International Journal of Information Security manuscript No.
(will be inserted by the editor)

Cyber-attack Triage using Incremental Clustering for Intrusion Detection Systems

Sona Taheri · Adil M. Bagirov · Iqbal Gondal · Simon Brown

Received: date / Accepted: date

Abstract Intrusion detection systems (IDSs) are devices or software applications that monitor networks or systems for malicious activities and signals alerts/alarms when such activity is discovered. However, an IDS may generate many false alerts which affect its accuracy. In this paper, we develop a cyber-attack triage algorithm to detect these alerts (so-called outliers). The proposed algorithm is designed using the clustering, optimization and distance-based approaches. An optimization based incremental clustering algorithm is proposed to find clusters of different types of cyber-attacks. Using a special procedure a set of clusters is divided into two subsets: normal and stable clusters. Then, outliers are found among stable clusters using an average distance between centroids of normal clusters. The proposed algorithm is evaluated using the well-known IDS data sets — Knowledge Discovery and Data mining (KDD) Cup 1999 and UNSW-NB15 — and compared with some other existing algorithms. Results show that the proposed algorithm has a high detection accuracy and its false negative rate is very low.

Keywords Computer network security · Intrusion detection system · Outlier detection · Cluster analysis · Incremental algorithm

S. Taheri
Internet Commerce Security Laboratory, School of Science, Engineering and Information Technology, Federation University Australia
E-mail: s.taheri@federation.edu.au

A. M. Bagirov
Internet Commerce Security Laboratory, School of Science, Engineering and Information Technology, Federation University Australia

I. Gondal
Internet Commerce Security Laboratory, School of Science, Engineering and Information Technology, Federation University Australia

S. Brown
Westpac Banking Corporation Australia

1 Introduction

Despite extensive research in intrusion detection [10,20,24], there have been a significant rising trend of cyber-attacks on government, companies, hospitals and universities [1,36, 40]. There are different types of devices and mechanisms within the security environment to protect the networks from malicious attacks. Two of the most popular and significant security countermeasures are intrusion detection system (IDS) and firewall (FW) [3,17]. The IDS detects possible intrusions and generates alerts once any suspected intrusion is determined. The FW looks outwardly for intrusions in order to stop them from happening. It screens network traffic and limits access between networks to prevent intrusion.

An IDS is considered a tool to be used in conjunction with the FW to increase the network systems security. One of the major problems associated with the IDSs is that they generate a large number of false alerts [9,10,12,22,37,38] which reduces their efficiency and accuracy. Therefore, a further research in the IDSs is still a topic worth consideration and new approaches are imperative to effectively defend networks against the attacks.

A false alert is either a false positive alarm or a false negative alarm [22]. In the first state, a false alert is an erroneous signal triggered by an IDS on an activity which is a normal network traffic. In the second state, the IDS identifies an activity as legitimate request while the activity is actually an attack. This case is the most serious and dangerous state. Most of the false negative alerts are outliers and could be harmful for victims. Therefore, outliers should be identified accurately to safe guard against undetected attacks.

An outlier is an inconsistent object (malicious activity/ attack) characterized by its dissimilarity from other objects (legitimate/normal activities) in a given data set. It is expected that outliers are far from the normal objects having

high density of neighbors, and outliers' behaviors are exceptional compared to the normal objects.

In this paper, we develop an algorithm for detection of outliers using optimization based incremental clustering approach. The algorithm clusters alerts generated by the IDS and determines outlier clusters (false alerts). The algorithm starts from one cluster and gradually adds more clusters. Using this algorithm, we find the sequences of the so-called "stable" clusters which are not changing or changing insignificantly in two or more successive iterations. The use of stable clusters of alerts reduces false alerts, resulting in higher reliability of the IDS. The incremental nature of the clustering algorithm and the use of optimization approach allow us to accurately identify isolated clusters which in the case of the IDS are mostly outliers. This fact makes the proposed algorithm well suited for the IDS.

It is anticipated that outliers are among stable clusters which are far away from most of normal objects. Moreover, we expect that outliers are usually clusters with the small number of objects. This gives us another filter to determine the set of candidate clusters for being outliers. The rest of clusters are considered as "normal". Using the average distance between centers of normal clusters we define a threshold and apply it to find a proximity of candidate clusters. Such clusters are accepted as outliers if the distance between their centers and the closest normal cluster's center is more than this threshold. We evaluate the performance of the proposed algorithm using the well-known data sets for IDSs — KDD Cup 1999 and UNSW-NB15 — and compare it with some concurrent algorithms. The main contributions of this paper are:

- Development of a cyber-attack triage algorithm based on optimization, incremental clustering and distance-based approaches;
- Evaluation and comparative analysis of the proposed algorithm.

The rest of the paper is organized as follows. Section 2 provides a brief overview of the literature on IDS, outliers detection and incremental clustering algorithms. The proposed outliers detection algorithm is introduced in Section 3. Numerical results are reported in Section 4 and Section 5 contains some concluding remarks and possible directions for future research.

2 Background and overview

In this section, first we present a brief overview of some existing algorithms for intrusion and outliers detection. Then, we provide a brief description of the incremental clustering algorithm.

2.1 Intrusion detection systems

The IDS is categorized as the network-based (NIDS) or the host-based (HIDS) [16,20,31]. In the first category, header fields of different network protocols are used to detect intrusions. The NIDS evaluates information captured from a network and analyzes the stream of packets which travel across the network. The HIDS focuses mainly on monitoring and analyzing the internals of a computing system as well as the network packets on its network interfaces.

In general, there are two approaches for intrusion detection [20,26,31]: signature-based and anomaly-based. The signature-based intrusion detection (SID) is designed on the basis of known attack patterns. In this case the new attack is checked for the presence of these patterns. If a match is discovered, then an alert indicating the existence of the corresponding attack is produced. The anomaly-based intrusion detection (AID) analyzes the deviation from normal activities. If a significant deviation is detected, then an alert is produced along with information about the nature of the deviation. Since the SID relies mainly on well-known intrusions it is incapable of detecting new intrusions [15,33].

Machine learning based IDS (ML-IDS) is one of the main techniques of the AID which aims to provide a learning based system to discover cyber-attacks (see [25] and references there in). Some of advantages of the ML-IDS over the conventional SID are [25]: it is easy to bypass the SID by doing slight variations in an attack pattern whereas the ML-IDS can detect the attack variants as it learns the behavior of the traffic flow; the CPU time is low to moderate in the ML-IDS as there is no need to analyze all signatures of the database as done by the SID; the ML-IDS can capture the complex properties of an attack behavior and improve the detection accuracy than the conventional SID. Furthermore, the ML-IDS are much faster than the later methods. Although the ML-IDS techniques are capable of detecting new cyber-attacks they suffer from different drawbacks [18, 22,38,41].

2.2 Outlier detection algorithms

There are various methods introduced in the literature to detect outliers including density-based, distance-based, clustering and optimization-based. The density-based approach works on the assumption that the density around a normal data object is similar to the density around its neighbors while the density around an outlier is considerably different to the density around its neighbors. The density-based method proposed in the paper [7] finds the outliers based on the density of a local neighborhood. It assigns a local outlier factor to a data object using k neighbors which measures the degree of an object being an outlier. The authors in the paper

[34] developed a generalization of the density-based outlier detection method using the kernel density estimation.

The distance-based method assumes that the normal data objects have a dense neighborhood and the outliers are far apart from their neighbors. The local distance-based outlier factor algorithm is proposed in the paper [43]. This algorithm uses the relative location of an object to its neighbors to determine the degree to which the object deviates from its neighborhood. The authors in the paper [44] introduced the self-adaptive neighborhood method to detect outliers. This method finds multiple valued neighbors of each data object by considering some characteristics of all objects.

The clustering-based methods find outliers as a side product of clustering algorithms [14,42]. They consider too distant clusters of small sizes as outliers. However, the noise is usually tolerated or ignored when these methods produce the result [19]. Different hybrid methods are proposed based on the combination of the clustering and other approaches [13,30,32]. For instance, the method proposed in [30] uses a combination of the clustering and distance-based approaches. It applies the k-means clustering algorithm on the initial set. Then, outliers are selected by applying the distance-based method using a certain threshold set by the user.

A multi-objective optimization algorithm to detect outliers is developed in [11]. This algorithm is based on the Particle Swarm Optimization method. The authors in the paper [28] introduced two algorithms for the outliers detection which involve solving the min-max optimization problems. The paper [35] presents the method of minimizing the sum of infeasibilities as a non-iterative outlier removal algorithm. An outliers detection algorithm based on the combination of the k-means clustering and binary optimization approaches is proposed in [13].

Most of the methods mentioned above are only able to find local solutions which can be far away from global solutions. Therefore, these methods are not continuously learning the behavior thereby leading to the possibility that a new intrusion will not be detected and/or a false alarm may be generated [8,31]. The proposed algorithm in this paper is able to find global or near global solutions, and therefore, it detects false alerts more accurately. The algorithm is a combination of the clustering, optimization and distance-based approaches. We apply the incremental clustering algorithm whose usage helps to solve two main problems: (i) to find accurate solutions to clustering problems; (ii) to identify the sequences of stable clusters which leads to determining the set of false alerts (outliers). Next, we present this algorithm briefly and refer to references [5,6,29] for more details.

2.3 Incremental clustering algorithm

Let A be a set of finite objects (points, records) $a^1,\ldots,a^m$, where $a^i \in \mathbb{R}^n$ for $i = 1,\ldots,m$. The hard clustering problem is the distribution of these objects into a given number k clusters $A^j \neq \emptyset, j = 1,\ldots,k$ such that

$$A^j \bigcap A^l = \emptyset, \text{ for } j,l = 1,\ldots,k\ , j \neq l, \text{ and } A = \bigcup_{j=1}^{k} A^j.$$

The cluster A^j is identified by its center $x^j \in \mathbb{R}^n, j = 1,\ldots,k$. The problem of finding these centers is called *the k-clustering* problem and is formulated as [4]

$$\begin{cases} \text{minimize} & f_k(x^1,\ldots,x^k) \\ \text{subject to} & x = (x^1,\ldots,x^k) \in \mathbb{R}^{nk}, \end{cases} \tag{1}$$

where

$$f_k(x^1,\ldots,x^k) = \frac{1}{m}\sum_{i=1}^{m} \min_{j=1,\ldots,k} \|x^j - a^i\|^2.$$

In the incremental clustering algorithm, a data set is static and clusters are computed incrementally. The algorithm starts with the calculation of the center for the whole data set and gradually adds one cluster center at each iteration. This algorithm involves the so-called auxiliary clustering problem to generate a set of starting cluster centers. Assume that the solution $(x^1,\ldots,x^{k-1})$ to Problem (1) for $k-1$ clusters is known. In order to solve the k-clustering problem we apply the special procedure introduced in [29] to generate a set S_1^k of starting points for the k-th cluster center.

The auxiliary clustering problem is solved starting from each point of S_1^k. As a result we obtain a set S_2^k of stationary points of the auxiliary clustering problem. In the next step of the incremental algorithm each point from the set S_2^k is added to the set of $k-1$ cluster centers from the previous iteration to obtain a starting point for solving the k-clustering problem (1). To solve this problem, the modified global k-means algorithm [4] is applied and a set of stationary points S_3^k is obtained. The best solution among these points is chosen to be a solution to the k-clustering problem. This algorithm terminates when the required number of clusters are computed. The incremental algorithm in addition to the k-clustering problem solves also all intermediate l-clustering problems, $l = 1,\ldots,k-1$. Such an algorithm allows one to find a good quality solution to clustering problems.

3 The proposed algorithm for detection of outliers

In this section, we design an algorithm for the detection of outliers generated by IDSs. In order to do so we introduce the notion of stable clusters and give some definitions.

3.1 Notations and definitions

Let

$$A^1_{p-1},\ldots,A^{p-1}_{p-1} \text{ and } A^1_p,\ldots,A^p_p$$

be the cluster distributions found by the clustering algorithm at the iterations $p-1$ and p, respectively. Take any cluster A^t_p, $t \in \{1,\ldots,p\}$ and consider the sets:

$$\bar{A}^l_t = A^l_{p-1} \bigcap A^t_p,\ l = 1,\ldots,p-1.$$

Define the numbers:

$$s^l_t = \frac{card(\bar{A}^l_t)}{card(A^l_{p-1})},\ l = 1,\ldots,p-1,\ t = 1,\ldots,p,$$

and a threshold $\alpha_1 \in (0,1]$. Here $card()$ is the number of objects in a set. Let $\bar{s}_t = \max\{s^1_t,\ldots,s^{p-1}_t\}$, $t = 1,\ldots,p$.

Definition 1 *Stable cluster(s).* If $\bar{s}_t \geq \alpha_1$, then the cluster A^t_p is called 1-step stable at the iteration p with respect to the iteration $p-1$. By extending this notion we define also a q-step stable cluster at the p-th iteration ($p > q$) which are 1-step stable clusters over the last q iterations.

It is expected that all outliers are among the stable clusters with small cardinalities. Next, we give the definition of a candidate outlier cluster. Let $\alpha_2 \in (0,1)$ be a given sufficiently small number and $A^1,\ldots,A^k$ be a k-partition of the set A.

Definition 2 *Candidate outlier cluster.* The cluster A^j is called a candidate outlier if

$$card(A^j) \leq \alpha_2 m.$$

All clusters which are not candidate outliers are called "normal" clusters.

Let $\bar{k}$ be a number of normal clusters, $\bar{k} \leq k$. Furthermore, let A^j and $A^{\hat{j}}$ be two normal clusters with the centers x^j and $x^{\hat{j}}$, respectively. The Euclidean distance between centers of these two clusters is defined as

$$d(x^j,x^{\hat{j}}) = \sum_{i=1}^{n} \left((x^j_i - x^{\hat{j}}_i)^2\right)^{1/2}.$$

Definition 3 *Average distance.* The average distance d_{ave} for the collection of $\bar{k}$ normal clusters is

$$d_{ave} = \frac{2}{\bar{k}(\bar{k}-1)} \sum_{j=1}^{\bar{k}} \sum_{\hat{j}>j}^{\bar{k}} d(x^j,x^{\hat{j}}).$$

Definition 4 *Outlier cluster.* The candidate outlier cluster is called an outlier if the distance between its center and the center of the closest normal cluster is greater than the average distance d_{ave}.

Now, we are ready to introduce the incremental clustering algorithm for outliers detection (ICAOD).

3.2 The proposed algorithm ICAOD

In the ICAOD, we first determine the distributions of data objects into a given number of clusters applying the algorithm described in Subsection 2.3. Then we find stable clusters using Definition 1 and select candidate outliers among them by applying Definition 2. The rest of clusters which are not candidate outliers are considered as normal clusters. Using centers of normal clusters, we calculate the average distance. After that, we compute the distance between the centers of each candidate outlier and the closest normal cluster. If this distance is greater than the average distance found, then the candidate outlier is identified as an outlier. We continue this process until the sets of outliers in two consecutive iterations of the incremental algorithm coincide. The proposed algorithm is given in its step-form in Algorithm 1, and its flowchart is presented in Figure 1.

Algorithm 1 Incremental clustering algorithm for outliers detection (ICAOD).

Input: Data set A, maximum number of clusters $k \geq 2$, $\alpha_1 \in (0,1]$ and $\alpha_2 \in (0,1)$.

Output: Collection of outlier clusters.

Step 1. (Initialization). Compute the center $x^1 \in \mathbb{R}^n$ of whole data object A. Set $l := 2$.

Step 2. (Computation of a set of solutions of the auxiliary clustering problem). Apply the procedure from [29] to find the set $S^l_1 \subset \mathbb{R}^n$ of starting points for solving the l-th auxiliary clustering problem. Then find the set $S^l_2 \subset \mathbb{R}^n$ of solutions to the auxiliary problem using starting points $y \in S^l_1$.

Step 3. (Computation of a set of cluster centers). For each $y \in S^l_2$ select $(x^1,\ldots,x^{l-1},y)$ as a new starting point, apply the modified global k-means algorithm to solve the l-th clustering problem (1) for $k = l$. Let $S^l_3 \subset \mathbb{R}^{nl}$ be a set of solutions to this problem.

Step 4. (Computation of the best solution). Compute values of the objective clustering function f_k for $k = l$ at each point from the set S^l_3. Find the best solution $(x^1,\ldots,x^l) \in \mathbb{R}^{nl}$ providing the least value of this function and the corresponding cluster distribution $D^l_1 = \{C^1,\ldots,C^l\}$.

Step 5. (Determination of stable clusters). Determine stable clusters from the set D^l_1 according to Definition 1 for a given α_1. Let D^l_2 be the set of stable clusters.

Step 6. (Determination of candidate outliers and normal clusters). For a given α_2, apply Definition 2 to determine the set D^l_3 of candidate outlier clusters. Define the set D^l_4 of normal clusters as $D^l_4 = D^l_1 \setminus D^l_3$.

Step 7. (Computation of the average distance). Compute the average distance d_{ave} using Definition 3 and the set D^l_4.

Step 8. (Determination of outlier clusters). Determine the set D^l_5 of outlier clusters from the set D^l_3 according to Definition 4.

Step 9. (Stopping criterion 1). If the set D^l_5 coincides with the set D^{l-1}_5, then **stop**.

Step 10. (Stopping criterion 2). Set $l := l+1$. If $l \leq k$, then go to Step 2. Otherwise, **stop**.

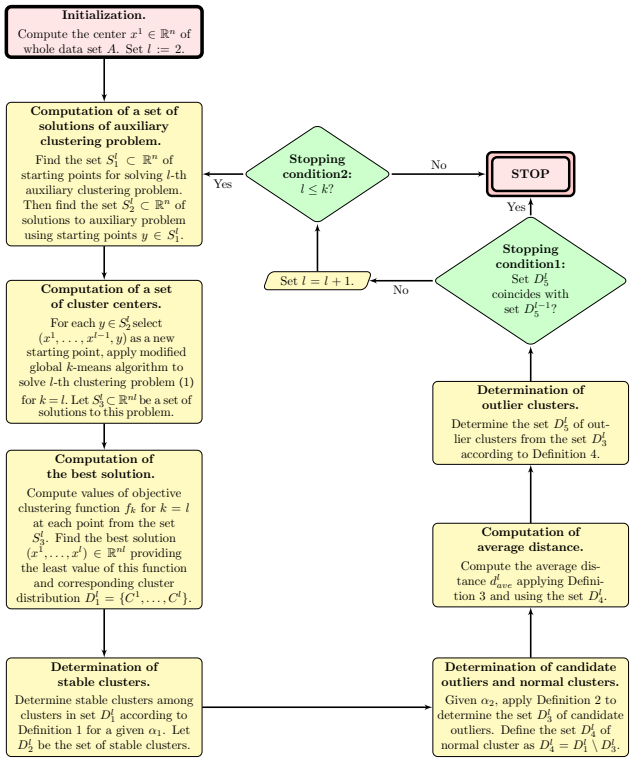

Fig. 1 Incremental clustering algorithm for outliers detection

3.3 Computational complexity of the proposed algorithm

To analyze the complexity of Algorithm 1, we estimate only the total number of distance calculations used by this algorithm. This is due to the fact that the algorithm involves distance calculations and additionally only simple operations to compute cluster centers. In order to select all starting points from the set S_1^l at Step 2 one needs $O(m^2)$ distance calculations. Let $\mathfrak{s}_1$ be a cardinality of the set S_1^l. Then the number of distance calculations to find the set S_2^l is $O(m\mathfrak{t}\mathfrak{s}_1)$ where $\mathfrak{t}$ is the maximum number of iterations used by the algorithm for solving the auxiliary problem. To compute the set S_3^l in Step 3 one needs $O(lmT\mathfrak{s}_2)$ distance calculations. Here, T is the maximum number of iterations used by the modified global k-means algorithm among all starting points from the set S_2^l, and $\mathfrak{s}_2$ is the cardinality of the set S_2^l. Note that $\mathfrak{t} \ll T$. We do not calculate any distance in Step 4. Moreover, Definitions 1 and 2 do not require any distance calculations and therefore, in Steps 5 and 6 we do not calculate distances. The number of distance calculations in Steps 7 and 8 is estimated as $O(l^2)$. Since $l \leq k$ and $\mathfrak{s}_1, \mathfrak{s}_2 \leq m$ we conclude the total number of distance calculations in Algorithm 1 can be estimated as $O(km^2T)$.

4 Numerical experiments

Using two well-known labeled data sets widely applied for the evaluation of IDSs in the literature, "KDD Cup 1999" and "UNSW-NB15", we present the performance of the proposed algorithm ICAOD and compare it with the following algorithms:

• Outlier detection methods mentioned in Section 2:

- *Density-based methods:* Local Outlier Factor (LOF), Kernel Density Estimation Outlier Score (KDEOS);
- *Distance-based methods:* Local Distance-based Outlier Factor (LDOF), Natural Neighbor (NAN);
- *Hybrid methods:* Combination of Clustering and Distance-based (CCD), Optimization based Clustering (OC).

• ML-IDS algorithms mentioned in [25]:

- Artificial Neural Network (ANN);
- Random Forest (RF);
- Support Vector Machines (SVM).

We use the following common performance measures in our experiments:

$$TPR = \frac{TP}{TP+FN}, \qquad FNR = 1 - TPR,$$
$$FPR = \frac{FP}{FP+TN}, \qquad TNR = 1 - FPR, \tag{2}$$

where

- True Positives (TP): malicious activity successfully classified as malicious;
- True Negatives (TN): normal activity successfully classified as normal;
- False Positives (FP): normal activity being classified as malicious;
- False Negatives (FN): malicious activity incorrectly classified as normal activity.

Note that an algorithm has a high ability of detecting the cyber attacks if it has a high true positive rate (TPR) and low false positive/negative rates (FPR/FNR).

In addition, we use the performance measure "balanced accuracy (BCR)" in the experiments. The reason for choosing the BCR rather than the regular accuracy is due to the better performance of the former measure in data sets where the number of different types of attacks is significantly disproportional. The BCR, the proportion of true results, is defined as

$$BCR = \frac{1}{2}(TPR + TNR). \tag{3}$$

4.1 Implementation of algorithms

The ICAOD is implemented in Fortran 95 and compiled using free compiler *gfortan*. We use R implementations of other algorithms, the outlier detection algorithms are available in [23], and the ML-IDS techniques are available in "GitHub" [2]. The computational experiments are carried out on a machine with 2.50 GHz Intel(R) Core(TM) i5-3470S and 8 GB of RAM and are based on the off-line evaluations.

In the ICAOD, where we apply the clustering algorithm to determine a distribution of data objects into a given number k of clusters, we utilize the parameters that were used in [29]. We set the maximum number of clusters $k = 120$ to allow the algorithm to reach the maximum possible purity. The other parameters in the ICAOD are α_1 and α_2. The first parameter is used to determine stable clusters. Note that this parameter should be chosen closer to 1 since it is required that a big portion of a cluster does not change over two successive iterations of the incremental algorithm. The experiments show that the optimal value is $\alpha_1 = 0.9$: the values greater than this do not change the list of stable clusters; the values smaller than 0.9 increase the number of stable clusters and, consequently, the number of outliers which lead to an increase in the rate of false positives.

Our experiments show that the value of the parameter α_2 cannot be greater than 0.03 as this allows the algorithm to choose clusters with a big portion of objects as candidate outliers which are not. In order to find the optimal value of this parameter, we select different values of $\alpha_2 \leq 0.03$ and find the candidate outliers corresponding to each of these values (see Figure 2). It can be observed from this figure that $\alpha_2 = 0.005$ is the optimal value: the values greater than this add more clusters to the list of candidate outliers which are not outliers and, therefore, increase the number of false positives; the values smaller than 0.005 remove some important outliers from the list of candidate outliers and, thus, increase the number of false negatives. The optimal values of these parameters, $\alpha_1 = 0.9$ and $\alpha_2 = 0.005$, are the same in both training and testing phases.

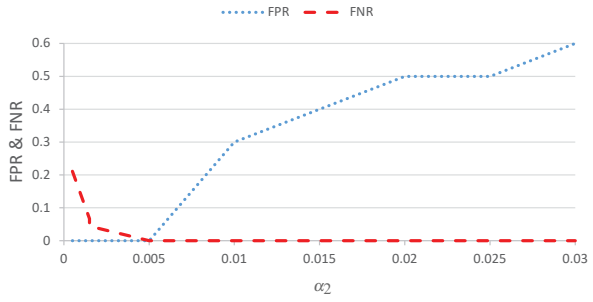

Fig. 2 Dependency of rates of false positive/negative on parameter α_2

4.2 Experiments using KDD Cup 1999

First, we give the brief description of this data set. Then, we present our experiments using this data.

4.2.1 KDD Cup 1999

The data set Knowledge Discovery and Data mining (KDD) Cup 1999 is available in the UCI machine learning repository [21]. This data set contains a variety of intrusions simulated in a military network. The corrected KDD data set has 42 features (35 numeric and 7 symbolic), where 4 features among the symbolic ones are nominal. To use this data in our numerical experiments, we converted the symbolic values into numeric ones. The last feature in the data set indicates if an alert is normal or attack. There are 22 different types of attacks. Each attack belongs to one of four main categories: denial of service (DOS), unauthorized access from a remote machine (R2L), unauthorized access to local superuser (root) privileges (U2R) and surveillance and other probing (PROB). The total number of records is 4,000,000. In our experiments, we create 4 subsets of the data set KDD Cup 1999 using its 4 main categories of attacks: U2R, DOS, R2L and PROB. The details of these subsets are given in Table 1, where m_A and m_N denote the number of data points in Anomalous and Normal activities, respectively.

Table 1 Description of subsets chosen from KDD Cup 1999

Subset		Anomalous		Normal
	Category	Type	m_A	m_N
1	U2R	bufferoverflow	30	10
2	DOS	teardrop	979	200
3	R2L	warezclient	1020	210
4	PROB	nmap	2316	500

4.2.2 Purity of clusters obtained by ICAOD

Each cluster may contain points from different classes. The purity gives the ratio of the dominant class size in the cluster to the cluster size itself (in percents). A large purity value implies that the cluster is a "pure" subset of the dominant class.

Let the data set A be a composed of the classes $\{C^1,\ldots,C^q\}$, and a clustering algorithm finds clusters $\{A^1,\ldots,A^k\}$ in this data set. The purity of the cluster A^p is defined as follows [39]:

$$purity(A^p) = \frac{1}{|A^p|} \max_{j=1,\ldots,q} |A^p \cap C^j| \times 100,\ p = 1,2,\ldots,k.$$

Results for the purity of clusters obtained by the ICAOD using the data set KDD cup 1999 are presented in Figure 3,

where in brackets we present the cluster number (iteration) and the corresponding value of purity. Red dots indicate the cluster numbers between which the purity is changed significantly. We can see that the purity changes drastically between iterations 54 and 55, considerably between iterations 55 to 68 and also 68 to 114 of the incremental clustering algorithm. This is due to the fact that the algorithm divides large classes to get clusters with the better purity.

Fig. 3 Number of clusters vs. purity of clusters

4.2.3 Stable clusters generated by ICAOD

The maximum number of clusters computed by the ICAOD is 120. According to results for the purity presented in the previous subsection we consider a cluster stable if its content does not change significantly after the 54-th iteration with respect to some tolerance. The incremental clustering algorithm starts to decompose large clusters after its 54-th iteration and improves the purity of clusters significantly. All small and remote clusters are not changed after this iteration. Therefore, all the stable clusters are among those obtained in the first 54 iterations of the algorithm.

We present the stable clusters generated by the ICAOD for the first 54 clusters in Figures 4 and 5. In these figures, $x-y$ shows that the cluster x stays stable over the last y iterations ($y = 1,2,\ldots,120$). In addition, the number $s \in [0,1]$ shows the portion of the cluster x which does not change its cluster label over the last y iterations.

These figures show that there are 23 stable clusters. Among these 23 stable ones 19 clusters stay exactly the same during these iterations (with $s = 1$). According to the procedure for finding starting cluster centers, the incremental clustering algorithm chooses starting centers from all clusters including stable ones, however the algorithm does not change the content of these clusters. This means that these clusters are located away from other clusters and can be considered as candidate outliers.

After finding the set of stable clusters we apply the rule on the cardinality of these clusters and keep only those which satisfy the condition in Definition 2. In this step we identify the set of candidate outliers. All stable clusters satisfy

Fig. 4 Stable clusters generated by the ICAOD

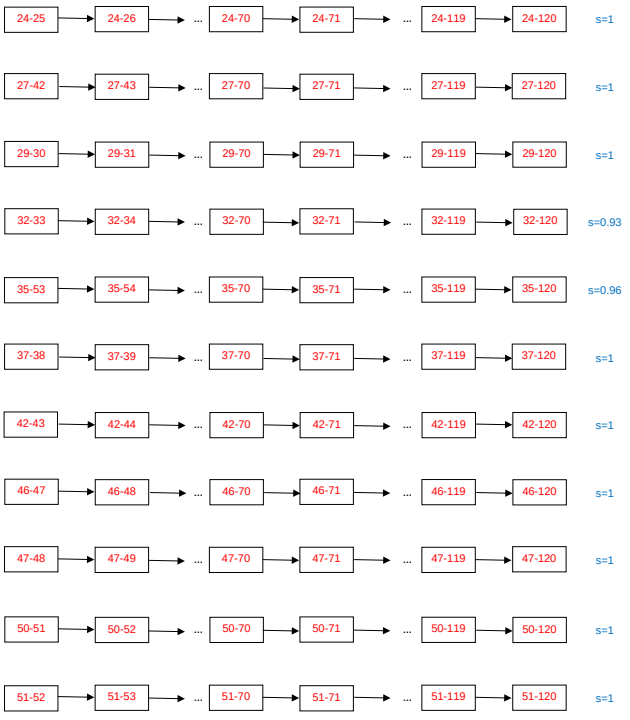

Fig. 5 Stable clusters generated by the ICAOD (cont)

this condition, and therefore they are considered as candidate outliers. Finally, we apply Definition 4 to the set of candidate outliers to determine how far these clusters are from normal clusters. This step identifies the clusters 2-6, 9-10, 15, 18-19, 24, 27, 29, 37, 46-47 and 50-51 as outliers.

4.2.4 Relationship between outliers and different attacks

In this subsection, we discuss the relationship between the outliers obtained by the ICAOD and different types of attacks. The outlier clusters found by the proposed algorithm contain the attacks from all four categories. The clusters 2-6 contain most of attacks from the category DOS (approximately 99.5 %). The clusters 9, 10, 15 and 16 contain all attacks from the category U2R. The attacks from the category R2L fall into the outlier clusters 18, 19, 24, 27, 29, 32, 35 and 42. Finally, the clusters 46, 47, 50 and 51 include all attacks from the category PROB.

4.2.5 Rate of false positive/negative generated by ICAOD

The implicit cost of a false alert in the cyber security domain is a serious issue. That is when a malicious activity is classified as normal activity and vice versa.
The TPR generated by the ICAOD on the data set KDD Cup 1999 is 0.9953 which means that the algorithm can detect 99.53 % of the attacks correctly in outlier clusters. This algorithm has very low false positive/negative rates: FPR is 0.0051 and FNR is 0.0047.

4.2.6 Comparison of Algorithms

The comparison on the performance of the ICAOD with some other existing algorithms is given in this subsection.

Outliers detection algorithms. In Tables 2 and 3, we present the performance of the ICAOD and outliers detection algorithms using the measures TPR and FPR, given in (2), and data subsets from Table 1 (rates are given as percentage). In all data subsets, the TPRs generated by the ICAOD is significantly higher than those obtained by algorithms LOF, LDOF and CCD. Moreover, the ICAOD has a larger TPR than KDEOS and NAN algorithms in these data subsets. The only exception is the OC algorithm whose TPR in the data subset 1 is slightly greater than that of the ICAOD. However, the latter algorithm outperforms the former one in the data subsets 2, 3 and 4. The ICAOD detects all outliers in the data subset 2, whereas the highest TPR among the other algorithms is only 94 (obtained by the KDEOS algorithm). Furthermore, the ICAOD and OC algorithms have generated the lowest FPRs in all data subsets compared to the other five algorithms.

Table 2 TPR for different outliers detection algorithms

Subset	ICAOD	LOF	KDEOS	LDOF	NAN	CCD	OC
1	86	73	80	75	83	73	88
2	100	81	94	83	85	79	93
3	99	76	93	85	79	78	90
4	99	78	85	81	83	77	91

Table 3 FPR for outliers detection algorithms

Subset	ICAOD	LOF	KDEOS	LDOF	NAN	CCD	OC
1	10	24	25	22	23	18	10
2	5	26	18	24	25	19	3
3	14	15	19	21	16	17	15
4	12	31	26	26	20	13	12

ML-IDS algorithms . To compare the accuracy of the ICAOD and other ML-IDS algorithms, we use the performance measure *BCR* given in (3). It can be observed from Table 4 that the ICAOD has the highest *BCR* in average in comparison with other ML-IDS algorithms: its *BCR* is significantly higher than that of the later algorithms in the data subset 1; it has the second highest *BCR* in the data subset 2; its *BCR*s are larger than those of others in the data subsets 3 and 4.

Table 4 *BCR* for ML-IDS algorithms

Subset	ICAOD	ANN	RF	SVM
1	88.0	40.9	75.2	61.1
2	97.5	98.4	92.6	90.0
3	92.5	86.6	82.2	73.0
4	93.5	92.0	86.8	90.9

4.3 Experiments using UNSW-NB15

In this subsection, we present the experimental results using the data set UNSW-NB15. First, we provide a brief discerption of this data.

4.3.1 UNSW-NB15

The raw network packets of the UNSW-NB 15 data set [27] is created by the IXIA Perfect Storm tool in the Cyber Range Lab of the Australian Centre for Cyber security to generate a hybrid of real modern normal activities and synthetic contemporary attack behaviors. Tcpdump tool is utilized to capture 100 GB of the raw traffic (e.g., Pcap files). The Argus, Bro-IDS tools are utilized and twelve algorithms are developed to generate totally 49 features (43 numeric and 6 nominal). To use this data in our experiments, we converted nominal values into numeric ones. The last feature in the data set indicates if an alert is normal or attack. This data set has nine families of attacks, namely: Fuzzers, Analysis,

Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The total number of records is 2,540,044. Similar to the paper [27], in our experiments, we take some part of the data set UNSW-NB15 and divide it into training and test sets. Details of these sets are given in Table 5.

Table 5 Description of subsets from UNSW-NB15

Subset	Total	Anomalous	Normal
Training	175342	119341	56000
Testing	82332	45332	37000
Total	257674	164674	93000

4.3.2 Relationship of purity with FP and FN

Figure 6 presents the relationship of FP and FN with the purity of clusters obtained by the ICAOD. Overall, values of both FP and FN decrease as the purity increases. The decrease in the value of FP is not significant and may oscillate, whereas the value of FN tends to decrease without any serious oscillations. In order to have a significant decrease in the values of FP and FN, one may consider a larger number of clusters since usually we get a higher purity as the number of clusters increases (see, for example , Figure 3).

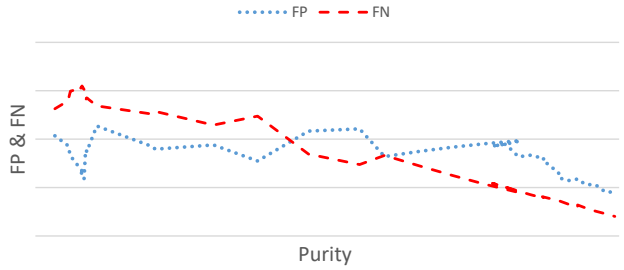

Fig. 6 Purity vs. FP and FN

4.3.3 Comparison of Algorithms

The experimental results comparing the ICAOD against several existing algorithms are given below.

Outliers detection algorithms. To evaluate the prediction ability of the ICAOD and other outliers detection algorithms using the data set UNSW-NB15, we first train the algorithms using the training data set and find outliers. Then these outliers are used to make predictions using data from the testing set. The results of TPR and FNR (in percentage) are presented in Figure 7 and for TNR and FPR (in percentage) in Figure 8.

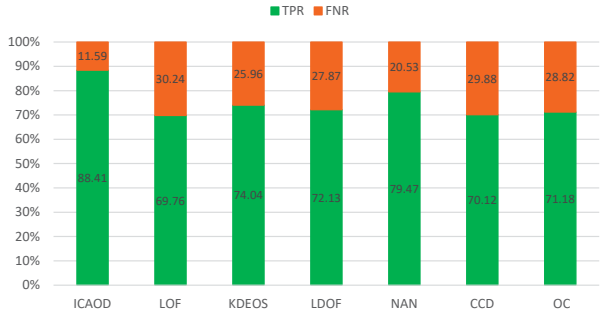

Fig. 7 *TPR* and *FNR* for outliers detection algorithms

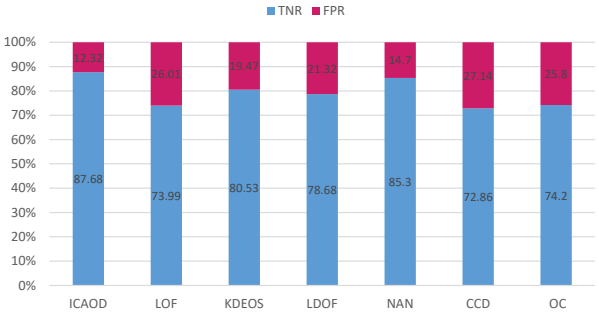

Fig. 8 *TNR* and *FPR* for outliers detection algorithms

From Figure 7 we observe that the ICAOD provides the best performance. It identifies 88.41 % of anomalous packets correctly and only 11.59 % of anomalous packets as normal. TPR and FNR for the algorithm NAN are 79.47 and 20.53 percentages, respectively. The LOF algorithm demonstrates the worst performance with 69.76 and 30.24 % for TPR and FNR, respectively. Furthermore, Figure 8 shows that the ICAOD has the highest TNR and it detects 87.68 % of the normal packets correctly, followed by NAN with 85.30 %. The CCD algorithm has the worst performance with the lowest TNR, 72.86 %.

ML-IDS algorithms . Figure 9 demonstrates the *BCR* of the ICAOD and other ML-IDS algorithms for different attacks of the data set UNSW-NB15. The results show that RF has the highest value of the *BCR*; the ICAOD has the second *BCR*; SVM is the third with a slightly small value of the *BCR* and ANN has the lowest *BCR*.

5 Conclusions

In this paper, we introduced an incremental clustering algorithm to detect outliers generated by an intrusion detection system. We gave definitions of the stable and outlier clusters. The proposed algorithm first finds the set of stable clusters, and then selects candidate outliers among them. Clusters which are not candidates are considered as normal clusters. Using an average distance between centroid of normal

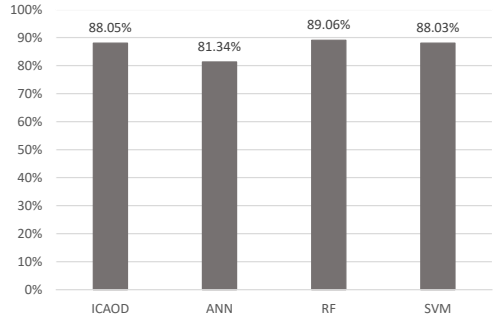

Fig. 9 *BCR* for ML-IDS algorithms

clusters, the algorithm determines the outliers representing anomalous events.

The performance of the proposed algorithm was demonstrated using the well-known data sets for IDSs, KDD Cup 1999 and UNSW-NB15. Results show that the algorithm is able to detect outliers with a high accuracy. In addition, using the performance measures true and false positive rates, the ICAOD was compared with some existing outliers detection methods. The experimental results confirm that the ICAOD achieves higher accuracy and detection rate than those of other outliers detection methods. Furthermore, we compared the ICAOD with three well-known machine learning based IDS (ML-IDS) algorithms. The results obtained show that the ICAOD has the highest balanced accuracy (*BCR*) in average in all data subsets from KDD Cup 1999 and the second highest in the data set UNSW-NB15. We did not present the comparison of algorithms based on the required CPU time as these algorithms were implemented using different platforms. However, we discuss computational complexity of the proposed algorithm which is rather polynomial.

Note that this paper is more theoretical and addresses mostly the algorithmic development. The aim of the proposed algorithm is to detect false alerts generated by the IDS in a network system. Further studies are needed to determine how this algorithm can be applied in multiple network routes and also how the running time and accuracy of the algorithm will be affected by increasing the number of packets analyzed. These will be the subjects of the future research.

Acknowledgements The authors are grateful to the anonymous reviewers for their constructive comments which greatly helped improving the quality of this paper.

Compliance with Ethical Standards

Funding: This research was conducted in Internet Commerce Security Laboratory (ICSL) funded by Westpac Banking Corporation Australia. In addition, the research by Dr. Sona Taheri and A/Prof. Adil Bagirov was supported by the Australian Government through the Australian Research Council's Discovery Projects funding scheme (DP190100580).

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

Conflict of Interest: The authors declare that they have no conflict of interest.

References

1. Global information security practices: survey key findings and trends. <URL: `http://www.pwc.com.`>, 2015.
2. Network-Intrusion-Detection-using-Machine-Learning. <URL: `https://github.com/Anshumank399/Network-Intrusion-Detection-using-Machine-Learning`>, 2018.
3. D.W. Archana and P.N. Chatur. Comparison of firewall and intrusion detection system. *International Journal of Computer Science and Information Technologies*, 5 (1):674–678, 2014.
4. A.M. Bagirov. Modified global k-means algorithm for minimum sum-ofsquares clustering problems. *Pattern Recognition*, 41(10):3192–3199, 2008.
5. A.M. Bagirov, B. Ordin, G. Ozturk, and A.E. Xavier. An incremental clustering algorithm based on hyperbolic smoothing. *Computational Optimization and Applications*, 61(1):219–241, 2015.
6. A.M. Bagirov, S. Taheri, and J. Ugon. Nonsmooth DC programming approach to the minimum sum-of-squares clustering problems. *Pattern Recognition*, 53:12–24, 2016.
7. M. Breunig, H. Kriegel, R. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.
8. J. Cannady and J. Harrell. A comparative analysis of current intrusion detection technologies. 2000.
9. E.M. Chakir, Ch. Codjovi, Y.I. Khamlichi, M. Moughit, and H. First Settat. False positives reduction in intrusion detection systems using alert correlation and data mining techniques. *International Journal of Advanced Research in Computer Science and Software Engineering IJARCSSE*.
10. P.T. Chen and Ch.S Laih. Idsic: an intrusion detection system with identification capability. *International Journal of Information Security*, 7(3):185–197, 2008.
11. A. Dickson and C. Thomas. Optimizing false alerts using multiobjective particle swarm optimization method. *IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems*, 2015.
12. N. Gupta, K. Srivastava, and A. Sharma. Reducing false positive in intrusion detection system: a survey. volume 7, 2016.
13. F. Hachmi, K. Boujenfa, and M. Limam. An optimization process to identify outliers generated by intrusion detection systems. *Security and Communication Networks*, 8(18):3469–3480, 2015.
14. M.F. Jiang, S.S. Tseng, and C.M. Su. Two-phase clustering process for outliers detection. *Pattern Recognition Letters*, 22(6):691–700, 2001.
15. K. Julisch. Clustering intrusion detection alarms to support root cause analysis. *ACM Transactions on Information and System Security*, 6(4):443–471, 2003.
16. P.U. Kadam and M. Deshmukh. Various approaches for intrusion detection system: an overview. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(11):6894–6902, 2014.
17. U. Kanika. Security of network using Ids and Firewall. *International Journal of Scientific and Research Publications*, 3 (6), 2013.

18. Z. Li, A. Das, and J. Zhou. Usaid: Yunifying signature-based and anomaly-based intrusion detection. In T.B. Ho, D. Cheung, and H. Liu, editors, *Advances in Knowledge Discovery and Data Mining*, pages 702–712, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
19. D. Lian, L. Xu, Y. Liu, and J. Lee. Cluster-based outlier detection. *Annals of Operations Research*, 168(1):151–168, 2009.
20. H.J. Liao, Lin Y.Ch. Lin, Ch.H.R, and K.Y. Tung. Review: Intrusion detection system: a comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.
21. M. Lichman. UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences. <URL: `http://archive.ics.uci.edu/ml`>, 2013.
22. R.P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. Mcclung, and et al. Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation. *DARPA Information Survivability Conference and Exposition*, 2:12–26, 2000.
23. J.H. Madsen. Distance and density-based outlier detection. <URL: `URL: https://github.com/jhmadsen/DDoutlier`>, 2018.
24. J. McHugh. Intrusion and intrusion detection. *International Journal of Information Security*, 1(1):14–35, 2001.
25. P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli. A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE Communications Surveys Tutorials*, 21(1):686–728, 2019.
26. A.S. Mujumdar. Analysis of signature-based and behavior-based anti-malware approaches. 2013.
27. M. Nour and S. Jill. Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). *Military Communications and Information Systems Conference, IEEE*, pages 1–6, 2015.
28. C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
29. B. Ordin and A.M. Bagirov. A heuristic algorithm for solving the minimum sum-of-squares clustering problems. *Journal of Global Optimization*, 61(2):341–361, 2015.
30. S.D. Pachgade and S.S. Dhande. A heuristic algorithm for solving the minimum sum-of-squares clustering problems. *International Journal of Advanced Research in Computer Science and Software Engineering*, 2(6):12–16, 2012.
31. V. Pareek, A. Mishra, A. Sharma, R. Chauhan, and S. Bansal. A deviation based outlier intrusion detection system. In Chaki N. Nagamalai D. Meghanathan N., Boumerdassi S., editor, *Recent Trends in Network Security and Applications*, pages 395–401. Springer, Berlin, Heidelberg, 2010.
32. H. Rizk, M. ElGokhy, and A. Sarhan. A hybrid outlier detection algorithm based on partitioning clustering and density measures. *2015 Tenth International Conference on Computer Engineering and Systems (ICCES)*, pages 175–181, 2015.
33. S. Rubin, S. Jha, and B.P. Miller. Automatic generation and analysis of nids attacks. In *Proceedings of the 20th Annual Computer Security Applications Conference*, pages 28–38, Washington, DC, USA, 2004. IEEE Computer Society.
34. E. Schubert, A. Zimek, and H.P. Kriegel. *Generalized outlier detection with flexible kernel density estimates*, pages 542–550. 2014.
35. Y. Seo, H. Lee, and S. Lee. Outlier removal by convex optimization for l-infinity approaches. In W. Toshikazu, H. Fay, and L. Stephen, editors, *Advances in Image and Video Technology*, pages 203–214. Springer, Heidelberg, 2009.
36. R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. *Security and Privacy*, pages 305–316, 2010.
37. G.P. Spathoulas and S.K. Katsikas. Reducing false positives in intrusion detection systems. *Computer and Securetiy*, 29(1):35–44, 2010.
38. G.P. Spathoulas and S.K Katsikas. Reducing false positives in intrusion detection systems. *Computers and Security*, 29(1):35–44, 2010.
39. P.N. Tan, M. Steinbach, and V. Kumar. *Introduction to data mining*. Addison-Wesley Longman Publishing Co., Inc., 2005.
40. M.F. Umer, M. Sher, and X. Bi. A two-stage flow-based intrusion detection model for next-generation networks. In *PloS one*, 2018.
41. D. Wagner and P. Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 255–264, New York, NY, USA, 2002. ACM.
42. C.H. Wang. Outlier identification and market segmentation using kernel based clustering techniques. *Expert Systems with Applications*, 36(2):3744–3750, 2009.
43. K. Zhang, M. Hutter, and H. Jin. A new local distance-based outlier detection approach for scattered real-world data. In T. Theeramunkong, B. Kijsirikul, N. Cercone, and T.B. Ho, editors, *Advances in Knowledge Discovery and Data Mining*, pages 813–822. Springer Berlin Heidelberg, 2009.
44. Q. Zhu, J. Feng, and J. Huang. Natural neighbor: A self-adaptive neighborhood method without parameter k. *Pattern Recognition Letters*, 80:30 – 36, 2016.