ABUAD INTERNATIONAL JOURNAL OF NATURAL AND APPLIED SCIENCES
ISSN: 2955-1021
AIJNAS Volume 5, Issue 1, 2025 pp 1-9
https://doi.org/10.53982/aijnas.2025.0501.01-j

https://journals.abuad.edu.ng/index.php/aijnas

Multi-Level Intrusion Detection System in Cloud Computing

[1*]Mogaji, S.A., [1]Folorunsho, O., [1]Akinsanya, S.E., Fagbuagun, O. A., [1]Hassan, J. B., [2]Olorunfemi, B. J
[1]Department of Computer Science, Federal University, Oye-Ekiti, Nigeria..
[2]Department of Mechanical Engineering, Federal University, Oye-Ekiti, Nigeria..
Corresponding Author: stephen.mogaji@fuoye.edu.ng

Abstract
An intrusion detection system (IDS) is essential for protecting private data, maintaining system integrity, and ensuring network security. This research focused on the development of a multi-level IDS using cloud computing technologies to enhance network security and employs a multi-layered strategy for intrusion detection, encompassing different levels for the scrutiny of network traffic, system behavior, and potential security risks. Cloud computing infrastructure forms the basis for deploying and expanding the IDS effectively. Anomaly-based intrusion detection systems have poor precision and recall, especially for unidentified attack types and this is seen as a research gap. Alternative Fuzzy C-Means Clustering (AFCM) was utilized in this work to group the training data into homogenous training subsets, train various Artificial Neural Networks (ANN) using those subsets, and then aggregate the results. The neural network was trained using the KDD Cup '99 dataset and was tested using real-time internet traffic. The SYN flood, ICMP flood, and Normal activity were used as test cases for the attack and normal activity, the results show that the SYN has a 98.9% true positive rate and 1.1% false negative rate of the 10000number of connections, ICMP has 99.9% true positive rate and 0.1% false negative rate of the 10000 number of connections and the normal activity has 88.09% true positive rate and 11.74% false negative rate of the 10000 number of connections. This is an improvement over other common anomaly-based intrusion detection systems.
Keywords: AFCM, ANN, Intrusion, Network Security, SYN Flood.

INTRODUCTION

Intrusion is an attack that improperly uses sensitive or private user data or that depletes CPU, speed, and storage. Traditional security techniques, such as firewalls, are inadequate.

An intrusion detection system (IDS) has been very suitable in identifying attacks and irregularities in networks. This security program gathers data from multiple network sources and examines it for symptoms of network intrusions (Daramola *et al.,* 2019).
By examining the network's data, an intrusion detection system (IDS) is a system that finds or identifies attacks. Network-based IDS and

Host-based IDS are the two primary categories of IDS based on deployment techniques
Intrusion detection involves the continuous monitoring of performance metrics to identify indications of potential attacks (Chiba *et al*., 2019).
Current approaches for detecting attacks in cloud environments are categorized into three primary classifications:
Signature-based, uses predefined rules or patterns to match network traffic with known attack signatures or indicators.

Anomaly-based: is an intrusion detection system that monitors system activity and categorizes it as either normal or abnormal to identify computer and network intrusions and misuse.

Hybrid detection methodologies. combines both anomaly-based and signature-based detection techniques to address the limitations of each approach (Alsoufi *et al.*, 2021).

In cloud computing environment, security could pose a serious threat to any network. There have been multiple successful attempts by hackers and intruders to take down network services and firm organizations. Many techniques, including the use of virtual private networks, firewalls, and encryption, are created to safeguard network infrastructure and communication over the Internet (Mogaji *et al.,*2022).

A cloud computing environment is a web-based computing environment where software, platforms, infrastructure, regulations, and other services are virtually provided by shared servers (Shafiq *et al*., 2022).

Technologies for virtualization having self-service capabilities are used by several cloud service providers (CSPs), such as Google, Amazon, and Microsoft. The initial requirement of cloud computing is virtualization (Bhardwaj and Krishna, 2021). Daily data increases are a result of a significant increase in IT technology.

A Cloud Federation (CF) serves as an effective solution to tackle these aforementioned challenges Elkhaldi and Abdullah (2022). The purpose of CF is to bring together Cloud Service Providers (CSPs) to fulfill the dynamic resource demands of users/applications, particularly for data-intensive workloads. Thus, by leveraging CFs, CSPs can leverage each other's resources to efficiently run Virtual Machines VMs, ultimately improving individual performance and enhancing user satisfaction. In cloud computing, a "Virtual Machine (VM)" is a software-based representation of a physical computer that, thanks to virtualization technology, enables users to run multiple operating systems and applications on a single physical server, effectively functioning as a "virtual computer" inside the infrastructure of a cloud provider. (Hema, & KanagaSubaRaja, 2023).

Several significant Information and Communications Technology (ICT) businesses are vying with one another to provide practical cloud computing services that can handle various business requirements (Jaber and Rehman 2020). From an economic standpoint, this historical change is advantageous because it allows them to better manage capital expenditures and spending on IT infrastructure. However, this has given rise to escalating worries about the security flaws and cyberattacks that these sophisticated systems might harbor (Devi *et al*., 2020)

Data mining and machine learning practices have been applied by several researchers. The biggest worries for the cloud are zero-day attacks (Rana and Batra 2021). Machine learning-based classifiers are typically used to distinguish between attack packets and legitimate packets.

Securing computer systems and networks is essential in today's networked and data-driven society. Traditional intrusion detection systems (IDS) are struggling to accurately detect and prevent many sorts of intrusions due to the rapid growth of cloud computing and the growing sophistication of cyber threats.

To increase the precision and effectiveness of intrusion detection, it is imperative to create a sophisticated intrusion detection system that makes use of cloud computing's capabilities and combines the advantages of Artificial Neural Networks (ANN) and Alternative Fuzzy C-Means (AFCM). The major goal of this research is to design and put into practice a multi-level intrusion detection system that can quickly recognize and react to various intrusion types in a cloud computing environment. To detect suspicious and malicious activity, this system will evaluate network traffic, system logs, and other relevant data sources using the algorithms of Artificial Neural Networks (ANN) and Alternative Fuzzy C-Means (AFCM). The system will efficiently manage massive volumes of data and carry out complex real-time analysis by utilizing the scalability and computational power provided by cloud computing.

The performance of the system is evaluated using Accuracy, false positive rate and computational efficiency metric.

Conventional intrusion detection systems (IDS) encounter difficulties in efficiently identifying and countering sophisticated network attacks due to their inherent limitations.

The objective of this study is to address these challenges by integrating cloud computing, artificial neural networks (ANN), and alternative fuzzy C-means (AFCM) into a multi-level IDS. This proposed approach aims to enhance the precision and efficiency of intrusion detection, thereby assisting organizations in bolstering the security of their networks and safeguarding sensitive data against malicious endeavors. The infrastructure provided by cloud computing allows for the IDS to be scaled up based on the size and intricacy of the network. Utilizing the capabilities of the cloud, the system becomes capable of managing substantial amounts of network traffic and adjusting to evolving demands. This scalability and adaptability are essential for effectively detecting intrusions in dynamic and ever-changing network environments.

Artificial Neural Networks (ANN) is a robust machine learning method capable of discerning patterns and behaviors from extensive network data (Abiodun *et al*., 2019). By training an ANN model using past intrusion data, the IDS can thoroughly comprehend typical and anomalous network behaviors. Consequently, the system becomes adept at detecting deviations and abnormalities, even when confronted with previously unseen attack patterns. Fuzzy clustering algorithms such as AFCM are instrumental in detecting network anomalies and categorizing similar network traffic. AFCM permits the identification of subtle deviations from normal behavior by assigning fuzzy membership degrees to data points. Integrating AFCM into the IDS design enhances the precision of intrusion detection and reduces false positives. By leveraging AFCM, the IDS becomes more adept at accurately identifying network anomalies and distinguishing them from regular network behavior (Abusitta *et al*., 2019).

Taking advantage of cloud computing resources for implementing the IDS brings several benefits. It allows for the utilization of computing power, storage capabilities, and network scalability as needed, thereby alleviating the strain on local infrastructure. By leveraging the cloud, IDS systems can process substantial volumes of network data in real-time, facilitating quicker and more effective intrusion detection. This cloud-based approach enables the IDS to operate efficiently and swiftly, leading to enhanced detection of intrusions. By harnessing the power of cloud computing and leveraging the algorithms of ANN and AFCM, the IDS becomes capable of real-time intrusion detection. This capability enables prompt response and mitigation measures to be deployed, thereby minimizing the potential harm caused by cyber-attacks. Real-time detection holds immense importance in contemporary network environments, where threats can arise and propagate swiftly.

REVIEW OF RELATED WORKS

This section provides a succinct summary of related research works that have attempted to enhance IDS performance by utilizing different techniques.

Wang *et al*., (2019) focused on employing an adaptive machine-learning strategy to mitigate alarm overload in distributed intrusion detection systems through the utilization of edge computing. The authors acknowledge the issue of excessive alarms in distributed intrusion detection systems and suggest an adaptive machine-learning technique to decrease false positives. They leverage edge computing to perform computations closer to the network edge, resulting in reduced latency and improved real-time decision-making. Experimental assessments are conducted to validate the effectiveness of the adaptive machine learning-based alarm reduction approach. The findings of the authors demonstrate enhanced alarm accuracy and a decrease in false positives, thereby increasing the efficiency and dependability of distributed intrusion detection systems.

Almiani *et al.* (2020) focused on developing a deep recurrent neural network (RNN) for intrusion detection in Internet of Things (IoT) systems. The study is published in the journal Simulation Modelling Practice and Theory. The authors recognize the importance of having reliable intrusion detection mechanisms in IoT environments and propose a deep RNN architecture to address this need. RNNs are selected for their ability to analyze sequential data and capture temporal relationships. The article details the design and implementation of the deep RNN model, which learns from IoT network traffic data to identify unusual patterns associated with intrusions. The deep RNN's capability to capture long-term dependencies improves the accuracy of intrusion detection. The performance of the deep RNN-based intrusion detection system is evaluated through experiments using real-world datasets. The results validate the efficiency of the proposed approach in detecting intrusions in IoT systems, with increased accuracy and detection rates.

Almiani *et al*. (2020) gave an overview of computational intelligence intrusion detection techniques specifically applied to mobile cloud computing environments. The research addressed the challenges posed by the dynamic nature of mobile cloud computing, such as limited resources, fluctuating network conditions, and the need for real-time detection, but didn't work on real-time internet traffic

Kumar *et al*. (2021) focused on creating a fog computing-based distributed ensemble intrusion detection system to protect Internet of Things (IoT) networks. The authors emphasize the importance of protecting IoT networks, propose a distributed ensemble design, and present experimental results that validate its efficacy.

In Ka *et al*. (2022), to attain an intrusion detection accuracy of 90.21%, the authors suggested a two-stage DL structure with GRU as the first stage and Dinoising Auto-Encoder (DAE) as the second stage.

In contrast, the authors of Fu *et al*. (2022) achieved an accuracy of 90.73% by using CNN-BiLSTM and Adaptive Synthetic (ADASYN) to handle class imbalance.

Ahmed *et al.* (2023) worked on an Anomaly-Based Intrusion Detection System using a One-Dimensional Convolutional Neural Network and suggested an anomaly-detection IDS model built on a one-dimensional Convolution Neural Network (CNN1D). This is a one-level intrusion detection system.

Alaketu *et al.* (2024) introduced predictive models for intrusion detection that leverage Big Data and Machine Learning (ML) methods, and they guaranteed the confidentiality, availability, and integrity of information systems. Before and after feature selection, the suggested method independently trained three ML classifiers using a large dataset (CIC-Bell-IDS2017). In order to identify the most pertinent collection of characteristics and normalize the data, big data analytics tools were also used for feature scaling and selection. Both performance evaluation and comparative analysis were conducted, and the findings indicated that the model's prediction accuracy had improved.

Kandakatla & Rajakumari (2024) suggest a unique method for improving cloud security through intrusion detection: The Hashing ECC-HMM (HECC-HMM). A strong framework for precisely identifying and categorizing security vulnerabilities in cloud environments is provided by the HECC-HMM, which combines the concepts of Hashing and Error-Correcting Codes (ECC) with the Hidden Markov Model (HMM). To test the efficacy of the HECC-HMM, extensive experiments were carried out utilizing two popular datasets: CICIDS and KDD. The findings show that the HECC-HMM can achieve detection accuracy of 98.5% on the CICIDS dataset and 96.7% on the KDD dataset, with corresponding false negative rates of 2.3% and 3.2% and false positive rates of 1.2% and 1.5%, respectively. Additionally, with a computational overhead of just 0.5 seconds per sample for CICIDS and 0.8 seconds per sample for KDD, the HECC-HMM demonstrates effective real-time intrusion detection capabilities. These numerical findings demonstrate how well the HECC-HMM approach works to improve cloud security by offering a dependable and effective way to reduce risks and protect cloud data and infrastructure from constantly changing cyber threats.

In Raji *et al*. (2024), the Generative Unique Adversarial Neural (GUNN) is presented by the authors. We collect the KDD-Cup 99 dataset from Kaggle and use the grey wolf optimization technique for pre-processing to eliminate irrelevant or noisy data and find useful features. Then, using a different feature selection framework-based support vector machine technique, the feature section of the data is chosen. Ultimately, the classification accuracy for the intrusion detection system (IDS) is 94.5%. The suggested approach uses high precision, recall, and F1 measurement to forecast IDS in cloud environments.

MATERIALS AND METHODS

System Architecture

This section outlines the methods, techniques, and approaches employed in the design and execution of the system. A more detailed explanation of the AFCM-ANN framework was provided in this section. To create clusters with similar properties, the training data were divided into several subsets using the AFCM clustering algorithm. Subsequently, Artificial Neural Networks (ANNs) were trained using these subsets. The membership grades for each data point in the created subgroups were retrieved and utilized to train a new artificial neural network (ANN) to deliver the final results.

The model received input from data and outputs projected data. There are two portions to the data. Training data is the first part, while testing data is the second. The model learns to do a certain task, a prediction, from the training data provided to it. To determine whether the model's projections are accurate after it has been trained, the projected values were contrasted with actual values from the testing data. Before feeding the best and most pertinent feature sets into the classifier during system training, the converted data is supplied to several feature selection methods (Abusitta *et al*., 2019).

The altered data was sent to the classifier model. A decision is made on the network data, to know if it is normal or an attack, as shown in Figure 1, Once there is an attack an alarm is raised and the data be resent for training.

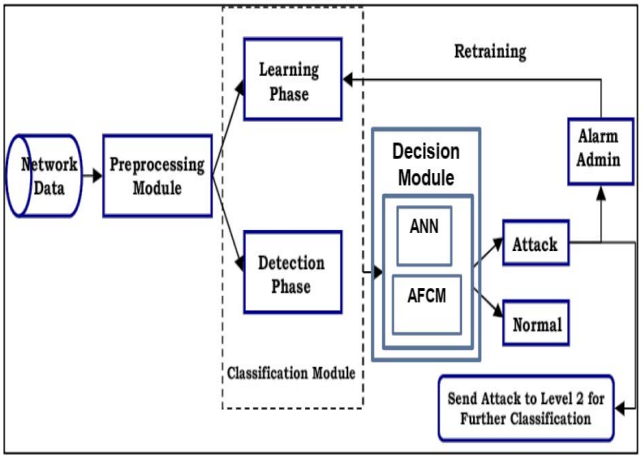

Figure 1: The System Architecture

Generation of Dataset and Preparation of Data

The KDD Cup '99 Dataset from Kaggle, which was created exclusively for intrusion detection was used to train the neural network. The trained dataset consists of 498000 labeled data points with 42 attributes for the fundamental TCP connection features, domain-specific features, and specific time-based statistics. The labels can be classified about into 4 categories and can either be normal or one of 22 attack labels. For the training, the duplicates would be eliminated from the original dataset. 11 traits out of the original 42 would be adopted. Table 1 highlighted the 11 qualities that were chosen. The ANNs would not be trained using root-to-local-style attacks for this study. Our findings led us to the conclusion that the majority of remote-to-local attacks in the dataset target proprietary protocols that were susceptible in 1999 but have since been patched or fixed.

Therefore, using prior attacks to train the ANNs would simply make our intrusion detection system less

effective at spotting current intrusions. We choose 9600 data points from the modified dataset for training. The remaining 2500 are classified as Normal, 3500 as DoS, 3550 as Probe, and 52 as user to root.

Table.1: The Description of The Dataset Attributes

S/n	Attribute	Description	Type	Domain
1	Duration	duration of the connection.	continuous	Type
2	Service	destination service (e.g.telnet,fip)	Discrete	real
3	src_bytes	bytes sent from source to destination	continuous	integer
4	dst_bytes	bytes sent from destination to source	continuous	real
5	Count	number of connections to the same host as the connection in the past two seconds	continuous	real
6	srv_count	number of connections to the same services as the connection in the past two seconds	continuous	real
7	dst_host_count	count of connection having the same designation host	continuous	real
8	dst_host_srv_count	count of connection having the same designation host and using the same service	continuous	real
9	dst_host_diff_srv_rate	% of connections to the current host having the same src port	continuous	real
10	dst_host_same_src_port_rate	% of connections to the current host having the same src port	continuous	real
11	dst_host_serror_rate	% of connections to the current host that have an s0 error	continuous	real

System Designs and Techniques

In this research, Artificial Neural Networks (ANN), and Alternative Fuzzy C-Means (AFCM) framework was employed. To create homogenous clusters, the training data is separated into several subsets using the Alternative Fuzzy C-Means clustering method. The ANNs are then trained using these subsets. The membership grades are then collected for all the data points in the created subgroups, and they are pooled to train a new ANN for the outcomes. The following actions are taken during the training phase:

Alternative Fuzzy C-Means Clustering

The main goal of this stage, which is stage one, is to create various homogeneous training datasets based on fuzzy membership values from the heterogeneous training dataset (Abusitta *et al*., 2019).

Therefore, another fuzzy c-means clustering approach is applied to the entire training data TR at this step.

$$1 - \exp(-\beta \left|\left|x_j - z_i\right|\right|^2) \qquad (1)$$

Here, the constant β is defined by,

$$\beta = \left(\frac{\sum_{j=1}^{n}\left|\left|x - \bar{x}\right|\right|^2 \text{----}}{n}\right)^{-1} \text{with } \bar{x} = \frac{\sum_{j=1}^{n} x_j}{n} \qquad (2)$$

Artificial Neural Network

The ANN module is used to comprehend the patterns in the dataset and assess the degree to which a particular network activity, such as an attack or a routine interaction, is associated with these patterns in stage 2 (Abiodun *et al*., 2019)

$$\text{inp(j)} = \sum_{i=1}^{n} x_j w_{ij} \qquad (3)$$

The activation function is then fed to the aforementioned input. Between the input and hidden layer, a bipolar sigmoid activation function is employed.

$$f(x) = \frac{2}{(1 + \exp(-x))} - 1 \qquad (4)$$

Fuzzy Aggregation

Another ANN is developed using the pooled information from the earlier ANNs after the ANNs in stage 1 have been trained using their datasets. The number of inputs for this ANN is equal to the number of outputs for the earlier ANNs. To create a new input for Stage 3, the outputs of the Stage 2 ANNs are multiplied by the relevant membership value.

$$Y_j^{TR} = \left[y_{j1}^{TR}, y_{j2}^{TR}, y_{jk}^{TR}\right], \; j = 1,2, \; n \qquad (5)$$

Here, n is the number of the training set of the ANNs in stage 2, and y_{jk}^{TR} is the output of the ANNs for the input point j.

Evaluation Metrics

To evaluate the techniques, the metrics of choice are accuracy, true positive rate, false negative rate, false positive rate, and ROC curve.

Given that the positive class refers to attack activity while the negative class refers to normal activities;

i. True positive (TP) represents the number of samples correctly classified as an attack;
ii. True negative (TN) signifies the number of samples correctly classified as normal;
iii. False positive (FP) represents the number of samples incorrectly classified as attack;
iv. False negative (FN) means the number of samples incorrectly classified as normal

The metrics can be defined as follows:

Accuracy: This gives the ratio of correctly classified samples to the total number of samples.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad (6)$$

True positive rate (TPR): This gives the ratio of the samples that were correctly classified as attack out of the total number that are actually attack.

$$TPR = \frac{TP}{TP+FN} \quad (7)$$

False negative rate (FNR): This gives the ratio of samples that were incorrectly classified as normal out of all samples that are actually attacked. A high FNR indicates that the model failed to detect a high number of attack activities.

$$FNR = \frac{FN}{TP+FN} \quad (8)$$

False positive rate (FPR): This gives the ratio of samples that were incorrectly classified as attack out of all samples that are actually normal. A high FPR indicates that the model raised a lot of false alarms on normal activities.

$$FPR = \frac{FP}{TN+FP} \quad (9)$$

Receiver Operating Characteristic (ROC) Curve:
This is the true positive rate versus false positive rate curve at different classification levels. The performance of the model can be assessed using the area under the curve (AUC).
A model with a high ROC AUC score means that it can achieve a high true positive rate while maintaining a low false positive rate.

SYSTEM IMPLEMENTATION AND TOOLS

The techniques proposed in the previous section were implemented using the Python programming language. Jupyter notebook environment was used as development environment. Training of the ANN model was performed with 15GB VRAM Tesla T4 GPU acceleration. The following libraries were used in the implementation:

i. **NumPy**: A library with support for multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
ii. **Pandas**: An open-source library that provides data structures and functions needed to work on structured data seamlessly. It was used for data analysis and data manipulation.
iii. **Scikit-learn:** A machine learning library that features various algorithms and functions for classification, regression, clustering, dimensionality reduction, and pre-processing. Its implementation of Random Forest and Logistic Regression, along with its pre-processing and scoring functions, was used in this study.
iv. **Matplotlib:** A plotting library for Python and NumPy. It provides functions for making visualizations such as line plots, bar graphs, scatter plots, histograms.
v. **Keras**: A Python-based high-level neural networks API that utilizes TensorFlow. It works with both CPU and GPU devices, supports both convolutional and recurrent networks, and makes prototyping simple and quick. The CNN and LSTM models were put into practice using it.
vi. **PyTorch**: An open-source machine learning library for computer vision and natural language processing that is based on the Torch library. This study made advantage of its transformer model implementation.
vii. **XGBoost**: An open-source software library that offers a productive gradient boosting solution for machine learning. It was employed in the GDBT model's implementation.
viii. **Huggingface Transformer**: A library for various transformer models for variety of tasks. It allows access to pretrained models hosted on the Huggingface platform.

RESULTS AND DISCUSSION

Results Analysis Description

The distribution of the data points into the 4 clusters for the AFCM-ANN framework's clustering phase as depicted in Figure 2. The clustering algorithm maintains homogeneity within the clusters and variability among the various clusters, as shown in Figure 2. Since each packet has 11 features, different types of packets are grouped together since their feature values are comparable. It can be observed that Cluster 0 is amply clustered with about 5500 packets, which consists of Probe packets and DoS, as seen in Figure 3 and Figure 5, respectively.

As seen in Figure 4, Cluster 1 has over 3500 packets, the majority of which are Normal packets. The cluster also includes about 900 DoS packets and about 45 U2R packets as seen in Figure 6. Cluster 2 contains around 100 packets, all of which are Probe packets as seen in Figure 3. As seen in Figures 5 and 6, Cluster 3 also has about 100 packets made up mostly of Normal packets with traces of U2R.

Four artificial neural networks were trained using clustered packets. Each ANN's accuracy and loss graphs are displayed as a function of epoch. ANN_0 accuracy

for 500 epochs is 54.16%. Due to closely similar feature values contained in the training data for some packets, one of the causes of this low accuracy may be the inclusion of all four types of packets in Cluster 0.

.

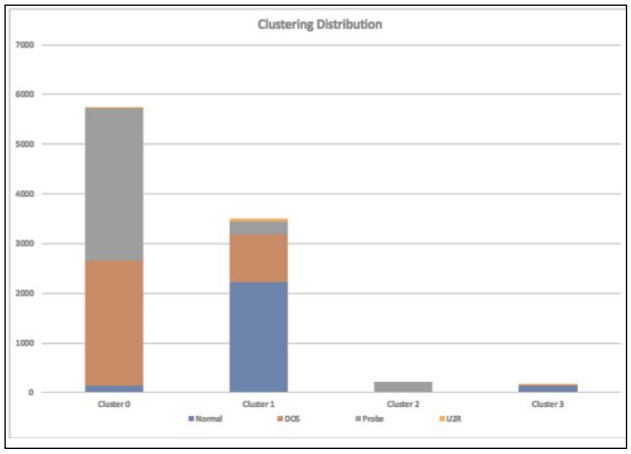

Figure 2: The accuracy and loss percentages for ANN_1 are 96.56% and 6.58%, respectively.

The accuracy and loss percentages for ANN_1 are 96.56% and 6.58%, respectively.

The prediction for these data points turns out to be more accurate than Cluster 0 because Cluster 1 predominantly consists of Normal packets. The accuracy for ANN_2 is 97.26%, and the loss percentage is 5.37%. These values are the result of the homogeneity of Cluster 2, which is made up entirely of Probe packets. The reported accuracy for ANN_3 across 500 epochs is 97.55%, and its loss percentage is the lowest of all the ANNs, at 4.98%. As soon as we have all the ANNs that have been trained across the four clusters, we create a fresh ANN for the Fuzzy Aggregation stage of the AFCM-ANN framework. As shown in Figure 5, this model's accuracy is 76.77%, and its loss percentage is 29.04% These values were acquired after this ANN was trained using all of the training data (TR), mixing Stage 2 ANN output with Stage I membership values.

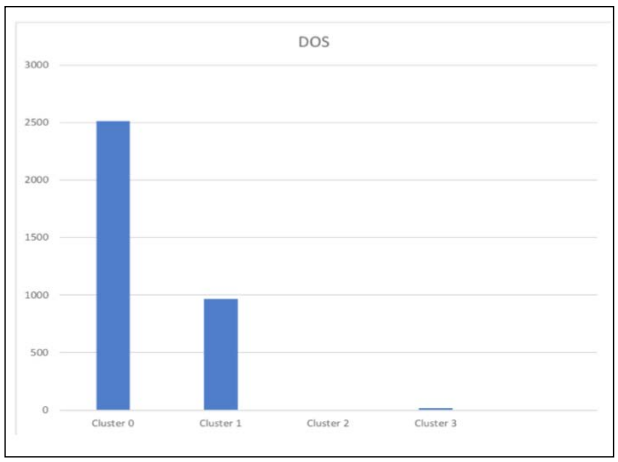

Figure 3: Cluster 0 (DoS Distribution)

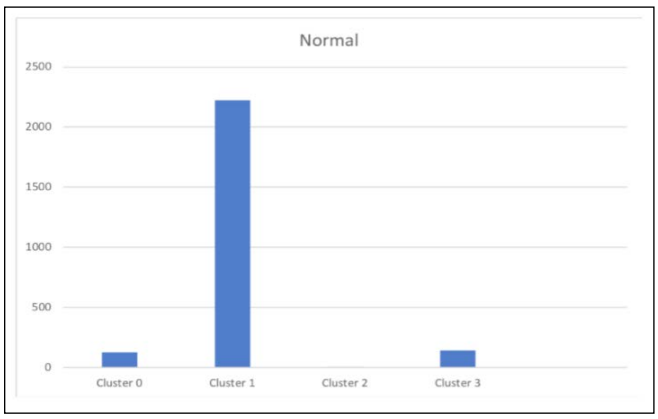

Figure 4: Cluster 1 (Normal Distribution)

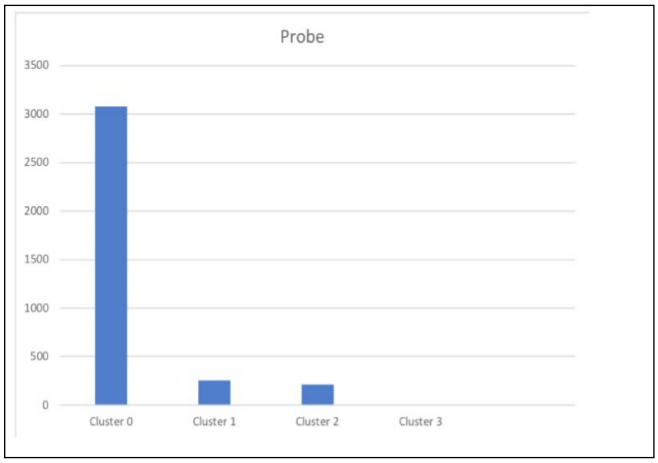

Figure 5: Cluster 3 (Probe Distribution)

DISCUSSION

The SYN flood, ICMP flood, and Normal activity were used as test cases for the attack and normal activity. The results show that the SYN has 98.9% true positive rate and 1.1% false negative rate of the 10000 number of connections.

ICMP has 99.9% true positive rate and 0.1% false negative rate of the 10000 number of connections and the normal activity has 88.09% true positive rate and 11.74%false negative rate of the 10000 number of connections.

SYSTEM PERFORMANCE EVALUATION

Three test cases were used to evaluate the system using standard internet workload, such as surfing YouTube. com, Wikipedia.org, etc., and two denial-of-service attacks. Additionally, one of the Video management systems (VMs) used wget to retrieve a sizable file from the Internet. Table 2 contains the outcomes of the aforementioned tests. The KDD Testing dataset was used to test the engine as well. Table 3 presents the findings. We can observe that the probe results are poor and that there is no U2R.

The engine cannot specifically identify the attack, but it recognizes the majority of probe and U2R connections as attacks and raises an alarm, which has the effect of banning that specific packet for security reasons.

- VMs is Video management system.
- A probe attack is created when the sender adds or modifies data bytes included in the original data that was sent.
- U2R is User to Root (Attack) Attacker has local access to the. victim machine and tries to gain super user privileges.

Table 2: The Normal and Attack Activity Results

Test Case	Total no of Connections	True Positive	False Negative	Standard Deviation
SYN Flood	10000	9890	110	0.2242
ICMP Flood	10000	9990	10	0.0094
Normal Activity	10000	8809	1174	15.4

Table 3: KDD Test Dataset Results

Test Case	Total No. of Connections	True Positive	False Negative
Normal	47893	47672	221
DoS	19385	16136	3249
Probe	2430	666	1764
U2R	202	0	202

CONCLUSION

In the present era, intrusion detection systems are fundamental to maintaining the security of cloud systems. The firewall serves as the first line of protection, but the presence of an IDS can aid in thwarting attacks if a security breach manages to get past the firewall or poses an internal threat. To accomplish this goal for various types of attacks, the suggested technique in this paper uses Artificial Neural Networks and Alternative Fuzzy c-means Clustering approach. The tests conducted on actual internet traffic have produced encouraging results for identifying assaults. The devastating effects that any single attack may have had can be avoided if the engine can distinguish clearly between an attacking and a non-attacking packet.

REFERENCES

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., ... & Kiru, M. U. (2019). Comprereview of artificial neural network applications to pattern recognition. IEEE Access, 7, 158820-158846

Abusitta, A., Bellaiche, M., Dagenais, M., & Halabi, T. (2019). A deep learning approach for proactive multi-cloud cooperative intrusion detection system. Future Generation Computer Systems, 98, 308-318.

Ahmed Tamer Assya , Yahia Mostafaa , Ahmed Abd El-khaleqa , Maggie Mashaly (2023). Anomaly-Based Intrusion Detection System using One-Dimensional Convolutional Neural Network. The 14th International Conference on Ambient Systems, Networks and Technologies (ANT), Leuven, Belgium. Procedia Computer Science 220 78–85.

Alaketu, A.A., Abiola, O. B., Obamiyi, S. E., Oguntimilehin, A., Badeji-Ajisafe, B., Olutosin Babalola, G. O., Olatunji, K. A., Akinduyite, C. O., and Okebule, T (2024). Comparative Analysis of Intrusion Detection Models using Big Data Analyticsand Machine Learning Techniques. The International Arab Journal of Information Technology, 21(2):326–337 (2024)

Almiani, M., AbuGhazleh, A., Al-Rahayfeh, A., Atiewi, S., & Razaque, A. (2020). Deep recurrent neural network for IoT intrusion detection system. Simulation Modelling Practice and Theory, 101, 102031.

Almomani, A., Alauthman, M., Albalas, F., Dorgham, O., & Obeidat, A. (2020). An online intrusion detection system to cloud computing based on NeuCube algorithms. In Cognitive Analytics: Concepts, Methodologies, Tools, and Applications, *IGI global*. pp. 1042-1059).

Alsoufi, M. A., Razak, S., Siraj, M. M., Nafea, I., Ghaleb, F. A., Saeed, F., & Nasser, M. (2021). Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review. *Applied sciences*, 11(18), 8383.

Bhardwaj, A., & Krishna, C. R. (2021). Virtualization in cloud computing: Moving from hypervisor to containerization—a survey. Arabian *Journal for Science and Engineering*, 46(9), 8585-8601.

Chiba, Z., Abghour, N., Moussaid, K., El Omri, A., & Rida, M. (2019). New anomaly network intrusion detection system in cloud environment based on optimized back propagation neural network using improved genetic algorithm. International *Journal of Communication Networks and Information Security*, 11(1), 61-84.

Daramola, C.Y., Ayogu, A.A., Folorunsho, O. and Akindolie, A.M. (2019) A Bayesian Based Classification Model for Network Intrusion Detection System. FUOYE *Journal of Pure and Applied Sciences.* 4(1):110-118.

Devi, B. T., Shitharth, S., & Jabbar, M. A. (2020, March). An Appraisal over Intrusion Detection systems in cloud computing security attacks. In 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 722-727). IEEE.

Elkhaldi, A. H., & Abdullah, G. A. (2022). The Effect of Cloud Computing's Advantages and Components on Time Savings and Data Privacy for the Quality of Electronic Banking Services. International *Journal of Professional Business Review*, 7(3), e0601-e0601.

Fu, Y. Du, Y, Cao, Z Li,Q and Xiang, W. 2022 "A deep learning model for network intrusion detection with imbalanced data," Electronics, vol. 11, no. 6, p. 898, 2022.

Hema, H., and KanagaSubaRaja, S. (2023). A Quantitative Approach to Minimize Energy Consumption in Cloud Data Centres using VM Consolidation Algorithm. *KSII Transactions on Internet and Information Systems*, 17, 2, (2023), 312-334. DOI: 10.3837/tiis.2023.02.002

Jaber, A. N., & Rehman, S. U. (2020). FCM–SVM based intrusion detection system for cloud computing environment. Cluster Computing, 23, 3221-3231.

Kandakatla R., and Rajakumari, k. (2024)»Optimal Feature Selection with Cryptographic Process for IDs based Cloud Environment,» *2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)*, Hassan, India, 2024, pp. 1-6, doi: 10.1109/IACIS61494.2024.10721989

Kao, M.T., Sung, D.Y., Kao, S.J and F.-M. Chang, F M, (2022) A novel two-stage deep learning structure for network flow anomaly detection, Electronics, vol. 11, p. 1531, 05 2022.

Krishnan R. and Raajan N., (2016) An Intellectual Intrusion Detection System Model for Attacks Classification Using RNN," International Journal of Pharmacy and Technology, vol. 8, no. 4, pp. 23157-23164, 2016

Kumar V., Das K.D., and Sinha D. (2021) "UIDS: A unified intrusion detection system for IoT environment," Evolutionary Intelligence, vol. 14, no. 1, pp. 47–59, 2021

Mogaji S.A, Ayeni, O.A, and Olutayo, V.A (2022) Analysis of Digital Forensics in the Implementation of Intrusion Detection using Snort *FUOYE Journal of Pure and Applied Sciences. FJPAS* Vol 7(1) ISSN: 2616-1419pg 100-108.

Raju, K. S., Rashmitha, P., Nagendra, K. V., Dharmireddi, S., Rekha M., and Sanaboina, S. P. (2024) Intrusion Detection System Using Generative Unique Adversarial Neural Network in cloud Environment,*2024 Ninth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)*, Chennai, India, 2024, pp. 1-5, doi: 10.1109/ICONSTEM60960.2024.10568683.

Rana, P., & Batra, I. (2021, April). Detection of attacks in cloud computing environment–a comprehensive review. In 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM) (pp. 496-499). IEEE.

Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2022). Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University-Computer and Information Sciences*, 34(7), 3910-3933.

Shamshirband, S., Fathi, M., Chronopoulos, A. T., Montieri, A., Palumbo, F., & Pescapè, A. (2020). Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues. *Journal of Information Security and Applications,* 55, 102582.

Wang, W., Du, X., Shan, D., Qin, R., & Wang, N. (2020). Cloud intrusion detection method based on stacked contractive auto-encoder and support vector machine. IEEE transactions on cloud computing, 10(3), 1634-1646.

Wang, Y., Meng, W., Li, W., Liu, Z., Liu, Y., & Xue, H. (2019). Adaptive machine learning-based alarm reduction via edge computing for distributed intrusion detection systems. Concurrency and Computation: Practice and Experience, 31(19), e5101.

Zeeshan A., Khan A., Shiang C., Johari A., and Ahmad F. (2020) "Network Intrusion Detection System: A Systematic Study of Machine Learning and Deep Learning Approaches," Journal of Transactions on Emerging Telecommunications Technologies, vol. 32, no. 1, pp. 1-29, 2020. DOI:10.1002/ett.4150.