Journal of Network and Computer Applications 66 (2016) 1–16

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

Ensemble based collaborative and distributed intrusion detection systems: A survey

Gianluigi Folino *, Pietro Sabatino

ICAR-CNR Istituto di Calcolo e Reti ad Alte Prestazioni Via P. Bucci, 87036 Rende (CS), Italy

ARTICLE INFO

Article history:
Received 22 August 2015
Received in revised form
16 March 2016
Accepted 17 March 2016
Available online 19 March 2016

ABSTRACT

Modern network intrusion detection systems must be able to handle large and fast changing data, often also taking into account real-time requirements. Ensemble-based data mining algorithms and their distributed implementations are a promising approach to these issues.

Therefore, this work presents the current state of the art of the ensemble-based methods used in modern intrusion detection systems, with a particular attention to distributed approaches and implementations. This review also consider supervised and unsupervised data mining algorithms, more suitable to work in an environment that requires the analysis of data streams in real-time. Sharing knowledge across multiple nodes is another of the key points in designing appropriate NIDSs and for this reason, collaborative IDS were also included in this work. Finally, we discuss some open issues and lessons learned from this review, which can help researchers to design more efficient NIDSs.

Contents

1. Introduction

Network Intrusion Detection Systems (NIDSs) are usually distinguished in *signature-based* or *misuse-based* intrusion detection systems and *anomaly-based* intrusion detection systems. In the

* Corresponding author.
E-mail addresses: folino@icar.cnr.it (G. Folino), pietro.sabatino@icar.cnr.it (P. Sabatino).

http://dx.doi.org/10.1016/j.jnca.2016.03.011
1084-8045/

misuse based approach, that can be summarized as "detect what I know", when an attack has been discovered, the different steps of the attack are encoded in a *signature*, then the signature is stored in a database and finally it is used by the system to detect the attacks. On the contrary, anomaly-based systems, first try to model the normal behavior of the system to be protected, and then generate an alarm when a significant deviation from this normal behavior is detected. However, it is problematic to define opportunely how significant the deviation is in these kinds of systems, as it is necessary to find a trade-off between generating a large number of false alarms and letting some attacks elude the system.

As a consequence of our interconnected society and also due to the increased speed of the underlining network connections (gigabits per second) (Lai et al., 2004), computer network activities, human actions, systems, etc. generate large amounts of data and network traffic (typical examples of these streaming data in this particular domain are network logs, credit card transactional flows, and sensor network data). Indeed, this aspect must be seriously taken into account in designing modern NIDSs, which have to handle these large and fast-changing data. Typically, data mining-based algorithms are among the most used to this aim (Dua and Du, 2011). However, the ever-changing nature, the high volume of these data, the large amount of storage necessary, etc. can put in crisis traditional sequential data mining algorithms, as they are not able to capture new trends in the stream. In fact, traditional algorithms assume that data is static, i.e. a concept, represented by a set of features, which does not change because of modifications of the external environment.

A more flexible approach is that of ensemble-based algorithms (Kuncheva, 2004), which have been stimulating an increasing interest in the intrusion detection community as they have some characteristics of interest in this particular domain. Indeed, they can be easily implemented in parallel/distributed architectures, improve the accuracy of a weak learner, can be specialized for the detection of one particular class and are particularly suitable to the case of unbalanced datasets.

In the last few years, distributed computing has considerably changed due to the emergence of new environments, such as grid and cloud computing. In particular, the Cloud Computing paradigm has enormous potential for intrusion detection systems; in fact, the availability of distributed resources on demand permits data and computational power to be accessed, previously hardly obtainable, and make it possible to deal with complex problems with real-time constraints. Furthermore, even for industries or institutions that do not have the opportunities to use expensive clusters, it is possible to design and to execute massively parallel algorithms also by exploiting the potentialities of the modern multicore CPU and GPU-based architectures.

The increase in the volume of data that need to be processed tends to push the computational load that the underlying hardware needs to sustain to its limits. Indeed, recall that in order to be really useful a NIDS should work in real time, and it needs to sustain the analysis of the data stream involved without saturating all the resources available.

Moreover, increased network connections also mean an increase in the number of alarms generated, especially in the case of signature-based NIDS. As a consequence, there is an ever increasing need for solutions that do not overwhelm human operators with a number of unmanageable alarms. Finally, NIDSs need to be capable of dealing with new undocumented attacks, for which a signature is not already available, i.e., the 0-day attacks.

In view of the previous considerations, this paper mainly reviews the ensemble-based algorithms based on data mining techniques that can be implemented in a parallel/distributed environment in as straightforward way as possible. Choice of reviewing mainly ensemble-based methods is due to the fact that they are not sufficiently treated in the literature for the task of intrusion detection.

More in detail, this review aims not only to analyze supervised data mining algorithms, but also works based on anomaly detection and, in particular on clustering suitable to operate in an environment that requires the analysis of data streams in real-time. Sharing knowledge across multiple nodes is another of the key points in designing appropriate NIDSs and for this reason, collaborative IDS were also included in this work.

1.1. A panoramic of surveys on Intrusion Detection Systems

In this section, we review the main surveys in the field of intrusion detection and remark on the differences from our work.

Many surveys give a general overview of network anomaly detection systems, for instance Raut and Singh (2014) and Kemmerer and Vigna (2002); an interesting survey of the field of anomaly-based network intrusion detection can be found in García-Teodoro et al. (2009), which also includes a panoramic of the principal anomaly-based intrusion detection techniques, namely statistical-based, machine learning-based and knowledge-based. For each of the techniques illustrated, the main systems present in the literature are described. Moreover, the main challenges of the field are outlined and discussed.

However, one of the most comprehensive introductions to the field of *Network Anomaly Detection* (NAD) is presented in Bhuyan et al. (2014). The main aim of this paper is to provide enough informations for a new researcher to acquire a certain familiarity with every aspect of the field. Accordingly, it contains a clear categorization of most common attacks encountered and of the main systems for network intrusion detection, according to the anomaly detection method and to the computational technique adopted. Based on this classification, an in-depth comparison is performed of the different architectures. A review of the principal tools that are useful to researchers in the network anomaly detection field is also included, as well as a comparative description of the datasets of network traffic publicly available.

Other surveys, described in the following, are more specific to a topic, types of data or technique. The problem of preprocessing and feature selection in the anomaly detection field is illustrated in detail in Davis and Clark (2011). The review covers both automated methods for features extraction and techniques used to reduce the dimensionality of the problem. In fact, the authors analyze the literature on this topic and identify a shift from techniques that rely on the analysis performed by human domain experts to automatic solutions that implement a broad range of techniques of analysis.

Generally, systems are categorized according to the detection paradigm considered and to the type of attack, i.e., denial of service, worms, etc. The work in Sperotto et al. (2010) concentrates on the intrusion detection systems that operate at the level of network flow. In Patel and Buddhadev (2014), the authors present a review of the main machine learning techniques employed in the network intrusion detection field. The techniques considered include rule-based learning, decision tree, Bayesian reasoning, neural networks, support vector machines and clustering. Moreover, a section devoted to nature-inspired techniques is also included, which covers artificial immune systems, genetic programming and swarm intelligence. In addition, a comparison of the performance of the different techniques is illustrated, reporting experimental results contained in the literature. Finally, the current challenges of the field are discussed.

In Hoque et al. (2014), a panoramic of tools useful to researchers in the intrusion detection field is presented. In particular, the main steps necessary to perform an attack are described. Scanning and sniffing tools are discussed, as well as an extensive

discussion of attack launching tools. Network monitoring systems and working attack detection systems are also covered. Finally, the main challenges that researchers in the field have to face are the subject of the closing considerations of the work. The same authors also reviewed the topics of port scans, scanning tools and launching attacks in Bhuyan et al. (2011c) and the main detection techniques in Bhuyan et al. (2011a) and Bhuyan et al. (2011b).

More correlated to our paper, two surveys of network intrusion detection systems from the distributed point of view can be found in Zhou et al. (2010) and Vasilomanolakis et al. (2015). Zhou et al. (2010) point their attention to coordinated attacks such as large-scale scans, worm diffusion and distributed denial of service and on the research in the field of collaborative intrusion detection that aims to detect and prevent these kinds of attack. The work describes different architectures of a collaborative intrusion detection system and also illustrates some algorithms used for the task of alert correlation. Vasilomanolakis et al. (2015) outline the general architecture of a collaborative intrusion detection system. The main building blocks of these systems and the methodology to integrate them in a common environment are described. In comparison with the latter two papers, our work is more focused on the ensemble-based paradigm, and on unsupervised and supervised data mining techniques. In addition, we analyzed the main distributed algorithms used for these topics, while the above-cited papers are more aimed at a general discussion of the possible collaborative and distributed architectures, without discussing the interesting ensemble-based techniques.

Another related work (Corona et al., 2009b) reviews many state-of-the-art solutions for computer security from the point of view of the information fusion and proposes a general scheme to perform information fusion for intrusion detection in computer systems. In addition, the work discusses strengths and weaknesses of currently used approaches and examines some interesting research issues. However, the paper is less specific than our, as it treats computer security in general, and furthermore it does not analyze parallel/distributed solutions.

Indeed, to the best of our knowledge, there are no up to date surveys in the literature on distributed and collaborative systems with particular emphasis on the ensemble approach.

1.2. Overview of the paper

The paper is organized as follows: Section 2 surveys NIDS systems based on the ensemble techniques, mainly analyzing data mining approaches: supervised, unsupervised and also hybrid solutions; in addition a subsection is devoted to discussing ensemble systems based on other techniques, i.e., Hidden Markov models; Section 3 provides a description of the distributed and collaborative approaches, in particular the main works concerning collaborative IDS, data mining distributed systems and high-performance implementation are illustrated; Section 4 discusses the strengths and weaknesses of the reviewed approaches and some open issues in the field; finally, Section 5 concludes the paper.

2. Ensemble-based techniques

Ensemble-based techniques permit one to combine multiple (heterogeneous or homogeneous) models in order to classify new unseen instances. In practice, after a number of classifiers are built usually using some parts of the original dataset, the predictions of the different classifiers are combined and a common decision is taken. Different schemas can be considered to generate the classifiers and to combine the ensemble, i.e. the same learning algorithm can be trained on different datasets or/and different algorithms can be trained on the same dataset.

The best known and used ensemble-based algorithm is the boosting algorithm, first introduced by Schapire (1990, 1995). In order to boost the performance of any "weak" learning algorithm, i.e. an algorithm that "generates classifiers which need only be a little bit better than random guessing" (Schapire, 1995), the boosting method adaptively changes the distribution of the training set according to how difficult each example is to classify. This algorithm, in common with many ensemble-based algorithms, has the drawback of needing to repeat the training phase for a number of rounds and that is really time-consuming in the case of large datasets or not practicable in the case of real-time requirements, i.e. in the intrusion detection domain. On the contrary, other variants of ensemble-based algorithms use functions to combine the classifiers that do not need to reuse the original training set. The majority vote is a classical example of this kind of combiner function, but there are many other functions with this property and they are named non-trainable combiners (Kuncheva, 2004). Combiner functions having this property considerably reduces the time requirements required to compute them, as it is not necessary to reuse the training set in this phase.

The information fusion approach, which combines information coming from different sources in order to build new aggregate features or to enhance the solution of a problem, may permit to solve better or more efficiently an IDS task (Dasarathy, 2003). In fact, different aspects can motivate the use of information fusion for this kind of problems, i.e., information may be present at multiple abstraction levels and may be collected from multiple sources, different abstraction levels can be present in the data, it is effective in increasing timeliness of attack identification and in reducing false alarm rates and usually obtain high detection rate, etc. However, as an additional computational overhead must be also taken into account to process these different sources, a fast and effective fusion of information is fundamental to enhance the value of the classification and clustering task and the computational cost of the process must be taken into account. Many fusion-based techniques can be applied to the intrusion detection task. An interesting classification (Bhuyan et al., 2014) considers three cases depending on whether the algorithm operates at the level of the data (data level), of the features (feature level), and of the final decision (decision level). Indeed, some methods try to consider the semantic groups in which the high dimensionality space of the features is divided or the hierarchical abstraction levels or the type of information contained. Furthermore, as the intrusion detection task can be computationally expensive, many of the information fusion-based approaches, which we review in the next section, adopt non-trainable functions to combine the classifiers.

IDSs benefit from using information fusion and ensemble-based algorithms for a number of reasons. First, these techniques perform well both when data are very scarce and when we have a huge amount of data; furthermore, they can be easily implemented in efficient computing environments such as parallel, multi-core, and GPGPU architectures and also on P2P and Cloud computing environments. In addition, they can easily model different abstractions or parts of a network, i.e., some models can be trained on some parts or on some levels of the network and finally combined together, to assure a better prediction.

More in detail, in the next subsections, we analyze different variants of ensemble and information fusion-based approaches for the intrusion detection task: supervised and unsupervised data mining approaches, hybrid techniques combining the previous two approaches and some interesting statistical techniques. A summary of the main characteristics (i.e., the technique adopted, the preprocessing phase used, the function combining the ensemble, the datasets on which the experiments are conducted, the metrics used for evaluation, etc.) of these works is given in Table 1 (data mining-based approaches) and in Table 2 (statistical approaches and other techniques).

Table 1
Summary table of papers reviewed, data mining techniques.

Author(s)	Year	Technique(s)	Ensemble	Dataset(s)	Preprocessing	Metric(s)
Supervised						
Perdisci et al. (2009)	2009	SVM	Average, product, minimum and maximum rule	DARPA, GATECH (Real traffic and injected attacks)	Clustering based Dhillon et al. (2003)	ROC curve
Borji (2007)	2007	ANN, SVM, Decision Tree, *k*-NN	Majority voting, average and "belief" function	DARPA 98	None	Detection rate, false positive rate
Aburomman and Ibne Reaz (2016)	2016	SVM, *k*-NN	Weighted majority voting chosen by PSO	KDD'99	None	Classification accuracy
Sivatha Sindhu et al. (2012)	2013	Decision Tree, C4.5	AdaBoost	KDD'99	Genetic Algorithm	True and false positive rate, precision, recall and *F*-measure
Unsupervised						
Zhang and Zulkernine (2008)	2006	Random forest	Majority voting	KDD'99	None	ROC curve
Jungsuk et al. (2009) and Song et al. (2013)	2009, 2013	Density based clustering, SVM	Maximum rule	Real traffic	Feature selection based on the results of Mukkamala and Sung (2003)	ROC curve
Bhuyan et al. (2012)	2012	Tree-based subspace clustering algorithm (TreeCLUS)	Majority voting, maximum rule	KDD'99, TUIDS	Information theoretic based	Detection rate, false positive rate
Giacinto et al. (2008)	2008	*k*-means, ν-SVC, Parzen-window density estimation	Maximum, minimum, mean and product rule	KDD'99	None	ROC curve
Hybrid						
Horng et al. (2011)	2011	SVM	None	KDD'99	Data mining based, BIRCH algorithm Zhang et al. (1996)	Accuracy, false positive rate
Nguyen et al. (2011)	2011	*k*-means algorithm, Decision Tree	Weighted mean	KDD'99	None	True positive rate, false positive rate
Elbasiony et al. (2013)	2013	Weighted variant of *K*-means	Random Forest	KDD'99	None	ROC curve
Singh et al. (2015)	2015	OS-ELM	Feature aggregation stage	NSL-KDD, Real traffic	Filtered, consistency, CFS subset evaluation and DBSCAN	Accuracy, true and false positive rate, true and false negative rate, F1-score, precision and execution time

Table 2
Summary table of papers reviewed, statistical approaches.

Author(s)	Year	Technique(s)	Ensemble	Dataset(s)	Preprocessing	Metric(s)
Ariu et al. (2011)	2011	HMM	Minimum, maximum, mean and geometric mean	DARPA, Real traffic	Random projection	ROC Curve
Song et al., 2009	2009	HMM	Weighted sum	Real traffic	None	ROC Curve
Khanna and Liu (2006, 2008)	2006, 2008	HMM	Weighted sum	DARPA, simulated subnetwork	None	ROC curve
Boro et al. (2012)	2012	C4.5, Näive Bayes, and decision table	Meta ensemble, weighted majority vote	KDD'99, TUIDS	Information gain based	Classification accuracy
Kumar and Selvakumar (2013, 2011)	2011, 2013	Adaptive neuro-fuzzy systems (ANFS), see Jang (1993), resilient back propagation (RBP), Riedmiller and Braun (1993)	Modification of AdaBoost	DARPA, CAIDA, Real traffic	Based on classification accuracy	Detection accuracy

2.1. Supervised data mining

The technique of n-grams has been extensively employed for different kinds of problems, i.e., the statistical analysis of texts (Leopold and Kindermann, 2002; Damashek, 1995) and the problem of speech recognition. A model of the text is usually obtained by a sliding window of n-characters that form an n-gram. At steps of one character, the n-grams captured constitutes a set that represents the text. The use of n-grams has a computational cost that grows exponentially in n, as the feature space grows as 256^n; therefore, this technique is computationally expensive. Nonetheless in protocols like HTTP, the content of a request is encoded in plain text, hence the name text-based protocols, and from here it follows that the technique of n-grams is clearly suitable to model the packet payload containing HTTP data, subsequently analyzed in search of anomalous content.

The pioneer work on applying n-grams to intrusion detection is described in Wang and Stolfo (2004), in which the PAYL network intrusion detection system is presented, while extensions of this work are discussed in Wang et al. (2006a, 2006b).

In the paper (Perdisci et al., 2009), an IDS based on an ensemble of SVMs is proposed. In this approach, the payload is modeled using the technique of 2_ν-grams. Namely, for a fixed ν, the payload is represented as a sequence of 2-grams extracted at intervals of ν bytes. Varying the value of ν, a representation of the payload in different feature spaces is obtained. It is worth noting that each feature space obtained in this way has dimension 256^2. In order to reduce the dimension of the feature space, the authors adopt a preprocessing stage, i.e., a clustering algorithm based on the technique originally proposed in Dhillon et al. (2003). In the training phase, an ensemble of SVMs, one model for each representation of the payloads, is build in order to model the normal traffic. In the detection phase, whenever a packet is received, each representation is classified by the corresponding SVM, giving as output an estimate of the probability that the payload belongs to normal traffic. The output probabilities are then combined using a non-trainable combiner, chosen among the average, the product, the minimum or the maximum. The output is then an overall estimate of the probability P of the payload being normal. Finally, the payload is then classified as normal if P is above a prefixed threshold. Experimental results were conducted on the well-known DARPA dataset and on the GATECH dataset, obtained by collecting seven days of real HTTP requests towards the website of the College of Computing School at the Georgia Institute of Technology, considered as normal traffic; subsequently, synthetic attacks generated by the polymorphic shell-code engine CLET[1] were added to the dataset. The approach is compared to PAYL and it results more effective in detecting polymorphic attacks and in detecting attacks with a very low false positive rate. However, the system has execution times higher than PAYL. A prototype of the system, also comprising the source code is available for download.[2]

In Borji (2007), a number of classifier algorithms, i.e., ANN, SVM, the Decision Tree and k-NN are trained and tested on a subset of the DARPA dataset and compared using detection and false positive rates. After that, the above-mentioned classifiers are combined in an ensemble by means of a combination rule, chosen among the majority voting rule, the average rule and the "belief" function, i.e., a function based on the estimation of the probabilities that a pattern assigned to a given data class actually belongs to that class or to other classes. Experimental results show that the ensemble-based approach tends to provide more reliable results compared with the approach based on a single classifier. In particular, the "belief" combination rule obtains better results.

In Aburomman and Ibne Reaz (2016), Aburomman et al. train two types of base classifiers, SVM-based and k-NN-based, on the KDD'99 dataset. After this preliminary stage, the base classifiers are combined in an ensemble by a weighted majority voting. Three techniques are employed in order to choose the weights of the combination function: one based on the particle swarm optimization (PSO), another based on a variant of the PSO, which uses local unimodal sampling in order to iteratively varying the search range of the parameters in an optimal way, and lastly the Weighted Majority Algorithm, introduced by Littlestone and Warmuth (1994). Experimental results show that the ensemble technique is able to improve the results obtained from the base classifiers alone in term of classification accuracy. Moreover, the best results are obtained by using the combination function, which adopt the technique based on the variant of the PSO.

An ensemble of neural networks is the basis of the architecture proposed in Sivatha Sindhu et al. (2012). In particular, the AdaBoost algorithm is used to train an ensemble on the KDD'99 dataset. In order to select the optimal subset of the features from the dataset, a genetic algorithm is employed, in which the individuals in the population are ensembles trained on a specific subset of the features and the fitness function is the accuracy on a validation set. It is well known that models based on neural networks tend to involve a certain complexity. To avoid this drawback, a decision tree is built by running the C4.5 algorithm on the classes obtained from the above-described technique. Experimental results show that the overall solution is better than the solutions based on neural networks and decision trees alone, with respect to different metrics, i.e., True Positive and False Positive Rate, Precision, Recall and F-measure.

[1] http://phrack.org/issues/61/9.html

[2] https://pralab.diee.unica.it/en/HMMPayl_and_McPAD

2.2. Unsupervised data mining

The supervised IDSs, described in the previous subsection, need to be trained on labelled datasets containing normal traffic and most of them also require that a considerable number of attacks are present in the training set. In this phase, usually the statistical behavior of the normal traffic is established and the parameters of the model are tuned accordingly. In many cases, a labelled dataset is difficult to obtain or only a few labeled tuples are available, and therefore unsupervised data mining algorithms are more suitable to cope with this problem. In the following, we review the main papers concerning unsupervised algorithms based on the ensemble model used in the intrusion detection field.

An unsupervised IDS framework based on the random forest algorithm is described in Zhang and Zulkernine (2008). The system comprises a preprocessing phase that analyzes the network traffic and generates a dataset. Then, an analysis is performed offline, as the computational cost of the algorithm employed, i.e., random forest, make it unfeasible for on-line analysis. The random forest algorithm (see Breiman and forests, 2001), is made up of an ensemble of regression trees and it generates a different tree for each bootstrap sample of the dataset. When an object needs to be classified, each tree gives a vote and the overall class is chosen by performing a majority voting. After the training stage, a proximity function between the instances is established and, in the detection stage, a number of outliers are individuated, based on the fact that the distance exceeds a prefixed threshold. The approach is evaluated on the well-known KDD'99 dataset and experiments prove that its accuracy is comparable to previously reported approaches, but it achieves a higher detection rate maintaining the false positive rate relatively low.

Another unsupervised approach based on data mining techniques is described in Jungsuk et al. (2009) and in Song et al. (2013). In the first paper, the overall architecture of an IDS is described, which consists mainly of three stages, i.e., filtering, clustering and modeling. During the training phase, the attacks are filtered and eliminated from the training data. This step is accomplished by an algorithm that employs a notion of density, which is determined by the ratio of attacks with respect to normal traffic; this parameter needs to be specified by the user. After the data has been filtered, the system performs a clustering of the training data. Again, this step is determined by a parameter specifying the number of clusters to be obtained. The clusters obtained represent different models of normal traffic. Finally, in the modeling stage, for each cluster obtained, a one-class SVM is trained. In the testing phase, the traffic instances are evaluated by the ensemble of SVMs obtained in the previous step. Instances are flagged as normal, if their representation falls inside one of the one-class SVMs. In Song et al. (2013), the authors extended the proposed method in order to tune the values of the above-mentioned parameters automatically without any intervention by the user. The architecture is evaluated on real traffic data obtained from Kyoto University. The two approaches are compared and the results show the superiority of the automatic approach, its efficiency in determining the suitable number of clusters and in estimating the ratio of attack instances with respect to normal traffic.

In Bhuyan et al. (2012), a fully unsupervised approach is presented based on a clustering technique. The clustering stage of the preprocessed data is performed by the TreeCLUS algorithm. This clustering technique is a tree-based subspace clustering algorithm that employs information theoretic concepts (see Amiri et al., 2011) in order to identify the most relevant features. Moreover, TreeCLUS performs a cluster stability analysis by using an ensemble and each element of the ensemble uses a different clustering stability measure to perform the task. The measures employed are *Dunnt index* (Dunn, 1974), *C-index* (Hubert and Schultz, 1976), *Davies Bouldin index* (Davis and Bouldin,), *Silhouette index* (Rousseeuw, 1987), *Xie-Beni index* (Xie and Beni, 1991). The results obtained by the classifiers are combined by a simple majority voting. Finally, the clusters obtained by the TreeCLUS algorithm are labeled by the CLUSLab algorithm, which also employs an ensemble approach, described in the following. After computing the cluster size, the compactness and the dominating features subset, it labels each cluster as normal or anomalous by using a maximum decision score-based function. The authors test the two algorithms on the well-known KDD'99 and on the real-world TUIDS datasets, and conclude that both the approaches outperform other existing unsupervised network anomaly detection techniques for the metrics of detection rate and false positive rate.

The approach adopted by Giacinto et al. (2008) is based on a modular ensemble. In practice, each classifier contained in the ensemble is specialized on the classification of the traffic concerning a specific service, e.g. web service, mail service, and so on. Three solutions, all density-based, for the base classifiers are tested: a Parzen-window approach (Duda et al., 2001), an approach based on the *K*-means clustering algorithm and ν-SVC, a solution inspired to SVM, see Schölkopf et al. (2001) for more details. Then, the base classifiers are combined in the ensemble by using simple non-trainable rules, i.e., maximum, minimum, mean and the product rule. Experiments on the KDD'99 dataset prove that the most efficient approach in terms of the area under the ROC curve is the one based on the ν-SVC technique.

2.3. Hybrid data mining techniques

In this section we collect techniques that combine the advantages of the supervised and unsupervised paradigms, or that include both a component based on misuse detection and on anomaly detection.

The approach (Horng et al., 2011) relies on the BIRCH clustering algorithm, see Zhang et al. (1996). In particular the KDD'99 dataset is split in to its five principal classes, Probe, Dos, U2R, R2L and normal traffic. The above-mentioned clustering algorithm is employed to build five clustering feature trees, one for each class. Roughly speaking, the clustering feature tree is a compact representation of the dataset, in which each leaf node corresponds to a cluster. After this compact representation of the dataset is obtained, a feature selection is performed. Four SVMs, one for each attack class, are then trained and combined together in an ensemble for subsequent testing, performed again on the KDD'99 dataset. Compared to solutions employing different techniques, such as decision trees and *K*-means, it shows slightly better or comparable performances.

In the approach proposed by Nguyen et al. (2011), data collected from heterogeneous sources such as network traffic packets, operation system audits, and system log files are preprocessed to form a single dataset. The dataset is labeled by means of domain expert knowledge, and other semi-automatic methods to constitute the IDS training set, which is split into a training and a validation set. Then, a clustering of the data is performed by using the *K*-means algorithm, which infers additional features. Subsequently, a decision tree algorithm is employed to build an ensemble of different classifiers. A weighted mean is used as combiner strategy for the ensemble, where the weights are selected according to the classification abilities of the single classifier established in the validation phase. Experiments are conducted on the benchmark dataset KDD'99 and they prove that the system is able to outperform different ensemble-based algorithms, i.e., bagging and boosting.

A hybrid approach to the intrusion detection problem is presented in Elbasiony et al. (2013). In particular, misuse and anomaly-based approaches are combined in the following way. First of

all, starting with a labeled dataset, the rules for detecting the attacks are generated by a random forest algorithm and then, these rules are employed in the misuse detection stage. Finally, the anomaly detection phase is accomplished by clustering a new training set using the *K*-means algorithm. Indeed, a weighted variant of the *K*-means algorithm is adopted, where the weights assigned to the features are extrapolated from the output of the random forest algorithm. In order to detect anomalous clusters, known attacks are injected in the training dataset and clusters containing a great number of injected attacks are labeled as anomalous. In the operational phase, the network traffic is first analyzed by the misuse detection stage and, if the traffic does not match any known attack, then it is passed to the anomaly detector that analyses it in order to discover a new kind of attacks. Experiments conducted on the KDD'99 dataset show that this system is able to outperform two other unsupervised anomaly detection frameworks, described in Eskin et al. (2002) and Zhang et al. (2008); it is worth noticing that the second is also based on a random forest algorithm.

An hybrid system based on neural networks is presented in Singh et al. (2015). The feature selection stage of the NIDS is based on an ensemble of three different techniques: filtered subset evaluation, consistency subset evaluation and CFS (correlation feature selection) subset evaluation, see Witten et al. (1991). Moreover, during the training stage, in order to reduce considerably the number of samples to be processed and the overall memory consumption, duplicate and similar connections are clustered by the DBSCAN algorithm. In practice, the centers of these clusters represent profiles of connections. Finally, the classification stage is performed by a neural network based algorithm, OS-ELM (Online Sequential Extreme Learning Machine), trained on the connection profiles that result from the previous stage. Experimental testing is conducted on NSL-KDD and on real traffic collected from the Kyoto University. Different tests are conducted in order to asses that each stage of the proposed architecture improve the overall efficiency with respect to the following metrics: detection accuracy, true positive, true negative, false positive, false negative rate, F1-score, precision and execution time.

2.4. Statistical approaches and other techniques

In this section, we will direct our attention to works that are based on statistical models for the analysis of the incoming network traffic. Among these techniques, the most relevant are based on Hidden Markov Models (HMM). Basically HMMs are a Markov process like Markov chains, but contrary to Markov chains, the transition probability functions between the states is not set a priori but it is determined in the training phase. After considering works based on HMMs, we briefly describe a system that employs the Näive Bayes Classifiers in the ensemble and one that exploits the use of Neyman–Pearson Hypothesis Testing in order to reduce the false positive detection rate.

2.4.1. Hidden Markov models

In Ariu et al. (2011) and Ariu and Giacinto (2010), an IDS based on an ensemble of HMMs is presented. Differently from the 2_ν-gram analysis of the payload used in the paper (Perdisci et al., 2009), discussed in Section 2.1, this approach models sequences of length *n*, avoiding the problems due to the approximation of the analysis based on 2_ν-grams. In practice, a window of *n* bytes slides over the payload and it is treated as a sequence that is inserted in the set that represents the payload. This selection produces a certain redundancy intuitively, and this is shown by a quantitative analysis in the paper. In order to alleviate this problem, only a subset of the sequences, selected randomly from the payload representation, is passed to the HMMs for further analysis; this approach has the obvious outcome of speeding up the IDS. Moreover, it provides an additional line of defense against attempts to evade the IDS, as the attacker has indeed no clue of which part of the payload is used for the analysis. The set of sequences obtained in the preprocessing phase is then passed to the classification stage of the IDS. In the training phase, the parameters of each one of the *K* HMMs are established in order to maximize the probability assigned by the HMM model to the packets of normal traffic in the training set. The difference in the *K* models is obtained by choosing different random initialization matrices. In the detection phase, each HMM produces a probability for the set of sequences extracted from the payload. Then, the *K* estimates are combined, similarly to the 2_ν-gram approach of Perdisci et al., using non-trainable combiners, i.e. the minimum, the maximum, the mean, and the geometric mean and consequently an output probability *P* is assigned. Finally, the payload is classified as suspicious if *P* is above a predefined threshold. For the purpose of testing the system, the authors employ three datasets of normal traffic, namely the classic DARPA'99, a dataset containing the HTTP requests sent to the web site of the College of Computing of the University of Georgia Institute of Technology (both these two datasets are the same as employed in the evaluation of McPAD, (see Section 2.1) and a dataset containing the HTTP requests sent at website of Department of Electrical and Electronic Engineering of the University of Cagliari. In addiction to the normal traffic datasets, a database of generic HTTP attacks (Ingham et al., 2007) and variants of the generic attacks generated by a polymorphic engine CLET (the same attacks used in the evaluation of McPAD) were employed. Moreover, simulate XSS-SQL attacks to the web server of the University of Cagliari were also added (Corona et al., 2009a). As for the accuracy, HMMPayl is able to outperform PAYL and McPAD and in addition, it is particularly efficient in detecting attacks based on Cross-Site Scripting and SQL-Injection, in spite of these attacks present payload statistics not significantly different from the normal traffic. However, the processing time of HMMPayl is significantly higher than PAYL, even if the authors point out that the MCS approach can be easily parallelized. A prototype of the system complete of source code is available for download.[3]

Although developed in a completely independent way, a similar approach to HHMPayl is described in Song et al. (2009). The proposed IDS, named Spectrogram, works at the level of the packet payload. In particular, it analyzes HTTP requests in order to discard irrelevant content. Then, the payload is modeled by employing a technique based on *N*-grams, and, in particular, the system adopts a sliding window of prefixed width to inspect the payload. Analogously to HMMPayl, an ensemble of Hidden Markov Models is employed to model the sequence of *n*-grams extracted from the payloads. During the training phase, the hidden parameters of the set of HMMs are determined by machine learning techniques. The user has to choose only two parameters, namely the gram-size and the number of HMMs in the ensemble. Experimental results are conducted on real traffic captured from a couple of operational web servers with added the traffic relative to a number of well-known attacks generated using available exploits. The evaluation considers how the choice of the main parameters (i.e., the gram-size and the number of HMMs) influences the performance of the system and, in addition, some techniques to optimally select the two above-mentioned parameters are proposed. Finally, compared to a work based on *n*-grams, i.e. Anagram (Wang et al., 2006b), the approach based on HMMs scale better with respect to the gram size, in spite of maintaining a reasonable computational load.

[3] https://pralab.diee.unica.it/en/HMMPayl_and_McPAD

Although in this review we direct our attention to intrusion detection systems that rely on the inspection of the network traffic alone, however there are other interesting approaches, named host-based intrusion detection systems, which are worth mentioning. In these kinds of approaches, many aspects of the system are monitored, from the CPU load to the user inputs. Examples of this paradigm are the two works described in the following. In the system proposed in Khanna and Liu (2006), HMMs are employed with the objective of generating models of normal activity for specific users; in practice, the models are linked to parameters such as CPU activity, systems call activity, system process activity, network activity, and session activity. Anomalous activities are then identified if they deviate from normal activities. In order to keep track of periodic changes in the user profiles, the system implements a concept drift detector based on the Kullback–Leibler divergence (Kullback and Leibler, 1951). In Khanna and Liu (2008), the same authors implement the system in a distributed environment, i.e. they consider a situation in which the data are collected from different systems in a subnetwork and then correlated in a central node. For further discussion on the topic of distributed intrusion detection systems, we refer to Section 3.

2.4.2. Other statistical techniques

A system based on data mining and statistical techniques is described in Boro et al. (2012). The authors pursue a meta-ensemble approach, namely a set of classifiers is combined by various simple non-trainable rules and finally different ensembles are combined using a weighted majority voting. First of all, in the preprocessing stage, nfdump is used to acquire the network flow. In the subsequent training phase, methods based on the concept of information gain, previously introduced in Kayacik Günes et al. (2005), are used in order to determine the most relevant feature for each class, into which the traffic is subdivided. The classification stage is then performed by three principal layers. The first layer consists of a set of basic classifiers, i.e., the C4.5 Algorithm, the Näive Bayes Classifiers and the Decision Table. Each classifier assigns a value, representing the probability of belonging to a specific class, to each instance of the traffic flow; then, in the second layer, the probabilities are combined using different functions: i.e., sum, mean, minimum, maximum, median and weighted average. The third layer combines estimates of the previous layer by using the weighted majority voting rule, where basically a weight is assigned to a particular rule based on its past performance. The method proposed is compared to Adaboost.M1, Bagging and Majority Voting and exhibits better performance for all the cases. Datasets used for the comparison comprehend the well-known KDD'99 and two additional datasets, part of the Tezpur University intrusion detection system (TUIDS) datasets. These datasets were generated by the authors deploying a subnetwork and launching attacks against it using exploits available at Packet Storm.[4]

The problem of detecting DDoS attacks is faced in Kumar and Selvakumar (2013). The solution proposed is based on an ensemble of supervised Adaptive Neuro-Fuzzy Systems (ANFS), see Jang (1993). These base classifiers are combined in an ensemble by a variation of the AdaBoost algorithm. Feature extraction and selection is performed at the traffic flow level, extracting six features from traffic dump, among the most relevant with respect to the type of attacks considered. The algorithm, moreover, incorporates a mechanism to reduce the false positive detection based on Neyman–Pearson Hypothesis Testing (Scott and Nowak, 2005). Experiments are conducted both on publicly available datasets (DARPA and CAIDA[5]) and on real traffic data captured from the servers at the University of Naples "Federico II". In the experiments, the system is able to outperform analogous solutions based on ensemble techniques as bagging and boosting. A similar approach is presented in Kumar and Selvakumar (2011). The main difference with respect to the above system is the choice of resilient back propagation (RBP) (Riedmiller and Braun, 1993), as the base classifier for the ensemble.

[4] https://packetstormsecurity.com/

[5] http://www.caida.org/projects/network_telescope/

3. Distributed and collaborative IDS

Attacks conducted against online systems which occur in multiple networks simultaneously (i.e. large-scale stealthy scans, worm outbreaks and distributed DDoS), are among the most dangerous. In addition, coordinated attacks are really difficult to detect using isolated intrusion detection systems since these systems usually monitor only a small portion of a network. Moreover, it is often the case that a particular IDS is more efficient in detecting certain type of attacks, while a different one is more suitable for other type of attacks. For instance, it is well known that misuse based IDSs are very efficient in detecting known attacks based on a database of signatures, but they are rather weak in detecting new attacks. On the contrary, anomaly based IDSs are able to detect a certain number of new attacks but they usually show a rather high false positive rate even in the case of known attacks. As already remarked, architectures of IDSs based on the anomaly detection paradigm implement relatively complex analysis techniques in order to detect outliers in the network traffic being monitored. For instance, these techniques can be either data mining based or statistical based, but in any case, in order to attain a suitable detection rate and to minimize the false positive rate, they usually involve a rather high computational complexity. As a consequence, in order to cope with the high volume data involved in a real time environment, the natural choice is to exploit the potentiality of modern parallel hardware. Finally, detecting anomalies in network traffic often requires the use of time-consuming computational techniques, e.g. data mining and statistical analysis, and it is difficult for them to operate in real-time. Therefore, it is necessary to distribute the computational load required using distributed environments, i.e., parallel machines, cloud computing, etc.

Recently, for the reasons described above, the interest in distributed IDS has grown and different methodologies have been proposed. In the following subsections, we will firstly direct our attention to Collaborative Intrusion Detection Systems (CIDS). Basically, in the CIDS approach, the aim is to correlate information on attacks coming from different subnetworks. Next, we will describe solutions based on data mining techniques that, although not all of them are set up and tested in a fully parallel environment, nonetheless the solutions and the algorithms employed are particularly suitable for a parallel implementation. Lastly, we will include a brief discussion on solutions whose implementation takes full advantage of modern readily available parallel hardware. Namely we will review works, which implement their algorithms on GPGPU capable hardware, clusters, etc.

3.1. Collaborative IDS

The approach to intrusion detection based on the collaborative paradigm has the potential to design IDSs capable of detecting intrusions that occur across large networks and to correlate alarms coming from different sensors. Moreover, CIDSs reduce computational costs by sharing intrusion detection resources among different networks and they mitigate the problem of false alarms that would be generated by a single IDS operating alone. The main components of a CIDS, in accordance to the work of Zhou et al. (2010), are a *detection unit* and a *correlation unit*. The detection unit consists of multiple sensors deployed in a network that

generates low level intrusion alerts. These low level intrusion alerts are combined by the correlation unit, then high level reports are generated and finally the real nature of the attacks is confirmed. The design of a CIDS faces then two main challenges, namely *system architecture* and *alert correlation* methodologies. From the point of view of the architecture, we can distinguish the *Centralized approach*, in which the data are collected by different sources but analyzed and correlated by a single central unit. This approach suffers from various drawbacks; first, as most of the work is done by a single unit, serious problems of reliability are present, since the failure of the central unit causes the CIDS stops from properly working. Moreover, a large computational power is required in order to prevent the degradation of the performance of the CIDS. In the *hierarchical approach*, the correlation units are structured in a pyramidal way, with layers that perform increasing complex analysis on the data acquired by the detection units, but the final analysis is always performed by a central unit, namely the top of the pyramid. This approach frees the central unit from a part of the computational load, with respect to the centralized approach, but suffers from similar analogous problems to the ones mentioned above. Finally, there is a *fully distributed approach* in which the detection and the correlation work is spread among different units. This approach can be set up by a P2P network, for instance, but it is often hard to coordinate a large number of nodes in order to attain a suitable detection accuracy and to guarantee the scalability of the global system. Indeed, the mechanism implemented to share information should be efficient enough to hinder the speed at which menaces spread across a network; moreover, it should remain so as nodes are added to the system. Just to give a glimpse of how hard this problem is, in some extreme cases, as in the case of the SQL-slammer worm, it has been estimated that as the worm began spreading through the Internet it infected over 90% of the vulnerable hosts just in ten minutes. In the following, we review the main collaborative IDS and a summary of their characteristics is reported in Table 3.

In Zhou et al. (2009), the authors focus their attention on the problem of alert correlation, with the purpose of improving the scalability without renouncing the accuracy of the system, in particular for DDoS attacks and worms. In order to accomplish the above task, the architecture of the system is based on two main ideas. First of all, to speed up the correlation of the raw alarms, exploiting the knowledge on the different types of attacks, the search is limited to specific "patterns", where pattern means a subset of features. For instance, the features of the network flow, "source address", "source port"' and "destination port" should be sufficient to correlate raw alarms for a broad class of DDoS attacks. Moreover, in order to deal with the large number of raw alerts coming from the various IDSs of the network, the alert correlation system is implemented in a fully distributed form, based on a P2P protocol. Each member of the alert network is made up of a detection unit and a correlation unit. The detection unit is responsible for collecting raw alarms relative to the local traffic. These alerts are analyzed by the correlation unit that implements a multi-dimensional clustering algorithm. In particular, the traffic is preliminarily filtered according to a combinations of key features or patterns of different attacks previously described, the filtered traffic is clustered and the traffic patterns are identified as suspicious by building a pattern lattice. Afterwards, the local results of this analysis are exchanged among the peers of the network for further analysis and correlation. The proposed system is tested on two data sets obtained from the Dshield.org website, which comprises a large number of firewall and NDIS logs collected from various platforms all over the world. In particular, one of the datasets employed in the testing contains data collected during the outbreak of the SQL-Slammer worm. The system is evaluated by conducting a large scale experiment on the PlanetLab

Table 3
Summary table of papers reviewed, CIDS. The column Evaluation indicates whether the framework is also evaluated on a real network.

Author(s)	Year	Technique	Correlation	Architecture	Dataset(s)	Evaluation
Zhou et al. (2009)	2009	Raw alarms aggregation	Clustering	Fully distributed	Internet Storm Center www.Dshield.org	PlanetLab network of the Princeton University
Boggs et al. (2011)	2011	n-grams analogously to the ANAGRAM algorithm (Wang et al., 2006b), models are encoded by Bloom filters	Exchanging models of traffic	Fully distributed	Real traffic from different sources	Real traffic from different sources
Hu et al. (2014)	2014	Ensemble of Gaussian mixture models	Particle swarm optimization, SVM	Fully distributed	KDD'99	None
McEachen and Wai Kah (2007)	2007	Aggregate features	Conversation Exchange Model	Centralized	M.I.T. Lincoln Lab 1999	Appositely set up subnetwork
Perdisci et al. (2010) and Roberto et al., 2013	2010, 2013	Statistical techniques	Hierarchical clustering: BIRCH Zhang et al. (1996)	Centralized	Malware samples	None

network of the Princeton University in comparison to a fully centralized approach using metrics as detection accuracy and message exchange rate. The experiments demonstrates that the fully distributed approach is more efficient than the centralized approach in terms of the time required to correlate the alerts. Moreover, the above-mentioned probabilistic approach is really efficient in the case of stealthy attack scenarios.

In Boggs et al. (2011), a method to correlate alerts collected by different IDSs is presented. The IDSs inspect the local traffic by analyzing the content of the payload of HTTP requests. The analysis is based on the technique of *n*-grams analogously to the ANAGRAM algorithm (Wang et al., 2006b). The detection stage in the ANAGRAM system is improved following the technique of STAND (Sanitization Tool for ANomaly Detection) (Cretu et al., 2008, 2007) by a sanitization phase of the training set; then, the technique introduced in Cretu-Ciocarlie et al. (2009) is employed to tune the sensor parameters automatically. Once the payload is processed, the models associated with the normal traffic and with the suspicious packets are stored by using Bloom filters (Burton, 1970), which constitute their signatures. It is worth noting that storing the models of traffic by using Bloom filter has also the positive effect of preserving the privacy of the users during the correlation phase in which signatures are shared across different sites. Cross-site correlation of the alerts is obtained by using the method described in Locasto et al. (2005), i.e. the Worminator system. After that signatures are shared across sites, correlation is performed locally, by comparing the unencoded local alerts with the Bloom filter representing the alerts coming from remote systems. The system is evaluated on the datasets collected from the web servers of three different locations: http://www.cs.columbia.edu, http://www.gmu.edu and http://www.cs.gmu.edu. During the testing period the prototype was able to detect a considerable number of application-specific attacks previously unknown as well as a wide range of well-known attacks without human intervention. Finally, the system shows a very low rate of false positives.

In Hu et al. (2014), another CIDS is presented. Its architecture is based on different local nodes, in which, during the training stage, the local network traffic is modeled by using an ensemble of gaussian mixture models. In practice, the ensemble is built by a variation of the AdaBoost algorithm, adapted to a streaming environment. Local parametric detection models are exchanged among the nodes and then they are combined on the local nodes in order to constitute a global model, by using a process based on particle swarm optimization and SVM to select the fusion function. Experimental tests, performed on the KDD'99 dataset, show the effectiveness of the system in improving the detection accuracy, in spite of exchanging a relative small quantity of data among the nodes, i.e., the parameters that determine the Gaussian mixture models.

It should be remarked that since only models of the traffic are shared and not network traffic itself, this guarantees the privacy of the various nodes involved.

An interesting point of view on the topic of correlating intrusion data coming from different nodes of a given subnetwork is proposed in McEachen and Wai Kah (2007). The solution proposed relies on the concept of *conversation exchange model* introduced in Ford (2004). This scheme has the scope of modelling the network traffic dynamic using aggregate traffic features and decision trees that encode, for each protocol, the way in which the packets are exchanged in the subnetwork considered. This process permits estimation of the distribution of the number of packets exchanged across the nodes of the subnetwork during nominal operations. At detection time, this distribution is monitored and techniques inspired by statistical mechanic and thermodynamics are used to detect anomalies in the distribution of the packets due to malicious activities. This approach is tested using different scenarios, i.e., different attacks on different segments of the network either occurring in different time slots or simultaneously. For each scenario, the system was able to detect the presence of attacks; however, it is worth remarking that the system detects deviations in the normal network dynamic. In order to detect the type of the attack or the nodes in which it has occurred, a deeper inspection at the level of packet content or of the network flow should be used, as analogous approaches described in this review.

In Perdisci et al. (2010), the authors present a system to cluster malwares based on their behavior. Simple statistical features are extracted from the HTTP traffic generated by the malware, i.e., the length of the GET requests, the number of requests, etc. Then, the models of the malware behavior are clustered in a two stage procedure. In the first stage, the BIRCH (Zhang et al., 1996) clustering algorithm is used, and its results are further refined by a hierarchical clustering. During the clustering process, cluster stability is performed by using the Davies–Boulding index in order to assess whether the clusters obtained are compact and well separated. Lastly, the signatures are generated from the clustering stages by using the Token-Subsequences algorithm (Newsome et al., 2005). Basically, for each cluster, signatures are generated considering substrings that are common to all HTTP payloads in that cluster. In Roberto et al. (2013), the same authors introduce some tweaks in the architecture, which presents a more balanced computational load and a better scalability in comparison with the original method. Experiments are conducted on many datasets of malware activities collected from various sources and the system proves to be able to generate signatures for different types of HTTP malwares.

3.2. Data mining-based approaches

As there are not many data mining distributed implementation of intrusion detection systems, this section also includes some works based on data mining that are particularly suitable to be implemented in a parallel/distributed environment. A summary of the main works, reviewed on the following, is reported in Table 4.

The work in Wang et al. (2014) describes a framework for the problem of adaptive intrusion detection over unlabeled HTTP traffic streams, which follows the autonomic paradigm proposed by IBM (Kephart and Chess, 2003). Indeed, it is able to self-label the incoming data stream, to update continuously the detection model and moreover, it is capable of adapting the detection model after a change happens in the data stream. The IDS works by inspecting HTTP packets at the payload level by essentially employing a 1-grams technique. After the preprocessing phase, in an initial stage, the data stream is clustered by an affinity propagation algorithm that employs a message passing technique named WAP (weighted affinity propagation). It is worth noting that clustering algorithms based on affinity propagation are graph-based and require a high computational load; therefore, they are generally more adapted to off-line analysis. However, the system adopts a message updating mechanism, which is suitable to be implemented in a distributed approach and that would extend its use to an online system. After the clustering phase is completed, the system analyzes the size and the sparseness of each cluster, in order to identify anomalous items. When a change is detected in the traffic, the models are rebuilt by restarting the clustering process. Three parameters control the rebuilding phase, i.e., the number of suspicious items, the length of the time window and the number of suspicious items since the last clustering stage. If one of these three parameters exceeds a certain threshold, the clustering phase is restarted. Finally, the system is compared with other existing clustering solutions: the *k*-Nearest Neighbor algorithm, the Principal Component Analysis (Jolliffe, 2002), a one class SVM method and a version of the system based on the

Table 4
Summary table of papers reviewed, data mining distributed (In the last column (Implementation): D indicates distributed implementation, S indicates suitable to distributed implementation).

Author(s)	Year	Technique(s)	Dataset(s)	Preprocessing	Metric(s)	Implementation
Wang et al. (2014)	2014	HTTP packet payload analysis based on 1-grams, WAP (Weighted Affinity Propagation) clustering	KDD'99, real traffic	None	ROC curve	S
Casas et al. (2012)	2012	DBSCAN algorithm (Ester et al., 1996), clustering ensemble: Evidence Accumulation for Ranking Outliers (EA4RO) Casas et al. (2012)	KDD'99, real traffic	Random Projections	ROC curve, detection accuracy	S
Leung and Leckie (2005)	2005	Grid density clustering (Nagesh et al., 2000)	KDD'99	None	ROC curve	S
Folino et al. (2010)	2010	Genetic Programming, Ensemble of Decision Trees	KDD'99	None	Detection rate, false positive rate, detection accuracy	D
Folino et al. (2016)	2016	Genetic Programming, Meta-Ensemble of Decision Trees	KDD'99, real traffic	None	Detection rate, false positive rate, detection accuracy	D
Huang et al. (2016)	2016	Ensemble of on line sequential extreme learning machines	KDD'99	None	Detection accuracy	D
Bukhtoyarov and Zhukov (2014)	2014	Genetic Programming, Neural Networks	KDD'99	None	Detection accuracy, false positive rate	D
Brahmi et al. (2012)	2011	Multi-Agent System, DBSCAN clustering, association rule mining	Real traffic	None	Detection rate and false positive rate	S

Sequential Karhunen–Loeve (Levey and Lindenbaum, 2000) transform. Testing is conducted on the KDD'99 dataset and on two large real HTTP traffic streams and the proposed system outperforms the other methods.

The approach in Casas et al. (2012) presents an IDS (named UNIDS, *Network unsupervised intrusion detection system*) based on the misuse detection paradigm. The solution proposed, based on a ensemble clustering method, is suitable to be easily implemented on parallel/distributed architectures. In addition, although the authors test it on data stream coming from a single source, they point out that it is well capable of dealing with data collected from different sources. The algorithm works in three phases: pre-processing, clustering and outlier identification. In the pre-processing phase, network traffic is captured from a single source using consecutive time windows of prefixed length. Afterwards, the captured traffic is aggregated into flows, and from each flow, some numerical features, i.e., source and destination network prefixes, traffic per time slot, etc. are extracted and a multi-dimensional vector representing the flow is obtained. A change detection algorithm is applied to the resulting vector and permits to flag some flows in the time slot as anomalous, which are passed to clustering phase. Then, in the clustering phase, a sub-space clustering algorithm is applied to the data, in the following way. First, the flows are aggregated using either IPsrc or IPdst aggregation keys and each aggregated flow is described by a number of traffic attributes. Multidimensional data, obtained as described before, are projected on multiple 2-dimensional subspaces of the feature space and a density-based DBSCAN algorithm (Ester et al., 1996) is performed on each subspace by using different notions of similarity or distance (i.e., a simple Euclidean distance, but also more complex statistical-based similarities). Finally, the results of the different clustering applied in each subspace are combined by an *evidence accumulation clustering* method (Fred and Jain, 2005); in particular, the algorithm of *Evidence Accumulation for Ranking Outliers (EA4RO)* is employed, better described in Casas et al. (2012). In the last phase, the top-ranked outlying flows are flagged as anomalies, using a simple thresholding detection approach.

Experimental results are conducted both on the well-known KDD'99 dataset and on traffic captured from two real networks. UNIDS is compared to traditional misuse based NIDS and it is comparable to the traditional approaches in its ability to detect unknown attacks; however, it is suitable to parallel computation, which permits a drastic reduction in the overall analysis time of the system.

An unsupervised approach based on clustering is introduced in Leung and Leckie (2005). The system adopts an adaptive grid algorithm, named fpMAFIA, based on the CLIQUE clustering technique (Agrawal et al., 2005), and its improvement pMAFIA described in Nagesh et al. (2000). The fpMAFIA algorithm basically employs *frequent-pattern tree* techniques in order to mine frequent item sets. This algorithm, based on the grid density clustering paradigm, starts by partitioning the features space in a grid, and part of the elaboration is basically performed by counting instances of the dataset in each element of the grid and establishing their relative density there, this involves the computation of the mean distance between two instances. Based on this elaboration grid elements are then merged to form clusters. The above analysis can then be implemented in parallel form by assigning, for instance, a certain number of grid elements to a single elaboration unit, e.g. a CPU core. The methodology proposed is tested on the KDD'99 dataset and it shows good performance in terms of elaboration speed and detection rate, but false positive rate is relatively high with respect to other methods proposed in the literature.

In Folino et al. (2010), Folino et al. propose an ensemble-based distributed data mining algorithm, which adopts a genetic

programming approach to build the classifiers composing the ensemble. The framework is used to improve the detection accuracy when classifying malicious or unauthorized network activity and the dataset is distributed across multiple nodes and the framework uses network profiles to improve the detection accuracy for the minority classes (i.e. the attacks). Experimental results show that the system outperforms the C4.5 algorithm and its boosting and bagging version as for the minority classes on the well-known KDD'99 dataset. The same authors (Folino et al., 2016) extend the above-cited approach by using ensembles specialized to detect particular types of attack and by adopting a non-trainable function to combine each specialized ensemble. The experiments are conducted on the recent ISCX dataset and demonstrate that the approach is able to cope with the intrusion detection task, even in the case of really unbalanced classes, by exploiting the advantages of the specialized classifiers.

Huang et al. (2016) propose an architecture based on an ensemble of on-line sequential extreme learning machines, which is a type of neural network that supports on line sequential learning, i.e., the predictor is updated as soon as new instances become available, see Liang et al. (2006) for more details. The approach combines different ensemble techniques such as Bagging, Subspace Partitioning and Cross Validation; in addition, in order to speed up the overall training phase, the data are distributed across the nodes of a cluster by means of the Hadoop framework. Experimental testing is conducted on the well-known KDD'99 dataset and the parallel implementation obtain a good scalability and a considerable reduction of the execution time and maintains the same accuracy as the sequential version of the approach.

Another distributed architecture based on neural networks is presented in Bukhtoyarov and Zhukov (2014). In this system, a probabilistic approach (more detail on this technique are given in Bukhtoyarov and Semenkina (2010) is used for the generation of a neural network on each node of a network. Then, the neural networks are combined in an ensemble, in which the combination function is evolved by employing genetic programming using the same technique presented in Bukhtoyarov and Semenkina (2010). Afterwards, each node classifies the traffic independently from the other nodes and it inquires the ensemble only in the case in which it is not “confident” in its prediction. “Confidence” is implemented by a function based on the signal level at the output of a neuron corresponding to the class determined by the individual neural network classifier; if the value of this function is below a prefixed threshold, then the individual classifier needs to require the help of the ensemble. The system is evaluated on the KDD'99 dataset in term of Detection Accuracy and False Positive Rate against standard ensemble algorithms and in general it is capable of equal or better score than the competition with optimal choice of the above-mentioned threshold for the confidence function.

The architecture in Brahmi et al. (2012) combines misuse based and anomaly detection techniques with the multi-agent paradigm. Indeed, the analysis of the traffic is subdivided among many agents and some of them are devoted to collecting and pre-preocessing the traffic; then, the traffic is filtered by the so-named misuse detection agent, which checks it by using some rules related to known attacks. Then, the filtered traffic is passed to the anomaly detection agent that analyze it by using a clustering algorithm that combines *K*-means and DBSCAN and identify normal and abnormal traffic. Finally, the result of this analysis is passed to the rule mining agent that, by using an association rule-based technique, builds new rules that will be used by the misuse detection agent. The system is implemented using the JADE (Java Agent DEvelopment Framework), and the agent platform and is suitable to be implemented in a distributed environment, however the experiments are conducted on a sequential version of the framework. Experimental testing is conducted on a real traffic dataset, in which the attacks are injected by the *Metasploit*[6] tool. The low bandwidth consumption and the network latency prove that the system is capable to cope with an increasing number of attacks and volume of network traffic. Moreover, the system is able to outperform largely used IDS systems, as SNORT, in term of detection rate and false positive rate.

3.3. High-performance implementations

Recently, efforts have been made in order to take advantage of modern parallel hardware in the analysis of network traffic for the purpose of intrusion detection (Vokorokos et al., 2014). In fact, in the last few years, massively parallel hardware is readily available in the form of off the shelf components, namely multi-core CPUs and graphic hardware capable of general purpose computations (GPGPUs).

Most of the works present in the literature try to boost the signature verification stage by using a parallel implementation. For instance, there are projects that expand existing solutions in order to take advantage of GPGPU hardware, i.e., `GNORT` that is based on the well-known open source solution `SNORT`. Others works adopt a signature-based solution from scratch, and concentrate their efforts on implementing a very efficient signature verification based on parallel pattern matching algorithms (Jamshed et al., 2012; Bellekens et al., 2014). This approach can be extended by distributing the signature verification among different nodes. In this case, a particular attention is devoted to the problem of synchronizing the elaboration between nodes, see Vasiliadis et al. (2011), Jamshed et al. (2012), Vokorokos et al. (2013), and Schaelicke et al. (2005).

An interesting work is presented in Bandre and Nandimath (2015), in which GPGPU hardware is paired to a cluster. In particular, computational tasks relative to signature matching are performed by the GPGPUs. The cluster is employed to process and mine the data streams of the alarms, and to assess the most common vulnerabilities and threats, by exploiting the advantages of the `Hadhoop` framework.

In order to take full advantage of a parallel implementation, a key point is to guarantee the scalability and a good load balancing. To this aim, the system proposed in Zheng et al. (2015) adopts a solution called negative pattern matching, which splices the preprocessed network traffic into small segments suitable to be analyzed in parallel and to perform a fine grained load balancing of the computational load. This technique ensures that no key information is lost in the process. Moreover, the rule set is subdivided into independent subsets, which can be applied independently, providing another means to parallelize the pattern matching process. Finally, the pattern matching is performed by a technique based on the ternary content addressable memory method. Experimental results show that the above-mentioned techniques are able to improve remarkably pattern matching performance and scalability in many real-world scenarios.

The approach in Alshawabkeh et al. (2010) uses an anomaly-based technique, which adopts a parallel local outlier algorithm. The technique used comprises a heavy phase of computation of the mutual distances of the points in some regions of the dataset. Therefore, a parallel implementation that increases the speed of the algorithm can be set up, as the computation of the distances can be performed independently. It would be interesting to implement more sophisticated solutions on parallel hardware and to test the systems on traffic data captured and processed in real time. For instance, modern database systems such as Oracle Database 10g (see Campos and Milenova, 2005) provide all the

[6] http://www.metasploit.com/

necessary instruments to easily implement an intrusion detection architecture, from the data preprocessing to the phase of clustering and analysis. The possibility of exploiting the potentialities of the Oracle grid computing infrastructure is really interesting. In fact, this infrastructure enables sharing the content of the databases, or in other words the network traffic to be analyzed using the power capacity of many computing nodes. Therefore, it is possible to distribute the computational load in a transparent way and to ensure the availability of flexible on-demand computing resources capable of guaranteeing scalability and reliability. This infrastructure could provide a good testbed to compare different solutions and techniques.

4. Discussion: lesson learned and open issues

As discussed in the introduction to the paper, a modern NIDS has to cope with the increased speed of the underlying network connections. Increased network connections naturally mean an increase in the number of alarms generated, especially in the case of signature-based NIDS that works at the packet level. As a consequence, there is an ever increasing need for solutions that do not overwhelm human operators with a number of unmanageable alarms; indeed, it would be desirable to implement a mechanism that aggregates elementary alarms and presents higher level information to the operators.

The increase in the volume of data to be processed makes it really hard for an NIDS to work in real-time using sequential hardware. Finally, a modern NIDS needs to be capable of dealing with new undocumented attacks, for which a signature is not available (0-day attacks).

In view of the above remarks, solutions based on algorithms that can be implemented in parallel/distributed environment should be preferred. It is well known that currently it is easy to deploy hardware solutions capable of executing massively parallel algorithms, by using multi-core CPUs and GPGPU hardware, using off the shelf components. In this review, ensemble-based framework were individuated to be particularly suitable to exploit easily the power of parallel hardware.

By analyzing the data mining algorithms present in the literature for the intrusion detection task, though generally the authors note that their solutions are suitable to be implemented in parallel environments, there is usually a lack of experiments that test the proposed solutions on real parallel hardware and on real network traffic.

Preprocessing steps and feature selection and extraction are key points in designing an efficient NIDS and the other part of the algorithm could benefit from an opportune design of these phases. In the reviewed works, we remarked the different preprocessing technique adopted, usually based on statistical, data-mining and information theoretic methods.

More in general, the works reviewed can be subdivided into two types. The first type considers the systems that inspect the traffic at the level of the packets payload. They usually concentrate on textual-based protocols, i.e. whose content is represented in plain text, such as for instance `HTTP`. In these cases the analysis of the payload is based on the technique of *n*-grams. The second type made up of systems that model the traffic at the level of network flow. The two types of approach have their weaknesses and strengths. First of all, the most computational intensive is generally the one that inspects package payload. It turns out that solutions based on the network flow are more adequate to modern environments involving high speed traffic and a consequent higher volume of data (Sperotto et al., 2010). On the other hand, packet inspection-based methods are particularly efficient in the detection of attacks that present anomalous requests, i.e., cross-site scripting. Network flow analysis is particularly useful when applied to complex attacks as the case of DDoS. For the reasons explained above, it should be interesting to try and implement a hybrid solution taking the best of the two approaches. It is worth noticing that in any case both the approaches require a high computational load, especially if the analysis stage is based on the anomaly detection paradigm.

Consequently, it would be desirable to set up an anomaly detection mechanism relying on traffic models that are based on aggregate features; usually, traffic is aggregated in flows and flow properties are used as features; moreover, from an higher level, it is important also to consider aggregations and statistics based on these features such as entropies of flow characteristics for individual source of IP addresses, number of flows with given characteristics, and so on. For instance, recall the approach (McEachen and Wai Kah, 2007) described in Section 3.1, in which aggregate features are employed in order to build up a model of traffic for an entire subnetwork. Then, after a first analysis of the aggregate features, an algorithm could analyze, at a deeper level, only the time windows in which the change detector individuates an anomaly. As this mechanism usually must work in real-time, a tradeoff between accuracy and speed should be reached.

In the collaborative IDS approach, the alarms are shared between different detection points. This is attained by sharing the raw alarms, i.e. those generated locally by each NIDS must be exchanged among different sites and in addition, a mechanism to collect and analyze them must be implemented. The main advantages of this methodology are the ability to detect complex and correlated activities such as malware diffusion, distributed DoS and scans that span entire subnetworks. Moreover, the sharing mechanism could enable exchange of informations among systems concerning attacks individuated in a node, but not currently observed on the other parts of the network. The current literature on the subject suggests that, in order to prevent bottlenecks, the collection and the analysis of the alarms should be implemented in a fully distributed way. In practice, each node should be able to collect, analyze and elaborate the alarms and then to share globally the final results. Typically, this approach uses P2P network with their relative drawbacks; in fact, particular attention must be paid to the sharing mechanism in order to guarantee scalability and performance.

Moreover, for the problem of managing new attacks, techniques based on anomaly detection and, in particular, on clustering algorithms are in principle suitable to operate in an environment that requires the analysis of data streams containing previously unseen anomalous content. Nonetheless, generally there is a lack of experimentation, in which the solutions are tested in a real-life environment; consequently, it is not always clear whether the solutions proposed are able to operate in real-time or in near real-time. It is also desirable to consider distributed clustering frameworks employing a mechanism for sharing the results across multiple nodes in a similar way to the alert correlation process described above; indeed, this sharing mechanism should be implemented again in a fully distributed way in order to prevent bottlenecks.

Sharing knowledge across multiple nodes either in the form of alerts or as the result of clustering and analysis, poses problems concerning the privacy of the involved subjects and institutions; in order to mitigate this problem as much as possible, the sharing mechanism should be based on abstract models of traffic, high level alerts, and should avoid the exchange of sensitive information such as the packet content. Recall for instance (Hu et al., 2014), in which only the parameters of the abstract models of traffic are exchanged, and (Boggs et al., 2011), in which signatures of different types of traffic, first encoded by Bloom filters for protecting users' privacy, are shared.

As a general consideration, it is really hard to compare the different methods, based on different implementations, proposed in the literature. A common testbed on which to perform extensive testing on different datasets or even better in real environments and in different scenarios, would be remarkably useful for the intrusion detection community.

5. Conclusions

This work mainly reviews data mining algorithms that present an approach to network intrusion detection suitable to take advantage of modern parallel/distributed and cloud environments. Therefore, the aspects of the general design of a network intrusion detection systems that play a key role in designing an efficient system are considered. With this aim in mind, solutions based on the ensemble paradigm are analyzed, as this kind of algorithm is easily implementable on distributed environments and presents many advantages in the intrusion detection field. Collaborative intrusion detection systems are particularly interesting as, by correlating information coming from different nodes, they permit the discovery of more complex attacks and generally present a lower number of false positive alarms. In fact, they share informations with the other nodes in a subnetwork, either in the form of correlated alarms or in the form of models for detecting the attacks and therefore they minimize the rate of false positive alarms and are able to detect complex attacks that spread across entire subnetworks.

In addition, in order to design a modern architecture capable of coping with the high amount of data involved in modern high speed networks, high-performance distributed frameworks must be also considered. Finally, an overlooked issue in the intrusion detection field is the lack of a common distributed testbed, which permits the different solutions to be deployed and compared in a real environment.

Acknowledgment

This work has been partially supported by MIUR-PON under Project PON03PE_00032_2 within the framework of the Technological District on Cyber Security.

References

Aburomman Abdulla Amin, Ibne Reaz Mamun Bin. A novel SVM-kNN-PSO ensemble method for intrusion detection system. Appl. Soft Comput. 2016;38:360–72.

Agrawal Rakesh, Gehrke Johannes, Gunopulos Dimitrios, Raghavan Prabhakar. Automatic subspace clustering of high dimensional data. Data Min. Knowl. Discov. 2005;11(1):5–33.

Alshawabkeh, Malak, Jang, Byunghyun, Kaeli, David, 2010. Accelerating the Local Outlier Factor algorithm on a GPU for intrusion detection systems. In: Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units, GPGPU-3, ACM, New York, NY, USA, pp. 104–110.

Amiri Fatemeh, Yousefi Mohammad Mahdi Rezaei, Lucas Caro, Shakery Azadeh, Yazdani Nasser. Mutual information-based feature selection for intrusion detection systems. J. Netw. Comput. Appl. 2011;34(4):1184–99.

Ariu, Davide, Giacinto, Giorgio, 2010. HMMP ayl: an application of HMM to the analysis of the HTTP payload. In: Tom Diethe, Nello Cristianini, John Shawe-Taylor (Eds.), Proceedings of the First Workshop on Applications of Pattern Analysis, WAPA 2010, September 1–3, vol. 11. Cumberland Lodge, Windsor, UK, pp. 81–87.

Ariu Davide, Tronci Roberto, Giacinto Giorgio. HMMPayl: an intrusion detection system based on Hidden Markov Models. Comput. Secur. 2011;30(4):221–41.

Bandrek, S.R., Nandimath, J.N., 2015. Design consideration of network intrusion detection system using Hadoop and GPGPU. In: 2015 International Conference on Pervasive Computing (ICPC), January, pp. 1–6.

Bellekens, Xavier J.A., Tachtatzis, Christos, Atkinson, Robert C., Renfrew, Craig, Kirkham, Tony, 2014. A highly-efficient memory-compression scheme for GPU-accelerated intrusion detection systems. In: Proceedings of the 7th International Conference on Security of Information and Networks, SIN 14. ACM, New York, NY, USA, pp. 302:302–302:309.

Hussain Bhuyan M, Bhattacharyya DK, Kalita JK. Survey on incremental approaches for network anomaly detection. Int. J. Commun. Netw. Inf. Secur. 2011a;3(3).

Bhuyan MH, Bhattacharyya DK, Kalita JK. Network anomaly detection: methods, systems and tools. IEEE Commun. Surv. Tutor. 2014;16(1):303–36 First.

Bhuyan, Monowar H., Bhattacharyya, D.K., Kalita, J.K., 2011. NADO: network anomaly detection using outlier approach. In: Proceedings of the 2011 International Conference on Communication, Computing& Security, ICCCS '11. ACM, New York, NY, USA, pp. 531–536.

Bhuyan Monowar H, Bhattacharyya DK, Kalita JK. Surveying Port Scans and their detection methodologies. Comput. J. 2011c;54(October (10)):1565–81.

Monowar H. Bhuyan, Bhattacharyya, D.K., Kalita, J.K., 2012. An effective unsupervised network anomaly detection method. In: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI '12. ACM, New York, NY, USA, pp. 533–539.

Bloom Burton H. Space/time trade-offs in hash coding with allowable errors. Commun. ACM 1970;13(7):422–6.

Boggs, Nathaniel, Hiremagalore, Sharath, Stavrou, Angelos, Stolfo, J. Salvatore, 2011. Cross-domain collaborative anomaly detection: so far yet so close. In: Sommer, Robin, Balzarotti, Davide, Maier, Gregor (Eds.), Recent Advances in Intrusion Detection, Lecture Notes in Computer Science, vol. 6961. Springer, Berlin, Heidelberg, pp. 142–160.

Borji, Ali, 2007. Combining heterogeneous classifiers for network intrusion detection. In: Cervesato, Iliano, (Ed.), Advances in Computer Science - ASIAN 2007. Computer and Network Security, Lecture Notes in Computer Science, vol. 4846. Springer, Berlin, Heidelberg, pp. 254–260.

Boro, Debojit, Nongpoh, Bernard, Bhattacharyya, Dhruba K., 2012. Anomaly based intrusion detection using meta ensemble classifier. In: Proceedings of the Fifth International Conference on Security of Information and Networks, SIN '12. ACM, New York, NY, USA, pp. 143–147.

Brahmi, Imen, Yahia, Sadok Ben, Aouadi, Hamed, Poncelet, Pascal, 2012. Towards a multiagent-based distributed intrusion detection system using data mining approaches. In: Proceedings of the 7th International Conference on Agents and Data Mining Interaction, ADMI'11. Springer-Verlag, Berlin, Heidelberg, 2012, pp. 173–194.

Breiman Leo. Random forests. Mach. Learn. 2001;45(1):5–32.

Bukhtoyarov, V.V., Semenkina, O.E., 2010. Comprehensive evolutionary approach for neural network ensemble automatic design. In: 2010 IEEE Congress on Evolutionary Computation (CEC), Barcelona, Spain, July, pp. 1–6.

Bukhtoyarov, Vladimir, Zhukov, Vadim, 2014. Ensemble-distributed approach in classification problem solution for intrusion detection systems. In: Corchado, Emilio, Lozano, José A., Quintián, Héctor, Yin, Hujun (Eds.), Intelligent Data Engineering and Automated Learning, IDEAL 2014, Lecture Notes in Computer Science, vol. 8669. Springer International Publishing, Salamanca, Spain, pp. 255–265.

Campos, M.M., Milenova, B.L., 2005. Creation and deployment of data mining-based intrusion detection systems in Oracle Database 10g. In: Fourth International Conference on Machine Learning and Applications, 2005. Proceedings. IEEE Computer Society, Los Angeles, California, USA, December.

Casas Pedro, Mazel Johan, Owezarski Philippe. Unsupervised network intrusion detection systems: detecting the unknown without knowledge. Comput. Commun. 2012;35(7):772–83.

Corona, I., Ariu, D., Giacinto, G., 2009. HMM-Web: a framework for the detection of attacks against web applications. In: IEEE International Conference on Communications, 2009. ICC '09, June, pp. 1–6.

Corona, Igino, Giacinto, Giorgio, Mazzariello, Claudio, Roli, Fabio, Sansone, Carlo, 2009. Information fusion for computer security: state of the art and open issues. Inf. Fusion, 10(4), 274–284, Special Issue on Information Fusion in Computer Security.

Cretu, G.F., Stavrou, A., Locasto, M.E., Stolfo, S.J., Keromytis, A.D., 2008. Casting out demons: Sanitizing training data for anomaly sensors. In: IEEE Symposium on Security and Privacy, 2008. SP 2008, May, pp. 81–95.

Cretu, Gabriela F., Stavrou, Angelos, Stolfo, Salvatore J., Keromytis, Angelos D., 2007. Data sanitization: improving the forensic utility of anomaly detection systems. In: Proceedings of the 3rd Workshop on Hot Topics in System Dependability, HotDep'07, USENIX Association, Berkeley, CA, USA.

Cretu-Ciocarlie, Gabriela F., Stavroum, Angelos, Locasto, Michael E., Stolfo, Salvatore J., 2009. Adaptive anomaly detection via self-calibration and dynamic updating. In: Recent Advances in Intrusion Detection, Springer, Saint-Malo, France, pp. 41–60.

Damashek Marc. Gauging similarity with *n*-grams: language-independent categorization of text. Science 1995;267(5199):843–8.

Dasarathy Belur V. Intrusion detection. Inf. Fusion 2003;4(4):243–5.

Davis David L, Bouldin Donald W. A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. PAMI—1 1979(2):224–7.

Davis Jonathan J, Clark Andrew J. Data preprocessing for anomaly based network intrusion detection: a review. Comput. Secur. 2011;30(6–7):353–75.

Dhillon Inderjit S, Mallela Subramanyam, Kumar Rahul. A divisive information theoretic feature clustering algorithm for text classification. J. Mach. Learn. Res. 2003;3(March):1265–87.

Dua S, Du X. Data Mining and Machine Learning in Cybersecurity. CRC Press, Taylor & Francis Group; 2011.

Duda RO, Hart PE, Stork DG. Pattern Classification. 2nd edition. New York, NY: John Wiley & Sons; 2001.

Dunn JC. Well-separated clusters and optimal fuzzy partitions. J. Cybern. 1974;4 (1):95–104.
Elbasiony Reda M, Sallam Elsayed A, Eltobely Tarek E, Fahmy Mahmoud M. A hybrid network intrusion detection framework based on random forests and weighted k-means. Ain Shams Eng. J. 2013;4(4):753–62.
Eskin, Eleazar, Arnold, Andrew, Prerau, Michael , Portnoy, Leonid, Stolfo, Salvatore J., 2002. A geometric framework for unsupervised anomaly detection. In: Barbará, Daniel, Jajodia, Sushil (Eds.), Applications of Data Mining in Computer Security, Advances in Information Security, vol. 6. Springer, US, pp. 77–101.
Ester, Martin, Kriegel, Hans-Peter, Sander, Jörg, Xu, Xiaowei, 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), vol. 96. AAAI Press, Portland, OR, USA, pp. 226–231.
Folino Gianluigi, Pizzuti Clara, Spezzano Giandomenico. An ensemble-based evolutionary framework for coping with distributed intrusion detection. Genet. Program. Evol. Mach. 2010;11(June (2)):131–46.
Folino, Gianluigi, Sabatino, Pietro, Pisani, Francesco S., 2016. Combining ensemble of classifiers by using genetic programming for cyber security applications. In: Applications of Evolutionary Computation—19th European Conference, EvoApplications 2016, 30 March–1 April. Proceedings. Lecture Notes in Computer Science. Springer International Publishing, Porto, Portugal.
Ford, David, 2004. Application of Thermodynamics to the Reduction of Data Generated by a Non-standard System. arxiv:cond-mat/0402325.
Fred Ana LN, Jain Anil K. Combining multiple clusterings using evidence accumulation. IEEE Trans. Pattern Anal. Mach. Intell. 2005;27(6):835–50.
Garcìa-Teodoro P, Dìaz-Verdejo J, Maciá-Fernández G, Vázquez J. Anomaly-based network intrusion detection: techniques, systems and challenges. Comput. Secur. 2009;28(1–2):18–28.
Giacinto Giorgio, Perdisci Roberto, Del Rio Mauro, Roli Fabio. Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Inf. Fusion 2008;9(1):69–82 Special Issue on Applications of Ensemble Methods.
Hoque N, Bhuyan Monowar H, Baishya RC, Bhattacharyya DK, Kalita JK. Network attacks: taxonomy, tools and systems. J. Netw. Comput. Appl. 2014;40(0):307–24.
Horng Shi-Jinn, Su Ming-Yang, Chen Yuan-Hsin, Kao Tzong-Wann, Chen Rong-Jian, Lai Jui-Lin, Dwi Perkasa Citra. A novel intrusion detection system based on hierarchical clustering and support vector machines. Expert Syst. Appl. 2011;38 (1):306–13.
Hu Weiming, Gao Jun, Wang Yanguo, Wu Ou, Maybank S. Online Adaboost-based parameterized methods for dynamic distributed network intrusion detection. IEEE Trans. Cybern. 2014;44(January (1)):66–82.
Huang Shan, Wang Botao, Qiu Junhao, Yao Jitao, Wang Guoren, Yu Ge. Parallel ensemble of online sequential extreme learning machine based on MapReduce. Neurocomputing 2016;174(Part A):352–67.
Hubert Lawrence, Schultz James. Quadratic assignment as a general data analysis strategy. Br. J. Math. Stat. Psychol. 1976;29(2):190–241.
Ingham, Kenneth L., Inoue, Hajime, 2007. Comparing anomaly detection techniques for HTTP. In: Kruegel, Christopher, Lippmann, Richard, Clark, Andrew (Eds.), Recent Advances in Intrusion Detection. Lecture Notes in Computer Science, vol. 4637. Springer, Berlin, Heidelberg, pp. 42–62.
Jamshed, Muhammad Asim., Lee, Jihyung, Moon, Sangwoo, Yun, Insu, Kim, Deokjin, Lee, Sungryoul, Yi, Yung, Park, Kyoung Soo, 2012. Kargus: a highly-scalable software-based intrusion detection system. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS 12. ACM, New York, NY, USA, pp. 317–328.
Jang J-SR. ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern. 1993;23(May (3)):665–85.
Jolliffe IT. Principal Component Analysis.Springer: Springer Series in Statistics; 2002.
Jungsuk Song, Takakura Hiroki, Okabe Yasuo, Yongjin Kwon. Unsupervised anomaly detection based on clustering and multiple one-class SVM. IEICE Trans. Commun. 2009;92(6):1981–90.
Kayacik Günes, H., Nur Zincir-Heywood, A., Heywood Malcolm, I., 2005. Selecting features for intrusion detection: a feature relevance analysis on KDD 99 intrusion detection datasets. In: Proceedings of the Third Annual Conference on Privacy, Security and Trust.
Kemmerer RA, Vigna G. Intrusion detection: a brief history and overview. Computer 2002;35(April (4)):27–30.
Kephart Jeffrey O, Chess David M. The vision of autonomic computing. Computer 2003;36(January (1)):41–50.
Khanna R, Liu Huaping. Control theoretic approach to intrusion detection using a distributed hidden Markov model. IEEE Wireless Commun. 2008;15(August (4)):24–33.
Khanna, Rahul, Liu, Huaping, 2006. System approach to intrusion detection using Hidden Markov Model. In: Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing, IWCMC '06. ACM, New York, NY, USA, pp. 349–354.
Kullback S, Leibler RA. On information and sufficiency. Ann. Math. Stat. 1951;22 (1):79–86 03.
Kumar PArun Raj, Selvakumar S. Distributed denial of service attack detection using an ensemble of neural classifier. Comput. Commun. 2011;34(11):1328–41.
Kumar PArun Raj, Selvakumar S. Detection of distributed denial of service attacks using an ensemble of adaptive and hybrid neuro-fuzzy systems. Comput. Commun. 2013;36(3):303–19.
Kuncheva Ludmila. Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons; 2004.
Lai, Haiguang, Cai, Shengwen, Huang, Hao, Xie, Junyuan, Li, Hui, 2004. A parallel intrusion detection system for high-speed networks. In: Jakobsson, Markus, Yung, Moti, Zhou, Jianying (Eds.), Applied Cryptography and Network Security, Lecture Notes in Computer Science, vol. 3089. Springer, Berlin, Heidelberg, pp. 439–451.
Leopold Edda, Kindermann Jörg. Text categorization with support vector machines. How to represent texts in input space. Mach. Learn. 2002;46(1–3):423–44.
Leung, Kingsly, Leckie, Christopher, 2005. Unsupervised anomaly detection in network intrusion detection using clusters. In: Proceedings of the Twenty-eighth Australasian Conference on Computer Science, ACSC '05, vol. 38. Australian Computer Society, Inc., Darlinghurst, Australia, pp. 333–342.
Levey A, Lindenbaum M. Sequential Karhunen–Loeve basis extraction and its application to images. IEEE Trans. Image Process. 2000;9(August (8)):1371–4.
Liang Nan-Ying, Huang Guang-Bin, Saratchandran Paramasivan, Sundararajan Narasimhan. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. 2006;17(6):1411–23.
Littlestone Nick, Warmuth Manfred K. The weighted majority algorithm. Inf. Comput. 1994;108(February (2)):212–61.
Locasto, E. Michael, Parekh, Janak J., Keromytis, Angelos D., Stolfo, Salvatore J., 2005. Towards collaborative security and p2p intrusion detection. In : Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC. IEEE, New York, USA, pp. 333–339.
McEachen, John C., Wai Kah, Cheng, 2007. An analysis of distributed sensor data aggregation for network intrusion detection. Microprocess. Microsyst. 31(4), 263–272. Special Issue with selected papers from the 11th IEEE Symposium on Computers and Communications (ISCC'06).
Mukkamala Srinivas, Sung Andrew H. Identifying significant features for network forensic analysis using artificial intelligent techniques. Int. J. Digit. Evid. 2003;1 (4):1–17.
Nagesh, H.S., Goil, S., Choudhary, A., 2000. A scalable parallel subspace clustering algorithm for massive data sets. In: 2000 International Conference on Parallel Processing, 2000. Proceedings. IEEE, Toronto, Canada, pp. 477–484.
Newsome, J., Karp, B., Song, D., 2005. Polygraph: automatically generating signatures for polymorphic worms. In: 2005 IEEE Symposium on Security and Privacy, May, pp. 226–241.
Nguyen, Hoa Huu, Harbi, Nouria, Darmont, Jérôme, 2011. An efficient local region and clustering-based ensemble system for intrusion detection. In: Proceedings of the 15th Symposium on International Database Engineering & Applications, IDEAS ' 11, ACM, New York, NY, USA, pp. 185–191.
Patel Kanubhai, Buddhadev Bharat V. Machine learning based research for network intrusion detection: a state-of-the-art. Int. J. Inf. Netw. Secur. 2014;3(3).
Perdisci Roberto, Ariu Davide, Fogla Prahlad, Giacinto Giorgio, Lee Wenke. McPAD: a multiple classifier system for accurate payload-based anomaly detection. Comput. Netw. 2009;53(6):864–81.
Perdisci Roberto, Ariu Davide, Giacinto Davide. Scalable fine-grained behavioral clustering of http-based malware. Comput. Netw. 2013;57(2):487–500.
Perdisci, Roberto, Lee, Wenke, Feamster, Nick, 2010. Behavioral clustering of http-based malware and signature generation using malicious network traces. In: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10, USENIX Association, Berkeley, CA, USA, pp. 26–26.
Raut Abhinav S, Singh Kavita R. Anomaly based intrusion detection—a review. Int. J. Netw. Secur. 2014;5.
Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE International Conference on Neural Networks, vol. 1, pp. 586–591.
Rousseeuw Peter J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 1987;20(0):53–65.
Schaelicke, Lambert, Wheeler, Kyle, Freeland, Curt, 2005. SPANIDS: a scalable network intrusion detection loadbalancer. In: Proceedings of the Second Conference on Computing Frontiers, CF'05. ACM, New York, NY, USA, 2005, pp. 315–322.
Schapire RE. The strength of weak learnability. Mach. Learn. 1990;5(2):197–227.
Schapire RE. Boosting a weak learning by majority. Inf. Comput. 1995;121(2):256–85.
Schölkopf Bernhard, Platt John C, Shawe-Taylor John C, Smola Alex J, Williamson Robert C. Estimating the support of a high-dimensional distribution. Neural Comput. 2001;13(July (7)):1443–71.
Scott C, Nowak R. A Neyman–Pearson approach to statistical learning. IEEE Trans. Inf. Theory 2005;51(November (11)):3806–19.
Sivatha Sindhu Siva S, Geetha S, Kannan A. Decision tree based light weight intrusion detection using a wrapper approach. Expert Syst. Appl. 2012;39(1): 129–141.
Singh Raman, Kumar Harish, Singla RK. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. Expert Syst. Appl. 2015;42(22):8609–24.
Song Jungsuk, Takakura Hiroki, Okabe Yasuo, Nakao Koji. Toward a more practical unsupervised anomaly detection system. Inf. Sci. 2013;231(0):4–14 Data Mining for Information Security.
Song, Yingbo, Keromytis, Angelos D., Stolfo, Salvatore, 2009. Spectrogram: a mixture-of-Markov-chains model for anomaly detection in web traffic. In: Network and Distributed System Security Symposium 2009. Proceedings, February 8–11, Internet Society, San Diego, CA, pp. 121–135.
Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A, Stiller B. An overview of IP flow-based intrusion detection. Commun. Surv. Tutor. IEEE 2010;12(3):343–56 Third.
Vasiliadis, Giorgos, Polychronakis, Michalis, 2011. Sotiris Ioannidis. MIDe A: a multi-parallel intrusion detection architecture. In: Proceedings of the 18th ACM

Conference on Computer and Communications Security, CCS '11, ACM, New York, NY, USA, pp. 297–308.
Vasilomanolakis Emmanouil, Karuppayah Shankar, Mühlhäuser Max, Fischer Mathias. Taxonomy and survey of collaborative intrusion detection. ACM Comput. Surv. 2015;47(May (4)):55:1–33.
Vokorokos, L., Ennert, M., Cajkovsky, M., Turinska, A., 2013. A distributed network intrusion detection system architecture based on computer stations using GPGPU. In: 2013 IEEE 17th International Conference on Intelligent Engineering Systems (INES), June, pp. 323–326.
Vokorokos Liberios, Ennert Michal, Čajkovský Marek, Raduővský Ján. A survey of parallel intrusion detection on graphical processors. Central Eur. J. Comput. Sci. 2014;4(4):222–30.
Wang, Ke, Cretu, Gabriela, Stolfo Salvatore J., 2006. Anomalous payload-based worm detection and signature generation. In: Valdes Alfonso, Zamboni Diego (Eds.), Recent Advances in Intrusion Detection. Lecture Notes in Computer Science, vol. 3858. Springer, Berlin, Heidelberg, 2006. pp. 227–246.
Wang, Ke, Parekh, Janak J, Stolfo, Salvatore J., 2006. Anagram: a content anomaly detector resistant to mimicry attack. In: Recent Advances in Intrusion Detection. Springer, Hamburg, Germany, pp. 226–248.
Wang, Ke, Stolfo, Salvatore J., 2004. Anomalous payload-based network intrusion detection. In: Erland, Jonsson, Alfonso, Valdes, Magnus, Almgren, (Eds.), Recent Advances in Intrusion Detection. Lecture Notes in Computer Science, vol. 3224. Springer, Berlin, Heidelberg, pp. 203–222.
Wang Wei, Guyet Thomas, Quiniou René, Cordier Marie-Odile. Autonomic intrusion detection: adaptively detecting anomalies over unlabeled audit data streams in computer networks. Knowledge-Based Syst. 2014;70(0):103–17.
Witten, I.H., Frank, E., Hall, M.A., 1991. Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science.
Xie Xuanli Lisa, Beni G. A validity measure for fuzzy clustering. IEEE Trans. Pattern Anal. Mach. Intell. 1991;13(August (8)):841–7.
Zhang, Jiong, Zulkernine, M., 2006. Anomaly based network intrusion detection with unsupervised outlier detection. In: ICC'06. IEEE International Conference on Communications, vol. 5, June, pp. 2388–2393.
Zhang Jiong, Zulkernine M, Haque A. Random-forests-based network intrusion detection systems. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. 2008;38 (September (5)):649–59.
Zhang, Tian, Ramakrishnan, Raghu, Livny, Miron, 1996. Birch: an efficient data clustering method for very large databases. In: ACM SIGMOD Record, vol. 25. ACM, pp. 103–114.
Zheng Kai, Cai Zhiping, Zhang Xin, Wang Zhijun, Yang Baohua. Algorithms to speedup pattern matching for network intrusion detection systems. Computer Communications, Elsevier Science Publishers 2015;62:47–58.
Zhou Chenfeng Vincent, Leckie Christopher, Karunasekera Shanika. Decentralized multi-dimensional alert correlation for collaborative intrusion detection. J. Netw. Comput. Appl. 2009;32(5):1106–23.
Zhou Chenfeng Vincent, Leckie Christopher, Karunasekera Shanika. A survey of coordinated attacks and collaborative intrusion detection. Comput. Secur. 2010;29(1):124–40.