Alerts Clustering for Intrusion Detection Systems: Overview and Machine Learning Perspectives

Wajdi Alhakami
Department of Computer Sciences, College of Computers and Information Technology,
Taif University, Taif, Saudi Arabia, KSA

Abstract—**The tremendous amount of the security alerts due to the high-speed alert generation of high-speed networks make the management of intrusion detection computationally expensive. Evidently, the high-level rate of wrong alerts disproves the Intrusion Detection Systems (IDS) performances and decrease its capability to prevent cyber-attacks which lead to tedious alert analysis task. Thus, it is important to develop new tools to understand intrusion data and to represent them in a compact forms using, for example, an alert clustering process. This hot topic of research is studied here and an understandable taxonomy followed by a deep survey of main published works related to intrusion alert management is presented in this paper. The second part of this work exposes different useful steps for designing a unified IDS system on the basis of machine learning techniques which are considered one of the most powerful tools for solving certain problems related to alert management and outlier detection.**

Keywords—Intrusion detection systems; alert clustering; taxonomy; survey; machine learning

I. Introduction

Intrusion Detection Systems (IDSs) are widely deployed into servers for data security purposes. However, these systems produce a lot of false positive alerts making the task of security analysts difficult such as taking suitable actions for them. This problem has received considerable attention from researchers given that manual intrusion alerts management is extremely fastidious and computationally expensive. Consequently, it will not be easy to manage alerts and to take appropriate actions for them. Thereby, the automation of the process of alert management is a necessity. It turns out that this problem becomes more difficult with the context of high-speed networks [1], [2]. Indeed, new severe issues related especially to the scalability, the real time constraints, and efficiency create a major challenge to the success of IDS [3], [4]. In particular, the big quantity of the security alerts leads to an expensive intrusion detection process. Evidently, the high-level rate of wrong alerts disproves the IDS performances and decrease its capability to prevent cyber-attacks that lead to tedious alert analysis task. In particular, alert clustering and outlier detection are crucial problems for taking suitable actions and for security threats understanding [5], [1], [2]. The content of this paper is a taxonomy and a survey related to the intrusion alert management. In fact, an understandable taxonomy, especially for beginner researchers, is presented. It is related to intrusion detection system with special emphasize on intrusion alert management problem. An informative description is presented for intrusion detection systems, for both misuse and anomaly-based detection, and for low- and high-level alert management (i.e. alert ranking, normalization, clustering, correlation, etc.) [6], [7], [8]. The second part of this paper is dedicated to expose how to design a possible IDS framework using machine learning techniques which are considered one of the most powerful tools for solving certain problems related to alert management and outlier detection. The rest of this paper is organized as follows. In the next section, a taxonomy related to the current area of research is presented. Section 3 is devoted to survey main existing works in the literature. Section 4 describes how can machine learning techniques be involved and considered as an interesting alternative to deal with such problem. Finally, we conclude the paper in the last section.

II. Taxonomy

This section presents a comprehensible taxonomy, especially for beginner researchers, for intrusion detection systems (IDS) and intrusion alert management problems. In particular, informative descriptions are given for intrusion detection techniques, alert management, alert clustering/classification and alert correlation.

1) Intrusion Detection Systems (IDSs): Intrusion detection systems are widely deployed into both hosts and networks to protect assets. To ensure security, most of developed IDSs apply mainly the so-called misuse-based (named also signature-based) or anomaly-based IDS techniques [9]. The key purpose of these techniques is to help the security administrator to fully recognize what the IDS is doing. Fig. 1 illustrate main intrusion detection techniques.

- **Misuse (Signature)-based detection**

Misuse-based intrusion detection techniques have been proven to detect effectively attacks without generating a great number of false alarms. For this reason, they are broadly approved in the most commercial systems. Such technique can be used to detect known attacks, so, we have to create signatures (patterns) for known attacks and store them into databases as a priori information. However, this kind of approach cannot detect unknown attacks and therefore they must be frequently updated with signatures of new attacks.

- **Anomaly-based detection**

Anomaly-based approaches (named behaviour-based) are used to detect unknown (novel or irregular) and known attacks on the basis of their profiles or statistical models. These models employ labeled data to model and train anomaly detection as a classification problem. This kind of approach try to find behavior (attack behavior) deviating from normal one. In general, techniques driven from pattern recognition (parametric

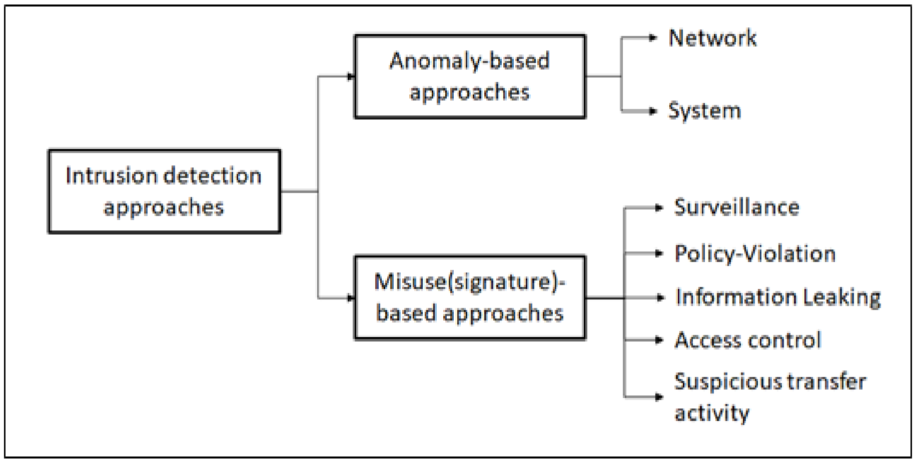


Fig. 1. Classification of intrusion detection approaches.

and non-parametric statistical models, neural networks, rule-based algorithms, hidden Markov model, etc.) are applied to identify normal data and abnormal (anomalous) ones. For example, the user can be notified about the existing of unusual behavior if the network activity deviates from its normal state. If normal behavior is well trained and well modeled, unusual behavior can be labelled and identified as intrusive. Thus, normal behavior should be well defined otherwise, if it is not, a lot of false alerts will be generated. Thus, the modeling problem, which is a difficult task, is very important for the design of anomaly-based approach. Finally, when compared to misuse-based approaches, anomaly-based ones are more efficient and faster, but, they generate a lot of false positive rate.

2) Intrusion detection alert management: Alert management function address two different processing: low-level processing and high-level processing (Fig. 2). Each level is able to accomplish specific objectives such as alert pre-processing, alert correlation, alert fusion, etc. Through management function, alerts are processed in order to help the security administrator to understand what the IDSs are addressing.

- **Low-level alert management**

The main purpose of low-level alert management is to facilitate further high-processing. It is required to achieve low processing such as updating alert attributes to a standardized format, scoring and prioritizing alerts, translating all alert's attributes into numerical values, etc. For instance, ranking alerts is required to mark and select only significant alerts for further investigation. In the literature, only few works have been proposed to deal with the importance of low level alert management for the overall system evaluation [3], [4]. Authors showed the importance of this step by defining various metrics to score and prioritize alerts such as applicability, sensor status, severity, alert relationship, etc.

- **Alert Normalization**

Alert attributes vary from one IDS to another. Thus, alerts from diverse sensors are encoded with different formats. For this reason, it is important to unify all information to make easy further processing. The normalization step will convert all diversified alert attributes into an appropriate unified representation such as the well known standard “Intrusion Detection Message Exchange Format” (IDMEF). Such standard is able to provide a flexibility for any possible extension since it is based on XML language. Each alert can be represented by the IDMEF data model as shown in Fig. 3.

- **Alert scoring/ranking/prioritizing**

Ranking and prioritizing alerts is a useful tool to evaluate the relative importance of such alerts. It helps in reduction the amount of incoming alerts by quickly discard irrelevant alerts. Some developed procedures [3], [4], [10] employed mainly the following properties: the integrity, the secrecy, and the availability metrics. In general, these properties are calculated easily since they are stored in the IDS database. As output, each alert is assigned with high or medium or low score. To achieve an appropriate scoring task as proposed in [3], [4], the pseudo-code for alert scoring in Fig. 4 can be used.

- **High-level Alert Management**

Automated and intelligent high-level alert management is a potential task helping the administrator to analyze properly alerts, and to save his time and effort. Alert management can be defined as a generalization function related to a variety of specific operations like alert classification (or clustering), alert fusion (or aggregation), and alert correlation. These operations are considered crucial since they provide an abstraction of a set of alerts. In the literature several approaches were proposed to solve the problem of alert management from different angles and some of them are presented in Section 3.

- **Alert Aggregation/Fusion/Merging**

The role of the aggregation function is to group alerts having same common characteristics such as the source IP, the target IP, the type of the attack, etc. The fusion/merging function attempts to combine a group of alerts into one hyper-alert. The latter should be representative of each belongs to the same cluster (or component). Some relevant methods [6], [7], [8] were developed in this context to help the analyst to take rapidly, through one global meta-alert, appropriate action against the seriousness of the attack.

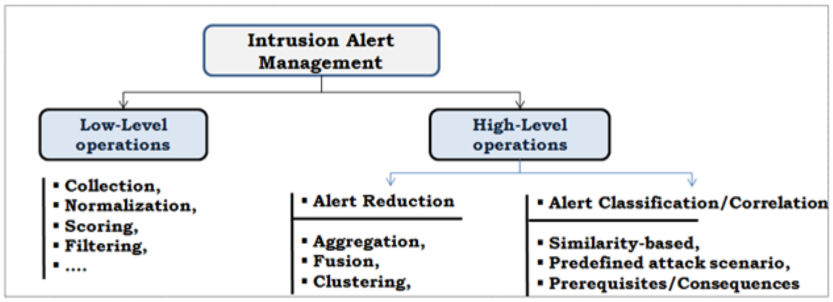


Fig. 2. Classification of Intrusion Alert Management Techniques.

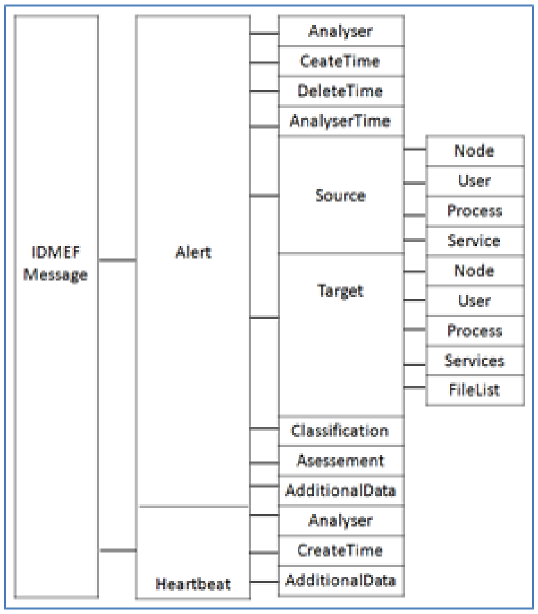
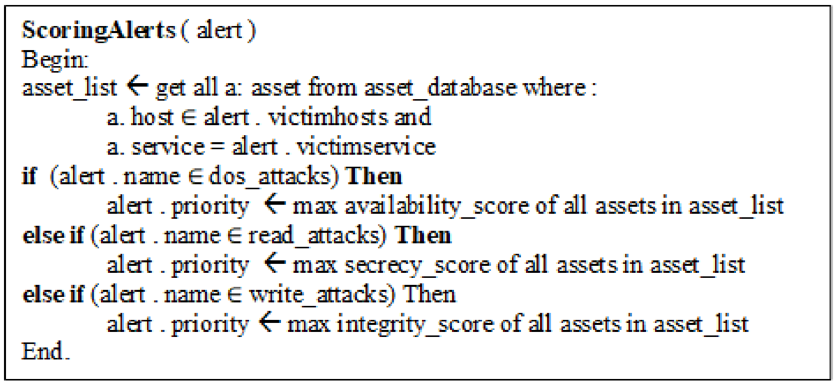


Fig. 3. The IDMEF data model.



```
ScoringAlerts ( alert )
Begin:
asset_list ← get all a: asset from asset_database where :
        a. host ∈ alert . victimhosts and
        a. service = alert . victimservice
if (alert . name ∈ dos_attacks) Then
        alert . priority ← max availability_score of all assets in asset_list
else if (alert . name ∈ read_attacks) Then
        alert . priority ← max secrecy_score of all assets in asset_list
else if (alert . name ∈ write_attacks) Then
        alert . priority ← max integrity_score of all assets in asset_list
End.
```

Fig. 4. Pseudo-code for alert scoring.

- **Alert Clustering/Classification**

This function attempts to cluster or classify alerts that match the same attack occurrence and share common features like source/target IP address or port number. Clustering output leads to reduce the number of alerts. Several criteria, as presented in [6], can be defined to achieve this task as a ”similarity relation” to connect alerts to specific cluster.

- **Alert correlation**

Alert correlation is used to discover any relationship between alerts in response to launched attacks in order to achieve a final goal. Some criteria [3] are defined to achieve this and provide different ways to study the relationship between the attacks.

III. Survey on Intrusion Detection Alert Management

It is noteworthy that several promising approaches have been developed, in the past decades, to solve the problem of intrusion alert management [11], [12]. Intrusion alert management methods can be categorized into four main categories: predefined attack scenarios-based approaches, similarity-based approaches, prerequisites and consequences-based approaches, and hybrid approaches. The main common objective of these approaches is to categorize alerts and to reduce false positive ones.

A. Predefined Attack Scenarios

This category takes into account only the known attack scenarios in order cluster alerts. These attack scenarios are learned in general from different datasets [13], [7], [14]. In order to construct the detected attack and to correlate alerts, each sequence of alerts must be compared with a known attack. The main advantage of this category is its ability to discover the causal relationship between attacks. However, the main drawback of this category is it is limited to known attacks and not unknown ones which is not very helpful for discovering and detecting new ones. Several solutions are suggested, like in [13], where attack scenarios through chronicle language is proposed. Another work is proposed in [7] that uses explicit rules to solve such problem. In [14] attack scenarios are constructed by comparing probability measures. This probability is defined as a metric and is usually calculated through a training data.

B. Similarity-based Approaches

The second category of approaches involves the definition of a similarity metric between alert’s attributes (e.g. source/target IP address, port number, etc.) to classify alerts [15], [16], [17], [18], [6]. Such metric may discover the relationships between different alerts. The obtained score, after calculating the similarity metric, decides if these alerts will be correlated or no. Although this category can be considered as effective for many cases; however, it cannot find the main causal relationship between alerts. For example, in [16], authors grouped alerts into common cluster using an clustering algorithm. A probabilistic-based distance method for alert clustering was also implemented in [15]. The suggested probabilistic model consists of a unified mathematical framework with appropriate similarity functions for each alert feature. As a result, alerts are grouped if the similarity measures are closely matched.

C. Prerequisites and Consequence-based Approaches

A third category has the role to match prerequisites with the consequences based on the dependencies between alerts [19], [20], [21], [22]. Two or more attacks are correlated if any of the prerequisites of the later attack match any of the consequences of the early one. With this principle, causal relationships between attacks can be successfully identified, and it is will possible to build a new attack scenarios by connecting each attack into a sequence of causal relations. The main advantage of this kind of approach is its simplicity to find out the casual relationship between alerts, but the process of discovering individual attacks is computationally expensive. A typical work is proposed in [21] which is based on logical predicates to construct prerequisites and consequences model of attacks.

D. Hybrid Approaches

Hybrid methods are proposed to overcome limitation of applying only single algorithm and to solve the alert management problems with several techniques at the same time [23], [24], [25], [26], [11]. Some interesting papers [12], [11] demonstrate that hybrid approaches provide better flexibility. For instance, a hybrid fuzzy-based anomaly IDS utilizing hidden Markov model (HMM) detector and a normal database detector to minimize false alert rate was developed in [24]. Another decision support system (DSS) for online network behavior monitoring is proposed in [23]. The developed classification model involves three phases: alert preprocessing, model constructing and rule refining. In [25], an effective algorithm is implemented for filtering false alert in network-based IDSs. The proposed filter involves three main components which are based on statistical properties of the alerts. In [8], [11], a collaborative architecture for multiple IDSs to detect real-time network intrusions is also developed. More advanced works are proposed in the literature such as the one published in [27] that enables alert aggregation and minimizing false alerts using an anomaly detector technique. In [26], authors proposed an interesting framework based on structured patterns technique for aggregating input alerts in real-time.

IV. Datasets and Evaluating Metrics

A. Datasets

For evaluation purposes, several challenging datasets are provided for researcher working in this field and several of these datasets are publicly available to be used. In the following, some of well known datasets are presented.

- **ISCX dataset [28]**

The ISCX (Information Security Centre of Excellence) is one of the widely used dataset. Records are defined by simulation and based on eleven features.

- **TUIDS dataset [29]**

The TUIDS dataset [29] was prepared by the University of Tezpur, in which several attack scenarios are performed. The used representative features are labeled into normal or attack.

- **KDDCup'1999 dataset [30]**

The KDDCup'1999 is the most important and used dataset for IDS performance evaluation. It is generated through several simulations and contains more than 4 million records. Each records is defined on the basis of 41 features either as normal or abnormal attacks.

- **CICIDS'2017 dataset [31]**

The CICIDS'2017 dataset is created for Cybersecurity and it contains both attack and normal scenarios as for the case of ISCX dataset.

- **Kyoto 2006+ dataset [32]**

This dataset is also a challenging benchmark used for real traffic data analysis. It involves 24 features.

B. Evaluating Metrics

When evaluating IDSs, several factors should be considered such as the cost, the ease-of-use, the speed, the memory/CPU, the effectiveness, and scalability. The performance is usually evaluated and expressed using the following metrics: True positives (TP), True negatives (TN), False positives (FP), False negatives (FN), sensitivity (or True positive rate: TPR), specificity (or True negative rate), and precision. These metrics are defined as follows:

- True Positive Rate (TPR) = $TP/(FN+TP)$.
- False Negative Rate (FNR) = $TN/(FP+TN)$.
- False Positive Rate (FPR) = $FP/(FP+TN)$.
- Accuracy = $(TN+TP)/(TP+FP+TN+FN)$.

V. Machine Learning (ML) Perspectives

Machine learning-based approaches (Bayesian approaches, Neural Networks, Statistical mixture models, SVM, Hidden Markov model, genetic algorithms, etc.) [33], [34], [35], [36], [37], [38], [39], [1], [2] have been proposed as a powerful techniques to solve several issues related to IDS, alert classification and intrusion detection problems. In particular, they are considered as effective tools for complex data modeling able to represent alerts in a compact form, to filter and to reduce the huge quantity of false alerts and to identify abnormal activities. Moreover, they offer high flexibility to train classifiers and to identify attacks based on a well predefined or extracted specific features. Their use, which is based on the using of a prior and newly acquired information, has proven to be of great importance in this growing area in order to improve the performance of IDS. In the literature, numerous machine learning-based algorithms were implemented for alert classification/clustering [35], [39], [2]. In particular, support vector machines (SVM) is widely employed since it is able to filter efficiently false alert and also it is considered by an important number of researchers in the context of intrusion alert management [40], [41], [42]. Indeed, an SVM-based network intrusion detector is implemented in [40] and its performance is well studied in [41]. A system for alert and attacks grouping is also developed by [43]. In the subsequent work, the expectation maximization (EM) algorithm is investigated to combine resulted groups into one single attack. Probabilistic models are also investigated online alert aggregation [44]. The later work used the so-called maximum likelihood method to estimate the statistical model's parameters. An anomaly-based algorithm that uses a discriminative machine learning model is implemented to detect intrusions attempts [45]. In fact, input intrusions can be modeled as outliers via a principled probabilistic approach. Moreover, finite mixtures models are mainly used to detect both previously seen (known) and unknown attacks. The same authors proposed another interesting classifier in [2] for online intrusion detection.

VI. Machine Learning (ML) based Intrusion Alert Clustering

According to the literature review, many works show that machine learning approaches can be very useful for intrusion alert clustering and outlier detection [44], [43], [46]. A lot of works share some common steps which are described in the following sub-sections. Thus, the main objective of this section is to present for the interested reader how can to design a unified ML-based solution that includes several steps. In particular, a case study will be described throughout next sections.

A. ML-based Framework Design

As shown in Fig. 5, a possible generic solution based on machine learning concepts for intrusion detection and management, alert clustering/classification and outlier detection is presented. Useful technical details related to the implementation of this solution are also provided.

- A preprocessing step: This step is necessary in order to unify and rank all alert information. Moreover, at this level, it is suitable to translate all alert's attributes into numerical values because some of them are in the form of non-numerical values such as SourceIPaddress, DestinationIPaddress, ServiceProtocol and AlertType. These attributes must be mapped into a numerical value.
- A feature extraction/selection step: This step is often used to simplify and accelerate further processing, it would be better to select only significant alert features. Determining an optimal feature set while preserving high accuracy is a challenging problem. To deal with this problem, several algorithms were developed in the literature. Most of them study the relationships between alerts.
- A clustering of normal/abnormal alerts step: Tthis is the main important step and the challenge question is how to develop an accurate intrusion detection model for both alert clustering and abnormal alert detection (outlier detection)? Many supervised and unsupervised machine learning algorithms have been applied to solve this issue. A good choice of a machine learning (ML) technique helps in obtaining effective clustering results.

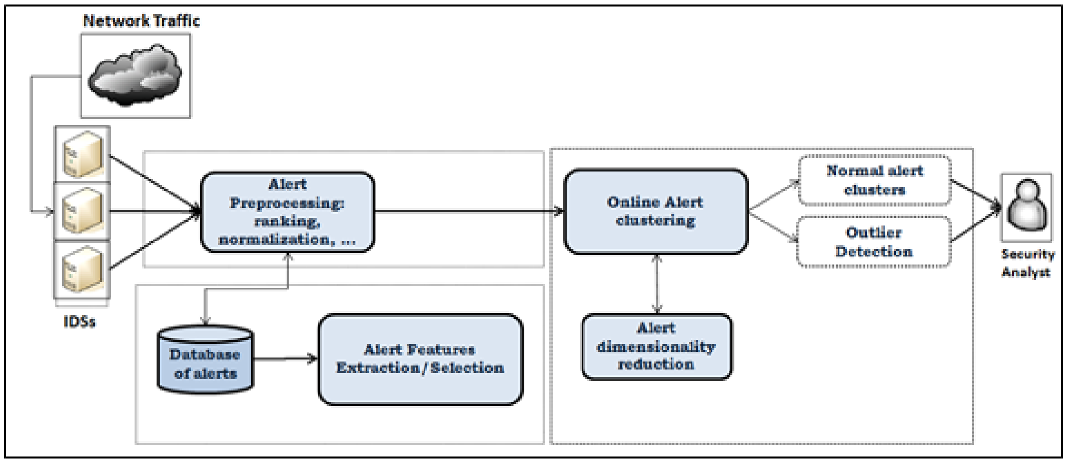


Fig. 5. General ML-based framework for alert clustering.

B. Problem Modeling

According to the literature review, many works suppose that attacks may be considered as a random processes generating alerts [44], [43], [46]. For instance, one of the most interesting techniques that can be applied is mixture models associated with maximum likelihood principle. More specifically, the method “expectation-maximization: EM” [44] can be a reasonable choice. This method has the advantage to avoid the restrictions imposed by other algorithms and can aggregate efficiently similar alerts. On the other hand, it would be more interesting if someone considers an effective strategy called “a bootstrap sampling strategy” within the EM algorithm in order to improve the overall process and to speed up the processing time of aggregating similar-alerts from the output of IDSs. As a result, an optimal representative set of input alerts will be determined according to some well defined criteria thanks to the strategy of bootstrap sampling.

1) Alert features selection: In order to study the relationships between alerts, it is indispensable to analyze their attributes (features). Among all of them, only few features contribute mainly to this relationship. Hence, identifying main features is a crucial step for further processing. To address this problem, several algorithms were developed, and many of them consider a lot alert features in the process which is very expensive. Furthermore, the correlation procedure cannot make use a big number of attributes given the restrictions imposed by the high-speed networks environment. Thus, determine an optimal feature set while preserving high accuracy is a challenging problem for alert management process. To meet this challenge, it seems adequate to follow a procedure that involves for instance two machine learning-based algorithms: principal component analysis (PCA) and a multi-class support vector machine. Why PCA ? since its components are orthogonal to each other and this characteristic has proven to be a useful statistical technique for dimension reduction and multivariate analysis [47] and guarantees a robust convergence and speedup training as confirmed in [48]. SVM is considers as a robust technique especially when dealing with big data. SVM is scalable and has high performance when compared to existing methods such as artificial neural networks (ANN). Now how can these two techniques be used ? First, PCA can be used to select an optimal subset of most relevant attributes. Then, a multi-class support vector machine can be applied to classify alerts into meaningful clusters based on the selected features. If the selected features are not sufficient and the clustering step fails, then, additional attributes are required to increase the clustering precision and accuracy. This process will be repeated until finding a good compromise between a high performance and a small number of alert features. The following algorithm can be used for alert’s attributes election.

```
  program- Alert’s Attributes Selection
    begin
        Run the PCA-algorithm;
        Rank Alert’s Attributes
        Determine initial subset of attr.
        Fa :=initial attributes ;
        Fa := {F1, F2,....,Fm}; (m < 41)
        Fr := {All Attributes} - Fa ;
        NumberSelectedAttributes :=  m;
    Repeat
        {Classify dataset using SVM in N
        classes:(Normal, DoS, U2R, R2L, Probe)}
        {Compute the classification accuracy}
        IF (Accuracy < \epsilon) then
            {Select best Attr. with best rank.}
            bestAttribute := fb;
            Fa :=  Fa + fb;
            NumberSelectedAttributes ++ ;
        Else
            {Return final selected Attr. }
            return Fa ;
        End IF
    Until convergence (accuracy is achieved)
end.
```

2) Dimensionality reduction: To speed up the step of alert clustering, it would be interesting to reduce the data dimension by taking, for example, into account a preprocessing step of data sampling. Among of the motivating techniques, the “bootstrap-sampling” [49] can be examined. Bootstrap is a data resampling method which was introduced as a tool for estimating the sample distribution of statistics. It is applied successfully in many pattern classification problems. The key idea of the Bootstrap is to generate new samples (random samples) to replace original data. The process of determining a random sample is repeated many times until finding an empirical distribution of the statistic. The process of sampling has to reduce the complexity of clustering algorithms (e.g the EM algorithm). From a technical point of view, the initial dataset will be replaced by only a small samples which are closely representative to the initial dataset. Then, the clustering algorithm will estimate statistics on one of these samples. With this manner, statistics can be calculated easily and in "real time". Details concerns the process of the bootstrap sampling combined with the EM algorithm are given as follows:

Let's consider that initial data contains n alerts noted by $A = (A_1, ..., A_n)$. The key problem is how to find randomly a representative Bootstrap sample from the initial dataset denoted $A^* = (A_1^*, \ldots, A_n^*)$. This problem is solved in the following way:

1) Step 1: From the initial sample $A = (A_1, ..., A_n)$, create an empirical probability distribution F which consists in placing the probability of $1/n$ for each alert (all alerts have the same probability).
2) Step 2: Given the empirical distribution function, F, (original data set), generate a new random sample of size n with replacement: this is called the "bootstrap resample".
3) Step 3: Calculate the model statistics through the EM algorithm for this resample (θ^*) instead of the original sample(θ).
4) Step 4: Repeat steps 2 and 3 B-times (B is the number of bootstrap samples) in order to generate B resamples and to obtain an approximation of the distribution. The size of B depends on the tests to be runned on the data.

Now, to estimate the optimal sample size, one can take advantage of some criteria [50]. These criteria can be applied easily in the context of intrusion alert clustering. The appropriate size of the strapped sample is determined as follow:

- K : represents the number of alerts having different attributes each others,
- $\pi_{\rm i}$ is the a priori probability of a particular alert.
- ϵ is a fixed small value,

The probability P_i that a particular alert "i" which is from the sample is given by: $P_i = 1 - (1 - \pi_i)^n$. If the condition $(n\pi_i > 4)$, the probability P_i can be approximated as: $P_i = 1 - e^{-n\pi_i}$. According to this condition, if we have many similar alerts then at least one of them should exist in the sample. It is equivalent to maximize the joint probability: $P_n = \prod_{i=1}^{K}(1 - e^{-n\pi_i})$. This problem is equivalent to minimize the derived of the logarithm the joint probability. Now, denote by n_0 the optimal bootstrap sample size. The value of n_0 is determined as:

$$\begin{cases} Size(n_0) = \sum_{i=1}^{K} \frac{\pi_i e^{-n_0\pi_i}}{1-\pi_i e^{-n_0\pi_i}} < \epsilon \\ n_0 > 4K \end{cases}$$

3) Intrusion alert clustering: At this stage, the challenge question is how to design an accurate intrusion detection model for both alert clustering and abnormal alert detection (outlier detection)? Many supervised and unsupervised machine learning algorithms have been applied to solve this issue. In particular, using an enhanced version of the EM algorithm which is combined with the bootstrap sampling making it an attractive solution. The EM is one of the most frequently used technique for estimating the probability density functions (PDF) in both univariate and multivariate cases. It is used especially to model a set of feature vectors by a mixture of statistical distributions. These distributions are then used to model the observation vectors. There are some researchers who have tried to apply EM-algorithm for alert clustering such as in [51], [52], [38], [53]. From a technical point of view, the distribution of the generated alerts can be approximated according to multivariate probability distribution given that an attack instance is considered as a random process. Let's consider an alert A consists of d attributes. For example, for the case of Gaussian distribution, the density function is defined as: $f\ (A_d; \theta_k) = \frac{1}{\sqrt{2\pi|\Sigma|}_k} \mathrm{e}^{-\frac{1}{2}(a_d-\mu_k)^T \Sigma_k{}^{-1}(a_d-\mu_k)}$ Where:

- $\theta_k = (\mu_k, \Sigma_k))$.
- a_d represents the dth feature of the alert a.
- μ_k and Σ_k are respectively the mean and the covariance matrix.

The posteriori probability to be calculated for each alert corresponds to the labeled classes that should be determined. The aim is to assign the best label L_n to each alert A where$L_n \in \{c_1, ..., c_k\}$, c_n are the classes of the mixture model and k is the number of classes. The output of the problem are: model's parameters associated to each class which are $\theta_k = (\mu_k, \Sigma_k)$; and a posteriori probability γ_k for each alert A. The algorithm is based on two main steps: E (expectation) and M (maximization). In E-step, incoming alerts are assigned to classes which leads to a compact partition A with K classes. The second step is the M step, where an optimal values of the parameters of the developed model is determined. The main steps of the EM algorithm are illustrated in the following pseudo-code.

```
begin
    1.  Initialization-step:
```

$$\pi_k := \frac{1}{K} \qquad (1)$$

```
    2.  Expectation-step:
```

$$\gamma_{nk}^{(q)} := P(c_k/A_n, \theta_k^{(q)}) := \frac{\pi_k^{(q)} f_k(A_n; \theta_k^{(q)})}{\sum_{l=1}^{K} \pi_l^{(q)} f_l(A_n; \theta_l^{(q)})} \qquad (2)$$

```
    3.  Maximization-step:
```

$$\begin{cases} \pi_k^q := \frac{\sum_{n=1}^{N} \gamma_{nk}}{N} \\ \mu_k^q := \frac{\sum_{n=1}^{N} \gamma_{nk} A_n}{\sum_{n=1}^{N} \gamma_{nk}} \\ \sigma_k^q := \frac{\sum_{n=1}^{N} \gamma_{nk} (A_n - \mu_k^{(q)})^2}{\sum_{n=1}^{N} \gamma_{nk}} \end{cases} \tag{3}$$

```
N: Number of alerts.
```

$$\begin{cases} \sum_{i=1}^{K} \pi_i = 1 \\ \pi_i \geq 0 \end{cases} \tag{4}$$

```
K :number of components (classes)
4.  Repeat step 2 and 3 until
convergence based on this criterion:
```

$$|\gamma_{nk}^{(q+1)} - \gamma_{nk}^{(q)}| < \epsilon \tag{5}$$

c) Online Classification Process

It is possible to extend the previous algorithm to an online fashion in order to allow alert classification in real-time. The online classification is a new version over the classical batch version where the parameters are re-calculated each time a new alert is coming. The main reason why an online algorithm is desirable is to avoid huge computational and memory savings. Thus, it does not require the whole data set to be available at each iteration. For online clustering, the mixture model parameter estimates from each previous iteration are used to initialize the next iteration. If a new alert is observed, it can be associated with an existing cluster or to a new created one. This process is performed on the basis of the most likely component using the obtained measures from the E-step of the EM-algorithm. If the alert cannot be assigned to an existing existing component (i.e. there is no similarity with previous alerts), it is assumed as a new alert instance and therefore a new component will be created for it. A possible pseudo-code for online alert classification can be summarized as follows:

```
program for Online Alert Classification

 begin
  While (new alert "a" is received) do
  - find most likely component for "a"

        $k^* := argmax_k(\gamma_k(\theta_k))$        (6)

  - add the new alert to the component

        $C_{old-k} := C_{k^*}$
        $C_{k^*} := C_{k^*} \cup \{a\}$

   - update the new statistics:

        $\pi_{k^*}$, $\mu_{k^*}$, $\sigma_{k^*}$        (7)

  if $|\theta_k - \theta_{k^*}| \geq$ Threshold

     - discard previous changes:

        $C_{k^*} := C_{old-k}$

     - create new component for new alert
     - update number of components
  else
     - accept changes

        $C_{k^*} := C_{k^*} \cup \{a\}$
```

Finally, redundant alerts in each cluster can be fused into a so-called "Meta-Alert". Redundant alerts are supposed have equal attribute values. Meta-Alerts are needed for the security expert reports and may be investigated further in order to detect more complex attack scenarios. A typical algorithm for meta-alert generating is given in the following:

```
program for Meta-alert generating

 begin
    MergeAlerts := 0;
    Repeat for each class Ck
      Repeat For each alert Ai in Ck
        IF (Attr(Ai) == Attr(Ai+1))
             Delete  alert  Ai;
             Meta-Alert := Ai+1;
             MergeAlerts := MergeAlerts++;
        End IF
      End Repeat
    End Repeat
 end.
```

4) Outlier detection: Outliers are anomalies' observations that do not conform to the normal behavioral of the dataset and deviate a lot from the other observations since their values are very different from the data values. Given that, some statistical parameters such as the mean and the standard deviation are sensitive to outlier detection; it is recommended to apply for example more robust well known distance measures such as the so-called "Mahalanobis distance" to filter false positive alerts (outliers). Mahalanobis measure is a multidimensional version of a z-score which is based on clustering between variables and depends on estimated parameters of the multi-variate distribution. It measures the distance of any alert to the center alert (multidimensional mean of the alert-class), given the covariance (multidimensional variance). The Mahalanobis distance is scale-invariant (not dependent on the scale of measurements) and takes into account the correlations of the data set. These properties are not retained by the classical Euclidean distance. Formally, the Mahalanobis distance between a particular multivariate alert vector "a" and the mean value μ is defined as:

$$MDist(a,\mu) = \sqrt{(a-\mu)^T \sum^{-1} (a-\mu)} \tag{8}$$

Where Σ is the covariance matrix of the "normal" data. All data alerts which are located far away from the mean alert (center of the data) are considered as outliers. In other word, multivariate outliers a_i have large distance value $MDist$. Formally, the Mahalanobis distance follows the chi-square distribution with p degrees of freedom χ^2_p. So, an alert is considered as outlier if we have $MDist(a, \mu) > \chi(0.975)$.

```
program for Alert-Outliers Detection
 {Input : different alert classes.
  Output: alerts as outliers. }
 begin
   - Compute mean value of each class;
   - Compute the covariance matrix
    Repeat for each class Ci,
      For each alert aj in Ci,
        - Calculate Mahalanobis distance
          between ai and mean(MDist).
          If( MDist > chi(0.975) ) Then
            - Set aj as Outlier-Alert;
            - Delete aj from class Ci;
          End IF
        End For
      End Repeat
 end.
```

VII. Conclusion and Discussion

The main important problem related to the developed intrusion detection systems (IDSs) in the literature is that they produce a lot of false positive alerts for the same attack. Therefore, it becomes too difficult to distinguish between normal and abnormal alerts, to classify them accurately and to take correct actions for them. A lot of promising researches have been developed but many of them have apparent limitations. It is noted that despite more than twenty years' efforts on the field of intrusion detection systems, a lot of issues still not yet solved. For example, some developed methods for IDSs are not qualified to recognize all kind of intrusions (anomalies), they cannot ensure that all alerts are true positives, and they fail to identify main malicious activities. In this study, a deep review of some well known clustering methods is presented. Then, a particular focus is dedicated for machine learning techniques which are considered as attractive alternatives to address main issues related to IDS systems. Moreover, this paper presents useful technical details for designing and implementing a unified framework by taking part some effective machine-learning algorithms. Such framework can be helpful for interested readers in this field of research.

References

[1] H. Sallay, A. Ammar, M. B. Saad, and S. Bourouis, "A real time adaptive intrusion detection alert classifier for high speed networks," in *2013 IEEE 12th International Symposium on Network Computing and Applications, Cambridge, MA, USA, August 22-24, 2013*, 2013, pp. 73–80.

[2] H. Sallay and S. Bourouis, "Intrusion detection alert management for high-speed networks: current researches and applications," *Security and Communication Networks*, vol. 8, no. 18, pp. 4362–4372, 2015.

[3] K. Alsubhi, I. Aib, and R. Boutaba, "Fuzmet: a fuzzy-logic based alert prioritization engine for intrusion detection systems," *Int. J. Netw. Manag.*, vol. 22, no. 4, pp. 263–284, 2012.

[4] F. Valeur, *Real-Time Intrusion Detection Alert Correlation*. University of California, Santa Barbara: Phd thesis, 2006.

[5] W. Alhakami, A. ALharbi, S. Bourouis, R. Alroobaea, and N. Bouguila, "Network anomaly intrusion detection using a nonparametric bayesian approach and feature selection," *IEEE Access*, vol. 7, pp. 52 181–52 190, 2019.

[6] F. Cuppens, "Managing alerts in a multi-intrusion detection environment," in *Proceedings of the 17th Annual Computer Security Applications Conference*, ser. ACSAC '01, 2001, pp. 22–31.

[7] H. Debar and A. Wespi, "Aggregation and correlation of intrusion-detection alerts," in *Recent Advances in Intrusion Detection*, 2001, pp. 85–103.

[8] J. Yu, Y. R. Reddy, S. Selliah, S. Reddy, V. Bharadwaj, and S. Kankanahalli, "Trinetr: An architecture for collaborative intrusion detection and knowledge-based alert evaluation," *Advanced Engineering Informatics*, vol. 19, no. 2, pp. 93 – 101, 2005.

[9] M. E. Whitman and H. J. Mattord, *Principles of Information Security*, 3rd ed. Boston, MA, United States: Course Technology Press, 2007.

[10] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, "A comprehensive approach to intrusion detection alert correlation," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 3, pp. 146–169, 2004.

[11] C. V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Computers and Security*, vol. 29, no. 1, 2010.

[12] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994 – 12 000, 2009.

[13] B. Morin and H. Debar, "Correlation of intrusion symptoms: an application of chronicles," in *In Proceedings of the 6th International Conference on Recent Advances in Intrusion Detection (RAID'03)*, 2003, pp. 94–112.

[14] O. Dain and R. K. Cunningham, "Fusing a heterogeneous alert stream into scenarios," in *In Proceedings of the 2001 ACM workshop on Data Mining for Security Applications*, 2001, pp. 1–13.

[15] A. Valdes and K. Skinner, "Probabilistic alert correlation," in *Recent Advances in Intrusion Detection*, 2001, pp. 54–68.

[16] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 4, pp. 443–471, 2003.

[17] S. Staniford, J. A. Hoagland, and J. M. McAlerney, "Practical automated detection of stealthy portscans," *Journal of Computer Security*, vol. 10, no. 1/2, pp. 105–136, 2002.

[18] O. M. Dain and R. K. Cunningham, "Building scenarios from a heterogeneous alert stream," in *IEEE Workshop on Information Assurance and Security*, 2001, pp. 231–235.

[19] F. Cuppens and A. Miège, "Alert correlation in a cooperative intrusion detection framework," in *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 2002, pp. 202–2015.

[20] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM conference on Computer and communications security*, 2002, pp. 245–254.

[21] P. Ning, Y. Cui, D. S. Reeves, and D. Xu, "Techniques and tools for analyzing intrusion alerts," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 2, pp. 274–318, 2004.

[22] S. J. Templeton and K. Levitt, "A requires/provides model for computer attacks," in *Proceedings of the 2000 workshop on New security paradigms*, New York, NY, USA, 2000, pp. 31–38.

[23] N.-Y. Jan, S.-C. Lin, S.-S. Tseng, and N. P. Lin, "A decision support system for constructing an alert classification model," *Expert Systems with Applications*, vol. 36, no. 8, pp. 11 145 – 11 155, 2009.

[24] X. D. Hoang, J. Hu, and P. Bertok, "A program-based anomaly intrusion detection scheme using multiple detection engines and fuzzy inference," *Journal of Network and Computer Applications*, vol. 32, no. 6, pp. 1219 – 1228, 2009.

[25] G. P. Spathoulas and S. K. Katsikas, "Reducing false positives in intrusion detection systems," *Computers and Security*, vol. 29, no. 1, pp. 35 – 44, 2010.

[26] R. Sadoddin and A. A. Ghorbani, "An incremental frequent structure mining framework for real-time alert correlation," *Computers and Security*, vol. 28, pp. 153 – 173, 2009.

[27] F. Maggi, M. Matteucci, and S. Zanero, "Reducing false positives in anomaly detectors through fuzzy alert aggregation," *Information Fusion*, vol. 10, no. 4, pp. 300 – 311, 2009.

[28] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357–374, 2012.

[29] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Packet and flow based network intrusion dataset," in *Contemporary Computing - 5th International Conference, IC3 2012, Noida, India, August 6-8, 2012. Proceedings*, 2012, pp. 322–334.

[30] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009*, 2009, pp. 1–6.

[31] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Madeira - Portugal, January 22-24, 2018.*, 2018, pp. 108–116.

[32] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, "Statistical analysis of honeypot data and building of kyoto 2006+ dataset for NIDS evaluation," in *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, BADGERS@EuroSys 2011, Salzburg, Austria, April 10, 2011*, 2011, pp. 29–36.

[33] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Comput. Netw.*, vol. 51, no. 12, pp. 3448–3470, 2007.

[34] F. Najar, S. Bourouis, N. Bouguila, and S. Belghith, "A fixed-point estimation algorithm for learning the multivariate ggmm: application to human action recognition," in *2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, 2018, pp. 1–4.

[35] S. Zanero and S. M. Savaresi, "Unsupervised learning techniques for an intrusion detection system," in *Proceedings of the 2004 ACM symposium on Applied computing*, 2004, pp. 412–419.

[36] S. Bourouis, A. Zaguia, N. Bouguila, and R. Alroobaea, "Deriving probabilistic SVM kernels from flexible statistical mixture models and its application to retinal images classification," *IEEE Access*, vol. 7, pp. 1107–1117, 2019.

[37] T. Pietraszek and A. Tanner, "Data mining and machine learning-towards reducing false positives in intrusion detection," *Inf. Secur. Tech. Rep.*, vol. 10, no. 3, pp. 169–183, 2005.

[38] F. Najar, S. Bourouis, N. Bouguila, and S. Belghith, "Unsupervised learning of finite full covariance multivariate generalized gaussian mixture models for human activity recognition," *Multimedia Tools and Applications*, pp. 1–23, 2019.

[39] P. Laskov, P. Dussel, C. Schafer, and K. Rieck, "Learning intrusion detection: supervised or unsupervised," in *IMAGE ANALYSIS AND PROCESSING, PROC. OF 13TH ICIAP CONFERENCE.*, 2005, pp. 50–57.

[40] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai, and C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Syst. Appl.*, vol. 38, no. 1, pp. 306–313, 2011.

[41] D. Fisch, A. Hofmann, and B. Sick, "On the versatility of radial basis function neural networks: A case study in the field of intrusion detection," *Information Sciences*, vol. 180, no. 12, pp. 2421–2439, 2010.

[42] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *The VLDB Journal*, vol. 16, no. 4, pp. 507–521, 2007.

[43] R. Smith, N. Japkowicz, M. Dondo, and P. Mason, "Using unsupervised learning for network alert correlation," in *21st conference on Advances in artificial intelligence*, ser. Canadian AI'08, 2008, pp. 308–319.

[44] A. Hofmann and B. Sick, "Online intrusion alert aggregation with generative data stream modeling," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 2, pp. 282–294, 2011.

[45] H. Sallay, S. Bourouis, and N. Bouguila, "Web service intrusion detection using a probabilistic framework," in *Progress in Systems Engineering*, vol. 1089, 2015, pp. 161–166.

[46] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Inf. Sci.*, vol. 177, no. 18, pp. 3799–3821, 2007.

[47] I. Jolliffe, Ed., *Principal Component Analysis*, ser. 3rd ed. Springer-Verlag, New York, 2002.

[48] E. Oja, "Neural networks, principal components, and subspaces," *Int. J. Neural Syst.*, vol. 1, no. 1, pp. 61–68, 1989.

[49] B. Efron, "Better bootstrap confidence intervals," *Journal of the American Statistical Association*, vol. 82, no. 397, pp. 171–185, 1987.

[50] C. Banga and F. Ghorbel, "Optimal bootstrap sampling for fast image segmentation: application to retina image," in *IEEE international conference on Acoustics, speech, and signal processing*, 1993, pp. 638–641.

[51] A. Hofmann and B. Sick, "Online intrusion alert aggregation with generative data stream modeling," *IEEE Trans. Dependable Sec. Comput.*, vol. 8, no. 2, pp. 282–294, 2011.

[52] S. Bourouis, Y. Laalaoui, and N. Bouguila, "Bayesian frameworks for traffic scenes monitoring via view-based 3d cars models recognition," *Multimedia Tools and Applications*, pp. 1–21, 2019.

[53] I. Channoufi, S. Bourouis, N. Bouguila, and K. Hamrouni, "Image and video denoising by combining unsupervised bounded generalized gaussian mixture modeling and spatial information," *Multimedia Tools Appl.*, vol. 77, no. 19, pp. 25 591–25 606, 2018.