ACIT'2017
The International Arab Conference on Information Technology
Yassmine Hammamet, Tunisia
December 22-24, 2017

ATwo Stage Intrusion Detection Intelligent System

Nevrus Kaja, Adnan Shaout and Di Ma
The University of Michigan – Dearborn, United States

***Abstract*—Security is becoming an inherited and amplified problem that keeps growing as new and emerging technologies hit the market. The problem of detecting and preventing malicious attacks is just one of the puzzle pieces that many enterprises prioritize in their operations. Challenges brought by security breachesor malicious attacks can put a high stress, almost un-survivable, on social and economic factors. In this paper, the usage of a two stage intelligent system in intrusion detection systems (IDS) will be introduced. Anomaly based IDS' are a feasible solution for many security problems, but the high rate of false positives makes them difficult to implement. The objective of the first stage in this proposed IDSis to simply detect whether there is an attack present through unsupervised learning. The second stage classifiesthese attacks in a supervised learning methodology, along with addressing and eliminating the number of false positives. The simulation of this approach results is an IDS able to detect and classify attacks at a 99.97% accuracy and lowers the false positives rate to 0%.**

Keywords—Intrusion detection systems (IDS), security, machine learnings, IPS, Knowledge discovery in databases (KDD)

I. INTRODUCTION

The advancements and innovations of different technological systems have brought many dependencies in networkstructures. With this dependency, one of the main questions that remains relevant is their security aspects. Very often, we open the television and watch unfortunate news about corporation hacking and other security breaches. These problems have huge financial losses for such corporation that many cannot even survive after these breaches. Privacy leaks also have a social impact to the wider society. Questions such as: is a system secure? Can it be hacked? Can we detect or prevent an intruder? are nowadays on the minds of every CEO[18].

Currently, artificial intelligence, machine learning, and other emerging technologies have taken the lead and are the hot trend. They are starting to solve many problems in innovative ways. The purpose of this paper is to apply these techniques and technologies to the security problem. Whether it is IoT, automotive, traditional networking or other technologies, there is always a dependency on data to analyze and detect intruders [18]. In this project, KDDdata is used toa number of different machine learning algorithms to build a two-stageintrusion detection system. KDD is a set of data gathered from MIT Lincoln labs, simulating a typical US Air Force LAN. They injected a set of attacks to such an environment which vary from DOS, R2L, U2R and probing [1]. In this paper,this data is used to build and test the proposed two stage IDS. The rest of this paper is organized as follow: Section 2 gives some background and relevant work on intrusion detection systems. Section 3 analyses the dataset and provides the feature engineering process. Section 4 gives the description of IDS and section 5 summarizes the performance results.

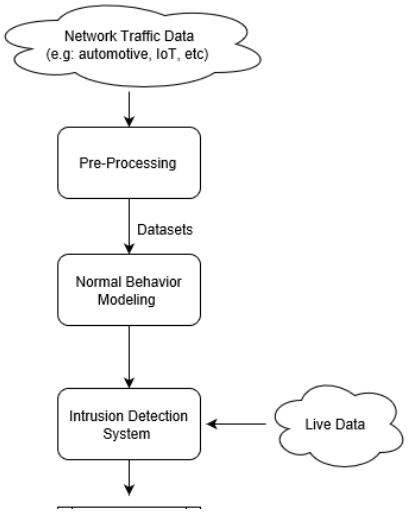

Figure 1: Intrusion Detection System Architecture

II. INTRUSION DETECTION SYSTEMS

What are intrusion detection systems (IDS)? IDS'are software applications that usually monitor a network traffic for suspicious activity or any malicious attacks. IDS systems are used in a wide variety of applications and as attacks are becoming more sophisticated. IDS are evolving in their nature as well to keep up with the progress. They usually work by modeling the normal behavior of a system and then comparing it with the current monitored state of the system. When an action significantly deviates from the normal behavior, then the system analyses such behaviors and categorizes the action into a certain anomaly [2]. This is often called anomaly based

detection and it works on the premises of real time data. In most general scenarios, attacks start with probes and sweeps against the network hardware. Intrusion Detection Systems (or Intrusion Prevention Systems - IPS) look at the data to identify these sweeps and probes and give an alert against an expected attackbehavior [2]. Generally speaking, IDS systemshave the architecture shown in Figure 1.

IDS are generally separated into the following two main categories: signature based systems and anomaly based systems [4, 5, 6]:

A. *Signature Based IDS*

In these kind of systems, the network traffic is monitored for probes and sweeps, which are preconfigured and predetermined attack patterns known as signatures [4]. These type of IDS are efficient for pre-known signatures but are inefficient when the IDS has no prior signature knowledge. Therefore, the main drawback of signature based systems is the ability to continuously update the signature database [3]. In additions, signature based systems are also not efficient towards attacks with self-modifying behavior.

B. *Anomaly based*

Anomaly based attacks work on the principle of modeling the normal behavior of the system. They continuously monitor the data to create the baseline model, then look for deviations from that. The benefit of these systems from the signature ones is the detection of malicious attacks even if those attacks were unknown to the system prior. But because there might be noise or other elements in the data, they generally have a problem with high false alarm rates [3, 4, 5, 6]. In this paper, we build an anomaly based, two-stage, intelligent, intrusion detection system with the main purpose of heavily reducingthe number of false positives (FP). We use an approach similar to the one which was used for a traffic light detection with two stage classifiers where false positives were reduced to almost zero [7].

C. *Related Work*

Other research projects have built IDS' with relevant performances but their number of false positives has been considerable. A summary of these project performances is provided in Table 1.

Table 1: Related Work

Research Project	Algorithm	Detection Rate
2017:Meena,Choudhary [21]	Bayes	92.7
2016: Subba at.al. [19]	ANN	95.05
2016: Subba at.al. [19]	Bayes	87.97
2016: Subba at.al. [19]	SVM	97.56
2015: Alom at.al. [20]	DBN	97.5

III. KDD CUP 99 DATA SET

KDD 99 cub data set is a standard and open source intrusion detection data set used in the 3rd International Knowledge Discovered and Data Mining Tool Competition [11]. To prepare this, MIT Lincoln labs set up an environment to gather TCP data dumps from a local area network (LAN) simulating a US Air Force network. They gathered about 7weeks' worth of raw data (tcpdump)for training and injected multiple attacks into it [9]. This data set is about 4 gigabytes of compressed data, comprised of around 5 million connections (4,900,000). In addition, there are also two weeks' worth of data for testing with around 2 million connections. A connection is defined as a sequence of TCP packets with a predefined start and end time from a source to a target IP [1, 9]. Each of these connections in the training data is labeled as "normal" or "attack" along with the specific attack type and each connection is about 100 bytes in size. Each of the specific attacks isclassifiedinto the following categories [1]:

- U2R: unauthorized access to local root privileges. (e.g. buffer overflow attack)
- R2L: unauthorized access from a remote machine. (e.g. guessing a password)
- Probing: unauthorized port scanning, surveillance or other probing.
- DOS – denial of service attacks. (e.g. syn flood)

Connections in the test data contain attacks that are not present in the training data. This makes the problem more realistic and allows to look at the performance of the intrusion detection systems onunknown type of attacks. There are 22 attack types in the training data and 17 additional new and unknown attacks in the test data[9].

Table 2: Attacks injected in the data

Attack Name	Attack Type	# of records
land	DOS	32
back	DOS	862
neptune	DOS	65825
pod	DOS	365
teardrop	DOS	956
smurf	DOS	854
ipsweep	Probe	685
satan	Probe	951
portsweep	Probe	635
nmap	Probe	1687
guess_passw	R2L	26
ftp_write	R2L	62
phf	R2L	64
imap	R2L	39
warezmaster	R2L	26
warezclient	R2L	75

spy	R2L	65
multihop	R2L	15
buffer_overflow	U2R	68
load_module	U2R	64
perl	U2R	35
rootkit	U2R	51
normal records	n/a	94,525
	Total	**167,967**

A. Data Pre-Processing

After doing some analysis on the dataset [9], it isdiscovered there are a significant amount of redundanciesin the dataset. Therefore, from the initial tests, it isdiscovered that training would lead to bias towards the more frequent records. In addition, the data itself is too large to process and often initial results lead tooverfitting. Therefore, pre-processing is needed to have a well-representeddata.To do this,first, correlated features need to be removed to avoid overfitting. Table 2shows all the attacks that are injected in the data along with the number of records for each of them.

In addition to attacks, the data comes with 41 features. To train a model which is able to infer well through a set of the most representative features, dimensionality reduction was needed. These steps are often called "feature engineering" and theyavoid under-fitting or overfitting. To start, the variance of feature valuesiscalculated. Features with low variances usually donot give much knowledge, therefore they were filtered outduring training. Equation (1) shows the variance.

$$\sigma^2 = \frac{\sum_{k=0}^{n} x^2}{N} - \mu^2 \qquad (1)$$

Table 3shows all the features along with their variance values.

Table 3: Data Features

Feature Name	Type	Variance Values
duration:	continuous	1.98E+06
protocol_type:	symbolic	1.57E-01
service:	symbolic	2.18E+02
flag:	symbolic	1.03+E01
src_bytes:	continuous	2.23E+11
dst_bytes:	continuous	4.50E+08
land:	symbolic	3.10E-04
wrong_fragment:	continuous	2.03E-02
urgent:	continuous	1.33E-03
hot:	continuous	8.61E-01
num_failed_logins:	continuous	2.26E-02
logged_in:	symbolic	2.47E-01
num_compromised:	continuous	2.30E-03
root_shell:	continuous	2.43E-03
su_attempted:	continuous	4.44E-04
num_root:	continuous	6.47E+01
num_file_creations:	continuous	4.58E-01
num_shells:	continuous	2.30E-03
num_access_files:	continuous	4.60E-03
num_outbound_cmds:	continuous	0.00E+00
is_host_login:	symbolic	4.88E-04
is_guest_login:	symbolic	2.76E-02
count:	continuous	1.65E+04
srv_count:	continuous	7.93E+03
serror_rate:	continuous	8.72E-02
srv_serror_rate:	continuous	8.90E-02
rerror_rate:	continuous	1.73E-01
srv_rerror_rate:	continuous	1.73E-01
same_srv_rate:	continuous	1.70E-01
diff_srv_rate:	continuous	6.72E-02
srv_diff_host_rate:	continuous	6.43E-02
dst_count:	continuous	8.84E+03
dst_srv_count:	continuous	1.25E+04
dst_same_srv_rate:	continuous	1.89E-01
dst_diff_srv_rate:	continuous	4.87E-02
dst_same_src_port_rate:	continuous	9.38E-02
dst_srv_diff_host_rate:	continuous	7.29E-03
dst_serror_rate:	continuous	7.46E-02
dst_srv_serror_rate:	continuous	7.95E-02
dst_rerror_rate:	continuous	1.45E-01
dst_srv_rerror_rate:	continuous	1.61E-01

As it can be seen from Table3 "land", "su_attempted", "num_outbound_cmds" and "is_host_login"have considerablelow variances, therefore they can be removed from consideration. Another method to reduce the number of features is to look at their correlation coefficient. Correlation coefficient shows how much the variances of two variables are associated with each other. In training, we are interested in the most representative features.Variables who look alike and change alike will not give much new knowledge and thus they can be removed from the data set

Equation (2)isused to calculate correlation coefficient among the given features.

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2 \sum(y-\bar{y})^2}} \qquad (2)$$

Table 4 shows pairs of features who are almost identically correlated.

Table 4: Correlated Features

Feature 1	Feature 2	Correlation
dst_srv_serror_rate:	dst_serror_rate:	0.95
dst_srv_rerror_rate	rerror_rate:	0.93
dst_srv_rerror_rate	srv_rerror_rate:	0.95

rerror_rate:	service:	0.91
num_compromised:	num_root	0.99
serror_rate	srv_serror_rate	0.97
srv_rerror_rate	rerror_rate	0.97

As it is observed from Tables3 and 4, all the 11 features (in red) can be removed from the data since they do not provide a distinct knowledge to the training model of the IDS. Therefore, the number of features was reduced from 41 to 30, which is more manageable.

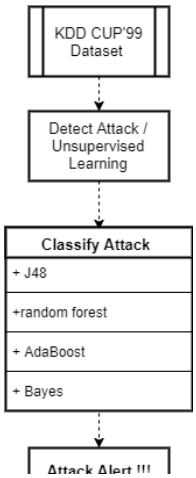

Figure 2: Intrusion Detection System Architecture

IV. MACHINE LEARNING BASED IDS

After "feature engineering" the next step is to design an intrusion detection system architecture based on machine learning algorithms. The contribution and novelty of this work is the ability to use a two stage (supervised/unsupervised) machine learning approach in the design of IDS. This is done in order to avoid false positives and accurately detect/prevent an attack [7]. The first phase objective is to simply detect whether there is an attack or not. The second phase is to classify the attack and provide an alert for it. Figure 2 shows the architecture block diagram of the proposed system.

K-Means algorithm is used in the first phase to cluster the dataand detect an attack [12].Clustering in this phase simply categorized the data into two categories: "attack" connections or "normal" connections. The design objective here is not to categorize the type of attacks but to simply detect if there is one. To achieve this there should be two clusters with good intra cluster similarity and low inter cluster similarity[12]. K-Means clustering is an unsupervised algorithm which processes ann observations into k clusters[13]. The general idea here is to look at the engineered features of the data and place them into two clusters. The placement of each point is decided by looking at the Euclidian distance between the point and the cluster center. Once a point is associated with a cluster, then the cluster centroid is recalculated accordingly[14]. The goalof the algorithm is to minimize the objective function which is known as the squared error function [15].

Figure 3 shows the results of data clustering for the purpose of attack detection after going through the K-Meansalgorithm.

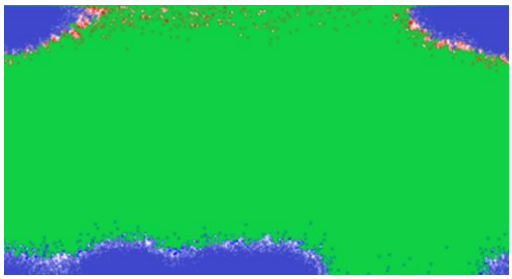

Figure 3: K-Means Clustering

As it can be seen from Figure 3, two clusters (with different colors) are well defined and the membership functions are according to the following numbers:

- Cluster 0 (Attack) : 37,847 or 23%
- Cluster 1 (Normal): 130120 or 77%

In addition, the above clusters do show the 4 types of attacks (corners), but the purpose of this phase is to simply detect attacks rather than classify them, so theseparation is ignored for this phase.

After detecting anattack in phase one, then the next step is to classify those attacks in phase two with a variety of machine learning algorithms. This is done to avoid false positives and correlate the results with the first phase. The following algorithms are used to test theirperformance regarding classifying the attacks:

A. J48

J48 is a classification algorithm based on decision trees. It uses the traditional C4.5 decision treeto build a decision trees based on information entropy [13]. While building the tree, the algorithm chooses the attribute according to the information gain, or whichever attribute gives the most effective split of the data [16]. The following are the algorithm steps:

1. For each attribute calculate the followingentropy:
$$Entropy = \sum_{i=1}^{C} -p_i * log_2(p_i) \quad (4)$$
2. Calculate information gain rate on that attribute according to the following equation

$$Gain(T,X) = Entropy(T) - Entropy(T,X) \quad (5)$$

3. Find the attribute with the highest information gain
4. Split the data on that attribute
5. Repeat steps 1-4 until you reach the leaf level

Table 5 shows the results of applying the above algorithm to the prepared data set.

Table 5: J48 Performance Results

Name	Value
Number of Leaves	711
Size of the tree	833
Correctly classified	99.9512%
Incorrectly classified	0.0488%
Mean absolute error	0.0001
Root mean squared error	0.0064
True Positive Rate	1.000
False Positive Rate	**0.000**
F-Measure	0.999

To explain some of the performance variables:

- Correctly classified:gives the % of connections that are correctly classified
- Incorrectly classified: gives the % of instances that are incorrectly classified
- Mean absolute error: measures the average magnitude of the errors without considering direction
- Root mean squared error:gives the average magnitude of the error
- True false positive: gives the number of instances predicted positive that are actually positive
- False positive rate: gives the number of instances predicted positive but are actually negative
- F-Measure: gives a combined measure of precision and recall

Features with the highest information gain in the data are as follow:

- srv_count
- same_srv_rate
- dst_diff_srv_rate

The number that is most interesting in this system is false positive (FP). Since one of the main drawbacks of anomaly based systems is the high number of false positives, thenew proposed system is designed with two stages to lower such performance element.

B. *Random forest*

If decision trees are combined through an ensemble learning method, then they form a random forest. Random forest is like bagging but it disrupts the downside of baggingwith a multitude of decision trees [17]. The output of random forest is the mode output among the decision trees. Table 6 shows the performance results obtained after running the data through therandom forest.

Table 6: Random Forest Performance Results

Name	Value
Number of Iterations	100
Correctly classified	99.9708%
Incorrectly classified	0.0292%
Mean absolute error	0.0001
Root mean squared error	0.0047
True Positive Rate	1.000
False Positive Rate	**0.000**
F-Measure	0.999
Computation Time	235.52s

C. *AdaBoost*

AdaBoost is another ensemble of machine learning algorithms used for classifications. This is similar to regular boosting algorithms where subsequent models attempt to correct prediction errors made by prior models. This ensemble algorithm uses short decision trees and adds them in series until the performance is not subsequently improved[16]. Table 7 provides the performance results from AdaBoost implementation.

Table 7: AdaBoost Performance Results

Name	Value
Correctly classified	97.8597%
Incorrectly classified	2.1403%
Mean absolute error	0.0478
Root mean squared error	0.1228
True Positive Rate	0.979
False Positive Rate	**0.005**
F-Measure	0.970
Computation Time	45.65s

D. *Naïve Bayes*

The last algorithm that was used in the intrusion detection system was Naïve Bayes. Naïve Bayes is a probabilistic classification algorithm based on Bayes theorem as shown in equation (6).

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \qquad (6)$$

Bayes theorem gives the probability of an event based on prior knowledge of conditions that lead to such event. Table 8 shows the results of applying the Naïve Bayes algorithm.

Table 8: Naive Bayes Performance Results

Name	Value
Correctly classified	92.748%
Incorrectly classified	7.252%

Mean absolute error	0.0063
Root mean squared error	0.0774
True Positive Rate	0.927
False Positive Rate	**0.000**
F-Measure	0.949
Computation Time	40.69s

V. Performance Summary

Out of the four algorithms tested in the second stage, results are summarized according asshown in figures 4 and 5.

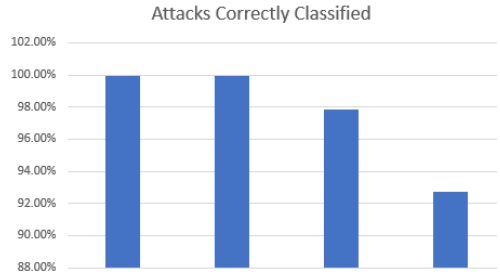

Figure 4: Attacks correctly classified

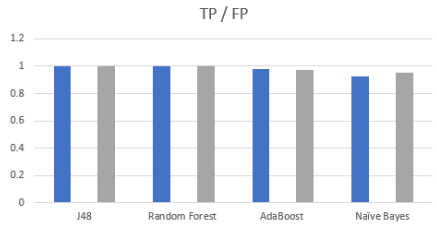

Figure 5: True and False Positives (TP/FP),F-Measure

As it can be seen from Figures 4 and 5, J48 is the best performing algorithm to detect and classify attacks.
Reference [16] describes J48 as one of the most practical and best-performing algorithms in machine learning, therefore this claim is validated in this research. It is important to mention that, this system is able to eliminate any false positive through its second stage. False positives are a syndrome of anomaly based intrusions, and the proposedIDS architecture in this paper is able to cope with them in an easy and simple way.

VI. Future Work

This work was performed to prove the concept of using a two stage, intelligent, hybrid systems in intrusion detection. The test data from MIT Lincoln lab provided a good test bed to explore and implement the proposed IDS architecture. Being able to use both supervised and unsupervised learning algorithms in a staged approach gives promising results, therefore the plan is to continue diving deeper with these methodologies. Automotive security is one of the hottest topics that is emerging nowadays and specially for connected data. Therefore being able to apply such methodologies in DSRC based, connected vehicle systems is the futurestep.

VII. Summary

In this project,initially,we have provided a background in intrusion detection systems and their value in overall security to detect a variety of attacks. Then the KDD CUP datasetwas analyzed and engineered to prepare it for use in a two stage intrusion detection system architecture. Reducing the number of features was necessary to learn efficiently and gainbetter performance results. Then an intelligent system which is based on two stage architecture was builtto detect and classify attacks. The first stage uses unsupervised learning to simply detect an attack and the second stage classifiesit accordingly using a supervised learning algorithm. From the performance evaluation of the experimental results, extremely high results were achieved. In addition false positives which are considered a syndrome of anomaly based attack detections were eliminated. In the future, we are planning to apply this architecture for automotive security.

References

[1] KDD Cup 1999 Data. (2017). [Network Traffic Dataset] Web.

[2] A Survey on Machine Learning Techniques for Intrusion Detection Systems. International Journal of Advanced Research in Computer and Communications Engineering, 2(11), pp.4349-4355.

[3] Chandolikar, N. and Nandavadekar, V. (2012). Efficient algorithm for intrusion attack classification by analyzing KDD Cup 99. 2012 Ninth International Conference on Wireless and Optical Communications Networks

[4] Stephen Northcutt , Judy Novak "Network Intrusion Detection", Third Edition, New Riders Publishing.

[5] Kayacik, G. H., Zincir-Heywood, A. N., "Analysis of Three Intrusion Detection System Benchmark Datasets Using Machine Learning Algorithms", Proceedings of the IEEE ISI 2005 Atlanta, USA, May 2005.

[6] Ozgur Depren, Murat Topallar, Emin Anarim, M. Kemal Ciliz. "An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks". Expert Systems with Applications 29 (2005) 713–722Expert Systems with Applications 29 (2005)713722.www.elsevier.com/locate/eswa.

[7] Kaja, N., Shaout, A. and Dehzangi, O. (2017). Two Stage Intelligent Automotive System to Detect and Classify a Traffic Light. To appear in the International Conference on new Trends in Computing Sciences.

[8] H. Güne Kayacık, A. Nur Zincir-Heywood, Malcolm I. Heywood,. " Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99Intrusion Detection Datasets". Dalhousie University, Faculty of Computer Science,

[9] Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, A. (2009). A detailed analysis of the KDD CUP 99 data set. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications.

[10] Ghanshyam Prasad Dubey, Prof. Neetesh Gupta, Rakesh K Bhujade " A Novel Approach to Intrusion Detection

System using Rough Set Theory and Incremental SVM". International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-1, Issue-1, 03,2011.

[11] Pervez, M. and Farid, D. (2014). Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs. The 8th Conference on Software, Knowledge, Information Management and Applications

[12] Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. Communications of the ACM, 39(11), pp.27-34.

[13] MacQueen J. B: "Some Methods for classification and Analysis of Multivariate Observations," in 5th Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, pp. 281-297.

[14] Velmurugan T., and Santhanam T., (2010): "Performance Evaluation of K-Means and Fuzzy C-Means Clustering Algorithms for Statistical Distributions of Input Data Points," European Journal of Scientific Research, vol. 46, no. 3, pp. 320-330.

[15] Borah S., and Ghose M. K., (2009): "Performance analysis of AIM-K-Means and K-Means in quality cluster generation," Journal of Computing, vol. 1, no. 1.

[16] Pandey, P. and Prabhakar, R. (2016). An analysis of machine learning techniques (J48 & AdaBoost)-for classification. 2016 1st India International Conference on Information Processing (IICIP).

[17] Abar, T., Ben Letaifa, A. and El Asmi, S. (2017). Machine learning based QoE prediction in SDN networks. 2017 13th Wireless and Mobile Computing Conference.

[18] Iguer, H., Medromi, H., Sayouti, A., Elhasnaoui, S. and Faris, S. (2014). The Impact of Cyber Security Issues on Businesses and Governments: A Framework for Implementing a Cyber Security Plan. 2014 International Conference on Future Internet of Things and Cloud.

[19] Subba, B., Biswas, S. and Karmakar, S. (2016). A Neural Network based system for Intrusion Detection and attack classification. 2016 Twenty Second National Conference on Communication (NCC).

[20] Alom, M., Bontupalli, V. and Taha, T. (2015). Intrusion detection using deep belief networks. 2015 National Aerospace and Electronics Conference (NAECON).

[21] Meena, G. Choudhary, R. (2017). A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA. 2017 ICCCE