UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Intrusion Alert Reduction Based on Unsupervised and Supervised Learning Algorithms

Oyinkansola Oluwapelumi Kemi Afolabi-B., Maheyzah @ MD Siraj
Faculty of Computing,
Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Johor, Malaysia
Email: kemiafolabi@rocketmail.com

Submitted: 1/8/2021. Revised edition: 28/9/2021. Accepted: 26/10/2021. Published online: 15/11/2021
DOI: https://doi.org/ 10.11113/ijic.v11n2.331

***Abstract*—Security and protection of information is an ever-evolving process in the field of information security. One of the major tools of protection is the Intrusion Detection Systems (IDS). For so many years, IDS have been developed for use in computer networks, they have been widely used to detect a range of network attacks; but one of its major drawbacks is that attackers, with the evolution of time and technology make it harder for IDS systems to cope. A sub-branch of IDS-Intrusion Alert Analysis was introduced into the research system to combat these problems and help support IDS by analyzing the alert triggered by the IDS. Intrusion Alert analysis has served as a good support for IDS systems for many years but also has its own short comings which are the amount of the voluminous number of alerts produced by IDS systems. From years of research, it has been observed that majority of the alerts produced are undesirables such as duplicates, false alerts, etc., leading to huge amounts of alerts causing alert flooding. This research proposed the reduction alert by targeting these undesirable alerts through the integration of supervised and unsupervised algorithms and approach. The research first selects significant features by comparing two feature ranking techniques this targets duplicates, low priority and irrelevant alert. To achieve further reduction, the research proposed the integration of supervised and unsupervised algorithms to filter out false alerts. Based on this, an effective model was gotten which achieved 94.02% reduction rate of alerts. Making use of the dataset ISCX 2012, experiments were conducted and the model with the highest reduction rate was chosen. The model was evaluated against other experimental results and benchmarked against a related work, it also improved on the said related work.**

***Keywords*—Alert, Intrusion Alert Analysis, Alert Reduction, Intrusion Detection System**

I. INTRODUCTION

Security and protection of information has been an ever-evolving process in the field of information security. One of the major tools of protection is the Intrusion detection systems (IDS). In the protection of networks today, the adoption of IDS is very crucial, vendors come up with different types of IDS and consumers purchase them in order to safely secure their networks or computers. Network owners employ the use of multiple IDS (homogenous or heterogeneous) and layers of security tools to achieve an extra mile of defense tactics, this leads to massive amount of alerts generated by IDS [1]. Due to these alerts, Network Administrators in charge of the networks or Security Analyst have found it difficult to sift through these huge number of alerts and the quality of the alert being received to determine the strategy or predict when an attack is underway.

Intrusion alert analysis is a very active area of research in the field of intrusion detection. The intrusion alert analysis system provides an overview of the possible intrusion attempts through the selection, aggregation, correlation and joint analysis of the alerts produced by distributed sensors installed at various locations in an organization [2]. Many research papers on intrusion alert analysis have been published in recent years [3], [4], [5], [6], leading to the enrichment of the field's literature. Different problems in IDS gave rise to the concept of intrusion alert analysis such as alert flooding [7], false-positives alert [8], non-relevant alert [9], unable to detect multistep attacks (isolated alerts) [10], [6], etc. Previous researchers analyzed the alerts in different ways and used them to better understand attack strategies and so on, examples are Attack step recognition (Alhaj, 2018), Alert reduction [11], Attack scenario construction [12], Alert correlation [13] etc.

As stated above intrusion alert analysis resulted from various IDS problems but one in particular would be researched into which is alert flooding. Alert flooding is when IDSs produce an unmanageable number of alerts that overwhelm security analysts, cost high computational overhead and waste system resources. Imprecise incident description, network incompatibility, and often a range of actual intrusions or illegal behaviors or malicious attackers, which appear to mislead the device supervisor from the main attack or attack target, may be the main reasons for this large number of alerts [7]. The need to reduce the impact of alert flooding on alert analysis is needed. The following section would go as follows section II would look into existing works, section III talks about the supervised and unsupervised algorithms used in the research, section IV talks about proposed methodology, section V talks on the experimental results and discussion and VI concludes.

II. LITERATURE REVIEW

This section starts with a general background into network security and its entails such as defending and monitoring systems and networks. This section investigates on one of the defense mechanisms which is the Intrusion Detection Systems (IDS); which is used to secure, monitor and defend the network and the problems plaguing it. The section then discusses an area which is found within IDS- Intrusion Alert Analysis, followed by a review into the core of alert analysis.

A. Intrusion Detection

As complex attacks have rapidly increase with each passing minute, the need for a means to detect these activities rapidly and effectively is needed, which brought about the use of IDSs. The works of Anderson (1980) and Denning (1987) were the first research into IDS. Anderson's research looked more into internal attacks of a network, while Denning developed a methodological framework that monitored the abnormal usage of a system. This led to the categorization of IDS based on what they monitor (host or network) and their techniques for monitoring (signature or anomaly). As more IDSs are developed, security administrators are faced with the challenge of analyzing a growing number of alerts arising from the study of multiple sources [14]. With the alerts produced by IDS it is possible to see the big picture of how each alerts received are connected and the attacks are conducted, which helps network administrators prepare for an attack before it happens by analyzing these alerts.

B. Intrusion Alert Analysis

Intrusion alert analysis is a very active area of research in the field of intrusion detection. It helps network administrators and security analyst detect attacks faster and better through the recognition of intrusion plans and strategies. Previous researchers analyzed the alerts in different ways and used them to predict attacks, examples are Alert correlation, Attack step recognition, Alert reduction, Attack scenario construction, etc.

The huge number of alerts in IDS can be challenging. This leads to alert flooding we focus on the disadvantages which is alert flooding where majority of the percentage of alerts are not true alerts [15], [16] e.g., false positives, redundant, etc. This makes analysis of the alerts less efficient and the quality of alerts low. Alert flooding can also come in the form of an attack where the sensor storage becomes full thus preventing further logging and the senor exceeds maximum alert throughput causing alerts to be lost or sensor malfunction [11].

Through the years, the reduction of alerts in alert analysis have been achieved by various techniques employed by different authors. Although the volume of alert is needed in making sense of the single isolated low-level alert, the quantity for alert analysis usually contain undesirable alerts which reduce the quality of a correlation or aggregation. Through this the alerts need to be cleaned so to speak to ensure proper analysis. Intrusion alert analysis resulted from various IDS problems but one in particular would be researched into which is alert flooding. Alert flooding is when IDSs produce an unmanageable number of alerts that overwhelm security analysts, cost high computational overhead and waste system resources. Imprecise incident description, network incompatibility, and often a range of actual intrusions or illegal behaviors or malicious attackers, which appear to mislead the device supervisor from the main attack or attack target, may be the main reasons for this large number of alerts [7]. The following sections would be looking into authors that have reduced alert.

1) Reduction Through False Positives

False positives have become daunting as network traffic rises. Several research and analysis have currently found that almost 99% of the alerts recorded by an intrusion detection system are not related to security issues [17], [18]. Literatures of alert analysis often time look for ways to make the quality of alerts for correlation or aggregation better by removing false positives.

The authors in [19], proposed a novel IDS alert correlator for Snort called EDGe, it uses statistical measures to find hosts that exhibit a repeated malicious multi-stage footprint which also detects malware family and variant. False positives were removed here through the EDGe algorithm. The downside of this work is that the authors designed the correlator to have small quantities of false positives with large quantities of false negatives. They decided to make a trade-off between false positives and false negatives.

In the thesis of [7], she proposed an effective Attack scenario construction to discover complete alerts relationship through identifying attack steps, attack stages and construction attack scenarios. These were achieved through two-tier feature selection and coarse grain cleaning. The first phase of the model aimed to identify attack steps, to achieve this, the use of three data mining techniques called Agglomerative Hierarchical clustering commonly known as Hierarchical

clustering, EM and Kmeans was used to group significant features from the originally gotten features and the generic features which is a subset of the original. Hierarchical clustering performed better as compared to the others.

2) Reduction through Duplicates, Redundant and Repetitive Alerts

The reduction of alerts can also be achieved through the removal of duplicates alerts, redundant alerts and repetitive alerts. This method also helps improve the quality of alerts and reduce alert data. Authors in [20] proposed a Purpose-Oriented Maximum Attack Sequence Pattern (PMASP) framework which uses frequent sequence mining for threat detection in alert correlation. Before correlation takes place, the authors perform pre-processing of the alert by removing redundant alerts based on time, alarm number (sid number), source IP and Destination IP with the simple rule of "If within 1 minute, there exist several alarm records with the same sid number, srcIP and desIP, we only retain the original one." After the removal the alerts are then sent to other components of the algorithm to dig out attack sequence.

Another method is through atomicity, which was brought forth by Zhang, Zhao, Luo, Xin & Zhu (2019). In their work they proposed a framework called Intrusion Action Based Correlation Framework (IACF) which aimed to improve alert aggregation and correlation. The framework uses a novel grouping method based on the concept of alerts having a strong intrinsic correlation among themselves which is called Atomicity. To better explain Atomicity better, they gave an example of a situation where a packet matches more than one signature which would trigger multiple alerts and another situation can have one attack with encapsulated instructions located in different packets which also triggers multiple alerts. It can be said that these two alerts have an "Intrinsic Correlation". Their work achieved a form of alert reduction through pruning of redundant actions and single attacks, which is believed reduces false positives.

3) Reduction of Alerts through Other Forms

The work of [12] developed an approach to support security management in alert correlation. Their work revolves around two steps, Offline Correlator, and the Online Correlator, which correlates historical alerts to recognize attacker's strategy and then association of upcoming alerts in real time according to the strategies revealed in the first step. A collection of IDS alerts is obtained as input by the offline correlator and it builds the cluster model of attack tactics to be used in the online correlation. By matching these alerts with the cluster model that the offline correlator developed, the online correlator analyzes the incoming alerts in real time and extracts useful information. In the offline correlator, the aggregation step creates connected components and the components that represent the exceptional situations are dropped. For a connected component to be represented as such they need to either contain only one alert or contain only alerts with the same signature. Their work focused more on the response time of the approach to a security event.

From the works above it can be seen that most of the related works focused mainly on filtering false alerts although only few based their works on reduction, it can be seen in their work alerts were filtered along the way. The research direction to focus alert reduction not only through false alerts filtration but to target other types of alerts as shown in the table above, using supervised and unsupervised algorithms.

III. UNSUPERVISED AND SUPERVISED ALGORITHMS

Self-Organising Map (SOM): SOM is an unsupervised neural network that generates a feature map that keeps the input data's topology based on their similarity. The work of Kohonen (1995) explains the underlying concept, architecture, and implementation approach of SOM. SOM-based unsupervised learning is a quick and easy approach to cluster data sets. When compared to other learning approaches, SOM is best suited to data classification because of their high speed and fast conversion rates [21]. This method is also thought to outperform other algorithms in terms of data representation due to its ability to preserve topological mappings between the input data. The use of SOM would be implemented for aggregating low-level alerts.

K-Means: Kmeans is a basic unsupervised learning technique that solves the well-known clustering problem by partitioning n objects into k segments depending on their properties, where $k < n$. The K-means algorithm implies that all attributes are independent and regularly distributed [22]. The primary idea behind this method is to identify k acceptable centroids, one for each cluster, and then arrange all data into the k subsets that have already been formed. The sum of distances or sum of squared Euclidean distances from the mean of each cluster is used to group them

X-Means: X-means clustering is a type of k-means clustering that refines cluster assignments by repeatedly trying subdivision and maintaining the best splits until a criterion such as the Bayesian Information Criterion (BIC) is met [23]. The Bayesian Information Criterion (BIC) or Schwarz Information Criterion (also SIC, SBC, SBIC) is a model selection criterion that favors the model with the lowest BIC among a finite number of models. It was developed by [23]. Xmeans was also used in the research to create an environment for no biases and for comparison of techniques performed better. Another reason it was used was due to it automatic method of choosing its clusters unlike Kmeans.

Classification Via Clustering: This technique merges the capabilities of both supervised and unsupervised machine learning techniques to classify data. The package under WEKA called Classification Via Clustering was adopted in this work. The package was created by Peter Reutemann, it is a meta classifier that falls under the Classification segment. In it, various number clustering algorithms can be selected for classification. It finds the best single cluster for each class. Weka is used to evaluate the classes-to-clusters relationship. The remaining clusters are left unlabeled, and a test instance

assigned to one of the unlabeled clusters is left unclassified. The version used was version 1.0.8. The clustering technique that was used to classify was Kmeans and EM. SVM and Naïve Bayes were chosen as classifiers to eliminate any form of biasness for the results of the classification via clustering. It was introduced into the research as a comparison element against the performance of the approach of classification via clustering.

Support Vector Machine (SVM): SVMs are supervised learning models that analyze data for classification and regression analysis with related learning algorithms. It was developed by Vapnik with peers at AT&T Bell Laboratories [24], [25].

Naïve Bayes: The naive Bayes classifier is a simple classifier which uses probability based on Bayes' theorem and strong (naïve) independence beliefs between features. They are one of the most basic Bayesian network models [26]. By assuming that characteristics are independent of class, the naive Bayes classifier dramatically simplifies learning [27].

Expectation Maximization (EM): In statistical models with unobserved latent variables, EM is an iterative method for finding (local) maximum likelihood or maximum a posteriori (MAP) estimates of parameters. Each instance is given a probability distribution by EM, which reflects the likelihood of it belonging to one of the clusters.

The idea of merging supervised and unsupervised algorithms was inspired from the work of [22] where they integrated the use of SOM algorithm and Kmeans to classify and cluster alerts for reduction.

IV. PROPOSED METHODOLOGY

Regarding the problem of alert flooding, this section provides an overview of the phases that would be used to achieve the solution concepts which are to target undesirable alerts such as false alerts, duplicates, low priority, etc. by the integration of supervised and unsupervised algorithms and techniques to reduce alerts through the removal of irrelevant and duplicate alerts and propose of alert filtration process to filter out false alerts. The research framework was divided into 3 phases. The first phase is data pre-processing and feature selection, the second phase would be data aggregation and reduction which would be divided among the removal of duplicates and false alerts and last phase would be evaluation of the research work.

A. ISCX2012 Intrusion Detection Evaluation Dataset

The dataset was developed by the University of New Brunswick's (UNB) Information Security Centre of Excellence (ISCX) lab under the research group Canadian Institute for Cybersecurity (CIC) in 2012. It contains 7 days of network activity 11th June 2010 till 17th June 2010, both normal traffic and malicious. This dataset is one of the more recent and appropriate datasets for alert analysis, although it was not developed for the sole purpose of evaluating alert analysis, but it can be used for it. This can be seen in the works of authors who have used this dataset, example [28], [29], [30], [31]. According to [28] reported that this dataset contains a larger network traffic compared to other existing benchmark IDS datasets.

Due to the size of the data, this research made use of a subset of it which was gotten from the work of [7]. The days 13th -14th June was used which are Sunday, Monday and Tuesday as shown in the figure above. The Sunday alerts contained a total of 23,834 alerts (rows) with 17 columns. The Monday alerts contained 28,950 alerts (rows) with 16 columns. The Tuesday alerts contained 70,318 alerts with 17 columns. These alerts were derived after the replaying the datasets through Snort which sums up to 90024 alerts and 17 columns.

B. Details of Research Framework

This section will discuss each phase of the research framework and how each of the phases will be achieved and process to achieve them. The first phase of the framework deals with data pre-processing which removed majority of the irrelevant alerts and feature selection, which would cover some conversion in the dataset going to be used, while the second phase would be alert reduction and aggregation and the third phase would be evaluation.

1) Phase 1: Data Pre-processing and Feature Selection

This phase contains the process data pre-processing and feature selection activities. The purpose of the pre-processing component is to supply missing alert attributes that are critical for use, as accurately as possible. This component would handle null values, missing and incomplete data. Data pre-processing has been known to contain stages such as data scaling, data representation [32], etc. This phase also deals with alert representation which includes date conversion which removes delimiters colons and slashes from dates, IP address conversion which involves the use of Eq. 3.1 to convert the IP addresses into a proper format for processing.

$$IP = \Big(\big(((X_1 \times 256) + X_2) \times 256\big) + X_3\Big) \times 256 + X_4 \quad (3.1)$$

The alert representation also used alert scaling to make the processing of the alerts easier. Using Eq. 3.2 as shown below.

$$X' = 0.8 \times \frac{(x - x_{min})}{(x_{max} - x_{min})} + 0.1 \quad (3.2)$$

Where: X' = the new value, x = raw value, x_{min} = the minimum value and x_{max}= the maximum value.

While the feature selection selected appropriates and significant attributes for the experiments. This phase investigated the use of feature selection to reduce the amount of dataset for simplification and easy and faster computation. Feature selection can be carried out through different techniques such as the wrapper method [33], filter method [34] and embedded method [35]. This research makes use of the one of the methods under filter method which is called ranking. It is the process of ranking features according to the value of a

scoring function, which is commonly used to determine feature importance [36]. Ranking features provide the opportunity for researchers to filter out less important features that would affect the productivity, results and analysis any work.

2) Phase 2: Alert Reduction and Aggregation

This phase handles the aggregation of alerts using clustering techniques and the reduction of the aggregated alerts. The aggregation correlated similar or closely related alerts low-level alerts together while the reduction helps reduce the volume of aggregated alerts. This phase drops the low priority alerts in the data as a means to reduce alerts. As the dataset ISCX 2012 is already has a feature called 'Priority' it is easier to drop such alerts based on the priority of the alerts. This phase further reduces the alerts by dropping out duplicates. The duplicates are alert rows that have the same information which makes one of the alerts a duplicate of the order. These alerts were produced as a result of the output from the IDS which was used to replay it. The duplicates were removed so as it would not affect the result analysis of the research. These was made possible by using the application called WEKA. Weka is free software that was developed at the University of Waikato in New Zealand.

As this research falls under the aspect of alert correlation, one of the methods of correlation involves the aggregation of alerts. Aggregation as seen in many works [37], [12], [38] is used to cluster alerts of similar low-level alerts for better processing and faster computation time. This research makes use of clustering techniques which authors such as [39] to group related alerts together for further analysis. This research makes the correlation technique in the work of [22]. In their work, they developed a two-stage classification system using a SOM neural network and K-means algorithm to correlate the associated alerts and to further classify the alerts into classes of true and false alarms. Their work make use of the SOMToolbox developed by CIS [40]. This research does not make use of this toolbox, but still implement the tools (SOM and Kmeans) as the paper did.

The aggregated alerts are then further processed for further reduction by removing the false positives. To eliminate false positives, the use of classification was adopted. The data was classified into true and false alerts to filter out the false alerts. The following techniques were used:

- Classification Via Clustering (CvC): This technique merges the capabilities of both supervised and unsupervised machine learning techniques to classify data. The package under WEKA called Classification Via Clustering was adopted in this work. The package was created by Peter Reutemann, it is a meta classifier that falls under the Classification segment. In it, various number clustering algorithms can be selected for classification. It finds the best single cluster for each class. Weka is used to evaluate the classes-to-clusters relationship. The remaining clusters are left unlabeled, and a test instance assigned to one of the unlabeled clusters is left unclassified. The version used was version 1.0.8. The clustering technique that was used to classify was Kmeans and Xmeans.
- Conventional Classification: Conventional classifiers are the already known classifiers such as SVM, Naïve Bayes, etc. These classifiers are introduced to the research to create an environment for comparison with the CvC and to eliminate any biasness or technique overfitting involved.

3) Phase 3: Evaluate and Benchmark

After experiments of different algorithms and techniques, the approach would be evaluated based on the metrics that would be discussed in the next sub-section. It would also be benchmarked against the techniques of the work of [22] as this was where the idea of the research was coined from.

TABLE IV-1. EVALUATION METRICS

Performance Measure	**Description**
Accuracy	Correctly classified as a legitimate transaction and fraudulent transaction $\frac{TP+TN}{TP+FP+TN+FN}$
True Positive Rate (TPR)	Measure the frequency of correctly predicted transaction of the model as normal $\frac{TP}{TP+FN}$
False Positive Rate (FPR)	Calculates the rate of incorrectly predicted alert $\frac{FP}{FP+TN}$
Precision	The ratio of positive occurrences correctly predicted to the total of positive observations predicted $\frac{TP}{TP+FP}$
F measure	The weighted average of recall and precision $F1-Score=\frac{2TP}{2TP+FP+FN}$
Reduction Rate	This measure would calculate the overall reduction in alerts. $\frac{(OD\ -ND)}{OD}\times 100$ OD = Original data size ND =New Data size OD – ND = Reduction
TP = True positives (properly categorized positive cases) TN = True negatives (properly categorized negative cases) FP = False positives (negative cases categorized as positives) FN = False negatives (positive cases categorized as negative)	

Evaluation of the study was on overall reduction of the alerts, the reduction based on the ranked features, the classification accuracy, number of false positives, true positives and error rate as shown in the Table III-1.

V. EXPERIMENTAL RESULTS AND DISCUSSION

This section explains how the experiments were conducted. The experiments in this research were carried out in five parts. The first 3 experiments serve as basis for the last two experiments. Experiment 1 - 3 contain the following phases: Pre-processing, Feature

Selection (Feature Ranking), Reduction through Low Priority, Reduction through Duplicates, Alert Aggregation and Reduction through Filtration. While experiment 4 and 5 only focus on Reduction through Filtration using the results from Experiment 1 – 3's results from the Alert Aggregation phase. Then result for each experiment would be given and a comparison amongst five experiments would be made. Due to space limitations, only the highest rate reduction of each experiment would be presented in this paper.

In experiments 1 -3, the data was preprocessed, the features were ranked using Information Gain and Gain ratio to select significant ones, low priority and duplicates were removed using WEKA, alerts were aggregated using the integration of supervised algorithms SOM + Kmeans and Xmeans and the filtering of false alerts was done using unsupervised algorithms CvC (Kmeans), CvC (EM) SVM and Naïve Bayes. The steps such as feature ranking and false alerts filtering had their results evaluated against each other to pick the best.

A. Experiment 1

From the original data, which was 90,024 alerts, after pre-processing, 82,815 alerts were passed onto the next stage which was feature ranking. After the features were ranked using Information Gain, the selected features were 'Classification', 'SigID', 'DgmLen', 'Source port', 'ScrIP', 'UID', 'TTL', 'Destination Port', 'Destination IP', 'Priority', 'Protocol' and 'Timestamp'. The data was passed to remove low priority alerts which resulted into 53,902 alerts remaining after the low priority alerts were dropped. Next the alerts were passed to rid the data or duplicate rows but due to its 'UID' being present not alert rows were considered duplicates, so no reduction occurred. The alerts were then aggregated and passed to be filtered by true or false. Picking the best classifier from the classifiers experimented on above which was Classification Via Clustering - Kmeans (Xmeans), a total of 12,228 alerts were left. It was determined that this experiment achieved a rate of reduction of 86.4%.The results of experiment 1 are shown below:

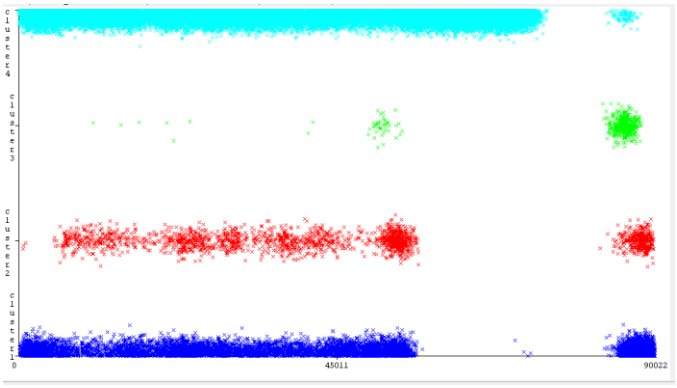

Fig. V-1. Result of Xmeans Aggregation on UID

TABLE V-1 EXPERIMENT 1 CONFUSION MATRIX

a	b	<---- Classified as
6963	5689	a = T
15606	25643	b = F

TABLE V-2. EXPERIMENT 1CLASSIFICATION MATRIX

TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy	Class
0.550	0.378	0.309	0.550	0.395	76.53%	T
0.622	0.450	0.818	0.622	0.707	23.74%	F

B. Experiment 2

From the original data, which was 90,024 alerts, after pre-processing, 82,815 alerts were passed onto the next stage which was feature ranking. After the features were ranked using Information Gain but dropping the 'UID' feature, the selected features were 'Classification', 'SigID', 'DgmLen', 'Source Port', 'Source IP', 'TTL', 'Destination Port', 'Destination IP', 'Priority', 'Protocol' and 'Timestamp'. The data was passed to remove low priority alerts which resulted into 53,902 alerts remaining after the low priority alerts were dropped. Next the alerts were passed to rid the data or duplicate rows which led to the alert to be reduced to 42,239 alerts. The alerts were then aggregated and passed to be filtered by true or false. Picking the best classifier from the classifiers experimented on above which was Classification Via Clustering - Kmeans (Xmeans) with correctly classified alerts of 82.1208%, a total of 5,947 alerts were left. It was determined that this experiment achieved a rate of reduction of 93.39%. The results of experiment 2 are shown below:

TABLE V-3. EXPERIMENT 2 CONFUSION MATRIX

a	b	<---- Classified as
5947	1707	a = T
5845	28740	b = F

TABLE V-4. EXPERIMENT 2 CLASSIFICATION EVALUATION

TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy	Class
0.777	0.169	0.504	0.777	0.612	81.88%	T
0.831	0.223	0.944	0.831	0.884	18.12%	F

C. Experiment 3

From the original data, which was 90,024 alerts, after pre-processing, 82,815 alerts were passed onto the next stage which was feature ranking. After the features were ranked using Gain Ratio, the selected features were 'Classification', 'Priority', 'Protocol', 'SigID', 'Source Port', 'TTL', 'Destination IP', 'Source IP' and 'Destination Port'. The data was passed to remove low priority alerts which resulted into 53,902 alerts remaining after the low priority alerts were

dropped. Next the alerts were passed to rid the data or duplicate rows which led to the alert to be reduced to 35,641 alerts. The alerts were then aggregated and passed to be filtered by true or false. Picking the best classifier from the classifiers experimented on above which was Classification Via Clustering - Kmeans (Xmeans) with correctly classified alerts of 82.8737%, a total of 5,436 alerts were left. It was determined that this experiment achieved a rate of reduction of 93.96%. The results of experiment 3 are shown below:

TABLE V-5. EXPERIMENT 3 CONFUSION MATRIX

a	b	<---- Classified as
5436	50	a = T
6054	24101	b = F

TABLE V-6. EXPERIMENT 3 CLASSIFICATION EVALUATION

TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy	Class
0.991	0.201	0.473	0.991	0.640	84.61%	T
0.799	0.009	0.998	0.799	0.888	15.39%	F

D. Experiment 4

This experiment was conducted with the use of conventional classifiers for the phase of Reduction through filtration. It was conducted this way to test the strength of the technique Classification via clustering against normal classifiers. It was also conducted to avoid biasness in techniques. The conventional classifier used in this experiment was SVM. This experiment was conducted using the output of experiment's 1 – 3 Alert Aggregation. The following sections show the results of the experiments.

Using SVM as a classifier, taking the best classification results from the prior experiments done on experiment 1's Xmeans with 94.7107% correctly classified alerts, experiment 2's Xmeans with 99.9953% correctly classified alerts and experiment 3's SOM + Kmeans with 99.9944% correctly classified alerts, experiment 2's classification results were picked as the best. But in terms of reduction in experiment 4, experiment 1's Xmeans achieved a rate of 88.29% reduction, experiment 2's Xmeans achieved a rate of 91.49% reduction and experiment 3's SOM + Kmeans achieved a rate of 93.90%. This leaves experiment 4's classification while using experiment 3's alert aggregation result of SOM + Kmeans with the highest rate of reduction.

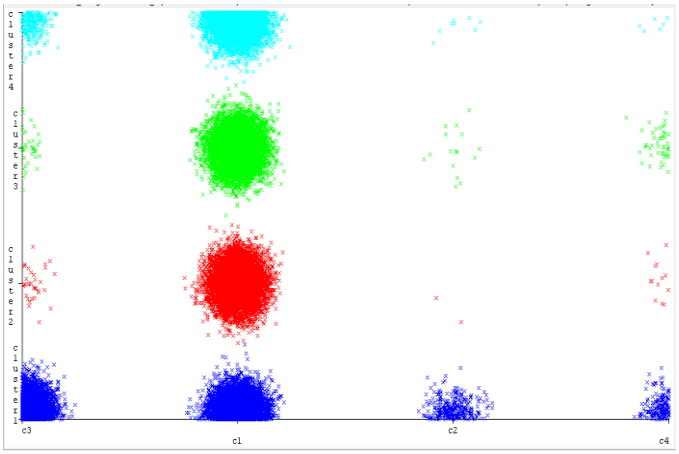

Fig. V-2. Experiment 4 Using Experiment 3 Result of SOM + Kmeans Aggregation Clusters

TABLE V-7. EXPERIMENT 4 CONFUSION MATRIX

a	b	<---- Classified as
5486	0	a = T
2	30153	b = F

TABLE V-8. EXPERIMENT 4 CLASSIFICATION EVALUATION

TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy	Class
1.000	0.000	1.000	1.000	1.000	84.61%	T
1.000	0.000	1.000	1.000	1.000	15.39%	F

E. Experiment 5

This experiment was conducted with the use of conventional classifier for the phase of Reduction through filtration. It was conducted this way to test the strength of the technique Classification via clustering against normal classifiers. It was also conducted to avoid biasness in techniques. The conventional classifier used in this experiment was Navie Bayes. This experiment was conducted using the output of experiment's 1 – 3 Alert Aggregation. The following sections show the results of the experiments.

Using Naïve Bayes as a classifier, taking the best classification results from the prior experiments done in experiment 5 among experiment 1's SOM + Kmeans with 99.8145% correctly classified alerts, experiment 2's SOM + Kmeans with 99.7254% correctly classified alerts and experiment 3's SOM + Kmeans with 99.206% correctly classified alerts, experiment 1's classification results were picked as the best. But in terms of reduction for experiment 5, experiment 1's SOM + Kmeans achieved a rate of 85.94 reduction, experiment 2's SOM + Kmeans achieved a rate of 91.51% reduction and experiment 3's SOM + Kmeans achieved a rate of 94.02%. This leaves experiment 5's classification while using experiment 3's alert aggregation result of SOM + Kmeans with the highest rate of reduction.

TABLE V-9. EXPERIMENT 5 CONFUSION MATRIX

a	b	<---- Classified as
5377	109	a = T
174	29981	b = F

TABLE V-10 EXPERIMENT 5 CLASSIFICATION EVALUATION

TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy	Class
0.980	0.006	0.969	0.980	0.974	84.61%	T
0.994	0.020	0.996	0.994	0.995	15.39%	F

Having conducted the series of experiments, the best performer for each experiment 1 according to overall rate of reduction is shown below:

TABLE V-11. OVERALL RATE OF REDUCTION FROM EXPERIMENTS 1 – 5

Experiments	**Model Structure**	**Overall Rate of Reduction**
1	Information Gain 1 (12), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	86.4%
2	Information Gain 2 (11), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	93.39%
3	Gain Ratio (9), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	93.96%
4	Gain ratio (9), Low Priority, Duplicates and SOM + Kmeans and SVM	93.90%
5	**Gain Ratio (9), Low Priority, Duplicates and SOM + Kmeans and Naïve Bayes**	**94.02%**

From the table above, this research evaluates on how its steps contributed to reducing alerts. The major steps in each of the experiments that had effects on the alert's reduction were the feature ranking reduction through priority, reduction through duplicates and reduction through filtration. From Table V-11, the number of features used are in parentheses and the steps in the experiment are listed alongside their overall rate of reduction. The model that achieves the highest rate of reduction is experiment 5 which uses Gain Ratio to select significant features, dropped low priority, dropped duplicates, aggregated the alerts using SOM + Kmeans and filtered out false alerts using Naïve Bayes. It was also deduced that the use of Gain Ratio which resulted in lesser features and the removal of UID contributed greatly to the reduction of alerts as Gain Ratio was able to pick the most significant features necessary for experimentation and the removal of UID helped eliminate duplicate alert rows.

F. Benchmarking

This research was benchmarked against the work of [22]. To benchmark against the work of [22], the experiment was checked against the rate of reduction gotten from their technique, as it was this technique that was used to construct the experiments of this research. Their work made use of a chunk of 1999 DARPA, with a total of 3,062 alerts. In comparison to the alerts of 90,024 alerts used in this research it is quite small, but their technique was adopted to be used on the dataset to see if it can reduce alerts. Benchmarking against the work of [22], we compared against the correctly classified alerts. As their work used a chunk 1999 DARPA dataset totalling 3,062 alerts. Their correctly classified alerts in DARPA part 1 was 95% while PART 2 was 99%.

TABLE V-12. BENCHMARKING WITH REFERENCED MODEL

	Method	Reduction Rate
[22]	**Hand-Picked Attributed (3), SOM + Kmeans and Kmeans**	**79.29%**
Experiment 1	Information Gain 1 (12), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	86.4%
Experiment 2	Information Gain 2 (11), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	93.39%
Experiment 3	Gain Ratio (9), Low Priority, Duplicates and Xmeans and Classification Via Clustering - Kmeans	93.96%
Experiment 4	Gain ratio (9), Low Priority, Duplicates and SOM + Kmeans and SVM	93.90%
Experiment 5	**Gain Ratio (9), Low Priority, Duplicates and SOM + Kmeans and Naïve Bayes**	**94.02%**

As seen above, the work of [22] did not address other undesirable alerts such as low priority, duplicate alerts and irrelevant alerts, they only focused on false alerts. This research focused on these alerts which serves as an addition to the benchmarked work. Also looking at the techniques used it can be seen from the benchmarked work that the use of handpicked features can be avoided if the use of feature raking tools such as Gain Ratio is used, this in turn eliminates biases from the path of the researcher. It is also observed that the power of conventional classifiers such as SVM, have more power in classification than the use of the technique of classification via clustering.

Although they worked achieved a high classification, researchers strive to go for the best in terms of results, which allows the conventional classifiers to come out on top.

VI. CONCLUSION

Alerts produced by IDS have been very crucial in the prevention of future attacks, prediction of oncoming attack and the correlation of precious attacks to future ones through the use of alert analysis carried out by various researchers. Although, the number of alerts produced by a single IDS system can be a daunting task for a security analyst to sift through much less multiple IDS systems. These alerts come in large numbers overwhelming the security analyst and taking up resources. Research have searched for ways to deal with this task by coming up with various techniques for reduction of

alerts. This research also focuses on the reduction of those alerts but in terms of what these alerts contain. Through research it was discovered that the alerts produced for analysis contain alerts that waste security analyst time and consume valuable resources such as time, hardware, computational time and overhead. This research aimed to look at what the types of alerts contained in these alerts, separate them and deal with them through reduction so as to allow security analyst focus on the high-risk alerts.

The research involved the use of Information Gain and Gain Ratio to select significant features, the use of unsupervised algorithms such as SOM, Kmeans and Xmeans to aggregate and classify alert and the use of supervised algorithms such as EM, SVM, Naïve Bayes and Classification via Clustering to classify alerts. After a series of experiments, the best model that produced the highest reduction rate was chosen which was the use of Gain Ratio to select feature, followed by the dropping of duplicates and low priority alerts, use of SOM + Kmeans to aggregate alerts and the use of Naïve Bayes to separate false alerts from true alerts.

ACKNOWLEDGMENT

This project report is dedicated to my mum and dad, for always being there for me, through my lows and my highs and for reminding me God has a purpose for me and loves me.

REFERENCES

[1] X. He, J. Wang, J. Liu, L. Han, Y. Yu and S. Lv. (2018). Hierarchical Filtering Method of Alerts Based on Multi-Source Information Correlation Analysis. *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, Hangzhou.

[2] A. A. Ramaki and R. E. Atani. (2016). A Survey of IT Early Warning Systems: Architectures, Challenges, and Solutions. *Security and Communication Networks*, 4751-4776.

[3] N. Hunnalli and V. Suryanarayananx. (2014). False Alarm Minimization Techniques in Signature-based Intrusion Detection. *Computer Communications*, 1-17.

[4] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser and M. Fischer. (2015). Taxonomy and Survey of Collaborative Intrusion Detection. *ACM Computing Surveys*, 1-33.

[5] R. Zuech, T. M. Khoshgoftaar and R. Wald. (2015). Intrusion Detection and Big Heterogeneous Data: A Survey. *Journal of Big Data*, 1-41.

[6] M. Chengpo, H. Houkuan and T. Shengfeng. (2006). A Survey of Intrusion-detection Alert Aggregation and Correlation Techniques. *Journal of Computer Research and Development*, 1-8.

[7] T. A. M. Alhaj. 2018. An Effective Attack Scenario Construction Model based on Two-Tier Feature Selection and Coarse Grain Cleaning.

[8] R. Sadoddin and A. Ghorbani. (2006). Alert Correlation Survey: Framework and Techniques. *Proceedings of the 2006 International Conference on Privacy, Security, and Trust: Bridge the Gap Between PST Technologies and Business Services*, New York.

[9] S. Salah, G. Macia-Fernandez and J. E. Diaz-Verdejo. (2013). A Model-based Survey of Alert Correlation Techniques. *Computer Networks*, 1289-1317.

[10] U. Zurutuza and R. Uribeetxeberria. (2004). Intrusion Detection Alarm Correlation: A Survey. *Proceedings of the IADAT International Conference on Telecommunications and Computer Networks.*

[11] G. Tedesco and U. Aickelin. (2008). Data Reduction in Intrusion Alert Correlation [Online]. Available: https://arxiv.org/ftp/arxiv/papers/0804/0804.1281.pdf.

[12] C. T. Kawakani, S. B. Junior, R. S. Miani, M. Cukier and B. B. Zarpelao. (2016). Intrusion Alert Correlation to Support Security Management. *Proceedings of the 12th Brazilian Symposium on Information Systems (SBSI)*, Florianopolis.

[13] B. L. Dalmazo, J. P. Vilela and M. Curado. (2018). Triple-similarity Mechanism for Alarm Management in the Cloud. *Computer & Security*, 33-42.

[14] F. Valeur, G. Vigna, C. Kruegel and R. A. Kremmerer. (2004). Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing*, 146-169.

[15] R. Lippmann, S. Webster and D. Stetson. (2002). *The Effect of Identifying Vulnerabilities and Patching Software on the Utility of Network Intrusion Detection*. Springer, Berlin, Heidelberg,

[16] M. B. M. S. @. Siraj, (2013). Hybrid of Structural-causal and Statistical Model for Intrusion Alert Correlation.

[17] E. Bloedorn, B. Hill, A. Christiansen, C. Skorupka, L. Talbot and J. Tivel. (2000). Data Mining for Improving Intrusion Detection. MITRE.

[18] T. Pietraszek. (2004). Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection. *International Workshop on Recent Advances in Intrusion Detection*.

[19] E. Raftopoulos and X. Dimitriopoulos. (2014). IDS Alert Correlation in the Wild With EDGe. *IEEE Journal on Selected Areas in Communications*, 1933-1946.

[20] X. Lu, J. Han, Q. Ren, H. Dai, J. Li and J. Ou. 2018. Network Threat Detection based on Correlation Analysis of Multi-Platform Multi-Source Alert Data. *Multimedia Tools and Applications*, 33349-33363.

[21] K. Labib and R. Vemuri. (2002). NSOM: A Real-time Network-based Intrusion Detection System Using Self-organizing Maps. *Networks & Security.*

[22] G. C. Tjhai, S. M. Furnell, M. Papadaki and N. L. Clarke. (2010). A Preliminary Two-stage Alarm Correlation and Filtering System using SOM Neural Network and K-means Algorithm. *Computers and Security*, 712-723.

[23] D. Pelleg and A. W. Moore. (2000). X-means: Extending k-means with Efficient Estimation of the Number of Clusters. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*.

[24] B. E. Boser, I. M. Guyon and V. N. Vapnik. (1992). A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*.

[25] H. Drucker, C. J. Burges, L. Kaufman, A. Smola and V. Vapnik. (1997). Support Vector Regression Machines. *Advances in Neural Information Processing Systems*, 155-161.

[26] A. McCallum. (2011). Bayesian Network Representa.

[27] I. Rish. (2001). An Empirical Study of the Naive Bayes

Classifier. *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*.

[28] S. S. M. Ahmed. (2014). Intrusion Alert Analysis Framework Using Semantic Correlation. Canada.

[29] A. Ammar. (2015). A Decision Tree Classifier for Intrusion Detection Priority Tagging. *Journal of Computer and Communications*, 52-58.

[30] M. H. Kamarudin, C. Maple, T. Watson and N. S. Safa. (2017). A New Unified Intrusion Anomaly Detection in Identifying Unseen Web Attacks. *Security and Communication Networks*.

[31] I. Ullah and Q. H. Mahmoud. (2017). A Filter-based Feature Selection Model for Anomaly-based Intrusion Detection Systems. *IEEE International Conference on Big Data (BIGDATA)*, Boston.

[32] H. W. H. Hua, M. M. Siraj and M. M. Din. (2017). Integration of PSO and K-Means Clustering Algorithm for Structural-based Alert Correlation Model. *International Journal of Innovative Computing*, 4-39.

[33] T. M. Phuong, Z. Lin and R. B. Altman. (2006). Choosing SNPs using Feature Selection. *Journal of Bioinformatics and Computational Biology*, 4(02): 241-257.

[34] J. Hamon. (2013). Optimisation Combinatoire Pour La Sélection De Variables En Régression En Grande Dimension: Application En Génétique Animale.

[35] E. Saghapour, S. Kermani and M. Sehhati. (2017). A Novel Feature Ranking Method for Prediction of Cancer Stages using Proteomics Data. *PLoS One*.

[36] V. Mehul. (2018). Feature Selection in Machine Learning: Variable Ranking and Feature Subset Selection Methods. 20 July 2018. [Online].

[37] G. P. Spathoulas and S. K. Katsikas. (2013). Enhancing IDS Performance through Comprehensive Alert Post-processing. *Computers & Security*, 176-196.

[38] M. Wu and Y. Moon. (2019). Alert Correlation for Cyber-Manufacturing Intrusion Detection. *47th SME North American Manufacturing Research Conference*, Penn State Behrend Erie.

[39] M. GhasemiGol and A. Ghaemi-Bafghi. (2015). E-correlator: An Entropy-based Alert Correlation System. *Security and Communication Networks*, 822-836.

[40] CIS. (2005). *SOM toolbox 2.0*.