Unleashing the Beast: Evaluating Adversarial Vulnerability of AI-driven Intrusion Detection

Hélène Orsini
CIDRE team, IRISA
Inria, CentraleSupelec, CNRS, U. Rennes
Rennes, France
helene.orsini@irisa.fr

Yufei Han
CIDRE team, IRISA
Inria, CentraleSupelec, CNRS, U. Rennes
Rennes, France
yufei.han@inria.fr

Valérie Viet Triem Tong
CIDRE team, IRISA
Inria, CentraleSupelec, CNRS, U. Rennes
Rennes, France
valerie.viettriemtong@centralesupelec.fr

***Abstract*—Machine Learning (ML)-based Intrusion detection systems (IDS) aim at avoiding, preventing, and informing about malicious intrusion into a system using Machine Learning-based techniques. Accurate as they are, recent studies show the dark side of the story: ML techniques can be prone to slightly perturbing the inputs, a.k.a. adversarial attacks. As a security-critical application, such an adversarial vulnerability raises reliability and trustworthiness concerns over deploying ML-based IDS in practice. Our work proposes to assess the adversarial robustness of ML-based IDS methods using a computationally efficient robustness evaluation protocol. Our results over a state-of-the-art ML-based IDS method and real-world log-based anomaly detection data demonstrate the existence of an adversarial vulnerability in the ML-based IDS approach. We show that the claimed high detection rate over adversarial noise-free cross-validation test may not be suitable to measure the practical usability of ML-based IDS methods.**

***Index Terms*—Intrusion Detection, Robustness, Adversarial Attack**

I. Introduction

Recent studies have witnessed the blossom of Machine Learning-driven intrusion detection systems (ML-IDS). Effective as a data-driven IDS solution, ML-IDS also inherits the adversarial vulnerability from ML techniques [1]. Slightly perturbing inputs to ML-IDS may drastically change the detection results. For security-critical applications, e.g., IDS, the adversarial vulnerability of ML methods raises concerns over the trustworthiness of ML-driven prediction/detection results.

In our study, we propose to assess the adversarial risk of ML-based IDS approaches by extending the adversarial robustness evaluation framework originally established in our previous work *AdvCat* [2]. Our motivation is two-fold. **First of all**, *AdvCat* is designed to handle both continuous and discrete inputs, which well suits the IDS applications taking discrete features, e.g., network logs and security event signatures, as input. Residing at the core of *AdvCat*, it adopts an "assessment-by-attack" strategy to craft inputs to mislead the prediction output of ML models using computationally efficient heuristic search approaches. **Secondly**, *AdvCat* only requires black-box access to the target ML-based IDS and organizes computational-efficient adversarial attacks. Our work proposes to assess the adversarial robustness of ML-based IDS methods following an "assess-by-attack" strategy. We introduce *AdvCat* as an adversary against ML-based IDS. In this attack scenario, the adversary can only access the logs and/or security event reports in the database where the ML-based IDS model is deployed for detecting anomalies (testing-time attack). **Instead of hiding his traces of malicious activities from the database and evading detection, we assume the adversary aims to insert a few new logs into the normal behavior logs dataset**. The adversary's goal is to trigger false alarms of the target IDS as much as possible, destroying its utility. Given the attack setting, an ML-based IDS is more robust if it can tolerate more inserted logs without significantly increasing false alarms.

Our main contributions can be summarized as below:

- We instantiate how to assess the adversarial robustness of ML-based IDS via adopting the *AdvCat*-based robustness assessment. We choose to evaluate *DeepLog*, a Deep Learning-based anomaly detection method used for intrusion detection [3]. Specifically, we tailor *AdvCat* to only insert a few system logs to the input of *DeepLog* to measure whether *DeepLog* can preserve its detection rate and keep the false alarm rate low for practical use.
- We organize a systematic and comprehensive measurement study using one public and another privately owned system log dataset. Our empirical observations show the adversarial vulnerability in the state-of-the-art intrusion detection model, despite the claimed high detection accuracy of *DeepLog*. By inserting no more than two well-chosen logs into the input of *DeepLog* with *AdvCat*, the false alarm rate of *DeepLog* increases by 30-50 times, which destroys the utility of *DeepLog* completely. It raises a trustworthiness concern over the use of AI-based IDS models in practices.

II. Related work

In [4], [5], Szegedy *et al.* demonstrated the adversarial vulnerabilities of ML models, notably deep neural network models. The attacker may hold white-box, grey-box, or black-box knowledge about the victim classifier depending on whether he has complete, partial, or no access to the parameters/architectures of the victim classifier [6]. In our study, we focus on adversarial attacks with black-box knowledge of the target ML model. Despite previous works of adversarial

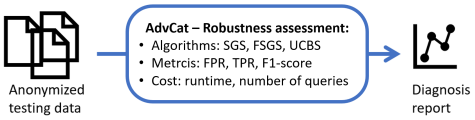

Fig. 1. Workflow of Robustness Evaluation using *AdvCat*

attacks, attacking ML-based IDS has two unaddressed bottlenecks. First, data in IDS applications usually contain both numerical and discrete features. For example, port numbers, IP protocols, and the availability of a specific network are all categorical attributes profiling network attack behaviors. Performing adversarial attacks on discrete data is, in nature, a mixed-integer non-linear programming problem, which requires solving an NP-hard combinatorial search task [7]. Second, the attackers usually have black-box access to the ML model deployed in a target IDS. He rarely accesses the architecture or parameter values of the ML model. According to [6], Such black-box attacks can only adjust modifications introduced to the testing inputs by observing the output of the target ML model. Designing feasible adversarial perturbation is a challenging task. AdvCat [2] proposes evaluating an ML model's robustness. It considers an ML model within black-box settings by modifying categorical data at the testing time. So far as known, AdvCat is the first research effort in quantitatively measuring the adversarial risk of ML models with discrete inputs and under the black-box setting.

III. Workflow

A. DeepLog as the target system

DeepLog adopts a Recurrent-Neural-Network (RNN)-based detection model for anomaly detection. It is applied over time series of system logs produced by an IT architecture. The RNN-based model learns the logs' sequential patterns corresponding to the IT architecture's normal behaviors. More specifically, given the previous 10 logs of normal behaviors $\{l_{t-10}, l_{t-9}, ..., l_{t-1}\}$, *DeepLog* predicts the next log at l_t. For a given log sequence containing 11 logs, if *DeepLog* 's prediction of the 11th log is not consistent with the observed one, the whole log sequence will be tagged as an abnormal sequence. *DeepLog* adopts the top-K prediction scheme. If the observed log appears in the top K predicted logs with the highest confidence scores, the corresponding log sequence is considered normal, and vice versa.

B. Threat model for robustness evaluation

We apply an "assess-by-attack" scheme in the robustness evaluation step in *AdvCat*. We assume an adversary launches evasion attacks against *DeepLog*. The outcome of the evasion attacks, e.g. the damage brought to the prediction accuracy of *DeepLog* and the computational cost of attacks, are used to measure the robustness of *DeepLog* quantitatively.

Adversary's knowledge: The adversary in *AdvCat* has only black-box access to the target AI-based IDS model. It means the adversary doesn't know either the parameter values or the architecture of the target IDS model. Furthermore, we assume the adversary can not access the training dataset nor interfere with the model's training process. However, the adversary knows all the unique logs that may appear in the training dataset of the target IDS model.

Adversary's capabilities: The adversary can insert new logs into the sessions of logs corresponding to normal activities in order to trigger false alarms. We assume the adversary only inserts a few logs into the set of logs, e.g., 1-2 logs.

Adversary's objective: The aim is to increase the false alarm rate of *DeepLog* model significantly. A high number of false alarms allows the attacker to hide and be longer to detect. And, It discredits the model.

C. Robustness evaluation framework

Figure.1 shows the workflow of robustness evaluation using *AdvCat*. *AdvCat* crafts input log sequences and submits these crafted inputs to *DeepLog*. Based on the returned prediction confidence from *DeepLog*, *AdvCat* can adjust its modification over the input log sequence or decide to stop attacks if the goal of attacks is reached. The loss of anomaly detection's utility after modifying the input log sequence is used to measure the adversarial robustness of *DeepLog*.

AdvCat crafts adversarial examples by finding two parameters: the position of the modification and the modification itself. It uses two different types of algorithms to do it. The first type is greedy algorithms with two implementations: Forward Stepwise Greedy Search (FSGS) and Stochastic Greedy Search (SGS). The former algorithm explores all the possibilities for both parameters. The latter doesn't iterate to find a candidate but chooses a set of candidates randomly, which is lower in time complexity. The other type is a multi-arm bandit exploration method named Upper Confidence Bound Search (UCBS). It uses Upper Confidence Bound (UCB) to guide the exploration in the combinatorial space of adversarial perturbations.

We evaluate the effectiveness of attacks using the following three metrics to quantitatively measure the adversarial robustness of the target IDS method. The larger the differences are, the more effective the attack is, and the more vulnerable the target IDS method is.

FPR/$\mathbf{D_F}$: The false positive rate (FPR) of anomaly detection using *DeepLog* over testing sessions after the attack and the difference of FPR before and after the attack.

TPR/$\mathbf{D_T}$: The true positive rate (TPR) of anomaly detection using *DeepLog* over testing sessions after the attack and the difference of TPR between before and after the attack.

F1/$\mathbf{D_{F1}}$: The F1-score of *DeepLog*'s detection after attack and the difference of F1-score between before and after the attack.

We also record the averaged query cost ($\underline{\text{AvgQ}}^{\text{q}}$) and the runtime cost ($\underline{\text{AvgR}}^{\text{r}}$) spent by each robustness assessment algorithm to attack *DeepLog*.

IV. Experiment

A. Experimental set up

Dataset. We run the evaluation on Deeplog with AdvCat on the HDFS dataset [3]. HDFS contains Hadoop-based execution

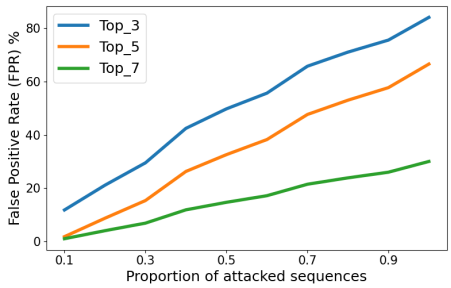

Fig. 2. Evolution of FPR in function of proportion of attacked sequence

sessions over 200 Amazon Elastic Computer Cloud nodes. The HDFS dataset comprises 4855 normal training sessions, 553366 normal testing sessions, and 15200 abnormal testing sessions. Each session contains up to 300 logs.

Detection. We conduct anomaly detection using *DeepLog* at the session level. We scan each session using a sliding window of 11 logs and perform *DeepLog* in each sliding window. If at least one sliding window is detected as abnormal, the whole session is considered to be abnormal. For each attacked sliding window, we constraint to inject at most two new logs. In our test, we randomly select 40% of the sliding windows for each session of logs to perform the log-inserting attack. Among these selected sliding windows, if one of the sliding windows is tagged as abnormal, we stop the attack and tag the whole session as abnormal. We adopt the top-K prediction of *DeepLog* (K=3,5,9).

TABLE I
ROBUSTNESS ASSESSMENT OF DEEPLOG ON HDFS DATASET

Attack	(%), Top-5, 40% of attacked sequences			Cost of attacks	
	FPR(D_F)	TPR(D_T)	F1(D_{F1})	AvgQ[q]	AvgR[r]
SGS	15.2(↑ 14.8)	99.8(0.0)	21.4(↓ 69.9)	324.2	0.416
FSGS	20.0(↑ 19.6)	99.8(0.0)	18.1(↓ 73.6)	569.7	0.578
UCB	36.7(↑ 36.3)	99.8(0.0)	90.5(↓ 81.0)	68.5	0.01

[q]Average query cost. [r]Average runtime cost

Table.I reports the three performance metrics of *DeepLog* with the top-5 prediction mode after attack and **the difference of FPR, TPR, and F1 between before and after the attack** (noted as D_F,D_T and D_{F1}) when it is exposed to the three attacks of *AdvCat*. We observe FPR of *DeepLog* becomes **30-50 times larger** after the attack. In parallel, F1-score of *DeepLog* shrinks to almost **1/5** of that without attacks. The TPR of *DeepLog* remains barely changed. These observations echo the attacks' goal, which significantly increases the false alarms of *DeepLog* by perturbing logs of normal sessions. As unveiled, *DeepLog* becomes utterly unusable after the attacks. In Figure.2, we show the variation of FPR of *DeepLog*'s detection with increasingly more sliding windows per session selected to attack. We increase the fraction of the selected windows from 10% to 100%. The y-axis shows the FPR values of top-3,5 and 7 prediction modes of *DeepLog*. As seen, the FPR of all three prediction modes increases quickly to over 10% even though only less than 30% of the sliding windows are selected to attack. If the attacker can arbitrarily choose any windows to attack in a session (the fraction of the selected window to be 100%), the FPR of *DeepLog* can reach over 60% and 80% for the top-3 and 5 mode, which is surprisingly high. Another interesting observation is the top-K prediction with a larger K is more robust to the attacks, as it offers a more stable decision by design.

We observe that the most efficient (for both number queries and runtime) algorithm is the UCB algorithm. Regarding the greedy algorithms, SGS implementation performs better.

One example demonstrating the attack over the HDFS system logs is given as below. For the session of receiving and storing two successive blocks of data, the produced logs are "Received blocks" - "Packet response for blocks" - "Received blocks" - "Packet response for blocks" - "Add stored blocks". The first 4 logs denote data transferring and confirmation of receiving the data. The last log indicates the block map of HDFS is updated to include the newly received blocks. The attack inserts "Add stored blocks" between the second and third logs, which interrupts the normal sequential pattern of logs and triggers a false alarm of *DeepLog*.

V. CONCLUSION AND DISCUSSION

We measure the adversarial robustness of a state-of-the-art ML-driven and system log-based intrusion detection method, namely *DeepLog*. We show injecting 1-2 new logs into the input log sequence of *DeepLog* can drastically increase the false alarm rates of *DeepLog*, which indicate a thoughtful adversarial vulnerability in ML-driven IDS applications. Our future study will extend the measurement study over more ML-based intrusion detection methods and different attack data. Our current study doesn't consider the problem space constraints of system logs, i.e., the injected logs may not appear in real attacks. We will focus on integrating the problem space constraints into adversarial perturbation on ML-based intrusion detection methods.

REFERENCES

[1] Alatwi, H. Ali, and C. Morisset. "Adversarial Machine Learning In Network Intrusion Detection Domain: A Systematic Review". arXiv, 6 décembre 2021. http://arxiv.org/abs/2112.03315.

[2] H. Orsini, H. Bao, Y. Zhou, X. Xu, Y. Han, L. Yi, W. Wang, X. Gao, and X. Zhang. "AdvCat: Domain-Agnostic Robustness Assessment for Cybersecurity-Critical Applications with Categorical Inputs". arXiv, 13 décembre 2022. http://arxiv.org/abs/2212.13989.

[3] Du, Min, F. Li, G. Zheng, and V. Srikumar. "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning". In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 1285-98. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017. https://doi.org/10.1145/3133956.3134015.

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. "Intriguing properties of neural networks". arXiv, 19 février 2014. http://arxiv.org/abs/1312.6199.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy. "Explaining and Harnessing Adversarial Examples". arXiv, 20 mars 2015. http://arxiv.org/abs/1412.6572.

[6] Brendel, Wieland, J. Rauber, and M. Bethge. "Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models". arXiv, 16 février 2018. http://arxiv.org/abs/1712.04248.

[7] Y. Wang, Y. Han, H. Bao, Y. Shen, F. Ma, J. Li, and X. Zhang. "Attackability Characterization of Adversarial Evasion Attack on Discrete Data". In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '20), pp.1415–1425.