- Creating an effective TinyML dataset or repurposing existing datasets for TinyML.
  Traditional ML models need a large amount of data that are hard to get. Collecting and labelling these large datasets is expensive and the data may be only used for special tasks. Although there are a number of well-known open-source datasets for training ML models, these public datasets are not suitable to train ultra-low-power models for embedded devices, and are relatively large for TinyML-specific use cases. Therefore, a specific TinyML dataset is one of the promising areas for future research. TinyML scholars could set up brand new datasets or take advantage of these existing datasets by repurposing them.
- Designing specific algorithms for TinyML-specific requirements relevant to AI security.
  Most ML algorithms are very vulnerable to perturbations and TinyML algorithms. What's more, some current algorithms are either computationally too intensive or overly complex for TinyML deployment [2]. Therefore, one interesting research direction is to empirically evaluate how robust the existing TinyML models/algorithms are and how to improve their robustness in terms of AI security.
- Protocol, benchmarks, standards and rules for TinyML referring to AI security.
  New endpoint security mechanisms are not only required to meet adequate security standards, but also such mechanisms must be as lightweight as possible. The TinyML community extends the existing MLPerf benchmark suite to TinyMLPerf with TinyML systems. The goal of TinyMLPerf is to provide a detailed description of the motivation and guiding principles for benchmarking of TinyML systems [L2]. Although the security protocol Object Security for Constrained RESTful Environments (OSCORE) is designed to tackle this challenge, more standards are needed for AI security when employing TinyML [3].

The work described in this article has been carried out in the frame of an ERCIM "Alain Bensoussan" Fellowship.

Links:
[L1] https://www.tinyml.org
[L2] https://mlperf.org

References:
[1] R. Sanchez-Iborra and A. F. Skarmeta: "TinyML-enabled frugal smart objects: challenges and opportunities," IEEE Circuits and Systems Magazine, vol. 20, no. 3, pp. 4–18, 2020, doi: 10.1109/MCAS.2020.3005467.
[2] S. Siddiqui, C. Kyrkou, and T. Theocharides: "Mini-NAS: a neural architecture search framework for small scale image classification applications", in TinyML Research Symposium, 2021, pp. 1–8.
[3] H. Doyu, R. Morabito, and M. Brachmann: "A tinyMLaaS ecosystem for machine learning in IoT: overview and research challenges," in International Symposium on VLSI Design, Automation and Test, 2021, pp. 1–6. doi: 10.1109/VLSI-DAT52063.2021.9427352.

Please contact:
Hui Han
Fraunhofer Institute for Experimental Software Engineering IESE, Germany
hui.han@alumnos.upm.es

IoT Malware Detection with Machine Learning

by Levente Buttyán (Budapest University of Technology and Economics) and Rudolf Ferenc (University of Szeged)

Embedded devices are increasingly connected to the Internet to provide new and innovative applications in many domains. However, these IoT devices can also contain security vulnerabilities, which allow attackers to compromise them using malware. We report on our recent work on using machine learning for efficient and effective malware detection on resource-constrained IoT devices.

Embedded devices connected to the Internet are threatened by malicious programs (viruses, worms, also known as malware). One of the most infamous examples for IoT malware is Mirai, which infected hundreds of thousands of IoT devices and launched the largest distributed denial-of-service attack against Internet-based services in 2016, but the IoT threat landscape includes many other malware families as well.

Anti-virus products developed for traditional IT systems have higher resource needs than that offered by embedded IoT devices. The required amount of free storage space and memory to run these products is often measured in gigabytes, which exceeds the capacity of typical IoT devices, such as WiFi routers, IP cameras, smart household appliances, and wearable devices. In addition, many existing anti-virus products do not even support the operating systems used on IoT devices. Therefore, they could not be installed, even if a particular IoT device met their system requirements.

Since malware detection is essentially a classification task, machine learning-based methods have been applied in this area in recent years [1]. Machine learning-based malware detection has several advantages. For example, these methods are not only able to detect previously seen malware, but they can also detect new, previously unseen malware if it is similar in some way to previously seen samples. Another advantage is that machine learning models can represent more concisely the characteristics of previously seen malware patterns than the signature databases used in traditional signature-based detection. This makes machine learning-based malware detection particularly well-suited for resource-constrained environments such as embedded IoT devices.

Hence, in our projects (MILAB [L1] and SETIT [L2]), we work on new machine learning-based malware detection methods tailored for resource-constrained IoT devices. In a recent paper [2], we have proposed SIMBIoTA (SIMilarity Based IoT Antivirus), an effective and efficient anti-virus solution for such devices. The operating principles of SIMBIoTA are similar to those of traditional signature-based anti-virus solutions, but SIMBIoTA uses TLSH hash values of known malware instead of raw binary signatures for detection purposes. TLSH is a similarity hash algorithm, and it is different from cryptographic hashes: similar inputs result in similar TLSH hash values, and SIMBIoTA takes advantage of this feature. More specifically, embedded IoT devices that use SIMBIoTA store only a few TLSH hash values of known malware, and they compare the TLSH hash value of new files to these stored hashes. If the TLSH hash of an unknown file is similar to that of a known malware, the unknown file is detected as malware.

We evaluated the detection performance of SIMBIoTA and measured a true positive detection rate of more than 90% on average, even for previously unseen malware samples. Moreover, in the experiments performed, its false positive detection rate was 0%. In terms of resource needs, SIMBIoTA requires just a few tens of kilobytes of storage space, which is certainly available even on resource-constrained IoT devices.

In a follow-up work, we also used TLSH hash values for malware detection on IoT devices, but in a manner different from that of SIMBIoTA. Our key observation is that, thanks to their well-defined structure, TLSH hash values can be used as feature vectors for training machine learning models, which can then be used for malware detection. We call the resulting antivirus solution SIMBIoTA-ML. We showed that this approach can result in interesting trade-offs in terms of detection performance and resource usage on IoT devices. More specifically, SIMBIoTA has lower storage requirements and false positive detection rate than SIMBIoTA-ML, but SIMBIoTA-ML outperforms SIMBIoTA in terms of true positive detection rate, achieving more than 95% on average (see Figure 1). We also showed that SIMBIoTA's database of TLSH hash values increases over time, which has an impact on its detection time. Specifically, the larger the database is, the longer it takes for SIMBIoTA to decide whether an unknown file is malicious or not. By contrast, we showed that SIMBIoTA-ML has a near-constant running time, which allows for better estimation of the delay introduced by the anti-virus solution, and this can be an advantage in case of real-time applications (e.g., cyber-physical systems). In addition, using our DeepWater [3][L3] machine learning framework, we compared the detection

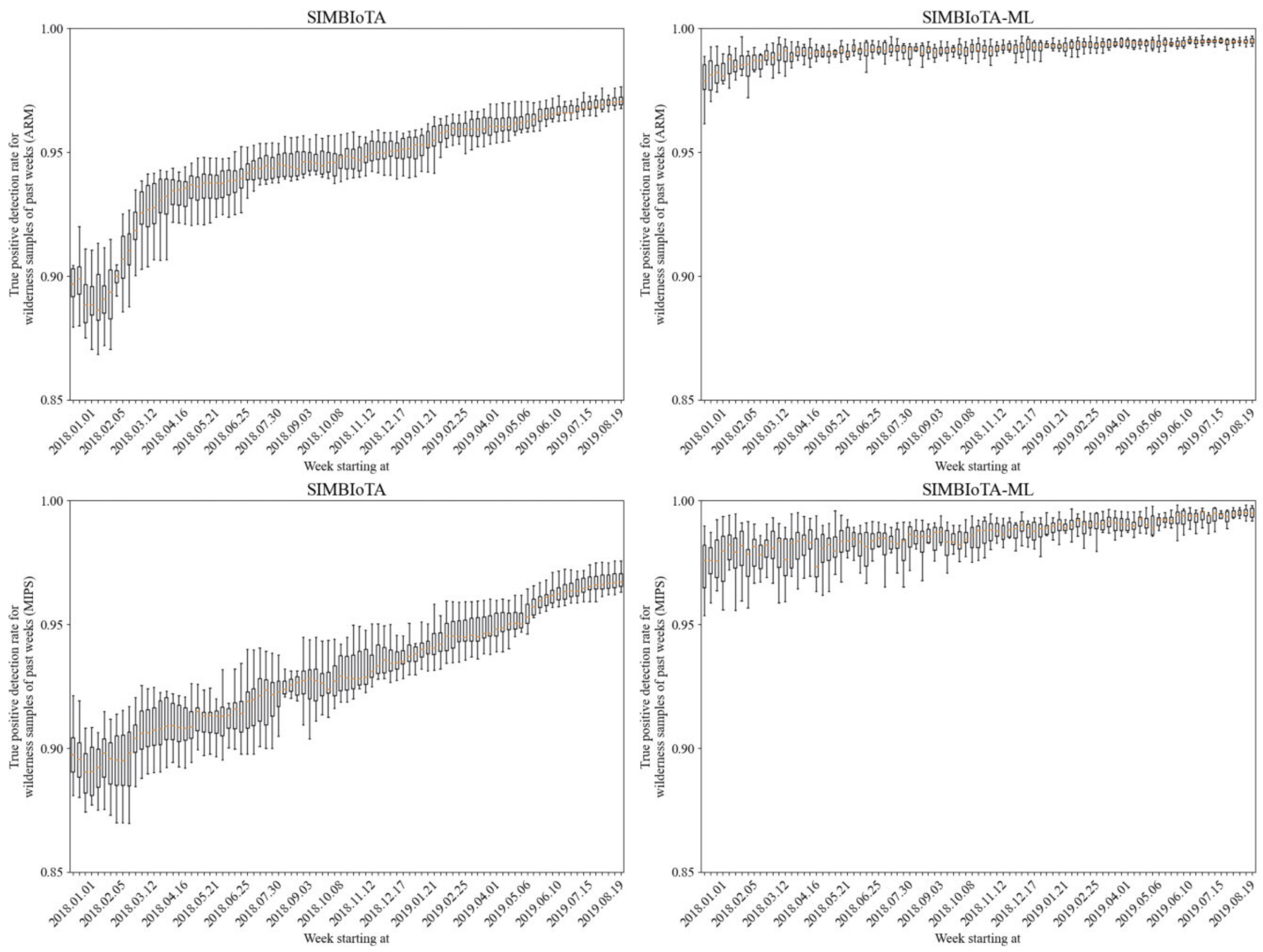

Figure 1: Box plot of the true positive detection rate of SIMBIoTA and SIMBIoTA-ML on ARM and MIPS samples.

performance of SIMBIoTA-ML when used with different machine learning models, and found that the best performance is achieved by the logistic regression model, which also turns out to be the least resource demanding in terms of memory usage and prediction time.

We performed all experiments using our IoT malware benchmark dataset called CUBE-MALIoT, which we made public at [L4]. This data set consists of 29,209 malicious samples developed for the ARM platform and 18,715 malicious samples developed for the MIPS platform. To the best of our knowledge, such a large dataset containing raw binaries of IoT malware was not previously available publicly to the research community. We hope that CUBE-MALIoT will become a de facto benchmark dataset in IoT malware detection in order to satisfy the need for the comparability and reproducibility of results of different research groups.

Links:
[L1] https://mi.nemzetilabor.hu/about-us
[L2] https://kwz.me/hfY
[L3] https://kwz.me/hfB
[L4] https://kwz.me/hfb

References:
[1] Ucci et al.: "Survey of machine learning techniques for malware analysis", Computers & Security. 81:123-147, 2019.
[2] Tamás et al.: "SIMBIoTA: Similarity-based malware detection on IoT devices", in Proc. of IoTBDS 2021, 58–69. SciTePress.
[3] Ferenc et al.: "Deep-water framework: The Swiss army knife of humans working with machine learning models", SoftwareX 12 (2020) 100551. Elsevier.

Please contact:
Levente Buttyán
Department of Networked Systems and Services, Budapest University of Technology and Economics, Hungary
buttyan@crysys.hu

Rudolf Ferenc
Department of Software Engineering, University of Szeged, Hungary
ferenc@inf.u-szeged.hu

Fighting Cybercrime by Introducing Trustworthiness and Interpretability in Deep Learning Malware Detection

by Giacomo Iadarola, Fabio Martinelli (IIT-CNR) and Francesco Mercaldo (University of Molise and IIT-CNR)

Cybercriminals can use a device compromised by malware for a plethora of purposes. Malicious intentions include the theft of confidential data, using the victim's computer to perform further criminal acts, or data ciphering to ask a ransom. Recently, deep learning is widely considered for malware detection. The main problem in the real-world adoption of these methods is due to their "black box" working mechanism i.e., the security analyst must trust the prediction without the possibility to understand the reason why an application is detected as malicious. In this article we discuss a malicious family detector, providing a mechanism aimed to assess the prediction trustworthiness and explainability. Real-world case studies are discussed to show the effectiveness of the proposed method.

In recent years, we have focused on cybercrimes perpetrated by malware that can compromise devices and IT infrastructures.. To fight cybercrimes due to malware spread, researchers are developing new methodologies for malware detection, with a great focus on the adoption of artificial intelligence techniques. One of the biggest controversies in the adoption of artificial intelligence models regards a lack of a framework for reasoning about failures and their potential effects. Governments and Industries are assigning critical tasks to artificial intelligence, but how can we ensure the safety and reliability of these models? In response to this need, the research community is moving toward interpretable models (the Explainable-AI field), able to provide the meaning of their predictions in understandable terms to humans [1]. One of the main reasons the adoption of these models in the real world is held back for effectively fighting cybercrimes exploiting malware is the lack of interpretation of the decision made by these models, which causes a lack of reliability certification, essential for achieving wide adoption.

We introduce a deep learning model for detecting malicious families in malware represented as images, with some interesting ideas for exploiting the activation maps, and provide model interpretability. We contextualise the proposed model on the Android platform, but the proposed method is platform independent.

The aim is to train a model i.e., a Convolutional Neural Network, for malware detection. For activation map drawing, we resort to the Grad-CAM [2] algorithm, able to generate a localisation map of the important regions in the image.

This mechanism provides to the security analyst a way to understand the reason why the model outputs a prediction. This aspect is fundamental for malware detection; in fact, malware belonging to the same family share parts of the code, hence areas of the images will be similar. For this reason, the activation map help to understand the area of the image symptomatic of a malicious behaviour exhibited by a family, thus providing trustworthiness.

We tested the model with a dataset composed of six malware families: Airpush, Dowgin, FakeInstaller, Fusob, Jisut and Mecor [3], obtaining a 0.98 accuracy.