Support vector machine for network intrusion and cyber-attack detection

Support Vector Machine for Network Intrusion and Cyber-Attack Detection

Kinan Ghanem[*], Francisco J. Aparicio-Navarro[†], Konstantinos G. Kyriakopoulos[*§], Sangarapillai Lambotharan[*], Jonathon A. Chambers[†]

[*]School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough, LE11 3TU, UK
[†]School of Electrical and Electronic Engineering, Newcastle University, Newcastle upon Tyne, NE1 7RU, UK
[§]Institute for Digital Technologies, Loughborough University London, London, E15 2GZ, UK
e-mails: {k.ghanem, k.kyriakopoulos, s.lambotharan}@lboro.ac.uk, {francisco.aparicio-navarro, jonathon.chambers}@ncl.ac.uk

***Abstract*—Cyber-security threats are a growing concern in networked environments. The development of Intrusion Detection Systems (IDSs) is fundamental in order to provide extra level of security. We have developed an unsupervised anomaly-based IDS that uses statistical techniques to conduct the detection process. Despite providing many advantages, anomaly-based IDSs tend to generate a high number of false alarms. Machine Learning (ML) techniques have gained wide interest in tasks of intrusion detection. In this work, Support Vector Machine (SVM) is deemed as an ML technique that could complement the performance of our IDS, providing a second line of detection to reduce the number of false alarms, or as an alternative detection technique. We assess the performance of our IDS against one-class and two-class SVMs, using linear and non-linear forms. The results that we present show that linear two-class SVM generates highly accurate results, and the accuracy of the linear one-class SVM is very comparable, and it does not need training datasets associated with malicious data. Similarly, the results evidence that our IDS could benefit from the use of ML techniques to increase its accuracy when analysing datasets comprising of non-homogeneous features.**

Keywords—Classification Algorithms; Cyber Security; Intrusion Detection Systems; Machine Learning Techniques; Network Security; Support Vector Machine; SVM

I. INTRODUCTION

Cyber-security threats are a growing concern in networked environments. Therefore, providing strong and reliable security mechanisms has become essential in all areas of society. An Intrusion Detection System (IDS) is an effective tool to identify the presence of attacks and intrusions in the protected system.

In [1], we previously presented an unsupervised anomaly-based IDS, based on the combined use of multiple metrics from multiple layers of the protocol stack to carry out the intrusion detection. Unsupervised IDSs are able to learn the difference between normal and malicious information autonomously. Anomaly-based IDSs construct profiles of normal network traffic, and calculate the level of deviation of outliners to identify attacks. In contrast to misuse IDSs, an anomaly-based system is able to identify previously unknown and zero-day attacks. However, this type of IDS tends to generate higher number of false alarms than misuse IDSs [2].

Machine Learning (ML) techniques have gained wide interest in tasks of intrusion detection. ML-IDSs are based on the definition of models that allow the classification of the analysed information [2]. One of the most attractive ML techniques is the Support Vector Machine (SVM) [3]. An SVM is a classification technique that has proven to be effective in a wide variety of problems, such as image processing [4], often providing considerable improvement over competing methods.

In the area of cyber-security, the use of an SVM can improve the accuracy of IDSs. The classifier that is created by this technique is useful to predict between two possible outcomes (i.e. malicious and non-malicious network traffic). The study of ML techniques in tasks of intrusion detection would allow us to identify a classification technique that could complement our anomaly-based IDS, acting in a hybrid manner as a second line of detection, and, at the same time, to facilitate the creation of a benchmark to compare the performance of our IDS against.

In this work, we assess the performance of our IDS against one-class and two-class SVMs. Although, a two-class SVM may be generally more accurate, if we were able to generate a robust one-class SVM, we would reduce the need for a thorough off-line dataset labelling process. A one-class SVM requires only a training dataset containing normal traffic. Therefore, the aim of this paper is twofold. First, to evaluate which of the SVM techniques produces the best detection results. The assessment is conducted between a one-class SVM and two-class SVM, using linear and non-linear with Radial Basis Function (RBF) forms. Second, to assess the performance of our unsupervised anomaly-based IDS [1] against the different SVM techniques in tasks of intrusion detection.

The remainder of the paper is organised as follows. In Section II, the most relevant previous work is reviewed. In Section III, the networks and evaluation datasets are described. A description of the SVM theory and the anomaly-based IDS is given in Section IV. Section V presents the experimental results. Finally, conclusions are given in Section VI.

II. RELATED WORK

In the area of cyber-security and IDSs, ML classification techniques are adopted to improve the efficiency of the detection systems distinguishing between malicious and non-malicious network traffic. In [5], the authors propose an IDS that uses a one-class SVM to analyse incoming Netflows. In contrast to common procedure, the authors train their system solely with malicious data collected using a honeypot. This one-class SVM-based IDS produces very high detection results.

The author of [6] presents an approach that combines a linear SVM, decision trees and Naïve Bayes to reduce the number of false alarms of the IDS, while analysing the KDD99 dataset [7].

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) Grant number EP/K014307/2 and the MOD University Defence Research Collaboration in Signal Processing.

The approach proposed in this work conducts an analysis, in which each of the techniques processes the data in sequential order. The SVM is trained based upon a new binary classification added to the dataset to specify if the instance is an attack or normal traffic.

In [8], the authors present an IDS that makes use of an SVM as a classification technique. The presented IDS consists of three main components, a one-class SVM that distinguishes between malicious and non-malicious network traffic during an initial analysis, a multi-class SVM that categorises the traffic classified as malicious into one of the four classes (i.e. Denial-of-Service, Remote to local, User to root and Probing attacks), and a final clustering process. The experiments presented in this work are also conducted using the KDD99. Instead of using the KDD99, in this paper, we have evaluated the classifiers using a number of different network traffic datasets, gathered from real networks, both wired and wireless, at Loughborough University.

The training process of an SVM has also been carefully researched. The authors of [9] propose a new approach for enhancing the training process of SVM when dealing with large training datasets. This work combines the use of SVM and clustering analysis to reduce the number of instances used during the computation of the support vector margin, which, in turn, reduces the training time without affecting the final results.

III. Testbeds and Network Traffic Measurements

In total, five datasets[1] have been gathered from an IEEE 802.11 network testbed, deployed in our laboratory. Similarly, another dataset[2] has been gathered from an Ethernet Local Area Network (LAN) office, at Loughborough University. All the network traffic has been gathered in pcap format using tcpdump.

A. IEEE 802.11 Network Testbed

A schematic representation of the WiFi network is shown in Fig. 1.a, comprising of an Access Point (AP), a wireless client accessing various websites on the Internet, a monitoring node and an attacker. The attacker implements two type of attacks; deauthentication attack and injection attack. Four of the datasets gathered from this network comprise both malicious and normal frames, while another dataset contains only normal frames.

On the one hand, we made use of the tool Airpwn [10] to implement different modes of the injection attack. This software can be found as part of the suite of penetration testing tools Aircrack [11]. Airpwn eavesdrops the transmitted frames in a WiFi network. If Airpwn identifies an HTTP request from a legitimate wireless node, it injects its own crafted HTML code using the spoofed MAC address of the AP. We have used two modes of the Airpwn attack. In the dataset *Attack01*, the attacker replaces the HTTP headers fields of the requested website. In the dataset *Attack02*, the attacker replaces the images in the website. Lastly, dataset *Attack03* comprises the two modes of the attack.

On the other hand, the deauthentication attack has also been investigated. The attacker injects spoofed deauthentication frames with the purpose of forcing the client to re-establish a connection with the AP. The suite of tools used to implement this attack, generating the *DeAuth* dataset, is also Aircrack. All these datasets[1] have been described in more detail in [12].

Among all the available metrics, five metrics have been experimentally selected as the most appropriate for detecting the implemented attacks. These are the Received Signal Strength Indication (RSSI), Injection Rate (RATE), Network Allocation Vector (NAV) value, Sequence Number (SEQ), and Inter-arrival or Delta time (ΔTime) between two consecutive frames. We have implemented the detection using metrics extracted only from the PHY layer and the MAC layer, which remain unencrypted in the network frames header, even when utilising WiFi encryption techniques.

B. Ethernet Local Area Network

The Ethernet LAN used for the implementation of a Port Scanning attack comprises a number of PCs in two distinct offices used to generate real background traffic, an attacker using the network mapping tool nmap [13], and a victim. A schematic representation of this LAN is shown in Fig. 1.b.

Port scanning, also known as probing, is a technique used to discover possible vulnerabilities in the network through the probing of open ports. This attack often precedes the execution of multi-stage attacks [14]. The *Probing* dataset[2] has been explained in more detail in [15].

In total, four metrics have been computed and aggregated per second to carry out the intrusion detection. These metrics are Communication Rate (COM), number of frames transmitted; Throughput (THR), number of transmitted bytes; Source Port Distribution (SPD), number of source ports; and Destination Port Distribution (DPD), number of destination ports. In contrast to the WiFi datasets, this dataset comprises metrics with non-homogeneous patterns that make the analysis more challenging.

Table I presents an overview of how the information in the datasets is distributed, as well as the proportion of malicious traffic in each dataset. By using all these datasets, comprising of different attacks under different networks, we want to provide variability to the experiments.

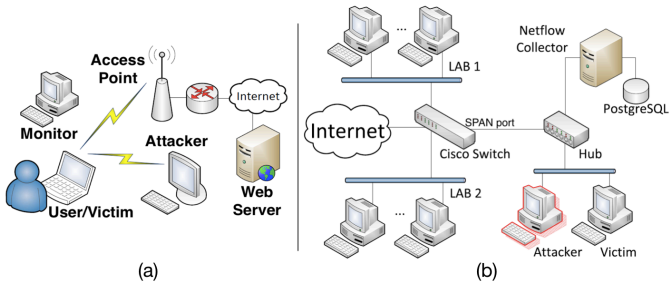

Fig. 1. Networks schematic; a) WiFi network used to implement the Airpwn and Deauthentication attacks; b) LAN used to implement the probing attack.

TABLE I. General Description of the Evaluated Datasets

Dataset	Total Instances	Normal Instances	Normal Instances (%)	Malicious Instances	Malicious Instances (%)
Normal	3631	3631	100	n/a	n/a
Attack01	1361	1350	99.2	11	0.8
Attack02	14493	13498	93.1	995	6.9
Attack03	12130	12016	99.1	114	0.9
DeAuth	228	164	71.93	64	28.07
Probing	700484	696638	99.4	4220	0.6

IV. Proposed Classification Methodologies

A. Support Vector Machine

The goal of an SVM is to find the optimal separating hyperplane which maximises the margin of the training data and minimises complexity and risk of overfitting. An SVM is easy

1. WiFi datasets available: https://figshare.com/s/9c116e0422eb5ddbe9ba
2. *Probing* dataset available: https://figshare.com/s/4bd0fe2dab7e09ce61dc

to implement, requires a small training dataset and is appropriate for extremely large dataset analysis [16]. An SVM also requires very limited time to perform the classification process, once the optimal classification hyperplane has been constructed. A complete tutorial on SVMs is presented in [3].

The SVM-based classifier that we present in this work has been developed based on Matlab and LibSVM [17]. The LibSVM is an integrated reliable software for support vector classification and distribution estimation, which includes several kernel functions. We have modified LibSVM to randomise the selection of samples during our experiments.

1) Linear Support Vector Machine

Let us consider a binary classification of d-dimensional feature space of linearly separable training samples $S = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_m, \boldsymbol{y}_m)\}$, where m is the number of samples in the training dataset, the input features $\boldsymbol{x}_i \in R^d$ are usually d-dimensional vectors describing the properties of the input samples, and the labels $y_i \in \{+1, -1\}$ are the binary output of the classification problem. An optimal discriminating function can be defined as:

$$\mathrm{f(x)} = \mathrm{sign}(\langle \boldsymbol{\omega} \cdot \boldsymbol{x} \rangle + b) = \begin{cases} +1 \text{ if } x \text{ belongs to class } A \\ -1 \text{ if } x \text{ belongs to class } B \end{cases} \quad (1)$$

where the vector $\boldsymbol{\omega}$ determines the orientation of a discriminant plane or the normal of the hyperplane, and the scalar b is the offset of the hyperplane from the origin in the vector space.

The target of the SVM is to find the optimal hyperplane via maximising the margin between classes. This can be obtained by solving the following quadratic optimisation problem:

$$\text{minimise } \langle \boldsymbol{\omega} \cdot \boldsymbol{\omega} \rangle$$
$$\text{subject to } y_i(\langle \boldsymbol{\omega} \cdot \boldsymbol{x_i} \rangle + b) - 1 \geq 0 \; i = 1, 2, \ldots, m \quad (2)$$

The dual of the optimisation problem in (2) can be written in terms of the Lagrangian multipliers a_i as follows [18]:

$$L = \sum_{i=1}^{m} a_i - \frac{1}{2} \sum_{j=1}^{m} \sum_{i=1}^{m} y_i y_j a_i a_j \langle \boldsymbol{x}_i . \boldsymbol{x}_j \rangle \quad (3)$$

The optimal Lagrangian multipliers are obtained as the maximiser of (3), $\boldsymbol{\omega}$ is determined as $\sum_{i=1}^{m} y_i a_i \boldsymbol{x}_i$ and b is obtained from Karush-Kuhn-Tucker (KKT) conditions.

2) Non-Linear Support Vector Machine

A linear SVM assumes that the different classes in the dataset are clearly distinguishable. In data with no possibility for linear separation, the use of a non-linear function may help to make the datasets separable. An SVM which uses kernel functions offers an efficient alternative solution to change the non-linear approach into a linear one by projecting the data into a highly dimensional feature space to allow the separation [3, 4].

A transformation function $\Phi: R^d \rightarrow H$ can be used to map the input into an Euclidean space H. This allows the value of the inner-product in space H to be computed without the non-linear mapping. This reduces the complexity of the computational problem, without any effect on Lagrangian optimisation theory.

$$L = \sum_{i=1}^{m} a_i - \frac{1}{2} \sum_{j=1}^{m} \sum_{i=1}^{m} y_i y_j a_i a_j K \langle \boldsymbol{x}_i . \boldsymbol{x}_j \rangle \quad (4)$$

The RBF is defined by (5), where the free parameter s gives the width of the kernel.

$$K\left(\boldsymbol{x}_i . \boldsymbol{x}_j\right) = exp\left(-\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 / s^2\right) \quad (5)$$

3) One-Class Support Vector Machine

The one-class SVM is a semi-supervised technique that constructs the classification model of normal behaviour during the training process using only one type of samples (i.e. training datasets comprising non-malicious data). The implementation of a one-class SVM is to find a hyperplane that maximises the distance of the data from the origin [19].

A one-class SVM uses an implicit transformation defined by the kernel function $\Phi(\mathrm{i})$ to project the data into a higher dimensional space. This approach separates the majority of the data from the origin, allowing only a few points to exist on the other side. The primary object of a one-class SVM is to achieve:

$$\min_{\omega, \varepsilon, \gamma} \frac{\|\boldsymbol{\omega}\|^2}{2} - \gamma + \frac{1}{v\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \varepsilon_{\mathrm{i}}$$
$$\text{subject to } \boldsymbol{\omega}^{\mathrm{T}} \Phi(x_{\mathrm{i}}) \geq \gamma - \varepsilon_{\mathrm{i}}, \varepsilon_{\mathrm{i}} \geq 0 \quad (6)$$

where $g(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{\omega}^{\mathrm{T}} \Phi(\boldsymbol{x}) - \gamma)$ assesses whether a sample point is inside or outside the estimated set, γ is the bias term, ε_{i} is the loss variable of point i that allows it to lie on the other side of the decision boundary, n is the size of the training dataset and v is the regularisation parameter. Varying v controls the tradeoff between ε_{i} and γ. Further details about different kernel functions in one-class SVMs can be found in [20].

In this paper, the one-class SVM requires to fine-tune three main parameters. These are $v, \{v \in \mathrm{R} : 0 < v < 1\}$, which has been empirically set to 0.01, $\gamma, \{\gamma \in \mathrm{R}: 0 \leq \gamma\}$, which has been set to 0.2, and lastly, ε_i which has been set to 0.1. It is worth mentioning that a trade-off between the three parameters plays a significant role in the performance of the SVM, especially in terms of false alarms, as will be shown in Section V.

B. Support Vector Machine Training and Classification

The comparison has been performed using both linear and non-linear forms of SVM with the RBF. Generally, the SVM classification process consists of the initial training phase and the classification phase. The training involves the construction of an accurate model based on the already labelled datasets.

We have used the different training datasets for the two-class SVM and one-class SVM. The one-class SVM has been trained using 100% of the dataset *Normal*. This dataset comprises non-malicious network traffic only. Then, the classification has been conducted using 100% of the remaining datasets. In the case of *Probing*, the training has been conducted using 35% of the dataset, previously labelled. We found that this training datasets comprised of normal traffic only. Then, the classification process is conducted using the remaining 65% of the datasets.

For the two-class SVM, the SVM has been trained using 35% of each dataset. The selection of the training dataset has been randomised to increase the probability of including a representative distribution of malicious and non-malicious network traffic. One model is constructed for each dataset after the training process. Then, the classification is conducted using the remaining 65% of the dataset. The main difference between the classification using two-class SVM and one-class SVM is that the two-class SVM requires the training datasets to be previously labelled.

C. Anomaly-Based Intrusion Detection System

Our unsupervised anomaly-based IDS [1], is based on the combined use of multiple metrics from multiple layers of the

protocol stack to carry out the intrusion detection. It uses the Dempster-Shafer (D-S) Theory of Evidence as a data fusion technique to merge evidence of information extracted from each of the metrics. The main purpose is to generate an overall belief on whether there is an attack present in the network or not.

D-S requires the assignment of belief values, also known as Basic Probability Assignment (BPA), which expresses the evidence attributed to the considered hypotheses. In [12], we proposed a novel statistical framework of assigning BPA values, which adapt the assignment of its evidence based on the current characteristics of the network traffic, without prior training. The proposed framework exploits a Sliding Window (SW) scheme to compute statistical parameters from the data, which are required to generate the different BPA values.

Three independent statistical approaches provide the belief values on the different hypotheses. The approach that assigns BPA values to the hypothesis *Normal* uses the distribution of the network traffic within the SW. The approach that assigns BPA values to the hypothesis *Attack* uses the Euclidean distance from a defined reference of normality (i.e. mean of information within the SW). Meanwhile, the BPA in the hypothesis *Uncertainty* is assigned based on the belief values assigned to *Normal* and *Attack* in the current SW. The different statistical techniques used by the IDS are described in more detail in [12].

V. Results and Analysis

This section evaluates the performance of the two-class SVM and one-class SVM against the performance of our IDS. The detection analysis is based on the performance metrics Detection Rate (DR), False Positive Rate (FPr), False Negative Rate (FNr), and Overall Success Rate (OSR). The experimental results are presented in Tables II-VI.

The detection results obtained using the linear two-class SVM technique are presented in Table II. Both the DR and OSR reach 100% for all the datasets, which indicates that the detection is completely accurate. Only in the case of *Probing*, the DR and OSR drop to 98.8% and 83.4%, respectively. For this dataset, the increase of FPr to 16.6% is also noticeable. Despite the increase of FPr for *Probing*, the results generated by the linear two-class SVM are very accurate. However, it is worth noting that this is a supervised technique, and requires training datasets, previously labelled, comprising both classes of data.

For the non-linear two-class SVM, the detection results obtained using this technique are slightly worse than the results generated by linear two-class SVM. However, the accuracy of non-linear two-class SVM can be considered acceptable. Only in *Probing*, does the OSR reach 81.67%. For the rest of datasets, the OSR reaches over 99.2%. The DR for most datasets reaches at least 93.51%, and only the dataset *Probing* generates false positives, reaching a FPr of 18.32%. The most noticeable result is the decrease in the DR for the dataset *Attack01*. After analysing the results, we identified that this decrease in the DR is caused by the metrics SEQ and ΔTime. The malicious and non-malicious instances for these metrics overlap and the non-linear SVM is unable to differentiate accurately between them. For our selected features, the two classes are linearly separable. Hence, the linear SVM is expected to outperform the non-linear SVM. All these results are presented in Table III.

Once again, the results obtained by the linear one-class SVM technique, shown in Table IV, are slightly worse than the results generated by the linear two-class SVM. However, the accuracy of this technique is very comparable to the two-class SVM. Additionally, this SVM only needs training datasets with normal data. Only for the dataset *Probing* is the OSR lower than 98%, reaching 88.91%. In the case of DR, all datasets generate results higher than 89%, with two cases reaching 100%. On the other hand, for the dataset *Probing*, the FPr improves from 16.6% to 11.05%, in comparison to the linear two-class SVM. This is due to the value of the parameters v, γ and ε_i. However, these values also increase the FNr as compared to the two-class SVM.

The results generated by the non-linear one-class SVM, shown in Table V, are the worst results overall. We can see a general decrease in the OSR and a noticeable decrease in the DR for the dataset *Probing*, reaching 61.37%. For all the datasets, this SVM technique also generates an increase in the FPr.

TABLE II. Detection results: Linear Two-Class SVM

Linear Two-Class SVM				
Dataset	**DR (%)**	**FPr (%)**	**FNr (%)**	**OSR (%)**
Attack01	100	0	0	100
Attack02	100	0	0	100
Attack03	100	0	0	100
DeAuth	100	0	0	100
Probing	98.78	16.6	1.22	83.4

TABLE III. Detection results: Non-Linear Two-Class SVM

Non-Linear Two-Class SVM with Gaussian Radial Basis				
Dataset	**DR (%)**	**FPr (%)**	**FNr (%)**	**OSR (%)**
Attack01	62.5	0	37.5	99.66
Attack02	99.52	0	0.48	99.97
Attack03	93.51	0	6.49	99.94
DeAuth	97.78	0	2.22	99.25
Probing	97.85	18.32	2.15	81.67

TABLE IV. Detection results: Linear One-Class SVM

Linear One-Class SVM				
Dataset	**DR (%)**	**FPr (%)**	**FNr (%)**	**OSR (%)**
Attack01	100	0	0	99.93
Attack02	89.25	0.41	10.75	98.85
Attack03	99.12	2.46	0.88	97.53
DeAuth	100	1.75	0	98.25
Probing	93.78	11.05	6.22	88.91

TABLE V. Detection results: Non-Linear One-Class SVM

Non-Linear One-Class SVM with Gaussian Radial Basis				
Dataset	**DR (%)**	**FPr (%)**	**FNr (%)**	**OSR (%)**
Attack01	100	14.62	0	85.38
Attack02	100	6.84	0	93.16
Attack03	100	5.82	0	94.18
DeAuth	95.38	3.07	4.62	95.61
Probing	61.37	1.15	38.63	98.61

Overall, the detection results generated by the linear two-class SVM technique are the most accurate results. Nonetheless, the accuracy of the linear one-class SVM performs comparably well without the need for labelled training datasets. Therefore, we compare the performance of our unsupervised anomaly-based IDS against these two linear SVM techniques.

From the detection results presented in Table VI, we can see that our anomaly-based IDS detects all the malicious traffic in the WiFi datasets. There are several false positive alarms that only reach 2.19% of FPr. Nonetheless, the accuracy of our IDS

is comparable to the detection results generated by the linear two-class SVM and linear one-class SVM techniques. Additionally, it is important to emphasise that these results are completely unsupervised, generated without any additional information about the nature of the network traffic dataset.

If we focus on the dataset *Probing*, there is a dramatic decrease in DR and OSR, and an increase in the false alarms. One factor that may be directly correlated to these results is the size of this dataset. Additionally, these results align with the reports indicating that the efficiency of IDSs that use statistical detection techniques decreases when non-homogeneous data are analysed [21]. In this case, where a non-homogeneous dataset is analysed, our anomaly-based IDS could benefit from the use of ML techniques to increase its detection accuracy.

TABLE VI. DETECTION RESULTS: UNSUPERVISED ANOMALY-BASED IDS

Unsupervised Anomaly-based IDS				
Dataset	**DR (%)**	**FPr (%)**	**FNr (%)**	**OSR (%)**
Attack01	100	0	0	100
Attack02	100	0.03	0	99.97
Attack03	100	0.06	0	99.94
DeAuth	100	2.19	0	97.81
Probing	18.82	15.99	81.18	83.52

VI. CONCLUSIONS

In this paper, we considered the SVM as a ML technique that could complement the performance of our IDS, or as an alternative detection technique. We have assessed the performance of our unsupervised anomaly-based IDS against one-class and two-class SVMs, using linear and non-linear with RBF forms. In order to provide variability to the experiments, the analysis has been implemented with a number of network traffic datasets, gathered from real networks, comprising different types of attacks. First, we have assessed which of the SVMs produces the best detection results. This assessment analysis gives insight into the detection performance of the SVM techniques. Then we have evaluated the performance of our IDS against SVM techniques in tasks of intrusion detection.

The results that we present show that the linear two-class SVM generates the most accurate results overall. This technique reaches 100% of DR and OSR for almost all the datasets. However, this SVM technique requires training data, previously labelled, comprising both classes of data. On the other hand, the accuracy of the linear one-class SVM performs comparably to the accuracy of the linear two-class SVM without the need for training datasets associated with malicious data. Only in the case of *Probing*, the OSR reaches 81.67%. For the rest of the datasets, the OSR reaches 99.25%. The DR for most datasets reaches at least 93.51%, and only *Probing* generates false positive alarms.

Our IDS detects all the malicious traffic in the WiFi datasets. However, the accuracy of the IDS when analysing the dataset *Probing* decreases drastically. For this dataset, the DR and FPr reach 18.82% and 15.99%, respectively. This may be due to the size and the non-homogeneous nature of the dataset. Additionally, it is important to emphasise that these results are completely unsupervised, generated without any additional information about the nature of the network traffic dataset.

Therefore, these results suggest that our anomaly-based IDS could benefit from the use of ML techniques to increase its detection accuracy. The use of linear SVM, both two-class and one-class with RBF forms, could potentially complement the performance of our IDS especially when non-homogeneous data are analysed.

REFERENCES

[1] K. G. Kyriakopoulos, F. J. Aparicio-Navarro, and D. J. Parish, "Manual and automatic assigned thresholds in multi-layer data fusion intrusion detection system for 802.11 attacks," in *IET Information Security*, vol. 8, no. 1, 2014, pp. 42-50.

[2] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," in *Computers & Security*, vol. 28, no. 1, pp. 18-28, 2009.

[3] C. J. Burges, "A tutorial on support vector machines for pattern recognition," in *Data mining and knowledge discovery*, vol. 2, no. 2, 1998, pp. 121-167.

[4] T. S. Hai, and N. T. Thuy, "Image classification using support vector machine and artificial neural network," in *Int. Journal of Information Technology and Computer Science (IJITCS)*, vol.4, no.5, 2012, pp. 32-38.

[5] P. Winter, E. Hermann, and M. Zeilinger, "Inductive intrusion detection in flow-based network data using one-class support vector machines," in *Proc. of the 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 2011, pp. 1-5.

[6] K. Goeschel, "Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive bayes for off-line analysis," in *Proc. of the Annual IEEE SoutheastCon conference*, 2016, pp. 1-6.

[7] University of California, Irvine (UCI) "KDD Cup 1999 Data", 1999. Available: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (Access date: 5 May, 2017).

[8] H. Lee, J. Song, and D. Park, "Intrusion detection system based on multi-class SVM," in *Proc. of the International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, 2005, pp. 511-519.

[9] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," in *Int. Journal on Very Large Data Bases*, vol. 16, no. 4, 2007, pp. 507-521.

[10] Airpwn Packet Injection Framework Website Available: http://airpwn.sourceforge.net/Airpwn.html (Access Date: 21 Feb, 2017).

[11] C. Devine, and T. Otreppe, "Aircrack", 2010. Available: https://www.aircrack-ng.org/ (Access date: 22 Feb, 2017).

[12] F. J. Aparicio-Navarro, K. G. Kyriakopoulos, and D. J. Parish, "An automatic and self-adaptive multi-layer data fusion system for Wi-Fi attack detection," in *International Journal of Internet Technology and Secured Transactions*, vol. 5, no. 1, 2013, pp. 42-62.

[13] G. Lyon, "Nmap: The network mapper – Free security scanner," Available: http://nmap.org/ (Access Date: 21 Jun, 2016).

[14] E. Bou-Harb, M. Debbabi, and C. Assi, "Cyber scanning: A comprehensive survey," in *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, 2014, pp. 1496-1519.

[15] F. J. Aparicio-Navarro, J. A. Chambers, K. G. Kyriakopoulos, Y. Gong, and D. J. Parish, "Using the pattern-of-life in networks to improve the effectiveness of intrusion detection systems," in *Proc. of the IEEE International Conference on Communications (ICC)*, 2017, pp. 1-7.

[16] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," in *Journal of network and computer applications*, vol. 28, no. 2, 2005, pp. 167-182.

[17] C.-C. Chang, and C.-J. Lin, "LibSVM: A library for support vector machines," 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin /libsvm.

[18] S. Boyd, and L. Vandenberghe, "Convex optimization," Cambridge university press, 2004.

[19] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," in *Neural computation*, vol. 13, no. 7, 2001, pp. 1443-1471.

[20] P. F. Evangelista, M. J. Embrechts, and B. K. Szymanski, "Some properties of the Gaussian kernel for one class learning," in *International Conference on Artificial Neural Networks (ICANN)*, 2007, pp. 269-278.

[21] D. Hongbo, "Data mining techniques and applications: An introduction," Course Technology Cengage Learning, 2010.