are allowed to build, run, deploy and manage applications. Examples include OpenShift and Windows Azure. Lastly, users are given access to the entire virtual machine in IaaS. Examples include Digital Ocean and Rocispace.

A. Abraham et al. (Eds.): ISDA 2020, AISC 1351, pp. 1383–1393, 2021.
https://doi.org/10.1007/978-3-030-71187-0_128

The major goal of cloud service providers is to use efficaciously and expeditiously services between the boundaries of agreement made by service provider and users [3]. Also, the price of installing with maintaining of cloud has decreased greatly.

Because the cloud is distributed in nature, it has become more prone to attacks both from within and outside of cloud environment and that is why users worry if data stored on cloud are safe or well protected [4]. Some of the malicious activities could either be from inside the cloud network or from outside of cloud network. Some of the attacks carried out by external attackers include DOS/DDOS attacks, phishing attacks etc.

Since the level of insecurity is growing every day, there is that need for a sophisticated security mechanism to ensure trusted and efficient security of tenants' resources in cloud. Hence the need for a Cloud Intrusion Detection System (CIDS) to take care of these malicious activities. It is a software or hardware component that monitors the system or network for policy violation or malicious activities [5].

We propose in this paper a Cloud Intrusion Detection System (CIDS) based on Ant Lion Optimization (ALO) feature selection and Naïve Bayes Classifier for classification.

Our major contributions in this paper are to:

- Classify cloud intrusion dataset using Naïve Bayes Classifier.
- Design an Ant Lion Optimization (ALO) algorithm-based feature selection system for detecting cloud intrusions in objective (i) above.
- Evaluate the feature selection technique's performance using standard metrics.

The remaining part of this work is organised as follows: Sect. 2 explain the related works while Sect. 3 discuss the proposed CIDS. Section 4 presents the result experiment and finally conclusion at Sect. 5.

2 Related Work

An architecture called VMGuard that monitors tenants' virtual machine (TVM) for the detection of malicious activities was presented by [6]. It performs the validation of processes at the VMM and also extracting traces of execution from processes that are running by emptying kernel debugging based VMI technique. The authors states that their method uses Bag on n-grams (BonG) together with frequency-inverse document for feature selection and Random Forest Classifier as the classification algorithm. Experimental result shows that the VMGuard is an efficient method for detecting hidden threats. However, this approach cannot detect new attacks because attacks that behave different from already learned behavior will not be detected. While, [7] emphasized on a type of security measures which allow cloud users to monitor and prevent their resources from attacks. These security measures must be capable of meeting the users' needs. They listed these security requirements to be: the size and services an organization offers, the type of application that runs and the number of virtual machines in the users virtual environment. Three components are put together to form the authors architecture. They are: Tenants Security Requirement Manager (TSRM), light weight IDS (LIDS), and remote advanced attack detection (RAAD). The result from experiment shows that the number of lost packets in FIDS is greater than that of the LIDS. However, this proposed

architecture have not been compared with other open source CIDS to test if it performs better than the existing system.

A cloud IDS using genetic algorithm was presented in [8]. This process increases the data transmission speed by optimizing the path through which data is conveyed. It also makes malicious activities nearly impossible. Four algorithms were used to optimize the network channel but result found GA to be more effective. Conversely, this method will only be more effective if the CIDS is network based. When NIDS and HIDS are hybridized, a complete security will be achieved. Meanwhile [9] introduces an approach that uses string matching algorithms to detect malicious activity. This model consists of the following components: the Enhanced Virtual Machine Monitor, IDS Engine Bock, Biography Engine, Signaling Analysis Bock, Remedy Report Block, Guest VM, System Call Interface Handler, and Improvised Hypervisor. After implementation, result obtained shows favorable results compared to traditional approaches.

In [10], an IDS that is multi-level and based on trust level of user is presented. They stressed that if the level of risk of a user is known; the required IDS will be selected and configured on the tenants Virtual Machine. The IDS is made up of three (3) different agents; dispatch agent, IDS manager agent and the rule-set manager. Xen Virtual Machine Monitor (VMM) was used as the environment for simulation, CentOS as the OS and snort 2.9.4.6 as IDS. Result from experiment has shown a great reduction in the time execution and in the rate of packet drop with a little decrease in accuracy. Nonetheless, the IDS is not setup dynamically in accordance to dynamic level of security. Likewise services with similar security problems cannot be classified efficiently.

There are several applications on feature selection [11, 12] and work on intrusion detection [13–15] are available in literature, which we are not considering for due to scope and limitation of work.

3 Proposed CIDS Model

The model proposed is set to address some of the setbacks notices from the review of related work. It is made up of four (4) major components as shown in Fig. 1:

- Data capture module: this module captures traffic for proper analysis. Tools such as wireshrak can be used for such purpose.
- Intrusion Detection Module: this consist of the ALO and the BC
- Storage contain the behavior database and the central log.
- An alarm is generated by alert system if intrusion occurs.

The model proposed is divided into two different parts: The feature selection part and the classification part. The NSL-KDD which is the adopted intrusion dataset was taken as input, and the nature inspired algorithm ALO now selected the best features for better classification. These selected features were moved to classification phase and the Naïve Bayes classifier was used as the classification algorithm which does the classification between normal and abnormal data. Five different evaluation matrixes were used to evaluate the proposed models performance. They are: Accuracy, False Positive Rate, Precision, Recall and F-Measure.

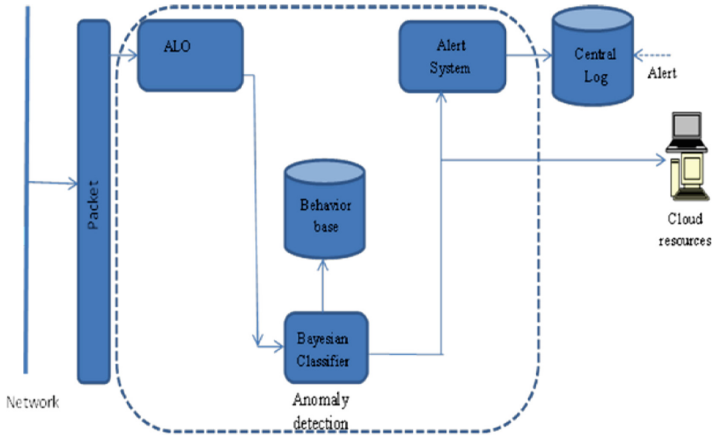

Fig. 1. Proposed model

3.1 Naïve Bayes Classifier

Naïve Bayes Classifier is an arithmetic classifier that predicts the occurrence of any given event belonging to two different classes (normal or abnormal) [16, 17]. The algorithm does it prediction by computing the chance of an event occurring for each class.

3.2 Ant Lion Optimization (ALO) Algorithm

The Ant Lion Optimization (ALO) is a nature inspired algorithm which mimics Antlions' way of hunting [18]. What it does is to wait patiently for pray in small round hole it dug moving backward into the sand. When an insect falls into the hole, the antlion grabs it, and then kills it for consumption.

Equation (1) shows the random walk of ants for each repetition

$$X(t) = [0, cumsum(2r(t_1) - 1), \ldots, cumsum(2r(t_n) - 1)] \tag{1}$$

Where total sum is cumsum, max iteration is n and t is the random walks' phase, and the stochastic function r(t) for which t = 1 if random number < ½ and 0 if otherwise.

For each stage of optimization, Ants position is updated using the random walk of ant. There is limit in the space of exploration; however, we will not be able to use Eq. (1) to assess ants' position. Equation (2) normalizes the random walk:

$$X_i^t = \left(X_i^t - a_i\right) * \left(d_i - c_i^t\right) / \left(d_i^t - a_i\right) + c_i \tag{2}$$

The snare of Antlion modeled with Eq. (3) and (4) is affected by the random walk of Ant in search of space.

$$c_i^t = Antlion_j^t + c^t \tag{3}$$

$$d_i^t = Antlion_j^t + c^t \tag{4}$$

c^t represents the least of all variables at t-th iteration, d^t determine the vector with the largest variables at t-th iteration, c_i^t is the smalest of all variables for i-th ant, d_i^t is the largest of all variables for i-th ant, and $Antlion_j^t$ shows the nominated j-th Antlions' position at t-th iteration.

Sliding prey towards |Antlion is modeled by Eq. (5) and (6) which reduces the radius of Antlions' random walk.

$$c^t = c^t / I \tag{5}$$

$$d^t = d^t / I \tag{6}$$

For Antlion to be able to hunt new targets, it needs to reposition itself to the position of the hunted prey. It is represented by Eq. (7):

$$Antlion_j^t = Ant_i^t, \quad \text{if } f\left(Ant_i^t\right) > f\left(Antlion_j^t\right) \tag{7}$$

t stands for the latest repetition, $Antlion_j^t$ indicates the position of selected j-th Antlion at t-th repetition, and Ant_i^t shows the position of i-th ant at t-th iteration.

$$Ant_i^t = R_A^t + R_E^t / 2 \tag{8}$$

Therefore, ALO algorithm is given as:

Algorithm 1: ALO Algorithm

```
Initialize the first population of ants and antlions randomly
Calculate the fitness of ants and antlions
Find the best antlions and assume it as the elite (determined optimum)
while the end criterion is not satisfied
    for every ant
        Select an antlion using Roulette wheel
        Update c and d using equations Eqs. (5) and (6)
        Create a random walk and normalize it using Eqs. (1) and (2)
        Update the position of ant using (8)
    end for
Calculate the fitness of all ants
Replace an antlion with its corresponding ant if it becomes fitter Eq. (7)
Update elite if an antlion becomes fitter than the elite
end while
Return elite
```

3.3 Dataset Description

The NSL-KDD dataset was used in conducting this experiment. It is modified from KDD cup 99 which is meant to overcome some of its drawbacks. These drawbacks include the dataset containing redundant records which makes the algorithm to be biased towards frequently occurring records and the high complexity of the dataset [19–21].

4 Evaluation of Proposed CIDS

4.1 Experimental Setup

At the feature selection phase, Matlat R2015a was used. The ALO_Toobox was downloaded, added to matlab library and run using the appropriate parameters. Waikato Environment for Knowledge Analysis (WEKA) an open source Java Machine Learning Application was used in the classification phase of this experiment. We used the complete NSL-KDD intrusion dataset as training data for the Naïve Bayes Classifier using 10-fold cross validation, 20-fold cross validation and 66% Split before feature selection (BFS) and after feature selection (AFS).

4.2 Result and Discussion

In order for the proposed CIDS to properly detect malicious activity in cloud, we used the complete training set of NSL-KDD dataset and 10-fold cross validation for testing purpose. We split the data into ten (10) disjoint subsets of almost equal sizes. From these ten (10) sets, one is used as test while others were used in building the classifier. The accuracy is gauged using the test set. In other to get a stabled result from the classifier, the algorithm was run ten (10) times with seeds (1, 2, 3, …, 10) and the average of the 10 prediction was calculated.

For comparison purpose, the algorithm was also run using 20-fold cross validation and 66% split. Detailed explanation of our result is presented in Table 1 and 2. This shows that there is a clear improvement in the performance of this model after feature selection with ALO.

Table 1. Summary of result for classification using 10-Folds, 20-Folds and 66% Split on Full NSL-KDD before feature selection

	Accuracy	FPR	Recall	Precision	F-measure	Kappa statistics	RMSE
10-folds	90.3827	0.1012	0.9033	0.9045	0.9033	0.8053	0.3045
20-folds	90.40661	0.1012	0.9037	0.9043	0.9037	0.8055	0.305
66% split	90.59316	0.0986	0.9057	0.9067	0.9057	0.8102	0.3028

Table 2. Summary of result for classification using 10-Folds, 20-Folds and 66% Split on Full NSL-KDD after feature selection

	Accuracy	FPR	Recall	Precision	F-measure	Kappa statistics	RMSE
10-folds	96.2238	0.0379	0.9616	0.9616	0.9616	0.9236	0.1927
20-folds	96.2222	0.0378	0.9619	0.9619	0.9619	0.9239	0.193
66% split	96.32029	0.0365	0.9625	0.9625	0.9625	0.9253	0.1905

4.3 Accuracy

Accuracy shows the level of correct prediction on intrusion datasets. Before feature selection, the accuracy gotten is 90.38% with the 10-fold cross validation, 90.40% with 20-fold cross validation and a little rise to 90.59% with 66% split. However, after feature selection, accuracy became 96.22% with 10-fold cross validation and 20-fold cross validation and an increase to 96.32% with 66% split. Figure 2 shows the accuracy achieved before and after feature selection with ALO.

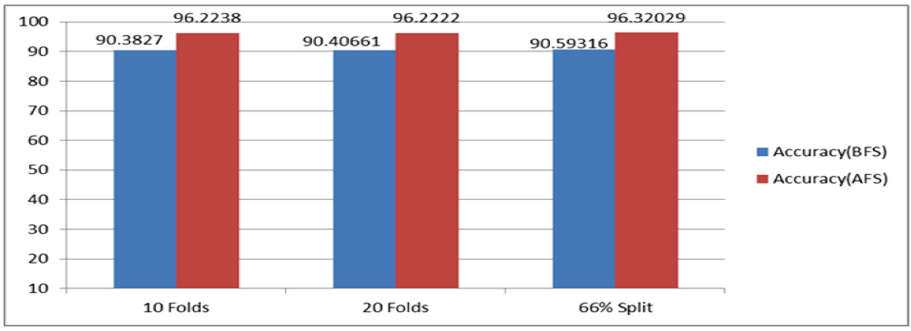

Fig. 2. Accuracy achieved before feature selection (BFS) and after feature selection (AFS) with ALO

4.4 False Positive Rate (FPR)

The FPR is that portion of dataset that are not malicious but are classified wrongly as malicious activity. A low FPR guarantees effectiveness of the CIDS. Figure 3 show a reduction in the false positive rate from 0.30 before feature selection to 0.19 after feature selection across all test option.

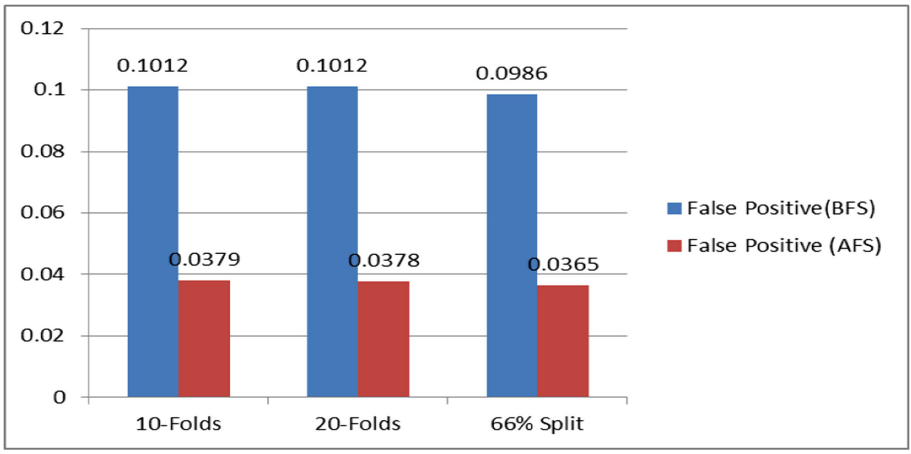

Fig. 3. False Positive BFS and AFS with ALO

4.5 Recall

This indicates the probability of correctly detection intrusion to the total number of intrusion in the network. The NSL-KDD dataset shows a high recall value of 0.9625 with test option of 66% split (Fig. 4).

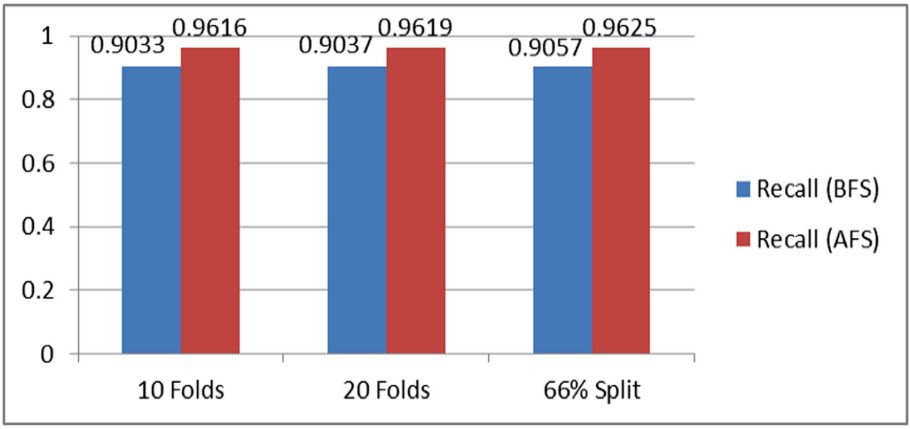

Fig. 4. Recall BFS and AFS

4.6 Precision

Precision represents the percentage of intrusion that has occurred and CIDS accurately detects them. In this experiment, NSL-KDD dataset shows a high Precision value of 0.9625 with test option of 66% split (Fig. 5).

4.7 Kappa Statistics

This gives the agreement between the true class and the classifications. Value 1 shows total agreement. NSL-KDD in this research shows a high kappa statistic value of 0.9253 with test option of 66% split. Figure 6 shows the results of kappa statistics the test options used.

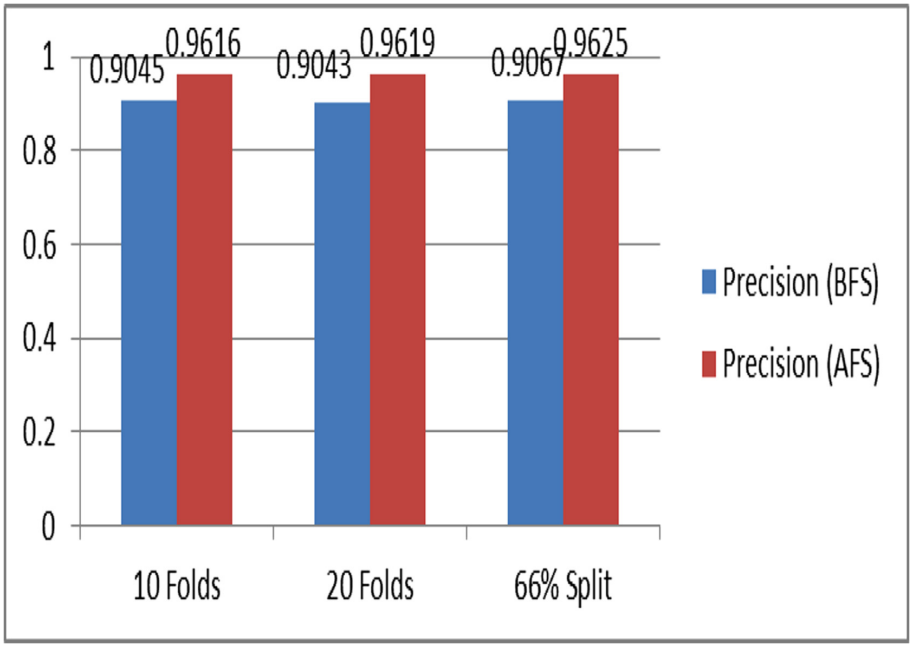

Fig. 5. Precision BFS and AFS

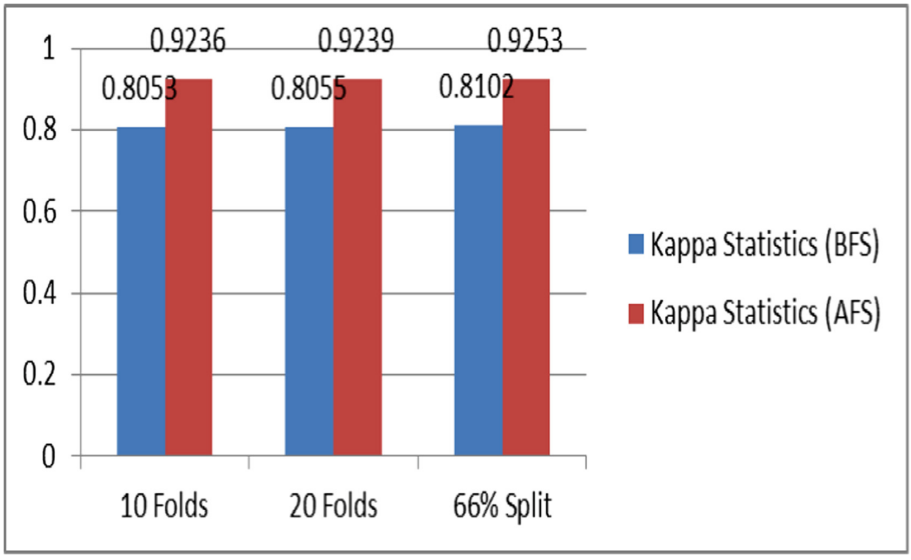

Fig. 6: Kappa Statistics BFS and AFS

5 Conclusion

Security has become a major issue in the area of cloud technology where users and operators of cloud technology are worried of the safety of cloud resources. Because of these security issues, CIDS is introduced into the cloud to minimize them. We have in this paper proposed CIDS which uses Ant Lion Optimization for feature selection and Bayesian classifier for classification between malicious and non-malicious activities. Results from experiment show that the proposed CIDS achieved high accuracy, low FPR and high detection rate.

Acknowledgement. The authors appreciate the Covenant University through its Centre for Research, Innovation and Discovery for Financial assistance and sponsorship.

References

1. Mehmood, Y., Ayesha, K.: Distributed intrusion detection system using mobile agents in cloud computing environment. In: 2015 Conference on Information Assurance and Cyber Security, pp. 1–8. IEEE (2015). ISBN 978-1-4673-7914-4
2. Abdulhamid, M., Shafie, M., Latiff, A.: A checkpointed league championship algorithm-based cloud scheduling scheme with secure fault tolerance responsiveness. Appl. Soft Comput. J. **61**, 670–680 (2017). https://doi.org/10.1016/j.asoc.2017.08.048
3. Hamid, S., Madni, H., Sha, M., Latiff, A., Coulibaly, Y.: Resource scheduling for infrastructure as a service (IaaS) in cloud computing: challenges and opportunities. J. Net. Comput. Appl. **68**, 173–200 (2016). https://doi.org/10.1016/j.jnca.2016.04.016
4. Mahajan, V., Peddoju, S.K.: Integration of network intrusion detection systems and honeypot networks for cloud security. In: Mahajan, V., Peddoju, S.K. (eds.) 2017 International Conference on Integration of Network Intrusion Detection Systems and Honeypot Networks for Cloud Security. 2017 International Conference on Computing, Communication and Automation, pp. 829–834 (2017). https://doi.org/10.1109/CCAA.2017.8229911
5. Nagar, U., He, X., Nanda, P., Tan, Z.: A framework for data security in cloud using collaborative intrusion detection scheme. In: ACM International Conference Proceedings Series, vol. 1338 (Part F) (2017). https://doi.org/10.1145/3136825.3136905
6. Mishra, P., Varadharajan, V., Member, S., Pilli, E.S., Tupakula, U.: VMGuard : A VMI-based security architecture for intrusion detection in cloud environment. IEEE Trans. Cloud Comput. **7161**, 1–14 (2018). https://doi.org/10.1109/TCC.2018.2829202
7. Hawedi, M., Talhi, C., Boucheneb, H.: Security as a service for public cloud tenants (SaaS). In: 9th International Conference on Ambient Systems, Networks and Technologies, ANT-2018 and The 8th International Conference on Sustainable Energy Information Technology, SEIT 2018, 8–11 May 2018, Porto, Portugal, vol. 130, pp. 1025–1030 (2018). https://doi.org/10.1016/j.procs.2018.04.143
8. Singh, T., Verma, S., Kulshrestha, V., Katiyar, S.: Intrusion detection system using genetic algorithm for cloud. In: Proceedings of the 2nd International Conference on Information and Communication Technology for Competitive Strategies, ICTCS 2016, pp. 1–6 (2016). https://doi.org/10.1145/2905055.2905175
9. Raj, R.S.: Securing cloud environment using a string based intrusion detection system (2017)
10. Zahra Salek, F.M.M.: Multi-level Intrusion detection system in cloud environment based on trust level. In: 6th International Conference on Computer and Knowledge Engineering, ICCKE 2016, 20–21 October 2016, Ferdowsi University Mashhad, pp. 94–99. IEEE (2016). ISBN 978-1-5090-3586-1

11. Oluranti, J., Omoregbe, N., Misra, S.: Effect of feature selection on performance of internet traffic classification on NIMS multi-class dataset. J. Phys. Conf. Ser. **1299**(1), 012035 (2019)
12. Abayomi-Alli, O., Misra, S., Matthews, V.O., Odusami, M., Abayomi-Alli, A., Ahuja, R., Maskeliunas, R.: An improved feature selection method for short text classification. J. Phys. Conf. Ser. **1235**(1), 012021 (2019)
13. Azeez, N.A., Bada, T.M., Misra, S., Adewumi, A., Van der Vyver, C., Ahuja, R.: Intrusion detection and prevention systems: an updated review. In: Sharma, N., Chakrabarti, A., Balas, V.E. (eds.) Data Management, Analytics and Innovation: Proceedings of ICDMAI 2019, vol. 1, pp. 685–696. Springer, Singapore (2020)
14. Azeez, N.A., Ayemobola, T.J., Misra, S., Maskeliūnas, R., Damaševičius, R.: Network intrusion detection with a hashing based Apriori algorithm using Hadoop MapReduce. Computer **8**(4), 86 (2019)
15. Shamshirband, S., Anuar, N.B., Kiah, M.L.M., Rohani, V.A., Petković, D., Misra, S., Khan, A.N.: Co-FAIS: cooperative fuzzy artificial immune system for detecting intrusion in wireless sensor networks. J. Net. Comput. Appl. **42**, 102–117 (2014)
16. Abdulhamid, S., Latiff, M., Chiroma, H., Osho, O., Abdul-Salaam, G., Abubakar, A., Herawan, T.: A review on mobile SMS spam filtering techniques. IEEE Access **5**, 15650–15666 (2017). https://doi.org/10.1109/ACCESS.2017.2666785
17. Modi, C.N., Patel, D.: A novel hybrid-network intrusion detection system (H-NIDS) in cloud computing. In: Proceedings of the 2013 IEEE Symposium on Computational Intelligence in Cyber Security, CICS 2013. 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013, pp. 23–30 (2013). https://doi.org/10.1109/CICYBS.2013.6597201
18. Mirjalili, S.: The ant lion optimizer. Adv. Eng. Softw. **83**, 80–98 (2015). https://doi.org/10.1016/j.advengsoft.2015.01.010
19. Osanaiye, O., Cai, H., Choo, K.R., Dehghantanha, A., Xu, Z., Dlodlo, M.: Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. EURASIP J. Wirel. Commun. Netw. **2016**, 130 (2016). https://doi.org/10.1186/s13638-016-0623-3
20. Chae, H., Jo, B., Choi, S., Park, T.: Feature selection for intrusion detection using NSL-KDD, pp. 184–187 (2013)
21. Deshmukh, D.H.: Improving classification using preprocessing and machine learning algorithms on NSL-KDD dataset (2015)