International Journal of Computer Applications Technology and Research
Volume 11–Issue 08, 341-350, 2022, ISSN:-2319–8656
DOI:10.7753/IJCATR1108.1008

Model for Enhancing Performance of Network Intrusion Detection based on Hybrid Feature Selection and Unsupervised Learning Techniques

Joseph Mbugua Chahira
Department of Information Science
Garissa University, Kenya

Jane Kinanu Kiruki
Department of Computer Science
Chuka University, Kenya

Abstract

As security threats change and advance in a drastic way, relevant of the organizations implement multiple Network Intrusion Detection Systems (NIDSs) to optimize detection and to provide comprehensive view of intrusion activities. But NIDSs trigger a massive amount of alerts even for a day and overwhelmed security experts as they require high levels of human involvement in creating the system and/or maintaining it. The main goal in this work is to enhances the structural based alert correlation model to improve the quality of alerts and detection capability of NIDS by grouping alerts with common attributes based on unsupervised learning techniques. This work compares four unsupervised learning algorithms namely Self-organizing maps (SOM), K-means, Expectation and Maximization (EM) and Fuzzy C-means (FCM) to select the best cluster algorithm based on Clustering Accuracy Rate (CAR), Clustering Error (CE) and processing time. The result inferred that the proposed model based on hybrid feature selection, PCA and EM is effective in terms of Clustering Accuracy Rate (CAR) and processing time for The NSL-KDD Dataset

Key words: Network Intrusion Detection, unsupervised learning, Clustering, alert correlation, Structural-based AC.

Introduction

Intrusion detection is a system for detecting intrusions and hence works as the major defensive mechanism in a network environment[1][2]–[4]. Its main goal is to automatically monitor network traffic and classify them as normal or suspicious activities and inform the Security Analyst or response system to take appropriate action before the intrusion compromises the network.

Network monitoring has been used extensively for the purposes of security, forensics and anomaly detection. However, recent advances have created many new obstacles for NIDSs. Firstly, they generate huge volume of low quality evidence and in different format produced by distributed IDS systems (Application, Network and Host based). Unfortunately, most of the alerts generated are either false positive, i.e. benign traffic that has been classified as intrusions, or irrelevant, i.e. attacks that are not successful . This results in slow training and testing correlation processes, higher resource consumption, lower accuracy and higher performance costs.

The unsupervised machine learning algorithms are applied when there is no class to be predicted but when the instances are to be subdivided into natural groups of instances determined by the features available to represent the items into clusters [3], [5]. The algorithms can be trained on unlabeled data or can be applied to the test or evaluation data without training. The trained clustering algorithms build internal representation of unlabeled

training data during training which apply to the test data set. The untrained clustering algorithms determines natural differences between subsets of data without prior insight into the data.

In order to improve the quality of alerts for analysis, some research in alert clustering for finding structural correlation have been done. In Structural-based AC (SAC), alerts are correlated based on similarity of attributes. Similarity index or function is used to determine the degree of relationships. Although it can discover known group of alerts or attack steps, research by [6][7]–[9] claimed that it cannot discover the causal relationships among alerts. The major problem in previous techniques is they relied heavily on Security Experts (SE) in developing and maintaining their correlation system. They are based on pre-defined rules or expert knowledge to manage and analyze the intrusion alerts and as a result, rules or knowledge for such systems need to be updated periodically as patterns of attacks change drastically [8], [10].

The aim of this work is to enhance the Structural-based AC model using machine learning technique to improve the quality of alerts and identify attack strategy. An intelligent hybrid clustering model is developed based on normalization, discretization and Improved Unit Range (IUR) technique to preprocess the dataset, EMFFS, Principal Component Analysis (PCA), SAC and proposed Post-Clustering algorithms is implemented to reduce the alerts dimensionality and optimize the performance and unsupervised learning algorithm to aggregate similar alerts and to reduce the number of alerts. In the proposed model the performance of various unsupervised learning techniques like Self-organizing maps (SOM), Expectation Maximization, K-means, hybrid clustering and Fuzzy c-means (FCM) is compared based on four measurements techniques applied are: (1) Clustering Error (CE) is the number of alerts that are wrongly clustered. (2) Error Rate (ER) is the percentage of wrongly clustered alerts, ER = (CE ÷ Total number of alerts observed) x 100, (3) Accuracy Rate (AR) is the percentage of alerts that are accurately clustered as they should be, AR = 100 – ER, and (4) Time is the algorithm processing time in seconds.

Related work

Collection mechanism and reduction of IDS alert framework (CMRAF) [11] was proposed to remove the duplicates IDS alerts and reduce the number of false alerts. They use information gain ratio algorithms to extract the similarities between set of alerts and provide the highest weight to the most effective features based on the class of alerts belonging to the algorithm.

Alert correlation using a novel clustering approach, [12], applied an incremental clustering approach to reduce the amount of alerts generated by IDS. Three attributes, destination IO, signature-id, and timestamp had been extracted and hashed by using MD5. The hash value from the next input tuple is checked against hash value of the existing clusters. The hashing technique is used to speed up the comparison in checking the similarities of alert attributes.

An improved framework for intrusion alert correlation by Elshoush *at el*, (2012), divided alert correlation into ten main components and contained them in the Data Normalization Unit, Filter-based Correlation Unit and Data Reduction Unit. Similar alerts are fused based on seven extracted features, namely Event ID, timesec, SrcIPAddress, DestPort, DestIPAddress, OrigEventName, and SrcPort in order to remove duplicate alerts created by the independent detection of the same attack by different sensors.

A probabilistic-based approach proposed [13], correlate and aggregate security alerts by measuring and evaluating the similarities of alert attributes. They use a similarity metric to fuse alerts into meta-alerts to provide a higher-level view of the security state of the system. Alert aggregation and scenario construction are conducted

by enhancing or relaxing the similarity requirements in some attribute fields. But similarity correlation is the only way for them to aggregate the alerts. They have to compare all the alert pairs and have to determine lot of thresholds with expert knowledge which lead to their huge volume of computing workload.

Methodology

In this study, the quantitative approach is preferred as the main method due to certain characteristics, such as performance measures, dataset evaluations and the usability of the results. This research has employed a deductive reasoning because it seems to be more appropriate to test the proposed solutions. It addresses the issues of improving the quality of alerts that are generated by multiple NIDSs and recognizing the attack strategy from the unrelated alerts. The identified problems under these issues and the coverage of each objective in this research are solved though these steps

Step (1) Read the pre-processed alerts as inputs.
Alerts that have been processed by Multi-Filter Feature Selection (EMFFS) Method are read from the database as inputs to clustering phase.

Step (2) Reduce the alerts high dimensionality.
All alerts with their attributes are dimensionally reduced using statistical PCA

Step (3) Adopt unsupervised learning algorithm which gives the highest accuracy. Expectation Maximization (EM), (K-means, FCM and SOM. unsupervised learning algorithm are tested and compared.

Step (4) Measure and validate the clustering and post-clustering performances. The performances of the proposed clustering system can be measured using predefined measurements.

Step (5) Save the analysis and experimental results. The analysis and experimental results are recorded and saved in the database. It includes the details on all of the identified clusters attack steps as well as the statistical analysis.

International Journal of Computer Applications Technology and Research
Volume 11–Issue 08, 341-350, 2022, ISSN:-2319–8656
DOI:10.7753/IJCATR1108.1008

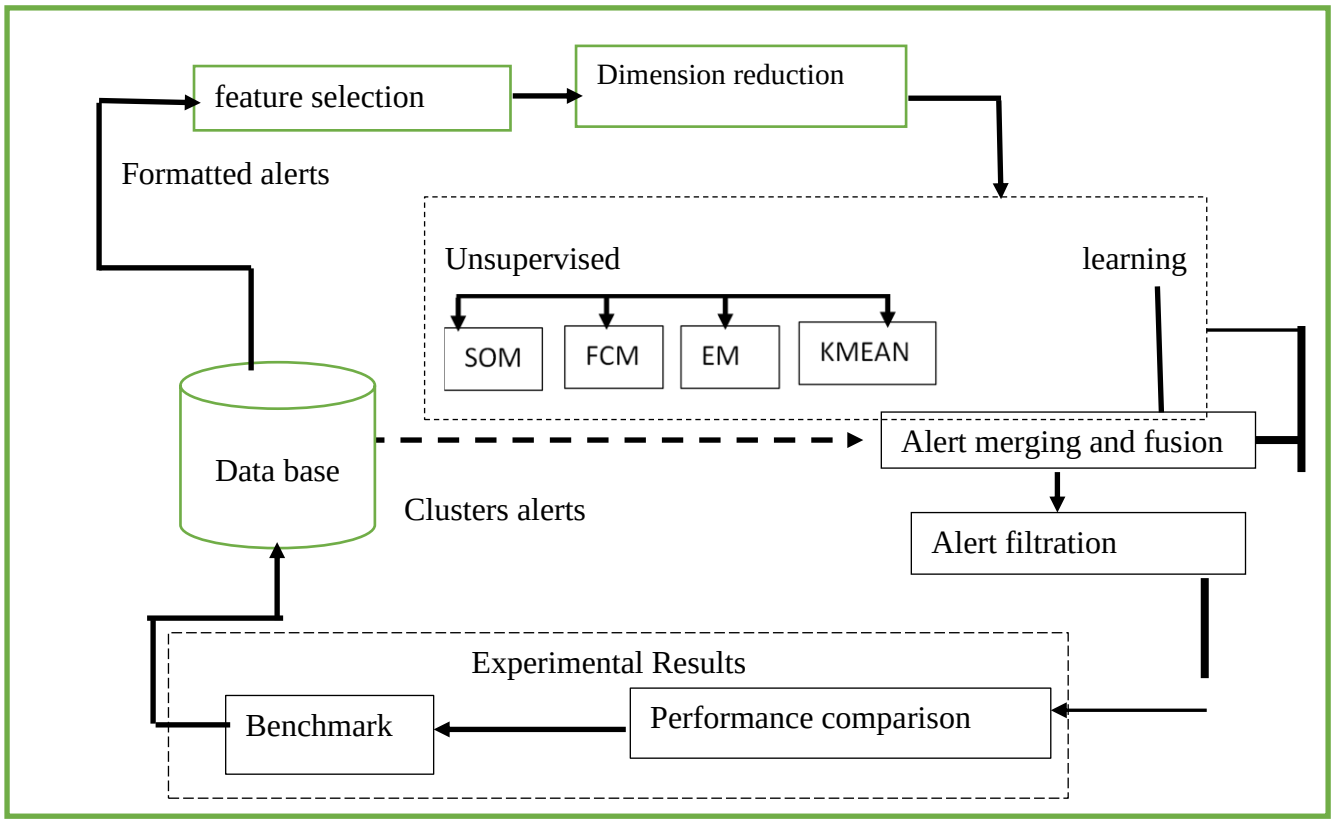

Figure 1: The flowchart of enhanced structural-based alert correlation model

Steps

Ensemble-based Multi-Filter Feature Selection (EMFFS) Method

The main aim of feature selection is to eliminates irrelevant and repetitive features from the dataset to make robust, efficient, accurate and lightweight intrusion detection system To achieve this objective, a model for network intrusion detection system based on Multi-Filter Feature Selection (**EMFFS**) Method is implemented developed by [14] to find the best set of features that are used in this work. The feature selection techniques integrated are Correlation Feature Selection (CFS) based evaluator with Best-first searching method, Information Gain (IG) based Attributes Evaluator with ranker searching method, and Chi Squared and Ranker searching method.

Unsupervised learning techniques

Self Organising Maps

The Self-Organizing Map [15], [16] is a neural network model for analyzing and visualizing high dimensional data and it belongs to the category of competitive learning network. The SOM defines a mapping from high dimensional input data space onto a regular two-dimensional array designed architecture as input vector with six input values and output is realized to two dimension spaces.

The SOM is a neural network trained with a competitive learning rule in an unsupervised manner. A competitive learning rule means that the neurons compete to respond to a stimulus, such as a connection vector (recall that a connection vector describes properties of a network connection, such as the destination port and number of packets sent). The neuron that is most excited by the stimulus, i.e. whose weight vector is most similar to the connection

vector, wins the competition. The winning neuron earns the right to respond to that stimulus in future, and the learning rule adjusts its weight vector so that its response to that stimulus in future will be enhanced, i.e. by moving the weight vector closer to the connection vector. This means that the next time that same connection vector is presented, the neuron that won the competition for that same vector last time will be more excited by it. During training, the SOM learns to project connection vectors that are close together in terms of Euclidean distance onto neurons that are close to each in the output grid. In this way, the SOM learns relationships between the connections a vector, expressing them as spatial relationships in the output grid. The training algorithm also ensures that the weight vectors of the neurons area good representation of the connection vectors in the training data. This is achieved by aiming for a low mean quantisation error, where the quantisation error is the distance between a connection vector and the winning neuron ‘s weight vector. The mean quantisation error is the average of this over all connection vectors in the training set

K - MEANS

The K-means algorithm, starts with k arbitrary cluster centers in space, partitions the set of the given objects into k subsets based on a distance metric [17]. The centers of clusters are iteratively updated based on the optimization of an objective function. This method is one of the most popular clustering techniques, which are used widely, since it is easy to be implemented very efficiently with linear time complexity (Biswas, Shah, Tammi, & Chakraborty, 2016). The principle goal of employing the K Means clustering scheme is to separate the collection of normal and attack data that behave similarly into several partitions which is known as K^{th} cluster centroids. In other words, K-Means estimates a fixed number of K, the best cluster centroid representing data with similar behavior. The algorithm initially has empty set of clusters and updates it as proceeds. For each record it computes the Euclidean distance between it and each of the centroids of the clusters. The instance is placed in the cluster from which it has shortest distance. Assume we have fixed metric M, and constant cluster Width W. Let di (C, d) is the distance with metric M, cluster centroid C and instance d where centroid of cluster is the instance from feature vector

Fuzzy c-means (FCM)

Fuzzy c-means (FCM) is an improvement of K-means algorithm has become very important in the application of intrusion detection. In fuzzy C-means is a clustering method that calculates the membership function between each test data instance and each cluster [10], [20]. The test data instance is allocated to the cluster which has higher membership [15], [21]. In fuzzy C-means, the individual data point can belong to several clusters at the same time. Nevertheless, the degree of membership is determined by membership grades which are assigned to each data point. For each x_i in dataset D the fuzzy C-means algorithm assigns membership grade u_{ij} which shows the degree of x_i membership in cluster j ($0 \leq u_{ij} \leq 1$). The membership grades are calculated for each example based on the minimization of an objective function which measures the distance between each data point and the cluster centers. If m is the size of the input dataset and K is the number of clusters, this objective function is calculated as follows:

K membership value to each center. After that, it finds higher membership and assigns the instance to higher membership cluster. In other words, the instance in test dataset will divided into two clusters according to the degree of membership to C_1 and C_2 in this case. In the above equation q is the fuzziness exponent and can be any real value greater than 1 depending on the kind of problem. c_j is the center of j-th cluster and its dimensions are equal to that of input vector x_i. Creating the clusters is done through an iterative optimization process for objective

function in which membership grades u_{ij} and cluster centers c_j are updated Once the Fuzzy C-means algorithm obtains the unlabeled dataset of magnitude m as input, it executes the above process and the output are two matrices: The Matrix U which consist of membership grades of each data example in each of the K clusters and matrix C which includes the cluster centers for K clusters then.

To create K disjoint subsets from the dataset based on matrix U, one subset for each individual example in the training dataset is determined based on its maximum membership grade i.e.

for each x_i: if $u_{iw} = \max \{u_{ij}\}$ then $x_i \in D_w$,

where i = 1, 2, . . ., m; j = 1, 2, . . ., K.

After calculating the subset for all examples, the training dataset is divided to K disjoint subsets D1, D2, . . ., DK. These K subsets are used to train classification techniques like ANN, SVM etc.

Expectation and Maximization Algorithm (EM)

The EM algorithm [22] [17]is a clustering technique in data mining and consists of two repeated steps, Expectation and Maximization. It is based on Gaussian finite mixtures model (GMM) for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables [23]. The EM algorithms alternates between performing an expectation (E) step, which computes the expectation of the log- likelihood evaluated using the current estimate for the parameters, and maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. The model consists of a set of k probability distributions that represent the data of each cluster while the number of iteration and log likelihood difference between two iterations are parameters that defines each of the k distributions. Initially, the algorithm makes guesses for these parameters based on the input data, then determines the probability that a particular data instance belongs to a particular cluster for all data using these parameter guesses. The distribution parameters are revised again and this process is repeated until the resulting clusters have some level of overall cluster 'goodness' or until a maximum number of algorithm iterations are reached.

Mathematically, the algorithm attempts to find the parameters θ, that maximize the probability function, log P (x; θ) of the observed data. It reduces the difficult task of optimizing log P (x; θ) into a sequence of simpler optimization sub problems, whose objective functions have unique global maxima that can often be computed in closed form. These sub problems are chosen in a way that guarantees their corresponding solutions $\varphi^{(1)}$ $\varphi^{(2)}$ ... and will converge to a local optimum of log P (x; θ). The Expectation step (E-step) of the algorithm estimates the clusters of each data instance given the parameters of the finite mixture. During the E-step, the algorithm chooses a function $f(g_t)$, that lower bounds log P (x; θ) everywhere, and for which $f(\varphi^{(1)}) = \log P (x; \varphi^{(t)})$. The Maximization step (M-step) of the algorithm tries to maximize the likelihood of the distributions that make up the finite mixture, given the data. During the M-step, the algorithm moves to a new parameter set $\varphi^{(t+1)}$, that maximizes $f(g_t)$. As the value of the lower-bound $f(g_t)$ matches the objective function at $\varphi^{(t)}$, it follows (9), so the objective function monotonically increases during each of the iterations in EM.

$$\text{Log P } (x; \varphi^{(t)}) = g_t (\varphi^{(t)}) \, g_t (\leq \varphi^{(t+1)}) = \log P (x; \varphi^{(t+1)}) \qquad \text{eqn 1}$$

Training data with the results of normalization and discretization techniques enter clustering step. The dataset will be divided into number of clusters in FCM, K-means, and EM to find the optimal results. Similarly, well test the SOM by simultaneously varying the epochs and lattice configuration. Two third of the dataset will be used for training and the rest is for testing.

The NSL-KDD Dataset

The simulated attacks in the NSL-KDD dataset fall in one of the following four categories [24].

i. Denial of service attack (Dos), where attempts are to shut down, suspend services of a network resource remotely making it unavailable to its intended users by overloading the server with too many requests to be handled. e.g. syn flooding. Relevant features include source bytes and percentage of packets with errors. Examples of attacks includes back, land, Neptune, pod, Smurf, teardrop

ii. Probe attacks, where the hacker scans the network of computers or DNS server to find valid IP, active ports, host operating system and known vulnerabilities with the aim discover useful information. Relevant features include duration of connection and source bytes. Examples includes IP sweep, n map, port sweep, Satan

iii. Remote-to-Local (R2L) attacks, where an attacker who does not have an account with the machine tries to gain local access to unauthorized information through sending packets to the victim machine in filtrates files from the machine or modifies in transit to the machine. Relevant features include number of file creations and number of shell prompts invoked. Attacks in this category includes ftp_ write, guess_ passwd, I map, multi hop, phf, spy, warezclient, warezmaster

iv. User-to-Root (U2R) attacks, where an attacker gains root access to the system using his normal user account to exploit vulnerabilities. Relevant features include Network level features – duration of connection and service requested and host level features - number of failed login attempts. Attacks includes buffer overflow, load module, Perl, rootkit

Experimentation, Results and Discussion

In implementation of the model, the researcher used WEKA Software. Three set of experiments were conducted and the results are tabulated in Table 1: In first experiment clustering with data preprocessing based on hybrid feature selection only (i.e., labeled as HFS), the second experiment clustering with PCA only (i.e., labeled as PCA), and the third experiment clustering with HFS and PCA (i.e., labeled as IPCA). The four measurements techniques applied are: (1) Clustering Error (CE) is the number of alerts that are wrongly clustered. (2) Error Rate (ER) is the percentage of wrongly clustered alerts, ER = (CE ÷ Total number of alerts observed) x 100, (3) Accuracy Rate (AR) is the percentage of alerts that are accurately clustered as they should be, AR = 100 – ER, and (4) Time is the algorithm processing time in seconds.

Table1: Clustering Performance based on Self-organizing maps (SOM), Expectation Maximization, K-means and Fuzzy c-means (FCM)

Mode	FCM				K Means				SOM				EM			
	CE	ER	AR	TI	CE	ER	AR	TI	CE	ER	AR	TI	CE	ER	AR	TI
HFS	74	17.5	82.6	1.3	57	13.4	86.6	4.4	135	31.8	68.2	4.2	45	10.6	89.4	1.9

PCA	133	31.3	68.6	3.6	141	33.3	66.2	5.2	170	40.1	60.0	6.5	86	20.3	79.7	2.7
IPCA	67	15.8	84.2	4.8	46	10.9	89.2	6.2	112	26.4	73.6	7.4	41	9.7	90.3	4.6

The number of clusters in FCM, K-Means, and EM were varied to find the optimal results. The SOM was tested by concurrently changing the epochs and lattice configuration. Two third of the dataset were used for training and the rest for testing. The optimum result on SOM (73.58%) was obtained after being trained for 2500 epochs using hexagonal 4 by 6 lattice type and produced 12 clusters. The SOM's best processing time both for training and testing was obtained after 7.4 seconds. Increasing or decreasing the processing changes the results if the dataset, epochs and lattice type are larger (Siraj et al., 2009c). The results of k-means clustering algorithm indicated that the performance depends on the number of clusters which are applied, and increasing or decreasing the cluster beyond the number of data types only lessens the efficiency of the model. Identifying the number of clusters therefore significantly changes to the results.

The research has to determine the number of clusters that are expected in advance in order to obtain good results. In this work several clusters were tested and the optimum results (89.2) were obtained at 22 clusters in a time of 6.2 seconds. However, the challenge of identifying the number of clusters in a dynamic network, is much more difficult since there is no base data to assist in deciding the number of clusters.[26] The best clustering algorithm was EM 90.3% and is arrived at 14 clusters in a time of 4.6 seconds. In respectively cluster, related alerts are clustered together and represent an attack step. The value of CE of FCM, K- Means, SOM and hybrid is larger, and hence a large number of alerts that belong together in one cluster are put into other different clusters. The result inferred that the proposed model based on hybrid feature selection, PCA and EM is effective in terms of clustering accuracy and processing time for this dataset.

Conclusion

The output is a hybrid machine learning approach for automated alert clustering and filtering based on EMFFS, Principal Component Analysis (PCA) and Expectation and maximization techniques that gives optimum results to aggregate similar alerts and to reduce the number of alerts compared to other unsupervised learning algorithms tested. The results are promising in terms of clustering accuracy rate (89.2) and processing time (6.2 sec). The model cannot reveal the memberships of attack stages like that of multi-stages attack which comprise of one/more attack steps.

REFERENCES

[1] L. Dhanabal and S. P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 6, pp. 446–452, 2015.

[2] K. Kumar, "Network Intrusion Detection with Feature Selection Techniques using Machine-Learning Algorithms," vol. 150, no. 12, pp. 1–13, 2016.

[3] S. T. Ikram and A. K. Cherukuri, "Improving Accuracy of Intrusion Detection Model Using PCA and Optimized SVM," vol. 24, no. 2, pp. 133–148, 2016.

[4] D. Perez, M. A. Astor, D. P. Abreu, and E. Scalise, "Intrusion Detection in Computer Networks Using

Hybrid Machine Learning Techniques," 2017.

[5] Y. Kumar, "AI based Hybrid Ensemble Technique for Network Security," no. Icaet, pp. 1–10, 2016.

[6] S. Upadhyay, "A Survey on IDS Alerts Classification Techniques," vol. 105, no. 12, pp. 27–33, 2014.

[7] M. Panda, A. Abraham, and M. R. Patra, "A hybrid intelligent approach for network intrusion detection," *Procedia Eng.*, vol. 30, no. 2011, pp. 1–9, 2012.

[8] A. I. Madbouly, A. M. Gody, and T. M. Barakat, "Relevant Feature Selection Model Using Data Mining for Intrusion Detection System," *Int. J. Eng. Trends Technol.*, vol. 9, no. 10, pp. 501–512, 2014.

[9] H. T. Elshoush and I. M. Osman, "An Improved Framework for Intrusion Alert," vol. I, pp. 4–9, 2012.

[10] T. A. Alhaj, M. M. Siraj, A. Zainal, H. T. Elshoush, and F. Elhaj, "Feature selection using information gain for improved structural-based alert correlation," *PLoS One*, vol. 11, no. 11, pp. 1–18, 2016.

[11] S. Chaurasia and A. Jain, "Ensemble Neural Network and K-NN Classifiers for Intrusion Detection," vol. 5, no. 2, pp. 2481–2485, 2014.

[12] C. Science, "A Hybrid Approach to improve the Anomaly Detection Rate Using Data Mining Techniques," no. July, 2015.

[13] F. Valeur, "Real-Time Intrusion Detection Alert Correlation," *Security*, no. June, p. 199, 2006.

[14] J. M. Chahira, "Model for Intrusion Detection Based on Hybrid Feature Selection Techniques," *Int. J. Comput. Appl. Technol. Res.*, vol. 09, no. 03, pp. 115–124, 2020.

[15] V. Sannady and P. Gupta, "Intrusion Detection Model in Data Mining Based on Ensemble Approach," pp. 1654–1658, 2016.

[16] A. Shrivastava, M. Baghel, and H. Gupta, "A Review of Intrusion Detection Technique by Soft Computing and Data Mining Approach," no. 3, pp. 224–228, 2013.

[17] M. M. Siraj, M. A. Maarof, and S. Z. M. Hashim, "Intelligent alert clustering model for network intrusion analysis," *Int. J. Adv. Soft Comput. its Appl.*, vol. 1, no. 1, pp. 33–48, 2009.

[18] B. Subba, S. Biswas, and S. Karmakar, "Enhancing effectiveness of intrusion detection systems : A hybrid approach."

[19] N. A. Biswas, F. M. Shah, W. M. Tammi, and S. Chakraborty, "FP-ANK: An improvised intrusion detection system with hybridization of neural network and K-means clustering over feature selection by PCA," *2015 18th Int. Conf. Comput. Inf. Technol. ICCIT 2015*, pp. 317–322, 2016.

[20] M. Amini, "Effective Intrusion Detection with a Neural Network Ensemble Using Fuzzy Clustering and Stacking Combination Method," vol. 1, no. 4, pp. 293–305, 2014.

[21] B. S. Harish and S. V. A. Kumar, "Anomaly based Intrusion Detection using Modified Fuzzy Clustering," vol. 4, pp. 54–59, 2017.

[22] M. M. Siraj, M. A. Maarof, and S. Z. M. Hashim, "A Hybrid Intelligent Approach for Automated Alert Clustering and Filtering in Intrusion Alert Analysis," *Int. J. Comput. Theory Eng.*, vol. 1, no. 5, pp. 539–545, 2009.

[23] Y. Wahba, E. ElSalamouny, and G. ElTaweel, "Improving the Performance of Multi-class Intrusion Detection Systems using Feature Reduction," *Ijcsi*, vol. 12, no. 3, pp. 255–262, 2015.

[24] M. R. Parsaei, S. M. Rostami, and R. Javidan, "A Hybrid Data Mining Approach for Intrusion Detection on Imbalanced NSL-KDD Dataset," vol. 7, no. 6, pp. 20–25, 2016.

[25] M. M. Siraj, M. A. Maarof, and S. Z. M. Hashim, "Intelligent clustering with PCA and unsupervised

learning algorithm in intrusion alert correlation," *5th Int. Conf. Inf. Assur. Secur. IAS 2009*, vol. 1, pp. 679–682, 2009.

[26] S. Duque and M. Nizam, "Using Data Mining Algorithms for Developing a Model for Intrusion Detection System (IDS)," *Procedia - Procedia Comput. Sci.*, vol. 61, pp. 46–51, 2015.