Techniques for Enhancing the Robustness of Similarity-based IoT Malware Detection Methods against Adversarial Examples

József Sándor
Technical University of Munich
jozsef.sandor@tum.de

***Abstract*—SIMBIoTA and SIMBIoTA-ML are two recently proposed lightweight IoT malware detection solutions. Despite their initial effectiveness, they are not necessarily robust against adversarial examples. This paper summarizes two previous works that address this vulnerability. The first solution applies adversarial training to SIMBIoTA-ML by extending its training set with some adversarial examples. The second solution, called PATRIoTA, modifies SIMBIoTA to detect malicious byte sequences embedded in adversarial examples. Both solutions can detect the created adversarial examples with more than 98% accuracy. Moreover, PATRIoTA is a more general solution, while adversarial training does not incur as much overhead.**

I. Introduction

Nowadays, we can find Internet of Things (IoT) devices in various aspects of our surroundings, including agriculture, healthcare, and even in our homes. By the end of 2023, the estimated number IoT devices worldwide reached 15 billion[1], and according to the forecasts, it is expected to double by 2030. While every computer system has security weaknesses, this is especially true for IoT devices due to their diversity and the lack of incentives for IoT vendors to create secure systems. Thus, IoT vulnerabilities create a substantial threat surface frequently exploited by adversaries, often through the use of malware. An infamous malware type attack is Mirai from 2016 [1], which infected hundreds of thousands of IoT devices and launched one of the largest DDoS attacks against Internet-based services. Successful attacks on IoT devices can lead to massive privacy breaches, economic losses, and even physical damages (see e.g. the proof-of-concept attack on a Jeep Cherokee carried out in 2015 and its potential consequences [2]). To prevent these attacks, it is crucial to detect malware before malicious content is executed. However, IoT devices have limited hardware resources; therefore, the detection process on IoT clients should be lightweight. According to the literature [3], [4], even well-performing malware detection systems are not necessarily robust against adversarial attacks. In this paper, we investigate the same issue, specifically, the robustness of two IoT malware detection solutions against adversarial examples (AEs). Additionally, we introduce two techniques to enhance their robustness. These solutions have been previously published in two separate papers. Essentially, this paper serves as a condensed summary of their contributions:

- In Section II, we present SIMBIoTA [5], a recently proposed similarity-based IoT malware detection solution, which fulfills the requirements of the IoT domain. It can detect malware entirely locally, providing fast and highly accurate malware detection. The other solution that we examine is SIMBIoTA-ML [6], which enhances SIMBIoTA's detection capabilities even further through the incorporation of machine learning.
- However, these two IoT malware detection methods may not inherently be robust against AEs. To substantiate this claim, we devised two strategies for creating AEs: namely, the *Chunker* and the *Disguiser* [7]. Both strategies involve appending some bytes to the end of existing malware binaries in such a way that the resulting AE is considered benign by SIMBIoTA and SIMBIoTA-ML. The AEs generated by the *Chunker* can deceive SIMBIoTA, whereas SIMBIoTA-ML exhibits some resilience against them. However, against the AEs generated by the *Disguiser*, both systems are helpless. In the second part of Section II, we provide a more comprehensive description of their operations and results.
- To address this problem and enhance the robustness of the two IoT malware detection methods, we propose two solutions: one for SIMBIoTA and one for SIMBIoTA-ML. The first technique involves adversarial training (AT) applied to SIMBIoTA-ML [7]. In this case, we augment the original training set with AEs and retrain our model on this extended training set. The second technique, named PATRIoTA [8], is a modification of SIMBIoTA designed to be more resilient against AEs. In Section III, we provide detailed explanations of these two countermeasures.
- In order to evaluate the effectiveness of the two robustness enhancing techniques, in Section IV, we measure the performance of SIMBIoTA-ML before and after AT. SIMBIoTA-ML, after AT, can detect AEs with the same accuracy as the original malware samples. Furthermore, we compare the malware detection capability of SIMBIoTA to PATRIoTA, which outperforms SIMBIoTA in each simulated scenarios.

[1] https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ (accessed on January 27, 2024)

II. Preliminaries

SIMBIoTA [5] (SIMilarity-based IoT Antivirus) is a lightweight IoT malware detection method and exploits the observation that the binary representation of malware and benign files differs drastically, additionally, malware binaries belonging to the same malware family resemble each other. To capture this (dis)similarity, SIMBIoTA uses TLSH difference metric [9]. TLSH, unlike cryptographic hash functions, produces similar output for similar input. Furthermore, we can measure the difference between two TLSH hashes using the TLSH difference metric, which results in a non-negative integer (higher value indicates a larger dissimilarity between two hashes). If an IoT device maintains its own database of TLSH hashes of malware samples, it can determine whether a suspicious file is malicious or not. To do this, it needs to compare the TLSH hash of the suspicious file to the stored malicious TLSH hashes. If there is a match, indicating that the suspicious file resembles a known malware, it can be considered malicious. Antivirus (AV) providers receive thousands of malware samples every day from various sources, resulting in vast malware databases. Even if we store only the TLSH hashes (35 bytes) of the samples, it still requires too much storage for an IoT device. Since malware samples belonging to the same malware family resemble each other, having one sample from the family allows us to recognize all the other members. Thus, we can compress knowledge without losing information if we can select a few representatives from the TLSH hashes of all the malware samples we have. SIMBIoTA precisely exploits this concept by constructing a similarity graph from the TLSH hashes of the malware samples. In this graph, the vertices represent the samples, and two vertices are connected with an edge if the TLSH difference between them is less than 40 (for a detailed explanation, see [10]). Subsequently, SIMBIoTA calculates a dominating set of the constructed graph, which is typically much smaller than the size of the original similarity graph. Only the TLSH hashes of the dominating set are then distributed to the IoT devices. With this construction, as evaluated in [5], SIMBIoTA required only 6-8 KB of storage capacity, and it could determine the malicious or benign nature of any file within 0.12-0.14 ms, it has ca. 95% true positive detection rate even on previously unseen malware samples, while maintaining a 0% false positive rate throughout the experiments. SIMBIoTA-ML [6] replaces the phases of similarity graph construction and dominating set calculation with the training of a machine learning model, which is trained with feature vectors extracted from both malware and benign samples. Only the fully-trained model is then sent to the IoT devices. With this solution, SIMBIoTA-ML has a true positive malware detection rate of ca. 95%, while having low false positive detection rate at the same time.

SIMBIoTA and SIMBIoTA-ML have rather simple detection methods, hence they may be vulnerable to adversarial attacks. In [7] we created two strategies (*Chunker* and *Disguiser*) that modify existing malware samples such a way that the original malicious functionality is preserved, while the TLSH value of the modified malware is different enough to be missclassified by the detection systems. More specifically, the *Chunker* appends a carefully chosen chunk of the original sample to itself with the goal of increasing the TLSH difference between the modified and the original samples above 40 (or beyond). *Disguiser* appends an appropriately chosen benign file to the malware binary and its goal is to decrease the TLSH difference between the modified malware and the benign file below 40 (i.e., to make the modified malware similar to the added benign file). These strategies are simple enough to be easily implemented by a real-world attacker. According to the measurements presented in [7] (as discussed in Section IV), both detection methods can be completely deceived by the *Disguiser* strategy, whereas SIMBIoTA-ML exhibits some robustness against the *Chunker* strategy.

III. Techniques for Enhancing the Robustness

Given that SIMBIoTA-ML incorporates machine learning, firstly we propose, in [7], a widely adopted solution in the machine learning domain to enhance its robustness against AEs: adversarial training (AT). During AT of SIMBIoTA-ML, we extend the original training set with some AEs generated from the original training set. Subsequently, we retrain our model with the extended training set, expecting that the newly trained model will correctly recognize both the elements of the original test set and the AEs created from the original test set. We do AT separately in the case of *Chunker* and *Disguiser*.

Our second approach, PATRIoTA [8] (PArticle Trained IoT Antivirus), modifies the operation of SIMBIoTA to enhance its robustness against AEs. In both the *Chunker* and *Disguiser* strategies, the actual malware is embedded in the samples. Consequently, PATRIoTA does not operate on the entire binary like SIMBIoTA; instead, it divides binaries into fixed-size parts, referred to as 'particles' hereafter. PATRIoTA constructs a similarity graph from the TLSH hashes of these malware particles and calculates the dominating set. Similar to SIMBIoTA, the elements of the dominating set are sent to the IoT devices. If a suspicious file contains a critical number of malicious particles, it is considered malware.

IV. Evaluation

Both AT and PATRIoTA can significantly improve the robustness of SIMBIoTA-ML and SIMBIoTA, respectively. In our experiments detailed in [7], [8], we measured the accuracy of the systems on different test sets using the regular Formula 1, where TP, TN, FP, and FN stand for true positive, true negative, false positive, and false negative values, respectively. For all the measurements, we used the same publicly available dataset[2], which contains thousands of ARM and MIPS executable malware samples. Furthermore, simulating the fact AV providers have knowledge about only a small fraction of the malware existing in the wilderness, we train our models only on the 10% of the data set and we test on the remaining 90%.

[2] https://github.com/CrySyS/cube-maliot-2021 (accessed on January 27, 2024)

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{1}$$

The measurements in Figure 1 show that the AEs created by the *Chunker* can decrease the accuracy of SIMBIoTA-ML to 93%. However, after AT, it rises back above 98%. The true power of AT manifests itself in the case of the other AE creation strategy, shown in Figure 2: the original SIMBIoTA-ML can be totally misled by the *Disguiser* strategy, but after AT, it can detect AEs with 98% accuracy. However, changing the composition of the training set may result in accuracy loss on the original test set, as we can observe in Figure 2. Seemingly, SIMBIoTA-ML is more sensitive to noise (i.e., AEs in the training set) in the case of MIPS samples; the detection accuracy of the updated model on the original test set drops to 82%. This phenomenon requires further investigation and may lead to additional research directions.

For the comparison of SIMBIoTA and PATRIoTA, in Figure 3, we measured the accuracy of both methods and we can observe that PATRIoTA outperforms the accuracy of SIMBIoTA in every cases. For more details, including the model size, detection time and training time overhead, we refer the reader to [7], [8]. In addition, while PATRIoTA was designed to be robust against AEs that were created from existing malware samples by appending extra bytes to them, we have the intuition that it is also robust against other strategies that create AEs that contain chunks of the original sample, as those chunks may result in particles that are similar to the particles of the original sample. In order to test this intuition, we measured the robustness of PATRIoTA against such a strategy. In particular, a very clever AE creation strategy against similarity-based malware detection was proposed in [11] that consists in modifying a few unused portions of a malware binary such that the TLSH difference between the modified and the original files is maximized, while the functionality of the original binary is fully preserved, the size of the modified file remains the same as that of the original one, and even the binary content is only slightly changed. We tested both SIMBIoTA and PATRIoTA with those samples, and SIMBIoTA recognized only 17% of them as malware, while PATRIoTA detected a remarkable 98% of them as malware!

V. Related Work

Solutions that combine machine learning with cloud-based approach scale well and can be applied also in the IoT field [12]. This construction is advantageous for resource constrained IoT devices, because resource heavy calculation and processing can be passed to cloud, and only a lightweight algorithm is needed on client side. They can use different ML models, including convolutional neural networks, recurrent neural networks, random forest classifiers, fuzzy and fast fuzzy pattern trees [13]. Moreover, there are many different approaches for adversarial attacks also in the context of malware detection [4]. From these approaches we can highlight *append* and *slack* attacks [3] for their simplicity. Append attacks generate bytes and add them to the end of malware binary. Slack attacks add or modify bytes in slack regions of a binary, which are gaps between neighboring sections of an executable file. Our presented strategies (*Chunker* & *Disguiser*) resemble the previously mentioned append attack. There are other solutions for generating and appending bytes to the end of a binary, including gradient-based approach [14]. Another more advanced technique is program obfuscation, which can change the binary representation of a program while preserving its functionality. In order to do so, ML solutions can be used, including reinforcement learning-based approaches, Generative Adversarial Networks (GAN) and Recurrent Neural Networks (RNN) [15]. Obfuscating existing malware samples may be a successful strategy, but we do not use it, because from the perspective of SIMBIoTA and SIMBIoTA-ML, obfuscated samples appear to be new malware, as their binary representations can be completely different from those of the original samples from which they were created. In other words,

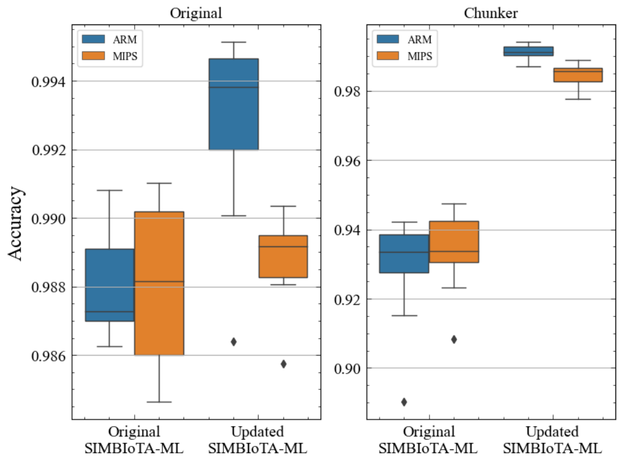

Fig. 1. Comparison of the accuracy of SIMBIoTA-ML before (Original) and after (Updated) AT, evaluated on the original test set and the AEs created by the *Chunker*, in the ARM and MIPS cases.

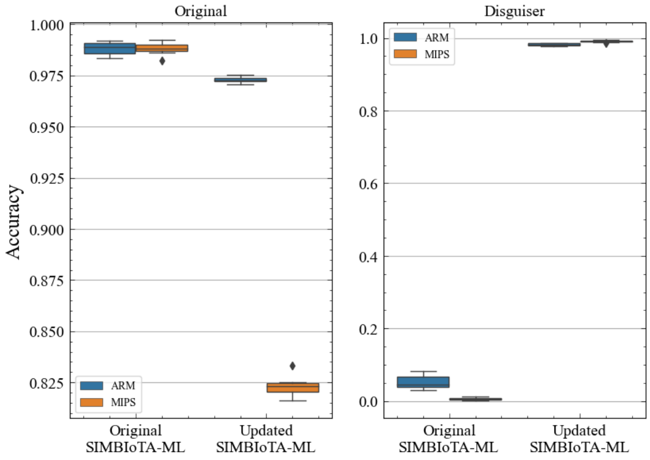

Fig. 2. Comparison of the accuracy of SIMBIoTA-ML before (Original) and after (Updated) AT, evaluated on the original test set and the AEs created by the *Disguiser*, in the ARM and MIPS cases.

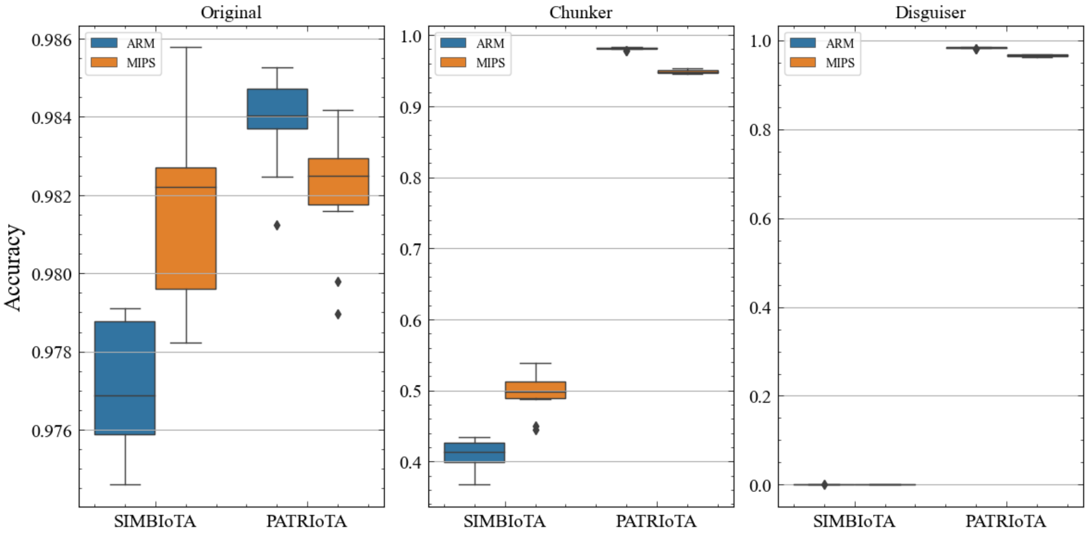

Fig. 3. Comparison of the accuracy of SIMBIoTA and PATRIoTA, evaluated on the original test set and the AEs created by the *Chunker* and *Disguiser*, in the ARM and MIPS cases.

obfuscated samples are considered new malware by SIMBIoTA and SIMBIoTA-ML, and their detection performance on them has already been measured in [6]. Finally, adversarial training is an effective way to increase the robustness of ML-based systems against adversarial examples, and it can also be applied in the malware detection domain. Several existing solutions use this technique to improve their malware detection systems [16]; however, we applied it first in the domain of ML-based IoT malware detection.

VI. Conclusion

To summarize this work, we presented two recently proposed IoT malware detection solutions: SIMBIoTA and SIMBIoTA-ML. We showed two adversarial strategies, *Chunker* and *Disguiser*, capable of misleading these solutions. To overcome this problem, we proposed two solutions that enhance the robustness of the systems against the devised adversarial strategies: adversarial training and PATRIoTA. Finally, beyond academic research, we offer an open-source Rust implementation of SIMBIoTA for Raspberry PI devices[3]. We encourage everybody who maintains such IoT devices to use it in the name of securing the IoT ecosystem.

References

[1] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, “Understanding the mirai botnet,” in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1093–1110.

[2] C. Miller and C. Valasek, “Remote exploitation of an unaltered passenger vehicle,” in *Black Hat USA*, 2015.

[3] O. Suciu, S. E. Coull, and J. Johns, “Exploring adversarial examples in malware detection,” *2019 IEEE Security and Privacy Workshops (SPW)*, pp. 8–14, 2019.

[4] K. Aryal, M. Gupta, and M. Abdelsalam, “A survey on adversarial attacks for malware analysis,” *ArXiv*, vol. abs/2111.08223, 2021.

[5] C. Tamás, D. Papp, and L. Buttyán, “SIMBIoTA: Similarity-based Malware Detection on IoT Devices,” in *IoTBDS*. SCITEPRESS, 2021, pp. 58–69.

[6] D. Papp, G. Ács, R. Nagy, and L. Buttyán, “SIMBIoTA-ML: Light-weight, Machine Learning-based Malware Detection for Embedded IoT Devices,” in *Proceedings of the 7th International Conference on Internet of Things, Big Data and Security - IoTBDS,*, INSTICC. SciTePress, 2022, pp. 55–66.

[7] J. Sándor, R. Nagy, and L. Buttyán, “Increasing the Robustness of a Machine Learning-based IoT Malware Detection Method with Adversarial Training,” in *WiseML’23: Proceedings of the 2023 ACM Workshop on Wireless Security and Machine Learning*, 2023.

[8] ——, “PATRIoTA: A Similarity-based IoT Malware Detection Method Robust Against Adversarial Samples,” in *IEEE International Conference on Edge Computing and Communications (EDGE)*, 2023.

[9] J. Oliver, C. Cheng, and Y. Chen, “Tlsh – a locality sensitive hash,” in *2013 Fourth Cybercrime and Trustworthy Computing Workshop*, 2013, pp. 7–13.

[10] L. Buttyán, R. Nagy, and D. Papp, “SIMBIoTA++: Improved Similarity-based IoT Malware Detection,” in *2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS)*. IEEE, 2022, pp. 51–56.

[11] G. Fuchs, R. Nagy, and L. Buttyán, “A Practical Attack on the TLSH Similarity Digest Scheme,” in *ARES ’23: Proceedings of the 18th International Conference on Availability, Reliability and Security*, 2023.

[12] H. Sun, X. Wang, R. Buyya, and J. Su, “Cloudeyes: Cloud-based malware detection with reversible sketch for resource-constrained internet of things iot devices,” *Softw. Pract. Exper.*, vol. 47, no. 3, p. 421–441, mar 2017.

[13] E. M. Dovom, A. Azmoodeh, A. Dehghantanha, D. E. Newton, R. M. Parizi, and H. Karimipour, “Fuzzy pattern tree for edge malware detection and categorization in iot,” *Journal of Systems Architecture*, vol. 97, pp. 1–7, 2019.

[14] 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 *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 533–537.

[15] D. Park, H. Khan, and B. Yener, “Generation & evaluation of adversarial examples for malware obfuscation,” in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019, pp. 1283–1290.

[16] D. Li, Q. Li, Y. F. Ye, and S. Xu, “Arms race in adversarial malware detection: A survey,” *ACM Comput. Surv.*, vol. 55, no. 1, nov 2021. [Online]. Available: https://doi.org/10.1145/3484491

[3] https://www.simbiota.io/ (accessed on January 27, 2024)