2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S) | 978-1-6654-3566-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/DSN-S52858.2021.00012

Systemically Evaluating the Robustness of ML-based IoT Malware Detectors

Ahmed Abusnaina†, Afsah Anwar†, Sultan Alshamrani†, Abdulrahman Alabduljabbar†,
Rhongho Jang◇, DaeHun Nyang‡, and David Mohaisen†

†University of Central Florida, ◇ Wayne State University, ‡ Ewha Women University.
ahmed.abusnaina, afsahanwar, salshamrani, jabbar @knights.ucf.edu;
r.jang@wayne.edu; nyang@ewha.ac.kr; mohaisen@ucf.edu

***Abstract*—The rapid growth of the Internet of Things (IoT) devices is paralleled by them being on the front-line of malicious attacks caused by malicious software. Machine learning (ML) algorithms, alongside the traditional signature-based methods, are typically used to detect malicious activities and behaviors. However, they are susceptible to malware evolution and sophistication, making them limited to the patterns that they have been trained upon. In this work, we systematically examine the state-of-the-art malware detection approaches using various representations, under a range of adversarial settings. Our preliminary analyses highlight the instability of the learning algorithms in learning patterns that distinguish the benign from the malicious. Our mutations with functionality-preserving operations, e.g., software stripping and binary padding, significantly deteriorate the accuracy of malware detectors.**

***Index Terms*—Internet of Things, Adversarial Machine Learning, Malware Detection**

I. Introduction

IoT devices are susceptible to vulnerabilities that have been disclosed decades ago, making them an easy target to well-known threats, e.g., Brickerbot botnets, and DDoS attacks [1]. To address those threats, the research community and the industry alike have proposed various detection systems. The underlying principles of those approaches vary from the traditional signature-based to more advanced Artificial Intelligent (AI)-based detectors. In a range of those detection systems, researchers relied on approaches that leverage static and dynamic analysis along with state-of-the-art techniques from the deep learning and natural language processing communities to efficiently generalize to new and unknown IoT malware samples responsible for those threats [2].

Considering that these techniques are heavily dependent on the data used for their training and testing, it is expected that they would have reduced reliability when tested in the wild. Moreover, the constant evolution of malware that employs obfuscation will further impact the performance of these static-based detectors over time. For example, packed software samples are known to be categorized as malicious by the industry-standard malware detectors [3]. Prior to having a practical detection system, it is essential to understand the shortcomings of state-of-the-art IoT malware detection systems under adversarial settings.

In this work, we examine state-of-the-art ML-based malware detection approaches. We consider techniques that represent the software as a sequence of binary, static features extracted from the disassembly, and control flow graphs. As shown in the prior work, these representations yield a promising detection performance, with higher than 99% detection accuracy.

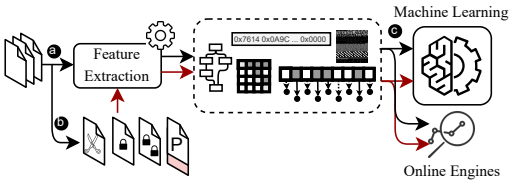

Fig. 1. The overall system pipeline. The software binaries are (a) represented using different state-of-the-art approaches, and (b) manipulated using functionality preserving operations, such as packing, stripping, and padding. The corresponding representations of the original samples and manipulated ones are then (c) tested against baseline pre-trained ML-based malware detectors and industrial detection engines.

Nevertheless, our preliminary results demonstrate that manipulating the malicious software with functionality-preserving operations, such as stripping and binary padding, significantly reduced the accuracy. Figure 1 shows the different phases of analysis strategy; feature representation, software manipulation, and evaluation of the ML-based detection systems and online industry-standard detection engines.

Contributions. Our contributions are summarized as follows:

1) *Validity of the baseline*: We examine the effectiveness of nine state-of-the-art malware detection representations as baselines for IoT ML-based malware detectors.
2) *Vulnerability to binary manipulation*: We evaluate the baseline detectors against three binary manipulation techniques, including packing, stripping, and padding.
3) *Vulnerability of industrial-standard malware detectors*: We evaluate the behavior of AI-based industrial malware detection engines against binary manipulation.

II. Dataset Overview

We collected a dataset of malicious and benign IoT software in binary forms for our study. The dataset was collected between November 2018 and December 2020, where 3,000 malware samples were retrieved from CyberIOCs [10], and VirusTotal [11]. On the other hand, 2,295 benign samples were compiled from source files on GitHub [12].

978-1-6654-3566-6/21/$31.00 ©2021 IEEE
DOI 10.1109/DSN-S52858.2021.00012

TABLE I

BASELINE CLASSIFIERS EVALUATION UNDER BINARY MANIPULATION (%). O FOR ORIGINAL, P FOR PACKED, S FOR STRIPPED, AND PP FOR PADDED. THE REFERENCES BELOW ARE: [1] [4], [5]; [2] [6]; [3] [7]; [4] [8], [9]

Representation	Benign				Malware			
	O	P	S	PP	O	P	S	PP
Binary–Image[1]	99.6	2.4	2.4	72.7	100	96.7	92.0	85.0
CFG–Adjacency[2]	88.0	98.9	88.0	88.0	91.5	73.2	91.5	91.5
CFG–Algorithmic[3]	99.5	99.6	99.5	99.5	99.0	88.8	99.0	99.0
Static–String[4]	98.7	2.4	98.7	98.7	100	100	100	100
Static–Symbols[4]	98.7	2.2	98.7	98.7	100	100	100	100
Static–Sections[4]	100	3.5	100	100	100	100	100	100
Static–Segments[4]	100	3.5	100	100	100	100	100	100
Static–Hexdumps[4]	98.3	1.7	98.3	98.3	100	92.8	100	100
Static–Combined[4]	99.8	3.3	99.6	99.8	100	100	100	100

TABLE II

THE ONLINE IOT MALWARE DETECTION ENGINES EVALUATION. O FOR ORIGINAL, P FOR PACKED, S FOR STRIPPED, AND PP FOR PADDED.

Engine	Benign				Malware			
	O	P	S	PP	O	P	S	PP
AegisLab	100	100	100	100	84.8	28.4	81.3	4.7
AVG	100	85.8	100	100	100	82.3	100	100
Avast	100	86.4	100	100	100	82.8	100	100
Comodo	100	100	100	100	80.6	29.2	79.2	4.0
Cynet	100	100	100	100	99.5	34.7	98.5	37
FireEye	100	100	100	100	99.3	94.5	99.3	95.3
Kaspersky	100	100	100	100	99.7	99.7	99.7	99.7
Microsoft	100	100	100	100	78.9	26.6	77.0	74.9
NANO-AV	100	100	100	100	98.0	33.3	97.0	0.51
Qihoo-360	100	100	100	100	98.3	33.8	97.3	23.5

Data Augmentation. The dataset samples are transformed to different representations (shown in Table I). First, samples were represented as images. We then used *Radare2* [13] to reverse-engineer the samples and obtain various features. The Hexdump representation is used to represent the "*.text*" section of the binaries, while graphs are used to represent the execution flow. We followed the literature to construct each feature representation, that are then used as baselines for ML-based IoT malware detectors.

III. EVALUATION

Experimental Setup. Towards evaluating the robustness of the state-of-the-art IoT malware detectors, the dataset is transformed using the nine representations (see Table I). Then, Random Forest (RF) is used as a baseline ML algorithm to implement the detectors. During the training stage, the dataset is split into 80% training and 20% testing.

Preliminary Results. Table I shows the performance of RF-based IoT malware detectors using different software representations against the original and binary manipulated samples, including binary packing, stripping, and padding (Figure 2). Except for image-based representation, padding and stripping do not affect the performance of the detectors. Whereas, packed software is most likely to be classified as malware.

Another common practice for inspecting software has been by using online scan engines, such as VirusTotal, with reports that contain the detection results of a pool of anti-virus engines. Those engines are fittingly considered as the up-to-date capability of industry-standard malware detectors. Table II show the evaluation of the original and binary manipulated datasets on ten industrial detection engines. Except for "AVG" and "Avast", the benign performance is 100% regardless to the binary manipulation technique. However, nine

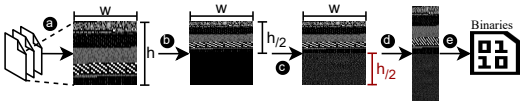

Fig. 2. Binary padding attack overview. (a) The software is represented as an $h \times w$ image. (b) The content of the image is then compressed into the size of $\frac{h}{2} \times w$. (c) Perturbation of size $\frac{h}{2} \times w$ in applied to the image. (d) The generated image perturbation is rescaled to the original size of the software, and then (e) reshaped to a 1-D vector represented the binaries to be appended.

out of ten engines degrade in performance under packing, while five engines malware detection rendered useless under binary padding. This emphasizes the usage of raw binaries for malware detection, regardless of their reachability.

Next, we will investigate practical adversarial attacks that generate executable and transferable malware, and analyze the performance of different ML architectures and online engines.

IV. CONCLUSION

Despite advances in AI-backed system defenses, the systems have been shown to be vulnerable. With this work, we systematically evaluated the state of a range of malware detectors, proposed by the research community and industry-standard. Our effort unveils the status-quo of the existing detectors, and brings forward various insights to consider when proposing detection systems, particularly, the ML model robustness.

Acknowledgement. This work was supported in part by NRF under grant 2016K1A1A2912757 and a CyberFlorida Collaborative Seed Award.

REFERENCES

[1] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *USENIX security symposium (USENIX Security)*, 2017, pp. 1093–1110.

[2] H. H. Pajouh, A. Dehghantanha, R. Khayami, and K. R. Choo, "A deep recurrent neural network based approach for internet of things malware threat hunting," *Future Gener. Comput. Syst.*, vol. 85, pp. 88–96, 2018.

[3] H. Aghakhani, F. Gritti, F. Mecca, M. Lindorfer, S. Ortolani, D. Balzarotti, G. Vigna, and C. Kruegel, "When malware is packin'heat; limits of machine learning classifiers based on static analysis features," in *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[4] D. Vasan, M. Alazab, S. Wassan, B. Safaei, and Q. Zheng, "Image-based malware classification using ensemble of cnn architectures (imcec)," *Computers & Security*, p. 101748, 2020.

[5] F. Mercaldo and A. Santone, "Deep learning for image-based mobile malware detection," *Journal of Computer Virology and Hacking Techniques*, pp. 1–15, 2020.

[6] Z. Xu, K. Ren, S. Qin, and F. Craciun, "Cdgdroid: Android malware detection based on deep learning using cfg and dfg," in *International Conference on Formal Engineering Methods*, 2018, pp. 177–193.

[7] H. Alasmary, A. Khormali, A. Anwar, J. Park, J. Choi, A. Abusnaina, A. Awad, D. Nyang, and A. Mohaisen, "Analyzing and Detecting Emerging Internet of Things Malware: A Graph-based Approach," *IEEE Internet of Things Journal*, 2019.

[8] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," in *Proceedings of ACM conference on data and application security and privacy*, 2016, pp. 183–194.

[9] A. Anwar, H. Alasmary, J. Park, A. Wang, S. Chen, and D. Mohaisen, "Statically dissecting internet of things malware: Analysis, characterization, and detection," in *International Conference on Information and Communications Security*. Springer, 2020, pp. 443–461.

[10] (2019) Cyberiocs. Available at [Online]: https://freeiocs.cyberiocs.pro/.

[11] (2019) VirusTotal. Available at [Online]: https://www.virustotal.com.

[12] Developers. (2019) Github. Available at [Online]: https://github.com/.

[13] (2019) Radare2. Available at [Online]: https://https://rada.re/r/.