amsqr at MLSEC-2021: Thwarting Adversarial Malware Evasion with a Defense-in-Depth

Alejandro Mosquera

Abstract

This paper describes the author's participation in the 3rd edition of the Machine Learning Security Evasion Competition (MLSEC-2021) sponsored by CUJO AI, VM-Ray, MRG-Effitas, Nvidia and Microsoft. As in the previous year the goal was not only developing measures against adversarial attacks on a predefined set of malware samples but also finding ways of bypassing other teams' defenses in a simulated cloud environment. The submitted solutions were ranked second in both defender and attacker tracks.

1 Disclaimer

The task of bypassing malware classifiers and defending against adversarially modified samples was taken as a personal challenge during my free time for learning purposes. Therefore I have not used any resources, tools, infrastructure, know-how or manpower from my employer, being all the the code and datasets referenced in this paper either open source or public domain. Attacks were only performed against the models provided by the contestants with an academic purpose and no commercial security software was used nor evaluated as part of this challenge.

2 Introduction

The adoption of artificial intelligence in cybersecurity is not without risks. Researches have many times demonstrated that adversarial attacks can successfully evade machine learning (ML) defenses. The Machine Learning Security Evasion Competition addresses countermeasures to adversarial behavior by raising awareness about the variety of ways ML systems may be evaded by malware and better defend against these techniques. There have been 3 editions so far in the last three years, including a similar event held at DEF CON 27, which shows that this is still a relevant and active research topic far from being considered solved. The challenge consisted of two different tracks focused on malware detection: On the one hand, contestants had the possibility of submitting a Docker image able of detecting malicious binaries while at the same time passing certain size, runtime, FN and FP requirements. On the other hand, contestants were encouraged to modify a set of 50 Portable Executable (PE) malware files in order to bypass the models submitted in the previous track with the following restrictions: Files had to be smaller than 2MB and droppers and self-extracting files were disallowed. Likewise, all the modifications had to be functionality-preserving which means that the malware should have exactly the same execution traces after modifications as evaluated by the organizer's execution sandbox. One important aspect is that while queries against defender models were not limited, there was one hour waiting time between sandbox submissions.

3 Defender track

Cyber-security researchers proved in previous editions of this challenge that all the ML defenses can be eventually bypassed (Ceschin et al., 2020), therefore a diverse defensive approach focused on slowing down certain attack paths by forcing attackers to produce extra model queries and sandbox submissions looked a-priori promising. This shares some similarities with the military concept of defense in-depth, that seeks to delay rather to prevent the advance of an attacker by placing different layers of defense instead of concentrating them in just one place. The submitted defender evaluated sequentially the following components:

3.1 Stateful defense

Most successful adversarial perturbations target PE file regions that usually do not affect functionality such as appending bytes to the overlay or adding new imports or sections (Demetrio et al., 2021a).

The stateful defense layer detects such attempts by calculating TSLH (Oliver et al., 2013) fingerprints of malware previously detected by the ML layers and comparing these against incoming samples. Rather than considering the whole executable, the fingerprint is only calculated on bytes of the PE file which are a priori harder to modify without breaking its functionality, such as the content of the first section.

3.2 EMBER baseline model

A gradient boosting classifier trained on EMBER (Anderson and Roth, 2018) dataset.

3.3 Rule-based anomaly detection

This layer considers as anomalies certain characteristics that malicious and adversarial PE files have in common but that differentiates them from most of clean files. The anomalies targeted here are as follows:

- Non-standard DOS stub.
- Duplicate PE sections (Quiring et al., 2020).
- Extra bytes between sections (Quiring et al., 2020).
- Extra bytes in overlay (Quiring et al., 2020).
- Wrong optional header CRC.
- PE EIP points to last section.
- PE with too many sections (> 20).
- PE is packed [1]
- PE contains obfuscated function names [2]
- PE contains base64 or hex-encoded executable.

3.4 EMBER-based EIP models

Two separate gradient boosting models trained on EMBER 2017 and 2018 datasets respectively with features extracted from the EIP section.

3.5 SOREL-based model

A gradient boosting classifier trained on SOREL (Harang and Rudd, 2020) dataset.

3.6 Non-deterministic SOREL-based model

The use of non-deterministic models has been proven a sucessful strategy against a variety of white-box and black-box attacks (Khan et al., 2019). For this reason, the last layer of defense consisted on a randomized ensemble of neural networks trained on SOREL (Harang and Rudd, 2020) dataset.

4 Attacker track

There were 6 defender systems unknown to the participants, however considering the fact that there are not many publicly available large-scale malware datasets besides EMBER and SOREL, attacks that would successful evade baseline models trained on these should have in principle better chances of evasion, or the very least least a better starting point than targeting the defenders directly. Therefore the chosen strategy was performing a mix of white-box and black-box attacks against EMBER and SOREL baselines while also extracting knowledge from the evaded defender systems in order to fine-tune the process. For most attacks universal adversarial perturbations (UAPs) (Labaca-Castro et al., 2021) were prioritized over sample-specific perturbations. The list of performed attacks are listed below:

4.1 Counterfit-based

Counterfit [3] was leveraged in order to perform optimized black-box adversarial attacks through Bayesian optimization (Shukla et al., 2019). The list of functionality-preserving PE modifications are as follows:

- Adding a new section with content from clean executables.
- Adding random imports.
- Appending strings from clean executables to overlay.
- Setting random PE header timestamp.
- Packing the executable with UPX.
- Unpacking an UPX-packed executable.
- Removing signature from certificate table.
- Signing the executable with a custom certificate.

[1] https://github.com/Yara_Rules/rules/blob/master/packers/packer.yar
[2] https://github.com/JusticeRage/Manalyze/blob/master/bin/yara_rules/suspicious_strings.yara
[3] https://github.com/Azure/counterfit

- Removing Rich header.
- Removing debugging information.
- Setting random optional header checksum.
- Adding a new section containing a code trampoline that redirects to the original entry point.

4.2 Greedy byte-based

Black-box attacks were performed by either appending random bytes to the overlay or to the DOS stub (Demetrio et al., 2021b), increasing the file size.

4.3 Packer-based

Custom PE packers for both .NET and x86 executables were used in order to execute the original code by applying either reflective loading or process hollowing techniques. While in-memory execution was allowed by the organizers, any approach that drops and executes the unmodified malware would have been disqualified.

4.4 Fuzzing-based

By fuzzing the PE header it was possible to discover that the modification of several header fields ignored by the Windows loader [4] would substantially impact SOREL and EMBER baseline classifiers. One of the most relevant was SizeOfCode, that while not being conclusive on its own, in combination with other perturbations would highly increase evasion chances. The inclusion of features derived from ignored/unused PE header fields makes the EMBER feature set (shared by both EMBER and SOREL datasets) particularly vulnerable to this type of attacks.

4.5 Explanation-based

Finally, in order to evade EMBER-based defenders and the submitted defender model described before, explanation-guided (Amich and Eshete, 2021) white-box attacks using both SHAP (Lundberg and Lee, 2017) and feature importance were used.

5 Results

The results at the end of the challenge were as follows: For the defender track, the proposed defense in-depth "A1" obtained the second best lowest number of conceded evasions (193), just 31 more than the winning system "secret" and 38 less than the third ranked team "kipple". In the attacker track, despite leading the classification for most of the competition, "amsqr" ended up dropping to the second place in the last week with 167 achieved evasions, 29 less than the best performing team "secret" and 52 more than the third best competitor "rwchsfde". When considering the number of model queries, there were substantial differences between the winning approach and the rest, "secret" only needed 600 queries while it took 3004 and 55701 to "amsqr" and "rwchsfde" respectively. The main limitation of the described attack workflow was the lack of a local execution sandbox, which caused that many unnecessary model queries had to be spent when uploading files to the sandbox provided by the challenge organizers, with the subsequent delay between submissions and results.

[4] https://web.archive.org/web/20110705011227/http://www.phreedom.org/solar/code/tinype/

Acknowledgments

The author thanks the 2021 Machine Learning Security Evasion Competition (MLSEC-2021) organizers, with special mention of Zoltan Balazs and Hyrum Anderson, and their sponsors (CUJO AI, VM-Ray, MRG-Effitas, Nvidia and Microsoft) for the opportunity to participate and raising awareness about adversarial attacks against ML-based malware detectors.

References

Abderrahmen Amich and Birhanu Eshete. 2021. Explanation-guided diagnosis of machine learning evasion attacks.

Hyrum S. Anderson and Phil Roth. 2018. Ember: An open dataset for training static pe malware machine learning models.

Fabricio Ceschin, Marcus Botacin, Gabriel Lüders, Heitor Murilo Gomes, Luiz Oliveira, and Andre Gregio. 2020. No need to teach new tricks to old malware: Winning an evasion challenge with xor-based adversarial samples. In *Reversing and Offensive-Oriented Trends Symposium*, ROOTS'20, page 13–22, New York, NY, USA. Association for Computing Machinery.

Luca Demetrio, Battista Biggio, Giovanni Lagorio, Fabio Roli, and Alessandro Armando. 2021a. Functionality-preserving black-box optimization of adversarial windows malware. *IEEE Transactions on Information Forensics and Security*, 16:3469–3478.

Luca Demetrio, Scott E. Coull, Battista Biggio, Giovanni Lagorio, Alessandro Armando, and Fabio Roli. 2021b. Adversarial exemples. *ACM Transactions on Privacy and Security*, 24(4):1–31.

Richard Harang and Ethan M. Rudd. 2020. Sorel-20m: A large scale benchmark dataset for malicious pe detection.

Daanish Ali Khan, Linhong Li, Ninghao Sha, Zhuoran Liu, Abelino Jimenez, Bhiksha Raj, and Rita Singh. 2019. Non-determinism in neural networks for adversarial robustness.

Raphael Labaca-Castro, Luis Muñoz-González, Feargus Pendlebury, Gabi Dreo Rodosek, Fabio Pierazzi, and Lorenzo Cavallaro. 2021. Universal adversarial perturbations for malware.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Jonathan Oliver, Chun Cheng, and Yanggui Chen. 2013. Tlsh – a locality sensitive hash. In *2013 Fourth Cybercrime and Trustworthy Computing Workshop*, pages 7–13.

Erwin Quiring, Lukas Pirch, Michael Reimsbach, Daniel Arp, and Konrad Rieck. 2020. Against all odds: Winning the defense challenge in an evasion competition with diversification.

Satya Narayan Shukla, Anit Kumar Sahu, Devin Willmott, and J. Zico Kolter. 2019. Black-box adversarial attacks with bayesian optimization.