2019 11th International Conference on Cyber Conflict:
Silent Battle
T. Minárik, S. Alatalu, S. Biondi,
M. Signoretti, I. Tolga, G. Visky (Eds.)
2019 © NATO CCD COE Publications, Tallinn

Addressing Adversarial Attacks Against Security Systems Based on Machine Learning

Giovanni Apruzzese
Department of Engineering
"Enzo Ferrari"
University of Modena and
Reggio Emilia
Modena, Italy
giovanni.apruzzese@unimore.it

Michele Colajanni
Department of Engineering
"Enzo Ferrari"
University of Modena and
Reggio Emilia
Modena, Italy
michele.colajanni@unimore.it

Luca Ferretti
Department of Engineering
"Enzo Ferrari"
University of Modena and
Reggio Emilia
Modena, Italy
luca.ferretti@unimore.it

Mirco Marchetti
Department of Engineering
"Enzo Ferrari"
University of Modena and
Reggio Emilia
Modena, Italy
mirco.marchetti@unimore.it

Abstract: Machine-learning solutions are successfully adopted in multiple contexts but the application of these techniques to the cyber security domain is complex and still immature. Among the many open issues that affect security systems based on machine learning, we concentrate on adversarial attacks that aim to affect the detection and prediction capabilities of machine-learning models. We consider realistic types of poisoning and evasion attacks targeting security solutions devoted to malware, spam and network intrusion detection. We explore the possible damages that an attacker can cause to a cyber detector and present some existing and original defensive techniques in the context of intrusion detection systems. This paper contains several performance evaluations that are based on extensive experiments using large traffic datasets. The results highlight that modern adversarial attacks are highly effective against machine-learning classifiers for cyber detection, and that existing solutions require improvements in several directions. The paper paves the way for more robust machine-learning-based techniques that can be integrated into cyber security platforms.

Keywords: *adversarial attacks, machine learning, deep learning, poisoning attacks, evasion attacks, intrusion detection*

1. INTRODUCTION

Solutions based on machine- and deep-learning algorithms are becoming pervasive in multiple fields [1], with documented successes for computer vision, speech processing, social media analysis and healthcare [2]. However, the application of these techniques to cyber security is still affected by several shortcomings that limit their effectiveness in real scenarios. Recent results evidence that utmost care and due diligence should be adopted when considering defensive methods based on machine learning to protect current organizations [3, 4, 5]. There are several motivations for these problems: attacks are relatively infrequent compared to the massive number of events generated by modern enterprises; they evolve rapidly, with consequences for possible ground truth for validation; and attackers are not constrained by rules as in artificial intelligence gaming. In this paper, we consider the additional problem presented by the inherent vulnerability of machine-learning methods to adversarial attacks, through which opponents can thwart the system by inducing the generation of incorrect or undesirable results [6]. This issue is aggravated by the multiple variations of malicious actions that can be performed during the training- or test-time of the machine-learning algorithms [7, 8].

Adversarial attacks against machine learning have been explored in image processing [9], but lack adequate analyses in the cyber security domain. The papers that evaluate the performance of cyber detectors in adversarial settings (e.g., [7, 10]) consider a limited number of cyber security problems, few machine-learning classifiers, and a restricted subset of adversarial attacks. The main focus is on spam and malware analysis [11, 12], while we consider this issue from a network intrusion detection perspective [13], where experimental evaluations and novel solutions are lacking [5]. We provide a comprehensive overview of adversarial attacks against cyber security applications of machine learning and propose a taxonomy of these threats in three areas: network intrusion detection, malware analysis, and spam and phishing detection. We present existing solutions to counter this menace, and propose an original method for mitigating attacks based on data poisoning. We have executed a large set of experiments to evaluate and compare the performance of cyber detectors under normal and adversarial settings. In addition, we have measured the effectiveness of some countermeasures, including the strategy proposed in this paper.

The remainder of this paper is structured as follows. Section 2 provides a thorough description of adversarial attacks in the cyber security sphere. Section 3 explores existing strategies for countering these threats and proposes our original methodology against poisoning attacks. Section 4 presents the experimental results and evaluations. Section 5 concludes the paper with final remarks and future work.

2. ADVERSARIAL ATTACKS AND CYBER SECURITY

To defend against cyber threats, security operators rely on techniques borrowed from the machine-learning domain [14, 15] because of their *anomaly detection* capabilities, which may identify novel attacks and which are not recognizable through *signature-based* approaches [16, 17]. Machine-learning algorithms can be divided into *supervised* and *unsupervised* techniques, depending on the requirement of the training phase, with a set of labelled data [18]. Both groups can solve cyber security problems [3], but supervised methods are appreciated due to their ability to provide actionable results, such as detecting an attack [4]. On the other hand, unsupervised techniques are employed for ancillary tasks such as data clustering [19]. All these methods present several open issues that must be considered when integrating them into security systems [18]. Here, we focus on the topic of adversarial attacks.

Adversarial attacks against machine-learning solutions represent a major limitation to the adoption of a fully autonomous cyber defence platform. These threats are based on the generation of specific samples that induce the model to produce an output that is favourable to the attacker, and leverage the intrinsic sensitivity of machine-learning models to their internal configuration settings [14, 20, 21]. Although adversarial perturbations affect all applications of machine learning, the cyber security field presents several characteristics that further aggravate this menace: there is a constantly evolving arms race between attackers and defenders; the system and network behaviour of an organization can be subject to continuous modifications. These unavoidable and unpredictable changes are denoted as the *concept drift* [22] problem, which decreases the performance of any model based on anomaly detection. Mitigations involve periodic retraining and adjustment processes that can identify behavioural modifications and recent related threats. While performing such operations is a challenging task in itself [18], it also facilitates the execution of adversarial attacks [23].

Many research results (e.g., [6, 24, 8]) show that machine-learning algorithms are unsuitable to face adversarial settings. The first examples of adversarial attacks date back to 2004 [25], but the advent of deep learning drew the attention of the research community to this issue [26]. Possible countermeasures have appeared in the computer vision literature [9], with several papers proposing solutions for improving the robustness of deep neural networks for image classification in adversarial environments [27]. However, the performance of machine-learning algorithms depends on their application contexts, hence it is of paramount importance to understand the effects of adversarial threats against cyber security detectors. We consider different classes of attacks by proposing a taxonomy inspired by the work of Huang et al. [6], where threats are classified on the basis of two properties: the *influence* determines whether an attack is performed at training-time or test-time; the *violation* denotes the type of security violation that may affect availability or integrity of the system.

- **Influence**
    - *Training-time*: these attacks include the manipulation of the training set used by the machine-learning model through the insertion or removal of specific samples that alter the decision boundaries of the algorithm. They are also known as *poisoning attacks*.
    - *Test-time*: These attacks assume that the detector has been deployed and aim to subvert its behaviour through the submission of specific samples during its operational phase.
- **Violation**
    - *Integrity*: often referred to as *evasion attacks*, these attacks aim to increase the false negative rate of the model by introducing malicious samples that are classified as benign. Hence, when successful, these stealthy threats do not cause any defensive action to be taken by the targeted organization.
    - *Availability*: these attacks make the targeted model useless, for example by causing overwhelming spikes of false alarms. For this reason, attacks of this type usually induce some sort of response action by the defending side, such as temporary shut-down and recalibration of the model.

A comprehensive classification of adversarial attacks requires a definition of the attacker model. According to Biggio et al. [24], we should consider the following main features.

- The **goal** is related to the security violation purpose of the adversarial attack.
- The **knowledge** denotes the information possessed by the attacker on the machine-learning system that may include the adopted algorithm, its parameters, and its training data set. Depending on the type of information,

we can distinguish between *black box* attacks (zero knowledge), *grey box* attacks (partial knowledge), and *white box* attacks (complete knowledge).

- The **capability** determines the type of actions that an attacker can perform against the targeted environment that includes, but is not limited to, the machine-learning system. As a strict requirement, it is important to specify which kind of access the attacker has to the cyber detector: he can have full access (that is, reading its output and modifying its internals), limited access (can only read its output) or no access at all.
- The **strategy** denotes the workflow pursued by the attacker to achieve his goal by leveraging previous knowledge and capabilities.

The attacker model distinguishes the adversarial attacks against cyber security systems from offences against other domains of application of machine learning. For example, most papers on image recognition [9, 28] assume that the attacker has complete knowledge and capability. These assumptions are unrealistic in cyber security applications for two reasons: cyber detectors are protected by multiple defence layers; if an attacker overcomes these barriers and can modify the detector, he can achieve his goals without relying on adversarial attack strategies. Thus, in the remainder of this paper, we consider attacks in which the attacker has limited or no access to the machine-learning system.

In Table 1, we classify the most important examples of adversarial attacks against three cyber security areas (Network intrusion detection, Malware analysis, Spam detection) representing scenarios where machine-learning methods are achieving appreciable results (e.g., [14, 15, 29]). In this table, columns indicate the cyber security problem while rows denote the adversarial attack class. Each cell reports the machine-learning algorithms that are tested against the related class of attacks. We remark that algorithms written in bold are evaluated for the first time in this paper. The existing literature focuses mainly on integrity attacks, with several algorithms evaluated for Malware analysis and Spam analysis. Few solutions exist and are tested in the Network intrusion detection context, and this observation motivates this paper. There are few documented attacks targeting the system availability, and there are no specific studies at test-time.

TABLE 1. MAPPING OF THE CATEGORIES OF ADVERSARIAL ATTACKS TO CYBER SECURITY PROBLEMS. LEGEND: RF=RANDOM FOREST; MLP=MULTI-LAYER PERCEPTRON; KNN=K-NEAREST NEIGHBOUR; NB=NAÏVE BAYES; SVM=SUPPORT VECTOR MACHINE; LR=LOGISTIC/LINEAR REGRESSION; DNN=DEEP NEURAL NETWORK.

		Network intrusion detection	Malware analysis	Spam analysis
Test-time	*Availability violation*	✗	✗	✗
	Integrity violation	RF [30] **MLP** **KNN** NB [31]	RF [32] SVM [7] LR [33] MLP [7]	SVM [34] LR [35] NB [35]
Training-time	*Availability violation*	✗	NB [36]	NB [11] Clustering [37]
	Integrity violation	**RF** **MLP** **KNN**	LR [38] DNN [39]	NB [25] DNN [39]

3. DEFENCES AGAINST ADVERSARIAL ATTACKS

Devising effective solutions against adversarial attacks is a challenging task. We present existing methods proposed in the literature that aim to mitigate these critical threats. Countermeasures can be divided into two groups: those conforming to the *security-by-design* paradigm that are effective against perfect-knowledge attacks; and methods that are only effective against partial- or zero-knowledge attacks. One of the main limitations of most solutions against adversarial attacks is that they may worsen the performance of the cyber detector in the absence of adversarial attacks, typically causing higher false positive rates (e.g., [40, 27, 41, 42]).

A. Defences Against Attacks at Test-time

Since there are no known examples of availability attacks at test-time, we focus on defences against attacks targeting the integrity of the system. These threats involve the creation of specific samples that evade the detection mechanism. For example, an opponent can alter a malicious sample to induce its classification as a benign sample. The security-by-design countermeasures aim to improve the machine-learning system capabilities to detect even adversarially manipulated samples.

- **Adversarial training.** These solutions train the model on datasets that include samples of possible adversarial attacks [43]. A recent proposal [42] suggests the adoption of a generative adversarial network (GAN) to automatically generate a similar dataset, achieving promising results. However, these approaches are not a "catch-all" solution, because it is simply unfeasible to obtain a dataset that contains all possible variations of realistic adversarial samples.
- **Robust optimisation.** The authors in [44] and [45] propose techniques aimed at smoothing the decision boundaries of the machine-learning algorithm, thus reducing the effects of adversarial samples. Similar solutions can help to mitigate some attacks, but expert opponents are still able to craft malicious samples that look like licit activities.
- **Feature selection.** Other proposals (e.g., [40, 5]) suggest training the detection model by considering only the subset of features that cannot be manipulated by an attacker. While this method can prevent certain types of evasion attacks, feature removal reduces the detection rates in non-adversarial scenarios [40].
- **Game theory.** These approaches represent the problem of adversarial attacks as a zero-sum game between the attacker and the defender, and work under several assumptions. They require a model of the attacker knowledge and capabilities that must be integrated into the machine-learning algorithm. The optimal defence course against the modelled attacker is found when the system reaches an equilibrium. An example of application to spam detection is described in [46]. The main limitation of these strategies is that they are only able to counter attacks that strictly conform to the considered attacker's model, because even small deviations nullify their effectiveness. Since the cyber security world is intrinsically unpredictable and fuzzy, most of these solutions are not applicable to real contexts.
- **Ensemble methods.** The paper by Biggio et al. [47] shows that it is possible to counter evasion attacks at test-time by devising systems composed by multiple classifiers. However, each classifier represents a weak link in the security chain because the misconfiguration of even one component can lead to poor results, as shown in [48].

Most black- and grey-box evasion attacks involve a *probing* step, in which the adversary aims to gather information on the detector by submitting specific inputs to the system and observing the subsequent response. Thus, existing defences address these malicious exploratory activities by providing misleading information to the attacker. For example, the authors in [47] suggest classifiers that are difficult to reverse-engineer or propose a randomization of the detector output. The problem of these solutions is that they tend to work against attackers with limited time or skill that adopt automated tools. Expert opponents can detect such deception activities and bypass them.

B. Defending Against Attacks at Training-time

Attacks performed at training-time alter the decision process of the machine-learning algorithm by modifying the configuration of the model before the training phases, that is, by manipulating the training dataset(s). Existing solutions focus on protecting the training dataset with the objective of minimizing the effects of adversarial perturbations. We identify the following two groups of security-by-design defences.

- **Data sanitization.** Poisoning attacks are countered through a data sanitization process that aims to detect and remove poisoned samples introduced in the training data [49]. The problem is that some assumptions of these approaches are not always applicable to the cyber security field. For example, the work in [50] assumes that each poisoning sample significantly affects the training process. This assumption is not valid in many situations in which an attacker introduces few samples just to avoid some specific detections of his interest. Other solutions [51] leverage the *machine unlearning* concept that allows the effects of poisoned data to be cancelled without the need to retrain the machine-learning model. The main limitation of this approach is that it needs to know which (poisoned) data to unlearn, that is, it requires the knowledge of which poisoned data samples have been introduced by the attacker. This is an unrealistic assumption in real cyber security contexts.
- **Ensemble methods.** The adoption of multiple-classifier systems can also be effective against attacks at training-time [50]. These solutions present the same advantages and problems characterizing their test-time version, that is, a misconfiguration of even one component can damage the results of the entire detection mechanism.

Defences against partial- or zero-knowledge attacks include the collection of training data from randomized sources [52] with the goal of making it harder for the attacker to devise effective adversarial samples; and the application of strategies to prevent the attacker from controlling the actual training dataset [52]. As an example of this latter group, we propose an original methodology based on the idea of generating the actual training set only at training-time. The approach introduces data transformation procedures on the training dataset. In this way, even if an adversary manages to poison the stored dataset by injecting malicious samples that are labelled as benign, the data transformation step ensures that the model is not trained on those exact poisoned samples. The expected result is that these samples will have a significantly smaller impact on the detector. The complete description of this solution is as follows.

We assume an organization that adopts a cyber detector relying on a supervised algorithm, which is periodically retrained. The training is based on a dataset X' that is stored on a dedicated database server. Let T be an invertible function with domain K so that:

$$T^{-1}(T(k)) = T^{-1}(k') = k, \quad \forall\, k \in K$$

The organization employs the transformation defined by T. More specifically, each time a new piece of data x is added to the dataset X', it is transformed as $T(x) = x'$. When it is necessary to retrain the detector, the dataset X' is retrieved and is inversely transformed through T^{-1}, providing the original training dataset, X.

Now, let us assume that an attacker obtains full access to the database server containing X'. The attacker attempts a poisoning attack by introducing some samples $\overline{x}$ in X' that are labelled as benign, and that represent malicious actions. (For example, the underlying code or network behaviour of a piece of malware, or a spam email). As the attacker is unaware of the data transformation, he does not try to infer the existence of a similar function by analysing the dataset and does not apply the data transformation T to the $\overline{x}$ samples. When the detector is retrained, these samples $\overline{x}$ will undergo the transformation T^{-1}, resulting in samples $\overline{x}^{-1}$ with different characteristics than those of the malicious actions that the attacker wanted to evade detection. This results in poisoning samples whose effect on the detector will be different from that desired by the attacker. We report the entire workflow of the proposed approach in Figure 1 and Figure 2.

FIGURE 1. WORKFLOW OF THE PROPOSED POISONING COUNTERMEASURE: OPERATIONS PERFORMED BEFORE THE (RE)TRAINING.

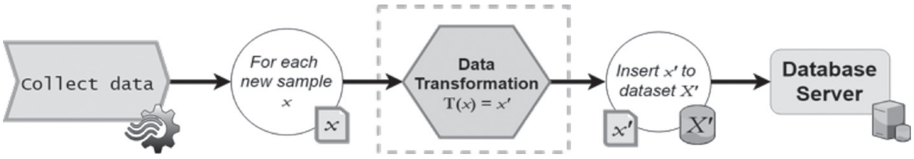
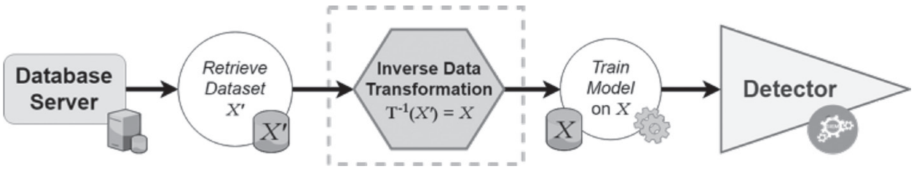


FIGURE 2. WORKFLOW OF THE PROPOSED POISONING COUNTERMEASURE: OPERATIONS PERFORMED AT (RE)TRAINING-TIME.



To provide an improved understanding of the proposed method, we present the following example. Consider an organization adopting a classifier C that analyses network flows [53] to distinguish between malicious and benign traffic; let $\hat{X}$ be the dataset of network flows used to train the classifier, and let $\hat{T}$ be a transformation that modifies a flow sample by multiplying the *flow_duration* by $d \in \mathbb{R}$, and dividing the *flow_exchanged_bytes* by $b \in \mathbb{R}$; conversely, $\hat{T}^{-1}$ modifies a flow sample by dividing its *flow_duration* by d and multiplying its *flow_exchanged_bytes* by b. With these assumptions, the dataset $\hat{X}$ is stored in the organization database as $\hat{X}'$. That is, every flow sample $\hat{x} \in \hat{X}$ is modified into $\hat{x}'$ by having the values of its *flow_duration* multiplied by d, and the values of its *flow_exchanged_bytes* divided by b. Therefore, every time the dataset $\hat{X}'$ is updated with a new set of flows, the flows are subject to the transformation denoted by $\hat{T}$. Consequently, whenever the classifier C undergoes a retraining process, each flow $\hat{x}' \in \hat{X}'$ will be inversely transformed by $\hat{T}^{-1}$ into its original version, $\hat{x}$.

Now, if an unaware attacker attempts to poison the stored dataset $\hat{X}'$ by inserting some adversarial samples $\hat{\hat{x}}$ that are wrongly labelled, he will not perform the transformation defined by $\hat{T}$, that is, the adversarial flows will not have their *flow_duration* and *flow_exchanged_bytes* modified. Hence, when the classifier, C is retrained, the adversarial samples $\hat{\hat{x}}$ will be transformed by $\hat{T}^{-1}$ into $\hat{\hat{x}}^{-1}$. As a practical example, if b=10 and d=2, and if an attacker introduces in $\hat{X}'$ the adversarial sample, $\hat{\hat{x}}$ having *flow_duration*=2 and *flow_exchanged_bytes*=240, then $\hat{T}^{-1}$ will modify it into $\hat{\hat{x}}^{-1}$ having *flow_duration* =1 and *flow_exchanged_bytes*=2400. Thus, this sample will have different effects on the retraining process of classifier C than the ones intended by the attacker.

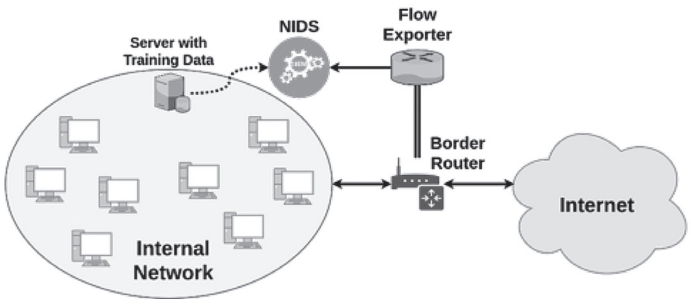
4. EXPERIMENTAL RESULTS

We present an original evaluation of integrity attacks performed at training- and test-time against network intrusion detection systems based on three supervised machine-learning algorithms that achieve appreciable detection performance [14]: Random Forest, Multi-layer Perceptron, K-Nearest Neighbour. We initially present the application scenario, the experimental testbed, and the baseline performance of the considered detectors. Then, we evaluate them in adversarial scenarios and assess the effectiveness of possible countermeasures.

A. Experimental Environment

We consider a typical context, shown in Figure 3, where the network of a large enterprise is monitored by a NIDS based on a machine-learning classifier that inspects the network flows of the border router [53]. The NIDS is periodically retrained with updated data stored on a dedicated database server.

FIGURE 3. SCENARIO ADOPTED FOR THE EXPERIMENTS.



The testbed is based on a publicly available collection of multiple datasets of network flows captured in a monitored environment with dozens of hosts, where some machines are infected with malware belonging to seven botnet families [54]. Overall, these datasets contain over 20 million network samples that are labelled as either legitimate or illegitimate. In the evaluation, we split each detector into several instances, each devoted to one botnet family. Each instance is trained on a training set containing 80% of the malicious samples of the related botnet family, while the remaining 20% is used in the test-set. We use a fixed 85:15 ratio of legitimate-to-illegitimate samples for each training- and test-set. The quality of each detector is measured through the traditional performance indicators *Precision, Recall* (or *Detection Rate*), *F1-score* and *Accuracy*:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \qquad Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP, TN, FP and FN denote true positives, true negatives, false positives and false negatives, respectively. A positive refers to a malicious sample. The values presented in Table 2 represent the average of the results for each detector. These detectors obtain an appreciable performance that is comparable to the state-of-the-art [14, 55].

TABLE 2. BASELINE PERFORMANCE OF THE CLASSIFIERS.

	Precision	Recall	F1-score	Accuracy
Random Forest	0.9774	0.9684	0.9729	0.9978
Multi-layer Perceptron	0.9616	0.9438	0.9526	0.9912
K-Nearest Neighbour	0.9558	0.9375	0.9466	0.9909

To measure the effectiveness of adversarial integrity attacks and their countermeasures, we introduce the *attack severity (AS)* metric, where attacks with higher (respectively, lower) magnitude will obtain *AS* scores that are closer to 1 (respectively, 0):

$$AS = 1 - \frac{Recall\text{ (after the attack)}}{Recall\text{ (before the attack)}}$$

B. Evaluation of Adversarial Attacks at Test-time

The first experiments involve integrity violations performed at test-time. We consider an attacker that has already established a foothold within the enterprise's internal network by compromising one or more machines with botnet malware; these bots communicate with an external Command and Control server. The attacker model is based on the following three assumptions: his goal is to evade detection in order to expand his control of the internal network [56]; he knows that the organization adopts a botnet detector based on machine learning, which is trained on malware samples that are similar to the variant used by the bots; he can interact with the controlled bots, but he cannot access the botnet detector. To achieve his goal, the attacker plans to slightly modify the network communications performed by the bots (e.g., small increments in the amount of exchanged data and in the communications duration) so that these small perturbations can induce misclassifications of botnet flows. We simulate this realistic attack scenario by altering the following flow-based features: *exchanged_bytes, duration, total_packets*. This process is repeated for all the samples of each botnet variants. Table 3 reports the average results for the three detectors considered, when tested against these adversarial samples. All these algorithms are severely affected by the adversarial attacks: the detection rate in the second column is about one-third of the original rate.

TABLE 3. EFFECTS OF THE EVASION ATTACK ON EACH CLASSIFIER.

	Recall (before the attack)	Recall (after the attack)	Attack Severity
Random Forest	0.9684	0.3429	**0.6459**
Multi-layer Perceptron	0.9438	0.3012	**0.6809**
K-Nearest Neighbour	0.9375	0.3121	**0.6671**

To defend against similar threats, we explore two of the countermeasures proposed in the literature: *adversarial retraining* and *feature removal*.

For the former case, we harden the detectors by inserting some of the adversarial samples that we manually crafted into their training sets (with the appropriate malicious label), and we repeat the training process. Then, we test the classifiers again on the respective adversarial datasets. The results of this evaluation are reported in Table 4, which compares the severity of the attacks before and after retraining. The decreased severity of the attack after retraining shows the validity of *adversarial retraining*. However, it should be observed that this technique does not guarantee detection against other types of adversarial perturbations.

TABLE 4. EVALUATION OF THE COUNTERMEASURE BASED ON ADVERSARIAL RETRAINING.

	Attack Severity	Attack Severity (after Retraining)
Random Forest	0.6459	**0.3842**
Multi-layer Perceptron	0.6809	**0.4089**
K-Nearest Neighbour	0.6671	**0.4772**

The defences based on *feature removal* aim to nullify the effects of evasion attacks by adopting a set that does not include features related to *duration, exchanged_bytes* and *total_packets*. By training each detector without these features, the results are optimal because the attack severity measure drops to 0. The problem with this approach is that it typically affects the detector performance in scenarios that are not subject to adversarial attacks. By comparing the performance in non-adversarial settings for each detector before and after retraining with the modified feature set, we obtain the results presented in Table 5. All the performance metrics considered fall well below acceptable values for any NIDS. It is possible to attenuate the performance drop by excluding only those features that have a small impact in the decision process of the detector, but this approach will not prevent all evasion attacks.

TABLE 5. EVALUATION OF THE COUNTERMEASURE BASED ON FEATURE REMOVAL IN NON-ADVERSARIAL SETTINGS, AND COMPARISON WITH THE BASELINE PERFORMANCE.

	Precision		*Recall*		*F1-score*		*Accuracy*	
	Original	New	Original	New	Original	New	Original	New
Random Forest	0.9774	**0.8561**	0.9684	**0.8885**	0.9729	**0.8719**	0.9978	**0.9711**
Multi-layer Perceptron	0.9616	**0.7934**	0.9438	**0.7561**	0.9526	**0.7743**	0.9912	**0.9816**
K-Nearest Neighbour	0.9558	**0.8298**	0.9375	**0.8091**	0.9466	**0.8193**	0.9909	**0.9838**

C. Adversarial Attacks at Training-time

We analyse the effects of poisoning attacks that focus on integrity violations. The attacker model considers an opponent who has compromised the targeted network and plans to infect other hosts with novel malware. He is aware that the network is monitored by a NIDS based on some supervised machine-learning algorithms, and he also knows that this detector is periodically retrained. His goal is to ensure that the deployed new malware variants evade detection mechanisms. The attacker has full access to the server that contains the training dataset, but he cannot interact with the detector. To reach his goal, the attacker plans to poison the training dataset through malicious samples representing the behaviour of the deployed malware variant, but that is classified with the benign label.

To simulate this attack scenario, we craft sets of malicious flows that slightly differ (to account for the novel malware variant) from those contained in the testbed, and we label them as benign. This procedure is performed by selecting the existing malicious samples and increasing their *duration* by [1-5] seconds, their *exchanged_bytes* by [1-1024], and their *total_packets* by [1-10]. Then, we inject some of these samples into each training dataset. We measure the effectiveness of a similar attack by comparing the performance of the detectors on the poisoned samples before and after the poisoned retraining phase. The results shown in Table 6 highlight that, before the poisoning attempt, the classifiers were able to identify the novel attack samples with detection rates comparable to other proposals against zero-day malware [17]. The performance of the same algorithms suffered a significant drop after a retraining phase with the poisoned data. The high attack severity score gives a clear idea of the impact of the effect.

TABLE 6. EFFECTS OF THE POISONING ATTACK ON EACH CYBER DETECTOR.

	Recall (before the attack)	Recall (after the attack)	Attack Severity
Random Forest	0.8834	0.2636	**0.7016**
Multi-layer Perceptron	0.8674	0.2777	**0.6798**
K-Nearest Neighbour	0.8611	0.2391	**0.7223**

We now evaluate the original methodology presented in Section 3.B by introducing a custom data transformation procedure on the training set, and then replicating the poisoning attack. For the sake of clarity, we consider a simple function $\hat{T}$ that multiplies the *duration* by $d \in \mathbb{R}$, and divides the exchanged_bytes by $b \in \mathbb{R}$. In this way, the poisoned samples introduced by the attacker are (inversely) transformed into samples that are different from the flows generated by the malware variant, because they have durations of $+[\frac{1}{d}, \frac{5}{d}]$ seconds (instead of +[1,5]) in which the hosts exchange $+[1*b, 1024*b]$ bytes (instead of +[1,1024]).

In Table 7, we compare the attack severity of the poisoning attempt before and after the application of the countermeasure, from which we can deduce that the proposed approach can significantly mitigate the effects of a poisoning attack.

TABLE 7. EVALUATION OF THE PROPOSED DEFENSIVE METHOD.
THESE RESULTS ARE OBTAINED BY SETTING d=2 AND m=5.

	Attack Severity	Attack Severity (with Data Transformation)
Random Forest	0.7016	**0.1587**
Multi-layer Perceptron	0.6798	**0.1741**
K-Nearest Neighbour	0.7223	**0.2830**

5. CONCLUSIONS

Machine- and deep-learning algorithms are adopted in many application domains, but in the cyber security field, they are affected by several open issues. In this paper, we consider adversarial attacks where the machine-learning model is compromised to induce an output favourable to the attacker. Literature on this subject is still immature, and most documented examples of adversarial attacks against security systems consider only few algorithms and few application areas. We present a taxonomy of adversarial attacks that evidences which cyber security areas and which machine-learning algorithms have been evaluated against what type of threat. This analysis evidences that there is space for novel research in the context of adversarial attacks against network intrusion detection systems based on machine learning. We are confident that the large set of original experiments and the novel way to address issues related to adversarial attacks presented here can pave the way for cyber detection platforms that are based on more robust machine-learning algorithms.

REFERENCES

[1] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature*, 2015.

[2] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, 2015.

[3] S. Dua and X. Du, Data mining and machine learning in cybersecurity, Auerbach Publications, 2016.

[4] M. Stevanovic and J. M. Pedersen, "On the use of machine learning for identifying botnet network traffic," *Journal of Cyber Security and Mobility*, 2016.

[5] J. Gardiner and S. Nagaraja, "On the Security of Machine Learning in Malware C8C Detection," *ACM Computing Surveys*, 2016.

[6] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein and J. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Security and Artificial Intelligence*, 2011.

[7] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto and F. Roli, "Evasion attacks against machine learning at test time," in *Joint Eur. Conf. Machine Learning and Knowledge Discovery in Databases*, 2013.

[8] N. Papernot, P. McDaniel, A. Sinha and M. Wellman, "Towards the science of security and privacy in machine learning," *arXiv preprint arXiv:1611.03814*, 2016.

[9] Please change: N. Papernot, P. McDaniel, A. Sinha and M. Wellman, "SoK: Security and privacy in machine learning," 2018 IEEE Europ. Symp. on Security and Privacy.

[10] B. Biggio, B. Nelson and P. Laskov, "Poisoning attacks against support vector machines," Proc. 29th Int. Conf. Machine Learning, 2012.

[11] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. Rubinstein, U. Saini, C. A. Sutton, J. D. Tygar and K. Xia, "Exploiting Machine Learning to Subvert Your Spam Filter," *Proceedings of the LEET, USENIX Association*, 2008.

[12] A. Demontis, M. Melis, B. Biggio, D. Maiorca, D. Arp, K. Rieck, I. Corona, G. Giacinto and F. Roli, "Yes, machine learning can be more secure! a case study on android malware detection," *IEEE Trans. Depend. Sec. Comput.*, 2017.

[13] F. Pierazzi, G. Apruzzese, M. Colajanni, A. Guido and M. Marchetti, "Scalable architecture for online prioritization of cyber threats," in *2017 NATO 9th Int. Conf. Cyber Conflict*.

[14] A. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, 2015.

[15] E. Blanzieri and A. Bryl, "A survey of learning-based techniques of email spam filtering," *Artificial Intelligence Review*, 2008.

[16] R. P. V. Sommer, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symp. Security and Privacy*.

[17] M. Alazab, S. Venkatraman, P. Watters and M. Alazab, "Zero-day malware detection based on supervised learning algorithms of API call signatures," in *Proc. 9th Australasian Data Mining Conf.*, 2011.

[18] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido and M. Marchetti, "On the effectiveness of machine and deep learning for cyber security," in *2018 10th Int. Conf. Cyber Conflict*.

[19] G. Apruzzese, M. Marchetti, M. Colajanni, G. Gambigliani Zoccoli and A. Guido, "Identifying malicious hosts involved in periodic communications," in *2017 16th IEEE Int. Symp. Network Computing and Applications*.

[20] M. Mannino, Y. Yang and Y. Ryu, "Classification algorithm sensitivity to training data with non representative attribute noise," in *Decision Support Systems*, 2009.

[21] I. H. Witten, E. Frank, M. A. Hall and C. J. Pal, Data Mining: Practical machine learning tools and techniques, 2016.

[22] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, 2014.

[23] A. Kantchelian, S. Afroz, L. Huang, A. C. Islam, B. Miller, M. C. Tschantz, R. Greenstadt, A. D. Joseph

and J. Tygar, “Approaches to adversarial drift,” in *Proc. 2013 ACM Workshop Artificial Intelligence and Security*, 2013.

[24] B. Biggio, I. Corona, B. Nelson, B. I. Rubinstein, D. Maiorca, G. Fumera, G. Giacinto and F. Roli, “Security evaluation of support vector machines in adversarial environments,” in *Support Vector Machines Applications*, 2014.

[25] N. Dalvi, P. Domingos, S. Sanghai and D. Verma, “Adversarial classification,” in *Proc,. 10th ACM Int. Conf. Knowledge Discovery and Data Mining*, 2004.

[26] M. Fredrikson, S. Jha and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in *Proc. 22nd ACM Conf. Computer and Communications Security*, 2015.

[27] N. Papernot, P. McDaniel, X. Wu, S. Jha and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” in *2016 IEEE Symp. Security and Privacy*.

[28] S. Huang, N. Papernot, I. Goodfellow, Y. Duan and P. Abbeel, “Adversarial attacks on neural network policies,” *arXiv preprint arXiv:1702.02284*, 2017.

[29] T. Chakraborty, F. Pierazzi and V. Subrahmanian, “Ec2: Ensemble clustering and classification for predicting android malware families,” *IEEE Trans. Depend. Sec. Comput*, 2017.

[30] G. Apruzzese and M. Colajanni, “Evading Botnet Detectors Based on Flows and Random Forest with Adversarial Samples,” in *2018 IEEE 17th Int. Symp. Network Computing and Applications*.

[31] C. V. Wright, S. E. Coull and F. Monrose, “Traffic Morphing: An Efficient Defense Against Statistical Traffic Analysis.,” in *Network and Distributed System Security Symp*., 2006.

[32] P. Laskov, “Practical evasion of a learning-based classifier: A case study,” in *2014 IEEE Symp. Security and Privacy*.

[33] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert and F. Roli, “Is feature selection secure against training data poisoning?,” in *Int. Conf. Machine Learning*, 2015.

[34] D. Lowd and C. Meek, “Adversarial learning,” in *Proc. 11th ACM Int. Conf. Knowledge Discovery in Data Mining*, 2005.

[35] D. Lowd and C. Meek, “Good Word Attacks on Statistical Spam Filters,” in *CEAS*, 2005.

[36] J. Newsome, B. Karp and D. Song, “Paragraph: Thwarting signature learning by training maliciously,” in *Int. Workshop Recent Advances in Intrusion Detection*, 2006.

[37] B. Biggio, I. Pillai, S. Rota Bulo, D. Ariu, M. Pelillo and F. Roli, “Is data clustering in adversarial settings secure?,” in *Proc. 2013 ACM Workshop Artificial Intelligence and Security*.

[38] C. Liu, B. Li, Y. Vorobeychik and A. Oprea, “Robust linear regression against training data poisoning,” in Proc. *10th ACM Workshop Artificial Intelligence and Security*, 2017.

[39] L. Munoz-Gonzalez, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu and F. Roli, “Towards poisoning of deep learning algorithms with back-gradient optimization,” in Proc. *10th ACM Workshop Artificial Intelligence and Security*, 2017.

[40] F. Zhang, P. P. Chan, B. Biggio, D. S. Yeung and F. Roli, “Adversarial feature selection against evasion attacks,” *IEEE Trans. Cybern*., 2016.

[41] B. Biggio, B. Nelson and P. Laskov, “Support vector machines under adversarial label noise,” in *Asian Conf. Machine Learning*, 2011.

[42] H. S. Anderson, J. Woodbridge and B. Filar, “DeepDGA: Adversarially-Tuned Domain Generation and Detection,” in *Proc. 2016 ACM Workshop Artificial Intelligence and Security*.

[43] A. Kantchelian, J. Tygar and A. Joseph, “Evasion and hardening of tree ensemble classifiers,” in *Int. Conf. Machine Learning*, 2016.

[44] H. Xu, C. Caramanis and S. Mannor, “Robustness and regularization of support vector machines,” *Journal of Machine Learning Research*, 2009.

[45] P. Russu, A. Demontis, B. Biggio, G. Fumera and F. Roli, “Secure kernel machines against evasion attacks,” in *Proc. 2016 ACM Workshop Artificial Intelligence and Security*.

[46] M. Bruckner and T. Scheffer, “Stackelberg games for adversarial prediction problems,” in *Proc. 17th ACM Int. Conf. Knowledge Discovery and Data Mining*, 2011.

[47] B. Biggio, G. Fumera and F. Roli, “Adversarial pattern classification using multiple classifiers and randomisation,” in *Joint IAPR Int. Workshops on Statistical Techniques in Pattern Recognition and Structural and Syntactic Pattern Recognition*, 2008.

[48] I. Corona, B. Biggio, M. Contini, L. Piras, R. Corda, M. Mereu, G. Mureddu, D. Ariu and F. Roli, “Deltaphish: Detecting phishing webpages in compromised websites,” in *Eur. Symp. Research in Computer Security*, 2017.

[49] G. F. Cretu, A. Stavrou, M. E. Locasto, S. J. Stolfo and A. D. Keromytis, “Casting out demons: Sanitizing training data for anomaly sensors,” in *Proc. 2008 IEEE Symp. Security and Privacy*, 2008.

[50] B. Biggio, I. Corona, G. Fumera, G. Giacinto and F. Roli, “Bagging classifiers for fighting poisoning attacks in adversarial classification tasks,” in *Int. Work Multiple Classifier Systems*, 2011.

[51] Y. Cao and J. Yang, “Towards making systems forget with machine unlearning,” in *2015 IEEE Symp. Security and Privacy*.

[52] A. D. Joseph, P. Laskov, F. Roli, J. D. Tygar and B. Nelson, “Machine learning methods for computer security,” in *Dagstuhl Manifestos*, 2013.

[53] “Netflow,” [Online]. Available: https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html.

[54] S. Garcia, M. Grill, J. Stiborek and A. Zunino, “An empirical comparison of botnet detection methods,” *Computers & Security*, 2014.

[55] M. Stevanovic and J. M. Pedersen, “An analysis of network traffic classification for botnet detection,” in *2015 Int. Conf. Cyber Situational Awareness, Data Analytics and Assessment*.

[56] G. Apruzzese, F. Pierazzi, M. Colajanni and M. Marchetti, “Detection and threat prioritization of pivoting attacks in large networks,” *IEEE Trans. Emerg. Topics. Comput.*, 2017.