An outlier ensemble for unsupervised anomaly detection in honeypots data

Lynda Boukela, Gongxuan Zhang, Samia Bouzefrane, Zhou Junlong

An outlier ensemble for unsupervised anomaly detection in honeypots data

Lynda Boukela[a,*], Gongxuan Zhang[a], Samia Bouzefrane[b] and Junlong Zhou[a]
[a]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[b]*CEDRIC lab, Conservatoire National des Arts et Métiers, Paris, France*

Abstract. Nowadays, computers, as well as smart devices, are connected through communication networks making them more vulnerable to attacks. Honeypots are proposed as deception tools but usually used as part of a proactive defense strategy. Hence, this article demonstrates how honeypots data can be analyzed in an active defense strategy. Furthermore, anomaly detection based on unsupervised machine learning techniques allows to build autonomous systems and to detect unknown anomalies without the need for prior knowledge. However, the unsupervised techniques applied for honeypots data analysis do not value the advantages of these tools' data, particularly the high probability that they include a large number of previously unseen anomalies with unexpected and diverse patterns. Therefore, in the present work, the aim is to improve the unsupervised anomaly detection in honeypots data by varying the data feature subset and the parameterization of the anomaly detection algorithm. To this purpose, an outlier ensemble with LOF (Local Outlier Factor) as a base algorithm is proposed. The ensemble outperforms existing solutions as depicted in the experiments where a detection rate higher than 92% is achieved.

Keywords: Outlier ensembles, network security, anomaly detection, honeypots

1. Introduction

Because computer networks are more and more accessible online, they are confronted with a plethora of security threats. Therefore, for a better defense by means of interacting with intruders, honeypots, a rapidly developing technology [25], whose value lies in being probed, attacked, or compromised [22], are widely used. These tools are of great importance because they gather data related to attackers' activities allowing in this way the extraction of valuable information and the detection of previously unseen attacks and diverse anomalies before harming any production system. Therefore, robust and appropriate techniques and algorithms for analyzing deeply honeypots data while valuing their advantages are required.

In honeypots data analysis, the majority of previous works is essentially conducted as part of a proactive defense strategy where attack patterns are learned after their occurrence. Thus, in this paper, it is demonstrated how honeypots data can be exploited to be part of an active defense strategy. Furthermore, previous works rely either on supervised or unsupervised machine learning techniques. The promising trend for anomaly detection is the use of machine learning techniques working in an unsupervised fashion. Indeed, these techniques help to build autonomous systems which detect both known and unknown

*Corresponding author: Lynda Boukela, School of Computer Science and Engineering, Nanjing University of Science and Technology, 200 Xiaolingwei Street, Nanjing 210094, China. E-mail: lyndaboukela@njust.edu.cn.

anomalies without the need for prior knowledge or training phase. Nevertheless, with the unsupervised techniques for honeypots data analysis, a small effort has been put in to exploit deeply the advantages of these data because the techniques either search in the full feature space of the data or use approaches that find anomalies as their side-products. Indeed, as it has been demonstrated in [6], anomalies can be hidden in lower-dimensional projections of the data, and these projections can be of different combinations and sizes. Furthermore, when using techniques such as clustering, the main objective is to group similar data points while the outliers are only side-products. Therefore, it is believed that using unsupervised methods which have outlier mining as their primary objective will permit to obtain better results. In the present work, it is referred to these techniques with the term "outlier-based techniques".

Outlier-based techniques that operate in an unsupervised fashion are numerous [17], and their efficiency has been demonstrated [2]. However, when working with these techniques on high dimensional data, as in the case of honeypots data that contain anomalies of diverse and eventually of unknown patterns, important challenges can be identified: sparsity of the data a.k.a. the curse of dimensionality, outliers that can be hidden in particular data subspaces, the choice among different accurate algorithms, etc. To address these challenges, the ensembles analysis approach has been extended to the unsupervised anomaly detection field, especially to the outlier-based techniques [4,5]. This approach is known as outlier ensembles, it combines results from different algorithms or iterations of one base algorithm fed with different parameters or applied to different data subspaces. Consequently, in this paper, an outlier ensemble is proposed as a solution for anomaly detection in honeypots data. To the best of our knowledge, there is no work on the use of outlier ensembles techniques for anomaly detection in honeypots data. Our contributions are as follows:

1. In order to include honeypots in an active defense strategy, a scheme to exploit their data appropriately is introduced.
2. The problem of previously unseen anomalies is addressed by using the outlier-based anomaly detection algorithm LOF. This algorithm works in a completely unsupervised fashion, without any need for prior knowledge or training phase.
3. An outlier ensemble that combines results of several executions of LOF with different parameters and different data subspaces is proposed to tackle the high dimensionality of the data and the anomalies hidden in subspaces.
4. Three different functions are explored to combine the results of the ensemble models.
5. The proposed outlier ensemble has been evaluated and compared to existing solutions.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 presents the proposed solution based on outlier ensembles for honeypots data exploitation and analysis. In Section 4, the experimental results are depicted and discussed. Finally, concluding remarks are given in Section 5.

2. Related work

Since their introduction, honeypots have been progressively gaining importance in network defense strategies. These deception tools allow collecting data that contain information of great usefulness. As can be seen in detail in Table 1, previous works on honeypots traffic data analysis aimed essentially to: (i) gain information about attacks and discover their patterns and root causes [7,8,16,21], (ii) predict attacks [14,23,26,27], and, (iii) detect and classify attacks [1,9,18–21].

Additionally, both supervised and unsupervised machine learning techniques have been used for analyzing honeypots data. Supervised techniques can be found in [1,7,9,23,26,27]. In [1], the authors

Table 1
Summary of works on honeypots data analysis

Ref.	Study objective	Method type	Used techniques	Data type	Results type
[1]	Data classification as normal or suspicious	Supervised	ANN, Decision trees, and k-NN	Record data	Labels
[7]	Statistical analysis: Temporal and spatial distribution, attacked protocols, involved autonomous systems ... etc.	Supervised	Data distributions and dependencies	Record data	Statistics
[8]	Attacks' root causes characterization	Unsupervised (applied on attack data)	Association rules learning. (Apriori algorithm)	Record data (Honeynet Research Alliance)	Association rules
[9]	Attack classification into severe and not-so-severe attacks	Supervised	Logistic Regression, Support Vector Machine (SVM), J48, Naive Bayes, PART, OneR, and k-Nearest Neighbors (k-NN)	Record data	Labels
[14]	Collaboration, attack pattern identification and attack prediction	Unsupervised (applied on attack data)	Sequential rule mining (TopKRules)	Record data (SABU platform)	Sequential rules
[16]	Attack patterns discovery	Unsupervised (applied on attack data)	Graph-based clustering	Time series of attacks (leurre.com project)	Plausible root causes
[18,19]	Attacks identification and characterization	Unsupervised	Change detection algorithm based on time series, Clustering ensemble and subspace clustering	Time series + Record data	Filtering rules + Anomalies ranking
[20]	Attack detection	Unsupervised (with profile building)	Principal component Analysis	Record data	Labels
[21]	Attack characterization	Unsupervised	Principal Componenet Analysis	Record data	Principal components + Extreme observations
[23]	Potentially vulnerable hosts prediction	Supervised	C4.5, BayesNet, Decision Table, and Naive-Bayes algorithms	Record data (LongTail Attack records)	Security rules
[26]	Attack prediction	Supervised	Analysis of the statistical properties of attack data with a framework based on the concept of stochastic cyber-attack process	Record data	Predictions
[27]	Attack prediction	Supervised	Extreme values theory, Time series theory, FARIMA + GARCH models	Time series	Predictions

have analyzed traffic from honeypots and production system in order to differentiate between internet normal traffic, anomalies and real attacks. For this purpose, the well-established supervised approaches k-nearest neighbor (KNN), Artificial Neural Networks (ANN) and Decision Trees have been used. The authors in [7] aimed at identifying and quantifying dependencies and distributions within honeynet data by investigating features such as: temporal and spatial distributions, attacked protocols, involved autonomous systems, etc. In [9], SSH attacks occurring in a honeypot are classified as "severe" or "not-so-severe" attacks. For this purpose, Logistic Regression, Support Vector Machine (SVM), J48, Naive Bayes, PART,

OneR, and k-Nearest Neighbors have been used. In [23], C4.5, BayesNet, Decision Table, and Naive-Bayes algorithms have been applied to predict potentially vulnerable hosts, hence allowing to define security rules for Software Defined Network (SDN) controllers and to block the subnet of the responsible IP. The authors in [26] analyzed the statistical properties of honeypot-captured cyber-attack data with a framework based on the concept of stochastic cyber-attack process. They confirm that exploiting the long-range dependence (LRD) that is exhibited by these attacks can help to achieve prediction accurately. The LRD has also been exploited in [27] for attacks prediction in terms of attack rate, for this purpose, the extreme value theory, the time series theory, and the gray-box models have been used.

Unsupervised techniques have been applied to analyze honeypots data in [8,14,16,18–21]. One of the earliest works is presented in [8]. Herein, the authors used the Apriori algorithm to extract association rules and to build clusters with the goal of finding the root causes of attacks occurring on distinct ports sequences. The algorithm has been applied with a set of traffic properties such as the number of destination IP addresses and the number of packets. In [14], sequential rule mining has been used to identify common attack patterns and to derive rules for predicting attacks. Additionally, exploiting a collaborative environment allowed attack prediction in multiple dimensions. To run their experiment, the authors used the alert sharing platform SABU. The authors in [16] proposed a framework for attack patterns discovery in honeypots data. The work emphasized on the temporal correlations between attacks where clustering, with an appropriate measure, was performed on time series of attacks. As explained, the ultimate goal is to help the analyst figuring out if a new imminent attack can be attributed to the very same group of attacks. In [21], the Principal Component Analysis (PCA) has been applied to data from low-interaction honeypots to characterize attackers' activities by finding and separating dominant groups of activities and finding outliers. In [20], the authors presented a method for detecting new attacks in honeypots traffic. A PCA model is built for the attacks seen in honeypots data, later, new observations are projected onto the residuals of the predefined PCA model and are tested against a predefined threshold. If a large deviation from the model is observed, the observation is considered as anomalous. Finally, recent works [18,19] used a two-phase algorithm, namely UNADA, to detect and characterize anomalies occurring in honeypots. In the first phase, the authors use an algorithm for change detection in time series to flag anomalous time-slots. In the second phase, they aggregate the flows of the flagged time-slot and describe them with a set of traffic features to perform clustering with DBSCAN and identify small-size clusters and outliers. To avoid the lack of robustness in cluster analysis techniques, the authors used the notions of clustering ensemble with subspace clustering, and multiple clusterings combinations based on evidence accumulations and inter-clustering associations.

As it can be noticed, most works on honeypots data analysis are essentially conducted as part of a proactive defense [7,8,14,16,21,23,26,27]. However, detecting intrusions and anomalies targeting the honeypots on time is also crucial. Indeed, this can allow triggering early warnings that will help to prevent threat propagation and to avoid the damage of production systems. Furthermore, using supervised techniques as in [1,9,23] has a major disadvantage because it is not possible to detect unknown attacks. Additionally, labeled data and model training are required. These issues do not exist with the unsupervised techniques. However, in the aforementioned works that rely on unsupervised techniques, several issues can be noticed. In [20], the method relies heavily on historical data. In [18,19] as explained by the authors, the data features considered are those characterizing well already encountered traffic anomalies, i.e. known anomalies. And, in [18–20], the approaches that have been used find outliers as their side-products. However, when mining for anomalies in honeypots data, consideration should be given to the characteristics and to the advantages of these data in order to analyze them deeply with appropriate techniques. Consequently, in this paper and unlike previous works, several aspects for detecting anomalies

in honeypots data are taken into account. Firstly, honeypots data consist mostly of anomalies. Therefore, a scheme to exploit them in a completely unsupervised way is proposed. Secondly, data described with a large number of diverse features are used and an appropriate algorithm has been applied because it is highly probable that some of these anomalies are previously unseen. Finally, the ensemble approach is explored in this work because when working on high dimensional data as in the case of honeypots data, the curse of dimensionality is encountered, and because each anomaly is generally characterized by few features only and certain feature subsets can be more relevant than others for detecting some anomalies [10]. Furthermore, since network anomalies are diverse, an algorithm with different parameterizations can detect different anomalies.

3. Honeypots data exploitation and analysis

In the field of computer networks, an anomalous event can be a consequence of:

- Intrusions and attacks such as Denial of Service (DoS) or Distributed DoS (DDOS) attacks, SQL injections, etc.
- Network failures such as traffic congestion, packets loss, etc.

Traditionally, signature-based intrusion detection systems are used to secure computer networks. With these systems, each time a new attack occurs, security experts design a signature for it and update the database of the intrusion detection system. Therefore, an activity is flagged as anomalous only if it presents a pattern that matches a known attack signature. As it can be concluded, these systems require a heavy human intervention for designing the attack signatures and do not cope with previously unseen anomalies. The second category of systems used for detecting anomalous activities in computer networks traffic is anomaly-based. In this case, normal activities are modeled and any activity that deviates from that model is considered anomalous. These anomaly-based techniques are also non-autonomous and rely heavily on prior knowledge about what constitutes normal behavior.

Honeypots are considered as traps for intruders usually used as part of a proactive defense strategy. Gaining deeper insight and understanding of the attacks after their occurrence is of major importance, however, detecting intrusions and anomalies targeting the honeypots on time, as part of an active defense strategy, is also crucial. Indeed, this would allow early detection of the attacks and triggering early warnings in order to enable human experts to take the necessary countermeasures, essentially preventing threat propagation and avoiding the damage of production systems.

Network traffic gathered with honeypots is different from the conventional network traffic. In fact, depending on the honeypots deployment and the level of their interaction with the intruders, the gathered data can be of tremendous value. Consequently, above-highlighted systems are not sufficient to deal with honeypots data, and, to design a well-suited scheme to exploit these data, their properties should be considered carefully. Firstly, the majority of the honeypots data is illegitimate. Secondly, it is highly probable that previously unseen anomalies occur, and the patterns of these anomalies can be discovered in the data. The main emphasis of our work is placed on the latter consideration, i.e., the detection of anomalous events occurring in honeypots data. However, a scheme to demonstrate how honeypots can be part of an active defense strategy and to address the problem of large amount of anomalous traffic is proposed, since with unsupervised anomaly detection techniques, that are exploited in this paper, it is assumed that anomalies represent a minority of the overall data.

As illustrated in Fig. 1, after gathering the traffic exchanged with the honeypots and pre-processing it, an appropriate conventional signature-based network intrusion detection system can be used to detect

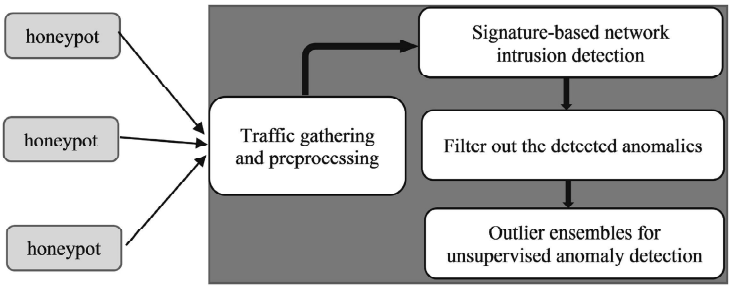

Fig. 1. Functioning of the scheme for exploiting honeypots data.

known anomalies in the data. After filtering out these anomalies, the remaining data will consist of normal activities and previously unseen anomalies and attacks. Subsequently, our anomaly detection solution can be applied to detect the remaining anomalous events.

The above-mentioned techniques, i.e. the signature-based detectors and the anomaly-based detectors which rely on normal traffic modeling are not able to cope with unknown attacks or need a model of the historical normal behavior. Furthermore, these techniques rely on a heavy human intervention. Due to these disadvantages, in recent years, the trend is for the analysis of network traffic with completely unsupervised machine learning techniques.

3.1. Unsupervised anomaly detection

In the data analysis field, anomaly detection is the process of finding unexpected patterns in data. These anomalies are also called outliers. When designing or applying an anomaly detection technique, various aspects should be taken into consideration [24] such as the nature of the input data (categorical or continuous, spatial, graph or sequence data), the output of the algorithm that can be either labels or scores, the types of anomalies to detect (point, collective or contextual anomalies), the availability of data labels, etc. Depending on the availability of data labels, machine learning techniques for anomaly detection are categorized into supervised, unsupervised, and semi-supervised. For our anomaly detection in honeypots data, point anomalies are targeted and the detection is made without relying on data labels, i.e. in an unsupervised fashion, and the results are formulated as labels.

Unsupervised anomaly detection techniques have gained a great interest in the research community especially in the field of network security due to their numerous advantages. Firstly, these techniques allow building autonomous systems which do not require a strong human intervention. Secondly, they operate by analyzing only the internal structure of the data, and for this purpose, properties such as distance and density measures are used. Consequently, these techniques work without a training phase and distinguish between normal and anomalous instances without the need for data labels. Finally, the important improvement brought by these techniques is their ability to detect new types of anomalies.

Numerous unsupervised anomaly detection algorithms have been developed, and they rely on the assumptions that anomalies are less frequent compared to normal instances and show rare and nonconforming patterns. Furthermore, several taxonomies were proposed to categorize unsupervised anomaly detection techniques. For instance, in [13] the authors classified unsupervised anomaly detection techniques into nearest-neighbors based, clustering-based, statistical, subspace-based, classification-based and others.

In our case, a nearest-neighbors based algorithm is used, namely LOF [15]. Nearest-neighbors based techniques are preferred to others since their primary objective is to seek for outliers while these deviating

points are only side-products of the other techniques such as the clustering-based. Numerous outlier-based algorithms have been proposed in the literature and making choice is not an easy task. However, the efficacy of LOF has been demonstrated, including in the network traffic analysis field [2], and, most importantly, the approach detects outliers lying in neighborhoods with different properties [15].

LOF is a density-based anomaly detection approach that assigns a degree of outlierness to each point in the data. To compute the degree of outlierness, firstly, a *kDistance*(p) is assigned to each point p that represents its distance to the k-th nearest neighbor. All points within this distance make the set of k nearest neighbors of p denoted as $N_k(p)$. At this point, it is essential to mention that a neighbor of p may not be unique, making the cardinality of $N_k(p)$ greater than k. The *kDistance* is then used to compute the reachability distance of point p with respect to point o, denoted *reachDist*$_k(p, o)$, as in the following:

$$\textit{reachDist}_k(p, o) = \max\{\textit{kDistance}(o), d(p, o)\} \tag{1}$$

Later, by replacing k with *MinPts*, the local reachability density of point p is defined as:

$$\textit{lrd}_{\textit{MinPts}}(p) = 1 \Big/ \left[\frac{\sum_{o \in N_{\textit{MinPts}}(p)} \textit{reachDist}_{\textit{MinPts}}(p, o)}{|N_{\textit{MinPts}}(p)|}\right] \tag{2}$$

Finally, the local outlier factor of a point p can be computed as follows:

$$\textit{LOF}_{\textit{MinPts}}(p) = \frac{\sum_{o \in N_{\textit{MinPts}}(p)} \frac{\textit{lrd}_{\textit{MinPts}}(o)}{\textit{lrd}_{\textit{MinPts}}(p)}}{|N_{\textit{MinPts}}(p)|} \tag{3}$$

The degree of outlierness depends on the number of local neighbors *MinPts* used to define the local density for each data point and this number is an input parameter of the algorithm. When a data point has a local outlier factor greater than a certain threshold t, which is a second input parameter, it is considered as an outlying point.

3.2. Outlier ensembles

An emerging and promising technique in the field of unsupervised anomaly detection is outlier ensembles [4,5]. The notion of ensembles analysis refers to the combination of results from different algorithms or iterations of one base algorithm fed with different parameters or different data subspaces. The main purpose of this approach is to improve the accuracy of the anomaly mining process by combining results from diverse algorithms and to address the curse of dimensionality when feature subspaces are used. Indeed, it has been demonstrated that making a consensus of results from different models can achieve better results than those of the base algorithms, including in anomaly detection [3]. However, when building an ensemble of unsupervised anomaly detection algorithms [4], careful consideration should be given to the diversity of the algorithms used to construct the ensemble, their accuracy and, the combination of their results.

In our case, to improve the unsupervised anomaly detection, an outlier ensemble with LOF as a base algorithm is proposed. Close to our work, the general-purpose feature bagging solution for outlier detection presented in [3]. In [3], diversity is induced by using different sets of features while in our case, the diversity of the ensemble is induced by using different sets of features, but also by feeding the outlier detection algorithm with different subsets of the full set of well-performing values of the parameter *MinPts* in terms of detection and false alarms rates. It is expected that these different parameters will contribute to improving the accuracy of the detection algorithm. Furthermore, unlike [3], the results of the sequential execution of LOF algorithm are combined with three different functions, from which,

Algorithm 1 Parameterization and Feature Subspace based Outlier Ensemble (PFSOE)

Input:
S: number of iterations
t: threshold
X: $n \times m$ tabular data
$MP = [v_1, v_2]$: set of values for *MinPts*
Output:
Labels: $n\times$ 1 array
Begin
for $i \leftarrow 1, S$ **do**
  Select randomly a distinct subset M_i from the set of m features of X;
  Use the feature subset M_i to create a subspace X_i of data X;
  Select randomly a distinct subset P_i from the set of values MP for the parameter *MinPts*;
  Use LOF with the subset P_i and subspace X_i to compute the degree of outlierness D_i of each instance in X;
end for
$D = \{D_1, D_2, \ldots, D_s\}$;
Labels = Combination (D, t);
end

the well-performing one, i.e. the one that achieves the optimal detection and false alarms rates, will be retained as our final combination function. The general approach of the proposed parameterization and feature subspace based outlier ensemble (PFSOE) is presented in Algorithm 1.

After collecting traffic traces exchanged with the honeypots, representative data with descriptive features should be extracted and constructed. As can be seen from the general algorithm, our technique uses tabular data $\boldsymbol{X}$ with n rows and m columns. $\boldsymbol{X}$ is a sliding window used to read honeypots data, each row in $\boldsymbol{X}$ represents a flow aggregated from the captured traffic and each column represents an attribute describing the flows. To avoid problems related to features belonging to different ranges, they are standardized by using the z-score normalization with the following formula:

$$Z = \frac{(X - \mu)}{\sigma} \tag{4}$$

where μ represents the mean and σ the standard deviation.

Once the data of the current window are normalized, an outlierness score is assigned to each data instance and finally, the decision on whether it is anomalous or not is made.

At each iteration i, a feature subset $\boldsymbol{M_i}$ is randomly selected from the original m features of $\boldsymbol{X}$, and a subspace $\boldsymbol{X_i}$ of the original data is produced. At each iteration, a subset of values $\boldsymbol{P_i}$ are selected randomly for the parameter *MinPts* from a set $\boldsymbol{MP} = [v_1, v_2]$ of predefined values. Subsequently, LOF, fed with the different *MinPts* values in $\boldsymbol{P_i}$, is applied to the created data subspace $\boldsymbol{X_i}$ and the degree of outlierness of each data point is calculated and saved in table $\boldsymbol{D_i}$.

At the end of the S iterations, a set $\boldsymbol{D} = \{\boldsymbol{D_1}, \boldsymbol{D_2}, \ldots, \boldsymbol{D_s}\}$ is obtained, where $\boldsymbol{D_i}$ is a vector that contains the degree of outlierness of each data point in $\boldsymbol{X}$ resulted from iteration i. To make a consensus on the results of the S iterations over LOF, the results are combined by using an appropriate combination function.

3.3. Combination function

The choice of a combination function is of tremendous importance [4,5]. In our case, in order to combine the results obtained from the different models of our ensemble, three combination functions are explored, namely, the averaging combination function, the maximum combination function, and the majority voting combination function. The three functions are defined as follows:

– The averaging combination function: herein, the average degree of outlierness $\boldsymbol{D_{final}}$ is defined as follows:

$$D_{final} = \frac{\sum_{i=1}^{S} D_i}{S} \tag{5}$$

The final degree of outlierness of each data point $D_{final}(j)$ is compared to the predefined threshold t. If $D_{final}(j)$ is greater than t, then it is labeled as anomalous. It is labeled as a normal data point otherwise as explained in Eq. (7). These results are then saved in the algorithm output *Labels*.

– The maximum combination function: in this case, the maximum degree of outlierness of each data point $D_{final}(j)$ is obtained as follows:

$$D_{final}(j) = \max_{i}(D_i(j)) \tag{6}$$

As for the averaging function, if the final degree of outlierness $D_{final}(j)$ of a point j is greater than a predefined threshold t, then it is labeled as anomalous. It is considered as a normal data point otherwise (Eq. (7)). These results are then saved in the output vector of the algorithm *Labels*.

$$Labels(j) = \begin{cases} 1 \textit{ if } D_{final}(j) > t \\ 0 \textit{ otherwise} \end{cases} \tag{7}$$

where t is a predefined threshold.

– The majority voting combination function: with this function, the decision on the maliciousness of each point is made at the level of each model of the ensemble. Thus, at the end of the S iterations, a set $\boldsymbol{L} = \{\boldsymbol{L_1}, \boldsymbol{L_2}, \ldots, \boldsymbol{L_s}\}$ is obtained, where $\boldsymbol{L_i}$ represents the labels of the data observations obtained from iteration i as follows

$$L_i(j) = \begin{cases} 1 \textit{ if } D_i(j) > t \\ 0 \textit{ otherwise} \end{cases} \tag{8}$$

where t is a predefined threshold.
Later, the final decision on the maliciousness of an observation is made by taking into account the decision of the majority of the models and saved in *Labels* as depicted in the following equation:

$$Labels(j) = \begin{cases} 1 \textit{ if } \sum_{i=1}^{S} L_i(j) > bound * S \\ 0 \textit{ otherwise} \end{cases} \tag{9}$$

where *bound* is the percentage of the ensemble models to consider as the majority.

4. Experiments and results

In this section, the honeypots data used in the experiments are presented, details are given on how the different parameters of the ensemble were fixed and the experiments conducted in order to choose the appropriate combination function are presented. Finally, the results of the cross-validation and the comparison of the performance of our work are presented and discussed.

4.1. Experimental setup

To evaluate the performance of our PFSOE algorithm, our experiments were performed on the Kyoto 2006+ dataset [12], which is open to the public at [11]. The dataset consists of daily traffic captured from 2006 to 2015. It is saved as text files that contain traces collected from a network of honeypots of different types that have been deployed at Kyoto University. The data has no missing values and it is described with 24 attributes, 14 statistical features detailing each observation and 10 additional features which

Table 2
Data attributes used in the experiments [12]

Feature name	Details
Duration	The length (seconds) of the connection
Source bytes	The number of data bytes sent by the source IP address
Destination bytes	The number of data bytes sent by the destination IP address
Count	The number of connections whose source IP address and destination IP address are the same to those of the current connection in the past two seconds
Same srv rate	% of connections to the same service in Count feature
Serror rate	% of connections that have "SYN" errors in Count feature
Srv serror rate	% of connections that have "SYN" errors in Srv count (the number of connections whose service type is the same to that of the current connection in the past two seconds) feature
Dst host count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose source IP address is also the same to that of the current connection
Dst host srv count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose service type is also the same to that of the current connection
Dst host same src port rate	% of connections whose source port is the same to that of the current connection in Dst host count feature
Dst host serror rate	% of connections that have "SYN" errors in Dst host count feature
Dst host srv serror rate	% of connections that "SYN" errors in Dst host srv count feature
Label (for evaluation)	% Indicates whether the session was attack or not; '1' means the session was normal, '-1' means known attack was observed in the session, and '-2' means unknown attack was observed in the session

Table 3
Properties of the data file used in the experiments

Data type	Original file	Reduced file
Normal instances	68998 (46.87%)	68076 (98.66%)
Known attacks	77268 (52.48%)	0 (0%)
Unknown attacks	956 (0.65%)	924 (1.34 %)

can be used for further analysis and evaluation of intrusion detection systems. For our experiment, the *20090822.txt* file that contains a high number of unknown anomalies [12] was chosen. Furthermore, as detailed in Table 2, the 12 continuous attributes out of the 14 statistical ones were retained and the attribute *Label* was used for the evaluation of our work. The evaluation attribute classifies each observation as normal, known attack or unknown attack with the labels 1, -1, -2, respectively. To fit the results of our algorithm, 1 is replaced with 0 to label normal instances, and 1 is used instead of -1 and -2 to label anomalous instances.

Since the major part of the data is attacks, it is assumed that any signature-based intrusion detection system can be used to filter out the known attacks as explained before and depicted in Fig. 1. Therefore, to conduct our experiments, the number of anomalies is reduced by keeping only the unknown malicious events. The properties of the data file before and after the changes were brought are described in Table 3. Furthermore, the number of observations has been adjusted to fit the size of the sliding window of our ensemble.

To conduct our experiments, the dataset was split into two equal sets: a training set and a test set. The training set was used in order to fix the range of values for the parameter *MinPts* and explore the different combination functions. The test set was used for the comparison of our work to existing solutions. In order to have a more reliable estimate of the performance of the algorithm, we have also used 10-fold

cross-validation to compare the proposed PFSOE and to choose the range of values for the parameter *MinPts* and the most accurate combination function.

To evaluate the performance and effectiveness of the proposed ensemble, the following evaluation metrics were used: Accuracy, true positives rate (TPR) and false positives rate (FPR). True positives (TP) are events labeled as anomalous that are indeed anomalies, while false positives (FP) are events labeled as anomalous that are, in reality, legitimate events. Normal events that were labeled as normal are called true negatives (TN), while the anomalous events that were labeled as normal are called false negatives (FN). Accuracy, true positives rate and false positives rate are defined as follows:

$$Accuracy = \frac{(TP + TN)}{(FP + FN + TP + TN)} \tag{10}$$

$$TPR = \frac{TP}{(FN + TP)} \tag{11}$$

$$FPR = \frac{FP}{(FP + TN)} \tag{12}$$

Furthermore, to illustrate the performance comparison, the area under the curve (AUC) and the Receiver Operating Characteristic (ROC) curves which depict the TPR as a function of the FPR were used.

In our experiments, the size of the sliding window $\boldsymbol{X}$ was fixed to 500 data instances, this window size allows considering events occurring over a period of time of several minutes (more than 15 mins) that is assumed sufficiently representative of the network traffic events. Each sliding window was analyzed sequentially for a fixed number of iterations $S = 49$ and feature subsets of a size that belongs to $[m/2, m]$ were built, where m is the size of the full feature set describing the data. The choice of these parameters was greatly influenced by [3].

4.2. Results and discussion

4.2.1. Selection of the MinPts values

Several experiments have been conducted to analyze the sliding windows of the training set in order to find the well-performing values of the parameter *MinPts*. LOF was applied on the sliding windows with a large set of values for *MinPts*. Furthermore, the algorithm has been applied on different data subspaces. After the manual inspection of the obtained results from each window, the range of well-performing values for *MinPts*, $MP = [38, 88]$, was chosen. Our choice of the values has been made by looking at the values of *MinPts* that performed well in the majority of the feature subsets. As explained previously, a randomly generated subset of values from the set MP is used in our ensemble, unlike what has been proposed in [15] where the authors used the whole set of values in the ensemble MP. The size of the subset used in our ensemble varies from 6 to 10. By using these subsets of the set MP instead of the full set, the time complexity of our ensemble can noticeably be lowered while taking advantage of the diversity of the parameter.

4.2.2. Selection of the combination function

In order to combine the results of the ensemble and make a consensus of the different models, three combination functions were explored, namely, the averaging combination function, the maximum combination function, and the majority voting combination function. This experiment has been conducted on the training data. Furthermore, to evaluate the performance of the different combination functions, it has been ensured that all the 49 models were the same when testing the functions, i.e. the same feature subsets and the same sets of values for *Minpts* in the different iterations were used.

Table 4
AUC of the different combination functions

Combination function	Area under the curve
Maximum combination function	0.8314
Averaging combination function	0.9480
Majority voting combination function, bound = 50%	0.9668
Majority voting combination function, bound = 60%	0.9670
Majority voting combination function, bound = 70%	**0.9701**
Majority voting combination function, bound = 80%	0.9133

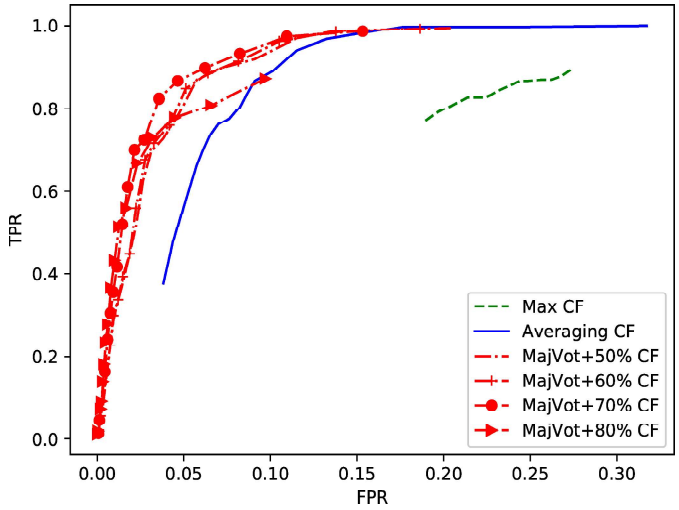

Fig. 2. ROC curves of the different combination functions.

The ROC curves, obtained with a variable threshold, for the TPR and FPR of the PFSOE algorithm executed with the three combination functions, are presented in Fig. 2. And, the AUC of the different combination functions are shown in Table 4. As can be seen, both the majority voting and the averaging functions outperform the maximum function. Indeed, this function causes a high number of false positives because a large LOF value can be assigned to non-anomalous observations by a specific model. From the results, it can also be noticed that the majority voting function outperforms the averaging function. The low performance of the averaging combination function can be explained by looking at the final outlierness scores of the data points. These scores are obtained by averaging the results of the different models which causes loss of accuracy if some models are of low accuracy. The majority voting function doesn't make any changes in the original scores, and, unlike the maximum combination function, it considers the decision of several models instead of only one model, hence it outperformed the two functions. However, as mentioned earlier, different upper bounds were applied to the majority voting instead of the 50% bound. In the same figure, it can be seen that the function's bound was set to 50%, 60%, 70%, and 80% and the bound that performed the best is the one set to 70%.

Since better results have been achieved with the majority voting based combination function, it has been retained to make the consensus of the results of the PFSOE.

4.2.3. Comparison

The performance of the proposed outlier ensemble was compared to both the base algorithm LOF [15], and the feature bagging solution presented in [3]. For LOF, the well-performing value of *MinPts*, i.e.

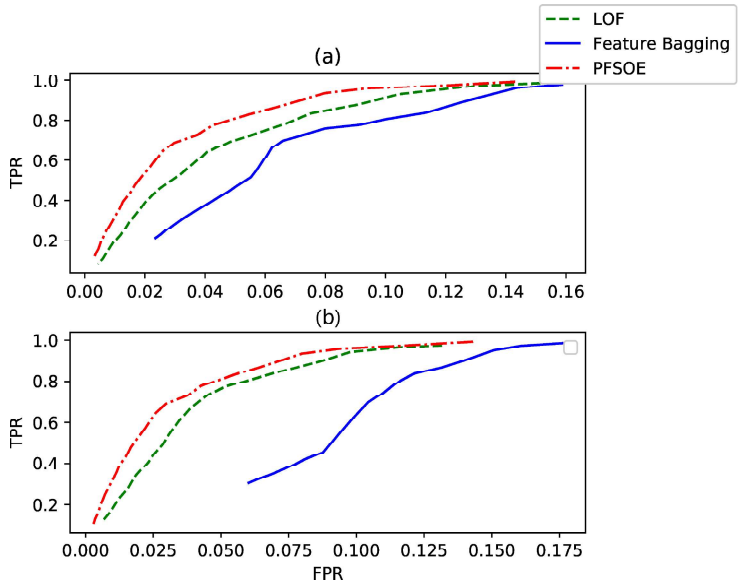

Fig. 3. ROC curves for the different approaches LOF, Feature Bagging and our outlier ensemble (PFSOE). In (a) both LOF and Feature Bagging were applied with MinPts = 68, while in (b) they were applied with the full set MP = [38, 88].

MinPts = 68, was used. In addition, the algorithm was applied with all the values of the set MP = [38, 88]. For the feature bagging solution, the same data subspaces as those constructed for the outlier ensemble has been used, and the results have been combined with the cumulative sum function as it performed effectively in [3]. Regarding the parameter *MinPts*, since this was not discussed clearly in [3], the approach has been applied twice by using the same values as the base algorithm LOF.

Figure 3 shows the computed ROC curves for the TPR and FPR obtained for the different approaches, LOF, Feature Bagging and, the proposed PFSOE. The curves were obtained with a variable threshold. The subplot (a) presents the results of executing LOF and Feature Bagging with the parameter *MinPts* = 68, while the subplot (b) depicts the results of executing the two approaches with the full set of values for *MinPts*, i.e. MP = [38, 88]. In both subplots, the outlier ensemble was fed with different subsets of $\boldsymbol{MP}$ and the results were combined with the majority voting function with an upper bound equal to 70%. As can be seen, in both scenarios, our solution achieved high detection rate and low false alarms rate, and it outperformed the other approaches. These results are due to the fact that our solution searches for anomalies in different data subspaces, outperforming in this way LOF which operates on the full data space, and thus, might suffer from the curse of dimensionality and fail to detect anomalies that lie in particular data subspaces. In addition to the different feature sets used in the proposed solution, the outlier detection algorithm is applied with different values for the parameter *MinPts* and uses a different combination function, this explains the fact that it outperforms the feature bagging approach. Indeed, the choice of the *MinPts* parameter is not an easy task, and a value might allow performing well in certain sliding windows but not in others, while with the proposed approach, by using various subsets of the well-performing values for *MinPts*, profit is taken from the performance of the algorithm with different well-performing parameters. From the same figure, it is noticed that the feature bagging didn't perform well, and was outperformed by LOF, these results are due to the combination function and to the fact that all the data features are significant [12].

It should be mentioned that the presented FPR and TPR are those obtained for the overall test set, and are not computed by averaging the obtained FPR and TPR from the different windows. However,

Table 5
Results of the 10-fold cross-validation, LOF and Feature Bagging were applied with MinPts = 68, the FPR, TPR and Accuracy are obtained with threshold $t = 2.8$, the AUC was computed for curves obtained with different values for the threshold

Expt	LOF				Feature bagging				PFSOE			
	FPR	TPR	Accuracy	AUC	FPR	TPR	Accuracy	AUC	FPR	TPR	Accuracy	AUC
1	0.086	1	0.905	0.984	0.165	1	0.827	0.973	0.084	1	0.907	0.986
2	0.082	0.939	0.902	0.956	0.127	1	0.857	0.956	0.072	0.853	0.911	0.960
3	0.107	1	0.879	0.981	0.159	1	0.827	0.967	0.096	1	0.889	0.986
4	0.112	0.95	0.885	0.952	0.170	0.95	0.827	0.935	0.0994	0.95	0.898	0.957
5	0.133	0.951	0.859	0.961	0.203	1	0.789	0.944	0.110	0.934	0.881	0.970
6	0.052	0.956	0.938	0.971	0.096	1	0.895	0.955	0.040	0.985	0.949	0.9812
7	0.117	0.849	0.856	0.938	0.146	0.925	0.828	0.936	0.099	0.844	0.873	0.949
8	0.075	0.644	0.903	0.944	0.126	0.710	0.853	0.913	0.061	0.756	0.917	0.966
9	0.063	0.919	0.931	0.979	0.140	1	0.855	0.959	0.057	0.946	0.937	0.986
10	0.099	0.989	0.887	0.973	0.173	1	0.816	0.943	0.096	0.978	0.890	0.977
Mean	0.0927	0.919	0.8945	0.96391	0.1506	0.958	0.8373	0.94812	0.0817	0.9249	0.9053	0.97182

Table 6
Results of the 10-fold cross-validation, LOF and Feature Bagging were applied with MinPts = [38, 88], the FPR, TPR and Accuracy are obtained with threshold $t = 2.8$, the AUC was computed for curves obtained with different values for the threshold

Expt	LOF				Feature bagging				PFSOE			
	FPR	TPR	Accuracy	AUC	FPR	TPR	Accuracy	AUC	FPR	TPR	Accuracy	AUC
1	0.112	1	0.879	0.982	0.283	1	0.709	0.956	0.084	1	0.907	0.986
2	0.101	0.957	0.882	0.957	0.192	1	0.794	0.958	0.072	0.853	0.911	0.960
3	0.131	1	0.855	0.980	0.226	1	0.761	0.952	0.0968	1	0.889	0.986
4	0.136	0.95	0.861	0.961	0.247	0.95	0.750	0.94	0.099	0.95	0.898	0.957
5	0.181	0.967	0.811	0.959	0.315	1	0.679	0.917	0.110	0.934	0.881	0.970
6	0.061	0.985	0.929	0.972	0.134	1	0.857	0.936	0.041	0.985	0.949	0.981
7	0.149	0.889	0.824	0.933	0.243	0.995	0.734	0.898	0.099	0.844	0.873	0.949
8	0.089	0.947	0.889	0.958	0.149	1	0.831	0.914	0.061	0.756	0.917	0.966
9	0.091	0.973	0.903	0.980	0.220	1	0.775	0.927	0.058	0.946	0.937	0.986
10	0.137	1	0.850	0.969	0.271	1	0.718	0.909	0.096	0.979	0.891	0.977
Mean	0.1191	0.9669	0.8686	0.9653	0.2282	0.994	0.7608	0.9309	0.0817	0.9249	0.9053	0.9718

in order to ensure a more reliable evaluation of the proposed outlier ensemble, 10-fold cross-validation has been conducted. The training phases of each experiment allowed to confirm that the majority voting combination function with an upper bound set to 70% and the set $MP = [38, 88]$ allowed obtaining the best results most frequently. Results of the 10 experiments of the cross-validation are presented in Tables 5 and 6. As can be seen, the proposed PFSOE algorithm is more accurate and robust than the two other solutions to which it has been compared.

5. Conclusion

In recent years, honeypots are gaining great interest. Therefore, efficient techniques for analyzing their data are necessary. In this paper, all aspects related to the honeypots data were taken into consideration. Due to the important number of anomalous events that constitute the data, a scheme to exploit them appropriately was proposed. The PFSOE, an outlier ensemble based on LOF for anomaly detection and deep analysis of the data, was presented. The solution works in an unsupervised fashion, therefore it deals with previously unseen anomalies. It also deals with the high-dimensionality characteristic of the data by working on data subspaces. Moreover, the presented outlier ensemble tackles the diversity of the

anomalies by searching for the anomalies in different data subspaces and applying the outlier detection algorithm with various parameters. Addressing all these challenges allowed the proposed solution to achieve better results when compared to the base algorithm LOF and an existing general-purpose outlier ensemble. Indeed, a detection rate higher than 92% for 8% of false alarms could be achieved in a completely unsupervised fashion. Additionally, the results of the cross-validation allowed demonstrating the robustness of the outlier ensemble. In the present work, the feature subsets are created randomly. However, as a future work, the aim is to improve the proposed detection method by partially feeding the ensemble with relevant subspaces learned throughout the detection process.

Acknowledgments

This work was funded by the National Natural Science Foundation of China, Grant number 61272420 and 61472189.

References

[1] A. Grégio, R. Santos and A. Montes, Evaluation of data mining techniques for suspicious network activity classification using honeypots data, in: *Defense and Security Symposium International Society for Optics and Photonics*, 2007.

[2] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur and J. Srivastava, A comparative study of anomaly detection schemes in network intrusion detection, in: *Proc. of the 2003 SIAM Int. Conf. on Data Mining*, 2003.

[3] A. Lazarevic and V. Kumar, Feature bagging for outlier detection, in: *KDD '05 Proc. of the Eleventh ACM SIGKDD Int. Conf. on Knowledge Discovery in Data Mining*, 2005, pp. 157–166.

[4] A. Zimek, R.J.G.B. Campello and J. Sander, Ensembles for unsupervised outlier detection: challenges and research questions a position paper, *ACM SIGKDD Explorations Newsletter* **15**(1) (2013), 11–22.

[5] C.C. Aggarwal, Outlier ensembles: position paper, *ACM SIGKDD Explorations Newsletter* **14**(2) (2012), 49–58.

[6] C.C. Aggarwal and P.S. Yu, Outlier Detection for High Dimensional Data, in: *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, 2001, pp. 37–46.

[7] D. Fraunholz, M. Zimmermann, A. Hafner and H.D. Schotten, Data mining in long-term honeypot data, in: *IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017.

[8] F. Pouget and M. Dacier, Honeypot-based forensics, in: *AusCERT Asia Pacific Information Technology Security Conference*, 2004.

[9] G.K. Sadasivam, C. Hota and B. Anand, Detection of severe SSH attacks using honeypot servers and machine learning techniques, *Software Networking* (1) (2017), 79–100.

[10] H.G. Kayacik, A.N. Zincir-Heywood and M.I. Heywood, Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets, in: *Proc. of the 3rd Annual Conference on Privacy, Security and Trust*, 2005.

[11] http://www.takakura.com/Kyoto_data/, Accessed June 2018.

[12] 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: *BADGERS '11 Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, 2011, pp. 29–36.

[13] M. Goldstein and S. Uchida, A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, *PLoS One* **11**(4) (2016).

[14] M. Husák and J. Kašpar, Towards Predicting Cyber Attacks Using Information Exchange and Data Mining, in: *14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2018, pp. 536–541.

[15] M.M. Breunig, H.P. Kriegel, R.T. Ng and J. Sander, LOF: Identifying Density Based Local Outliers, in: *SIGMOD '00 Proc. of the 2000 ACM SIGMOD Int. Conf. on Management of Data*, 2000, pp. 93–104.

[16] O. Thonnard and M. Dacier, A framework for attack patterns' discovery in honeynet data, *Digital Investigation* **5**(1) (2008), 128–139.

[17] P. Gogoi, D.K. Bhattacharyya, B. Borah and J.K. Kalita, A survey of outlier detection methods in network anomaly identification, *The Computer Journal* **54**(4) (2011), 570–588.

[18] P. Owezarski, A Near Real-Time Algorithm for Autonomous Identification and Characterization of Honeypot Attacks, in: *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security – ASIA CCS '15*, 2015.

[19] P. Owezarski, Unsupervised classification and characterization of honeypot attacks, in: *10th Int. Conf. on Network and Service Management (CNSM) and Workshop*, 2014, pp. 10–18.
[20] S. Almotairi, A. Clark, G. Mohay and J. Zimmermann, A technique for detecting new attacks in low-interaction honeypot traffic, in: *Fourth International Conference on Internet Monitoring and Protection*, 2009, pp. 7–13.
[21] S. Almotairi, A. Clark, G. Mohay and J. Zimmermann, Characterization of attackers' activities in honeypot traffic using principal component analysis, in: *2008 IFIP Int. Conf. on Network and Parallel Computing*, 2008, pp. 147–154.
[22] S. Lance, Honeypots: Tracking Hackers, Addison Wesley, 1st edn., 2002.
[23] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa and B. Yang, Predicting Network Attack Patterns in SDN using Machine Learning Approach, in: *2016 IEEE Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2016.
[24] V. Chandola, A. Banerjee and V. Kumar, Anomaly detection: a survey, *ACM Computing Surveys* **41**(3) (2009).
[25] W. Fan, Z. Du, D. Fernandez and V.A. Villagra, Enabling an anatomic view to investigate honeypot systems: a survey, *IEEE Systems Journal* (99) (2017), 1–14.
[26] Z. Zhan, M. Xu and S. Xu, Characterizing honeypot-captured cyber attacks: statistical framework and case study, *IEEE Transactions on Information Forensics and Security* **8**(11) (2013), 1775–1789.
[27] Z. Zhan, M. Xu and S. Xu, Predicting cyber attack rates with extreme values, *IEEE Transactions on Information Forensics and Security* **10**(8) (2015), 1666–1677.