Evaluating the effectiveness of Adversarial Attacks against Botnet Detectors

Giovanni Apruzzese, Michele Colajanni, Mirco Marchetti
Department of Engineering "Enzo Ferrari"
University of Modena and Reggio Emilia
Modena, Italy
{giovanni.apruzzese, michele.colajanni, mirco.marchetti}@unimore.it

***Abstract*—Classifiers based on Machine Learning are vulnerable to adversarial attacks, which involve the creation of malicious samples that are not classified correctly. While this phenomenon has been extensively studied within the image processing domain, comprehensive analyses are scarce in the cybersecurity field. This is a critical problem because cyber-detectors are being increasingly integrated with machine learning methods, making them suitable targets for skilled attackers leveraging adversarial samples to evade detection. In this paper, we propose a thorough analysis of realistic adversarial attacks performed against network intrusion detection systems that focus on identifying botnet traffic through machine learning classifiers. Our large campaign of experiments involves the most recent public datasets, representing multiple realistic network scenarios. Moreover, we evaluate the impact of these attacks against state-of-the-art detectors relying on different machine learning algorithms, providing a clear overview of this problem. The results outline the fragility of these methods. Our study represent a stepping stone for devising suitable countermeasures to the menace of adversarial attacks against cyber-detectors.**

***Index Terms*—Adversarial samples, machine learning, intrusion detection, flow inspection, botnet**

I. Introduction

A recent trend across multiple application domains, including cybersecurity, is the adoption of machine learning techniques. The appreciable results of these methods are not limited to scientific research, as many enterprises have started to integrate their commercial products with these novel techniques [1], [2]. This growing diffusion has led to question the reliability of machine learning in adversarial settings, where specific inputs are created with the explicit goal of subverting the machine learning algorithms [3], [4]. While this topic is extensively studied in domains such as computer vision and text processing [5], it lacks thorough analyses from a cybersecurity perspective [6]. In this paper, we study adversarial perturbations in the context of network intrusion detection systems (NIDS). Our objective is to provide a comprehensive evaluation of the impact of these malicious actions against state-of-the-art detectors of botnets, due to their relevance in the current cybersecurity landscape [7]. More specifically, we perform an extensive experimental campaign of realistic adversarial attacks, involving different machine learning techniques and multiple large datasets of network flows. Furthermore, we also evaluate the effects of some proposed defensive mechanisms. Thus, our broad analysis captures a multitude of scenarios that represent the heterogeneous nature of modern networked systems. To the best of our knowledge, this is the first paper that presents such a vast range of novel experiments on this subject. The results highlight the vulnerability of all the considered detectors to adversarial samples, and the limitations of the analysed defensive proposals. We are confident that our study will benefit the cybersecurity field, by inducing more research efforts to be focused on devising much needed effective countermeasures against this critical threat.

The remainder of this paper is structured as follows. Section II compares this paper with related work. In Section III we describe the realistic application scenario of our analysis. Section IV presents the testbed and evaluation methodology. We devote Section V to the experimental results. Conclusions are drawn in Section VI.

II. Related Work

Adversarial examples have been extensively studied within the image processing field (e.g.: [4], [5], [8]), but their evaluation in the cybersecurity domain is still an open research problem [9]. This is the main motivation behind our work, since a clear understanding of the impact of adversarial attacks against cyber defence systems based on machine learning is of crucial importance for the development of modern cybersecurity technologies.

Related literature has mostly focused on analysing the effects of these threats against malware and spam detectors [10]–[15], but few papers address this problem from a network intrusion detection perspective [16]. The

978-1-7281-2522-0/19/$31.00

authors of [17] present an analysis of attacks against Network Intrusion Detection Systems (NIDS), but they do not consider machine learning techniques. Other works raise the awareness on this subject by proposing mechanisms to evade machine learning-based NIDS, but they do not provide any experimental evidence to sustain their claims [18]. Previous work also include research efforts [19], [20] based on the deprecated [21] KDD-99 dataset[1], and cannot be considered as a good representation of modern large network environments. Other papers only consider attack scenarios where the adversary has extensive or perfect knowledge of the detector, which is an unrealistic assumption in a true cybersecurity context [22]–[24]. More recently, Apruzzese et al. showed effective realistic adversarial attacks against botnet detectors based on machine learning, but this work only focused on a single dataset and consider a limited subset of machine learning algorithms [16], [25].

To the best of our knowledge, this is the first paper that presents an extensive evaluation of *realistic* adversarial attacks performed against botnet detectors based on *multiple* machine learning algorithms, which are deployed in *different*, *recent* and *heterogeneous* network scenarios. Thus, the present study portrays a much needed and representative overview of the current state-of-the-art of machine learning botnet detectors in adversarial settings.

III. Application scenario

To conduct a meaningful evaluation of the impact of realistic adversarial attacks, it is necessary to describe the characteristics of the considered network environment and of the considered attacker.

A. Scenario

We assume a realistic network scenario of a medium-large organization comprising up to hundreds of devices that generate large amounts of daily traffic (>10GB/day). We also assume that the network perimeter is monitored by a flow-based[2] NIDS that adopts a supervised machine learning algorithm to detect traces of botnet activities. We remark that analysing network flows instead of raw traffic packets is a growing and successful practice for NIDS, due to the reduced resources needed for their storage and computation, as well as reduced privacy concerns [26], [27] with respect to deep inspection of full packet traces. We assume that some machines within the network are infected by a botnet-related malware (for example through a zero-day attack, successfull spear phishing attempts, or insider threats) that communicates with an external Command and Control (CnC) server.

B. Attacker Model

We describe the model of the considered attacker by following the guidelines proposed in related literature [9], outlining their *goal*, *knowledge*, *capabilities* and *strategy*.

1) Attacker Goal: The attacker aims to evade the botnet detector in order to maintain access to the network and perform additional malicious activities (e.g.: [28]).

2) Attacker Knowledge: The attacker is aware that the target organization adopts some defensive mechanisms that analyze network traffic through supervised machine learning algorithms; he also assumes that the training set may contain some malware samples with similar characteristics to the one used to infect the compromised hosts. However, the attacker has no complete knowledge of the dataset used to train the machine learning algorithm, of the the feature set, and of the classifier's internal parameters and configuration. We remark that these are realistic assumptions for any commercial cybersecurity appliance.

3) Attacker Capabilities: The attacker can issue commands to the infected machines through the CnC server. However, he does not have direct access to the detector, this includes both read and write operations. For example, he cannot use the detector as an "oracle" [3] by submitting certain inputs and reading the respective outputs.

4) Attacker Strategy: The attacker attempts to evade detection by performing an *integrity* attack [9]. More specifically, he modifies the (malicious) network communications generated by the infected devices and the corresponding CnC server. These alterations need to keep the internal logic of the employed piece of malware intact, as their sole purpose is to make the (malicious) samples different from the ones that have possibly been used to train the detector; furthermore, they need to be stealthy enough to avoid triggering detection through other defensive mechanisms [29]. Possible perturbations include small increments to the length of the communications, or the insertion of some pieces of junk data in the transmitted packets.

We highlight that our assumptions portray a realistic scenario: models that involve attackers with perfect (or near-perfect) knowledge are unrealistic, as the NIDS is usually protected by multiple layers of defenses. Adversaries that have possess such confidential information – or that have direct access to the detector – are also capable of launching attacks of higher magnitude that are out of the scope of this paper. Also adversaries that are able to generate a brand new malware characterized

[1]KDD99 Dataset: https://www.unb.ca/cic/datasets/nsl.html

[2]NetFlow: https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow

by different communication patterns between bots and CnC will still be able to evade detection. Hence we focus on the wide majority of attackers that leverage and customize common malware toolkits.

IV. Evaluation methodology

Conducting a thorough evaluation of machine learning-based NIDS in realistic adversarial settings is a complex procedure. Here, we present the datasets and the algorithms used to develop the considered detectors. Then we describe the methods used to generate the adversarial attacks and to measure their effectiveness on these detectors; finally, we assess the effectiveness of feature removal as a defense against adversarial attacks.

A. Testbed description

In our evaluation we rely on 4 recent, public and labelled datasets of network traffic that include communications generated by botnet-related pieces of malware: `CTU-13`[3] [30], `IDS2017`[4] [21], `CIC-IDS2018`[5] [21], `UNB-CA Botnet`[6] [31]. As these datasets involve different types of attacks, we only consider those portions that include botnet-related traffic. Some of these datasets are readily available in netflow format; for those that only include raw packet data, we generate the corresponding flows through Argus[7]. We summarize the meaningful metrics of each dataset in Table I, which reports the total amount of packets, internal hosts, flows and number of botnet families included. We exclude those families that present less than 100 samples, as their scarcity may lead to the creation of training sets that would generate poor detection results [25], [31], [32]. We can observe that the considered datasets are a valid representation of medium-to-large network scenarios.

Table I: Datasets metrics.

Dataset	Packets	Devices	Botnet Flows	Legitimate Flows	Botnet Families
CTU-13	855 866 143	150	443 906	19 199 170	6
IDS2017	5 776 888	111	1 966	189 067	1
CIC-IDS2018	13 486 990	450	283 429	760 824	1
UNB-CA Botnet	14 502 782	369	238 415	345 113	10

Our experiments involve multiple botnet detectors, each based on a different machine learning classifier among the following: Random Forest (RF), Decision Trees (DT), AdaBoost (AB), Multi-Layer Perceptron (MLP), K-Nearest Neighbor (KNN), Gradient Boosting (GB), Linear Regression (LR), Support Vector Machines (SVM), Naive Bayes (NB), ExtraTrees (ET), Bagging (Bag), Stochastic Gradient Descent Linear Classifier (SGD). We focus on these algorithms since previous work demonstrate their applicability to the task of identifying botnet traffic [6], [32]–[34]. As each dataset may include multiple botnet families, we develop our detectors by training every classifier on each individual botnet family per dataset. This is motivated by the fact that machine learning methods tend to perform better when they address a specific problem (that is, a specific botnet family) rather than being used as a catch-all solution [25], [32], [34]. More formally, let A be the number of considered machine learning algorithms (i.e., $A = 12$), let D be the number of involved datasets ($D = 4$), and let D_f^i the number of botnet families included in dataset D^i ($0 < i \leq D$); then, we devise a total of $C = A \cdot \sum_{i=1}^{D} D_f^i$ detectors (in our case, $C = 216$). We develop such a large amount of detectors in order to gauge how different machine learning classifiers, trained in different network environments, respond against adversarial attacks.

Each detector is trained, validated and evaluated individually. To train each classifier, we adopt sets of features used by related literature on flow-based classifiers [16], [27], [35]. We use 80% of the available malicious samples of each individual botnet family for training, and the remaining 20% for testing; benign samples are randomly chosen to compose sets with a benign-to-malicious samples ratio of 90:10. After performing multiple grid-search operations to determine its optimal configuration settings, each classifier is validated through 3-fold cross validation. We measure the performance of each detector through the *Precision*, *Detection Rate* (DR, or Recall) and *F1-score*, which are computed as follows:

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

$$F1\text{-}score = 2 * \frac{Precision * DR}{Precision + DR}$$

where TP (FP) denotes true (false) positives, and FN denotes false negatives. We consider a positive to be a malicious sample. Those detectors that obtain a score lower than 0.9 for any of these metrics are discarded, as such values are inadequate for NIDS deployed in real contexts.

B. Devising realistic adversarial attacks

To reproduce and gauge the effects of the adversarial attacks described in Section III-B4, we generate datasets of adversarial samples with the following procedure. For each detector (which is related to a specific botnet family in a given dataset), we consider all its corresponding malicious samples and randomly modify the values of

[3]CTU-13 Dataset: https://www.stratosphereips.org/datasets-ctu13
[4]IDS2017: https://www.unb.ca/cic/datasets/ids-2017.html
[5]CIC-IDS2018: https://www.unb.ca/cic/datasets/ids-2018.html
[6]Botnet dataset: https://www.unb.ca/cic/datasets/botnet.html
[7]Argus software: https://qosient.com/argus/argusnetflow.shtml

up to 4 different features: *flow_duration*, *sent_bytes*, *received_bytes*, *exchanged_packets*; the alterations involve small increments of [1..120] for *flow_duration* (in seconds), and [1..1024] for the remaining features. To maintain consistency, we also update the corresponding derived features (e.g.: *bytes_per_second*). The obtained adversarial datasets (which include only the manipulated malicious samples) are then used to test each classifier. The effectiveness of these attacks is measured through the following *Attack Severity* (AS) score:

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

Higher (lower) values of AS imply attacks in which greater (lower) amounts of adversarial samples have evaded detection.

C. Countermeasure: feature removal

A possible strategy to counter adversarial attacks is to nullify the effects of modifications applied to a given feature by excluding it. Hence, to defend against a wide ragne of adversarial attacks a defender could ignore all the features that can be easily manipulated by the attacker [14] without modifying the logic of the botnet family. However, this procedure might have detrimental effects on the performance of the detectors in non-adversarial settings, for example by triggering more false alarms. We evaluate the quality of this countermeasure by comparing the performance of each classifier in the absence of attacks before and after the application of this technique. That is, we re-train and re-test each classifier with feature sets that do not include the *flow_duration*, *sent_bytes*, *received_bytes*, *exchanged_packets*, as well as all the corresponding derived features.

V. Experimental Results

The experimental evaluation has a threefold objective:

1) confirm that the proposed detectors exhibit appreciable performance in non-adversarial settings;
2) evaluate the impact of our evasion attacks by testing these detectors against the adversarial samples;
3) assess the effectiveness of the considered countermeasure.

The following subsections present the results of our large experimental campaign by addressing each of these points.

A. Performance in non-adversarial settings

We begin by determining which detectors reach a performance that complies with real-world requirements. Thus, we train and test the 12 machine learning approaches considered in this paper against the malware families included in the 4 reference datasets, as described in Section IV-A.

Table II: Performance in non-adversarial settings.

Dataset	**F1-Score** (std. dev.)	**Precision** (std. dev.)	**Recall** (std. dev.)
CTU-13	0.957 (0.029)	0.958 (0.031)	0.956 (0.028)
IDS2017	0.996 (0.002)	0.999 (0.001)	0.993 (0.003)
CIC-IDS2018	0.999 (<0.001)	0.999 (<0.001)	0.999 (<0.001)
UNB-CA Botnet	0.991 (0.017)	0.992 (0.021)	0.991 (0.017)
Average	0.986 (0.011)	0.987 (0.012)	0.985 (0.011)

To avoid a negative bias caused by ML approaches that are not suitable for the detection of a given botnet family or that do not perform well on a given dataset, results of Table II only consider the subset of all possible detectors that achieve appreciable performance for the considered detection task. The inclusion criteria is for all performance metrics (F1-Score, Precision and Recall) to be equal or above 0.9. By applying this inclusion score we selected a total of 145 different detectors: 54 detectors for the CTU-13 dataset, 8 for IDS2017, 8 for CIC-IDS2018 and 75 for UNB-CA Botnet. We can observe that several (≈20%) detectors do not achieve suitable performance scores. This is motivated by the fact that some classifiers may not be appropriate for the given network environment.

Aggregated experimental results obtained by the 145 selected detectors are outlined in Table II.

In this table, rows indicate a specific dataset, while columns represent the value of F1-Score, Precision and Recall metrics. Each cell contains the average value of a given metric, together with its standard deviation enclosed in parentheses. We can see that the selected detectors achieve good detection performance, comparable to state-of-the-art solutions and consistent with results achieved in related work [16], [32], [33].

Besides average and standard deviation, we also provide a graphical depiction of the considered performance metrics through the boxplot diagrams shown in Figure 1, representing the distribution of the results for each dataset. This figure includes four different diagrams, one for each dataset. Each diagram shows three boxplots, representing the results for F1-Score, Precision and Recall, respectively. Boxplots show that the performance of all algorithms are comparable and consistently good across all datasets. Results are particularly promising for the CIC-IDS2018 dataset, where all the detectors achieve a F1-score that is very near to 1. We highlight that in

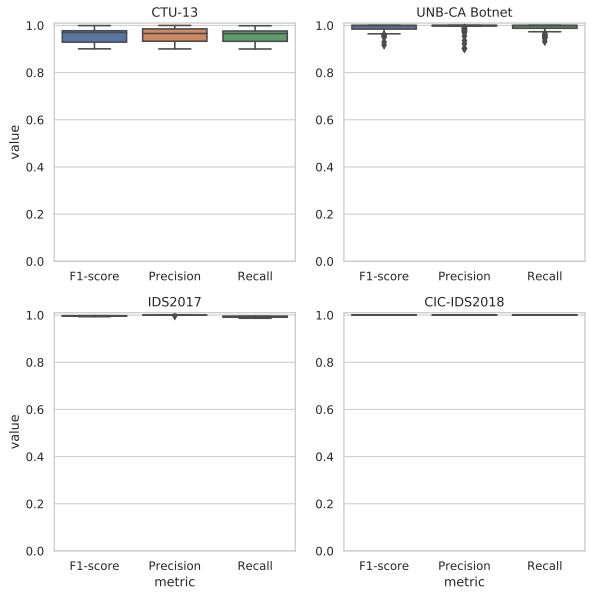

Fig. 1: Distribution of F1-Score, Precision and Recall for all detectors and all datasets.

previous work similar results led authors to conclude that ML-based detectors can be successfully applied to real network environments [32], [33], but that these previous work did not account for adversarial attacks.

The 145 detectors included in this evaluation represent our baseline for the subsequent experiments, and thus will be subject to the adversarial attacks and defenses based on feature removal.

B. Adversarial Attacks evaluation

We now evaluate the performance of our baseline detectors in the considered adversarial setting. We generate the adversarial datasets and test all the 145 detectors against adversarial attacks by following the procedure explained in Section IV-B. Aggregated experimental results are outlined in Table III. This table compares, for each dataset, the average (and standard deviation) Recall of the baseline detectors with the Recall obtained on the adversarial samples. We focus on the Recall metric since it reflects the number of malicious samples that the detector is able to identify. The last column of Table III shows the Attack Severity (see Equation 1), which expresses the effectiveness of adversarial attacks in reducing the Recall of a detector.

From Table III, it is clear that even the simple but realistic adversarial samples considered in our experimental evaluation manage to cause a significant drop in the Recall of ML-based cyber detectors. A more

Table III: Effects of the adversarial attacks.

Dataset	**Recall baseline** (std. dev)	**Recall adversarial** (std. dev)	**Attack Severity** (std. dev)
CTU-13	0.956 (0.028)	0.372 (0.112)	0.609 (0.110)
IDS2017	0.993 (0.003)	0.656 (0.102)	0.327 (0.103)
CIC-IDS2018	0.999 (< 0.001)	0.564 (0.112)	0.436 (0.112)
UNB-CA Botnet	0.991 (0.017)	0.588 (0.218)	0.328 (0.212)
Average	0.985 (0.011)	0.545 (0.136)	0.425 (0.134)

reflective comparison of the effects of adversarial perturbations on the Recall for all classifiers of each dataset is proposed in Figure 2. For each dataset, the first

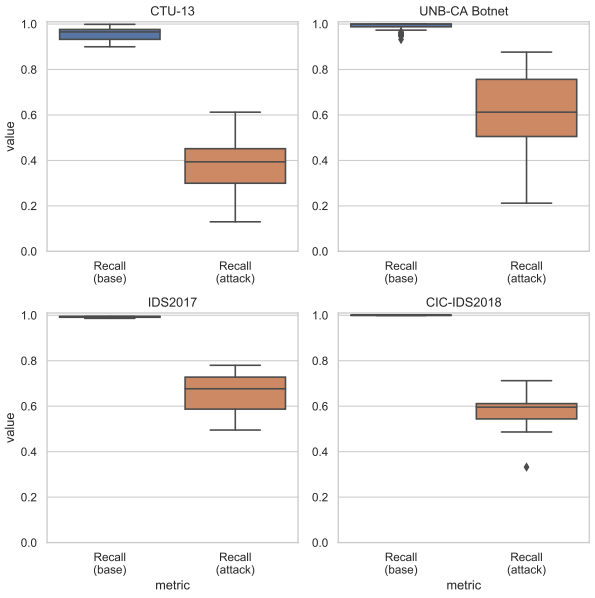

Fig. 2: Comparison between the distributions of the Recall metric in the non-adversarial (baseline) and adversarial (attack) scenarios for all datasets.

boxplot represents the distribution of the Recall across all detectors in non-adversarial settings, while the second boxplot shows results for the same metric in case of adversarial attacks. From these boxplots we can see that the detrimental effects of adversarial samples change significantly for different datasets and detectors. As an example, for the UNB-CA Botnet dataset the Recall of the more resilient ML-detector (based on the MLP algorithm) falls from 0.932 to 0.597, while the Recall for the most impacted ML-detector (based on SVM) for the same dataset falls from 0.914 to 0.198. The

Attack Severity is even worse for the CTU−13 dataset: in this case the Recall for the less affected ML-detection algorithm (based on RF) drops from 0.967 to 0.439. The effects of adversarial attacks against different datasets is summarized in Figure 3, which compares the Attack Severities for all detectors across all datasets.

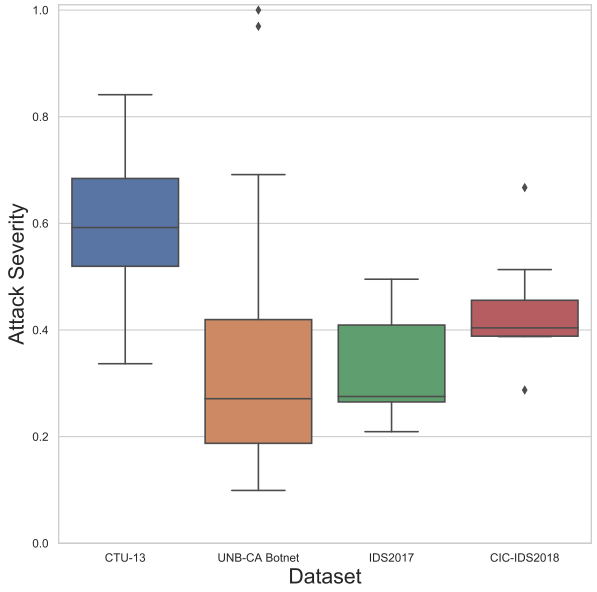

Fig. 3: Comparison of the distribution of Attack Severity for all the detectors among the 4 different datasets.

We remark that none of the tested 145 detectors that exhibited good performance in the non-adversarial settings is able to maintain F1-Score, Precision and Recall above 0.9 while analyzing adversarial samples. Hence despite good classification results achieved in previous papers, it is clear that adversarial samples represent a huge menace for real-world applications of ML-detectors to the problem of botnet detection.

C. Countermeasure effectiveness

Next, we assess the effectiveness of defensive strategies based on feature removal. To this purpose, we train the 145 ML-detectors by only considering features that are not affected by our adversarial attacks, as described in Section IV-C, and evaluate their F1-Score, Precision and Recall. Experimental results are summarized in Table IV.

By comparing Tables II and IV it is clear that feature removal causes a considerable reduction in the average detection performance. To also compare the distribution of F1-Score, Precision and Recall of the different detectors across all datasets we propose the boxplot diagrams of Figure 4, that can be directly compared with Figure 1.

Table IV: Detection results for ML detectors with feature removal.

Dataset	**F1-Score** (std. dev.)	**Precision** (std. dev.)	**Recall** (std. dev.)
CTU−13	0.803 (0.092)	0.810 (0.089)	0.799 (0.101)
IDS2017	0.503 (0.304)	0.777 (0.388)	0.596 (0.306)
CIC−IDS2018	0.859 (0.164)	0.814 (0.212)	0.942 (0.128)
UNB−CA Botnet	0.691 (0.276)	0.645 (0.285)	0.808 (0.209)
Average	0.714 (0.209)	0.761 (0.2235)	0.786 (0.186)

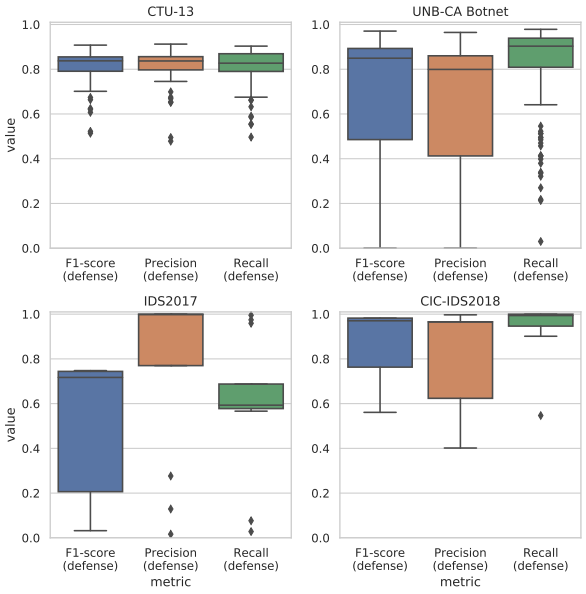

Fig. 4: Distribution of F1-Score, Precision and Recall using feature removal for all detectors and all datasets.

We can observe a significant decrease for all the considered metrics. In particular, we highlight that the lower Precision leads to a relevant number of false positives, which are extremely undesirable for modern cyber defence platforms. This reduction in quality is explained by the fact that the removed features have a meaningful impact to the underlying mechanisms of the baseline detectors; thus, their exclusion causes an important performance drop, with most detectors obtaining scores that are well below acceptable in real contexts. We can conclude that despite its ability to nullify the considered attack scenario, similar defensive strategies are inefficient and their application is discouraged in actual production environments. These results further motivate the need for novel defensive approaches that

are both effective and applicable to modern network scenarios.

We conclude this section by presenting the results obtained by the 5 best detectors for each dataset in non adversarial settings, which are provided in Tables V through VIII.

These tables show the average performance of each considered algorithm throughout all the steps performed in our experimental evaluation (baseline, attack and defense). The purpose of these tables is to highlight how even algorithms that achieve near-perfect results in non-adversarial settings can be heavily affected by such simple evasion mechanisms. Moreover, after the application of feature removal, only three algorithms (RF, AB and MLP) are able to achieve acceptable values of F1-Score, Precision and Recall, but only when applied to the `CIC-IDS2018` dataset.

VI. Conclusions

The application of machine learning algorithms to cybersecurity must face the problem posed by adversarial attacks. In this paper, we provide a thorough analysis of adversarial perturbations from a Network Intrusion Detection perspective. We evaluate the impact of realistic adversarial attacks against multiple state-of-the-art botnet detectors, based on different machine learning algorithms. Additionally, we study the effects of *feature removal*, which is a known defense against adversarial attacks. All experiments leverage several recent and public datasets of network flows, which represent realistic network environments. To the best of our knowledge, the present work is the first to analyze this threat by considering such a large amount of algorithms and datasets. The results highlight the high effectiveness of adversarial samples against all of the considered classifiers, and that feature removal is not a sufficient defense. Our contribution paves the way for the development of much needed defensive approaches that can improve the robustness of machine learning detectors in adversarial scenarios.

References

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[2] R. Ramprasad, R. Batra, G. Pilania, A. Mannodi-Kanakkithodi, and C. Kim, "Machine learning in materials informatics: recent applications and prospects," *NPJ Computational Materials*, vol. 3, no. 1, p. 54, 2017.

[3] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Sok: Security and privacy in machine learning," in *Proc. IEEE Europ. Symp. Secur. Privacy*, Apr. 2018, pp. 399–414.

[4] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Europ. Symp. Secur. Privacy*, Mar. 2016, pp. 372–387.

[5] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE T. Neur. Net. Lear.*, 2019.

[6] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, "On the effectiveness of machine and deep learning for cybersecurity," in *Proc. IEEE Int. Conf. Cyber Conflicts*, May 2018, pp. 371–390.

[7] H. Kettani and P. Wainwright, "On the top threats to cyber systems," in *Proc. IEEE Int. Conf. Inf. Comp. Tech.* IEEE, Mar. 2019, pp. 175–179.

[8] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. Leung, "A survey on security threats and defensive techniques of machine learning: A data driven view," *IEEE access*, vol. 6, pp. 12 103–12 117, Feb. 2018.

[9] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Elsevier Pattern Recogn.*, vol. 84, pp. 317–331, 2018.

[10] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Springer Joint Europ. Conf. Mach. Learn. and Knowl. Discov. Databases*, Sept. 2013, pp. 387–402.

[11] B. Biggio, G. Fumera, and F. Roli, "Pattern recognition systems under attack: Design issues and research challenges," *Int. J. Pattern Recogn. Artif. Intel.*, vol. 28, no. 07, p. 1460002, 2014.

[12] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar, "Adversarial machine learning," in *Proc. ACM Workshop Secur. and Artif. Intell.*, Oct. 2011, pp. 43–58.

[13] Z. Abaid, M. A. Kaafar, and S. Jha, "Quantifying the impact of adversarial evasion attacks on machine learning based android malware classifiers," in *Proc. IEEE Int. Symp. Netw. Comput. Appl.*, Oct. 2017, pp. 1–10.

[14] F. Zhang, P. P. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE T. Cybernetics*, vol. 46, no. 3, pp. 766–777, 2016.

[15] A. Demontis, P. Russu, B. Biggio, G. Fumera, and F. Roli, "On security and sparsity of linear classifiers for adversarial settings," in *Proc. Springer Joint. Int. Workshops Statist. Tech. Pattern Recognit. and Struct. Syntactic Pattern Recognit.*, Nov. 2016, pp. 322–332.

[16] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Addressing adversarial attacks against security systems based on machine learning," in *Proc. IEEE Int. Conf. Cyber Conflicts*, May 2019, pp. 1–18.

[17] I. Corona, G. Giacinto, and F. Roli, "Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues," *Elsevier Inform. Sciences*, vol. 239, pp. 201–225, 2013.

[18] J. Gardiner and S. Nagaraja, "On the security of machine learning in malware c&c detection: A survey," *ACM Comput. Surv.*, vol. 49, no. 3, p. 59, 2016.

[19] Z. Lin, Y. Shi, and Z. Xue, "Idsgan: Generative adversarial networks for attack generation against intrusion detection," *arXiv 1809.02077*, 2018.

[20] T. S. Sethi and M. Kantardzic, "Data driven exploratory attacks on black box classifiers in adversarial domains," *Elsevier Neurocomputing*, vol. 289, pp. 129–143, 2018.

[21] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. Springer Int. Conf. Inf. Syst. Secur. Privacy*, Jan. 2018, pp. 108–116.

[22] F. Kreuk, A. Barak, S. Aviv-Reuven, M. Baruch, B. Pinkas, and J. Keshet, "Deceiving end-to-end deep learning malware detectors using adversarial examples," *arXiv:1802.04528*, 2018.

[23] J. Clements, Y. Yang, A. Sharma, H. Hu, and Y. Lao, "Rallying adversarial techniques against deep learning for network security," *arXiv:1903.11688*, 2019.

[24] B. Kolosnjaji, A. Demontis, B. Biggio, D. Maiorca, G. Giacinto, C. Eckert, and F. Roli, "Adversarial malware binaries: Evading deep learning for malware detection in executables," in *Proc. IEEE Europ. Conf. Sign. Proc.*, Sep. 2018, pp. 533–537.

Table V: CTU-13 top5 algorithms results.

	Baseline			Attack		Defense		
Algorithm	*F1-score*	*Precision*	*Recall*	*Recall*	*Attack Severity*	*F1-score*	*Precision*	*Recall*
RF	0.9694	0.9722	0.9668	0.4390	0.5461	0.8564	0.8498	0.8641
AB	0.9722	0.9748	0.9696	0.4074	0.5803	0.8446	0.8487	0.8410
MLP	0.9458	0.9454	0.9462	0.3141	0.7261	0.7235	0.7734	0.6886
KNN	0.9296	0.9273	0.9320	0.2982	0.6806	0.6992	0.7265	0.6767
Bag	0.9745	0.9799	0.9693	0.4007	0.5869	0.8477	0.8516	0.8442

Table VI: IDS2017 top5 algorithms results.

	Baseline			Attack		Defense		
Algorithm	*F1-score*	*Precision*	*Recall*	*Recall*	*Attack Severity*	*F1-score*	*Precision*	*Recall*
AB	0.9972	1	0.9945	0.7455	0.2504	0.7172	0.9779	0.5663
MLP	0.9959	0.9972	0.9945	0.5991	0.3975	0.7169	0.9344	0.5816
KNN	0.9959	1	0.9918	0.5512	0.4442	0.4292	0.2764	0.9591
ET	0.9972	1	0.9945	0.7333	0.2626	0.7456	1	0.5943
GB	0.9945	1	0.9891	0.7221	0.2699	0.7476	1	0.5969

Table VII: CIC-IDS2018 top5 algorithms results.

	Baseline			Attack		Defense		
Algorithm	*F1-score*	*Precision*	*Recall*	*Recall*	*Attack Severity*	*F1-score*	*Precision*	*Recall*
RF	0.9999	0.9999	0.9999	0.5965	0.4034	0.9822	0.9653	0.9996
AB	0.9997	0.9999	0.9996	0.5632	0.4365	0.9709	0.9969	0.9463
MLP	0.9997	0.9999	0.9995	0.7123	0.2873	0.9696	0.9939	0.9465
KNN	0.9998	0.9999	0.9998	0.4866	0.5132	0.8225	0.7564	0.9012
ET	0.9999	0.9999	0.9999	0.6023	0.3976	0.9822	0.9653	0.9996

Table VIII: UNB-CA Botnet top5 algorithms results.

	Baseline			Attack		Defense		
Algorithm	*F1-score*	*Precision*	*Recall*	*Recall*	*Attack Severity*	*F1-score*	*Precision*	*Recall*
RF	0.9974	0.9997	0.9951	0.6856	0.3110	0.8912	0.8584	0.9283
KNN	0.9496	0.9479	0.9516	0.6167	0.3507	0.8144	0.7555	0.8871
ET	0.9993	0.9999	0.9987	0.6831	0.3160	0.8897	0.8544	0.9294
MLP	0.9215	0.9113	0.9321	0.5978	0.2756	0.7393	0.6779	0.8325
AB	0.9955	0.9971	0.9939	0.6840	0.3118	0.8926	0.8595	0.9303

[25] G. Apruzzese and M. Colajanni, "Evading botnet detectors based on flows and random forest with adversarial samples," in *Proc. IEEE Int. Symp. Netw. Comput. Appl.*, Oct. 2018, pp. 1–8.

[26] G. Apruzzese, M. Marchetti, M. Colajanni, G. G. Zoccoli, and A. Guido, "Identifying malicious hosts involved in periodic communications," in *Proc. IEEE Int. Symp. Netw. Comput. Appl.*, Oct. 2017, pp. 1–8.

[27] M. Stevanovic and J. M. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in *IEEE Int. Conf. Comput., Netw. and Commun.*, Feb. 2014, pp. 797–801.

[28] G. Apruzzese, F. Pierazzi, M. Colajanni, and M. Marchetti, "Detection and threat prioritization of pivoting attacks in large networks," *IEEE T. Emerg. Top. Com.*, 2017.

[29] F. Pierazzi, G. Apruzzese, M. Colajanni, A. Guido, and M. Marchetti, "Scalable architecture for online prioritisation of cyber threats," in *Proc. IEEE Int. Conf. Cyber Conflicts*, May 2017, pp. 1–18.

[30] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Elsevier Comp. Secur.*, vol. 45, pp. 100–123, 2014.

[31] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *Proc. IEEE Conf. Comm. Netw. Secur.*, Oct. 2014.

[32] B. Abraham, A. Mandya, R. Bapat, F. Alali, D. E. Brown, and M. Veeraraghavan, "A comparison of machine learning approaches to detect botnet traffic," in *Proc. IEEE Int. Joint Conf. Neur. Netw.*, Jul. 2018, pp. 1–8.

[33] M. Stevanovic and J. M. Pedersen, "An analysis of network traffic classification for botnet detection," in *Proc. IEEE Int. Conf. Cyber Situat. Awar., Data Analyt., Assessment*, Jun. 2015, pp. 1–8.

[34] F. V. Alejandre, N. C. Cortés, and E. A. Anaya, "Feature selection to detect botnets using machine learning algorithms," in *Proc. IEEE Int. Conf. Elect. Commun. Comp.*, Feb. 2017, pp. 1–7.

[35] A. Pektaş and T. Acarman, "Deep learning to detect botnet via network flow summaries," *Springer Neural Comput. Appl.*, pp. 1–13, 2018.