8

Efficient AI-based Attack Detection Methods for Sensitive Edge Devices and Systems

Daniel Hirsch[1], Falk Hoffmann[1], Andrija Neskovic[2], Celine Thermann[2], Rainer Buchty[2], Mladen Berekovic[2], and Saleh Mulhem[2]

[1]NXP Semiconductors, Germany.
[2]Universität zu Lübeck, Germany

Abstract

An increasing number of edge devices store and process sensitive user data, presenting an attractive target for attackers. This trend of data storage and processing at the edge is expected to continue. As secure devices are integrated into new systems with increased device operation times, exposure to environmental stress also increases significantly. Especially, for standalone micro-Edge devices the relevance of this is increasing. Enhanced protection mechanisms are required and AI-based approaches are promising candidates.

In this contribution, we examine the requirements for such mechanisms and the sensing capabilities of state-of-the-art secure devices. Based on these capabilities and attack models, a dataset for training and validation is generated. Considering the requirements and the available dataset, a selection of applicable algorithms is defined. The selected algorithms are evaluated and compared based on the obtained results and computational loads, as the basis for future work.

Keywords: artificial intelligence, machine learning, security, attack detection, edge AI, micro-Edge, autonomous security, AI security.

8.1 Introduction and Background

Edge Computing (EC) is one of the most practical computing concepts used in day-to-day life applications. The architecture of edge computing is illustrated in Figure 8.1. EC is divided into three levels: Edge/IoT device, Edge device/node, and Cloud level. The core idea of EC is to perform computations and storage directly at the end-user level [1]. i.e., at the network edge [2, 3]. Handling sensitive data becomes prominent. The extraction of these sensitive data and the manipulation of security-relevant features of IoT and edge devices represent a lucrative target for attackers. Therefore, the need to securely protect and handle these data becomes important. Protection mechanisms that work towards this goal can be deployed on all three levels.

Security features are usually implemented to protect these data assets. The most straightforward way to identify manipulation or attacks is by checking both environmental and the device's internal sensors. Another way is to observe the logical monitoring and protection mechanisms that trigger a device reset or limit further use of the device. In extreme cases, device operation is temporarily or permanently blocked.

False alarms may be triggered in cases where sensor information is directly used without any further evaluation of severity, application relevance,

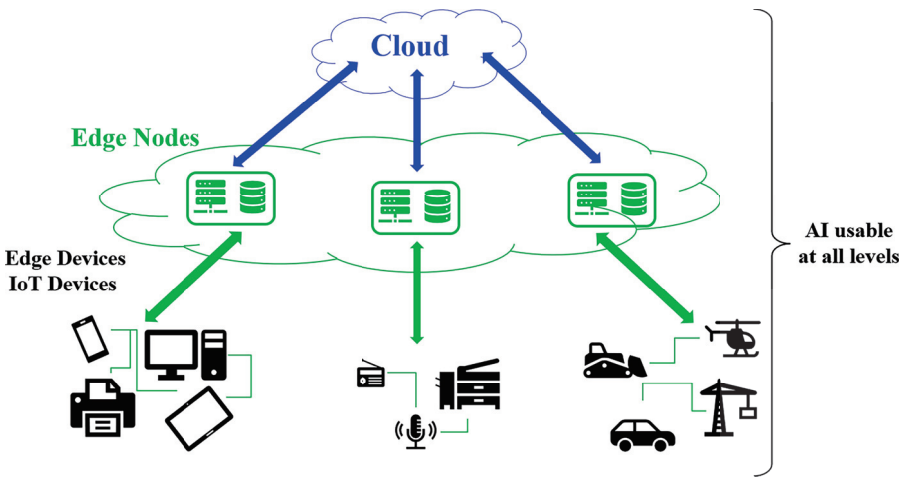

Figure 8.1 Architecture of Edge System

or statistical analysis of environmental effects. The consequence of such false alarms may be severe, leading to DoS attacks, for example. As secure devices are being integrated into an increasing number of systems with extended duty cycles, even up to permanent power-on conditions, the exposure to environmental stress increases significantly. A more advanced evaluation of sensor events and more flexible reactions need to be considered.

To ensure correct functionality of these systems and the integrity of user data, the evaluation of the security mechanisms with the help of AI algorithms represents a promising alternative to conventional approaches. To identify applicable algorithms for attack detection, an evaluation of the requirement specifications is carried out. However, due to the field of application and the special limitations in the physical domain of IoT and other edge devices, the requirements are challenging.

Relevant attacks on the edge

Studying security attacks on electronic devices and systems is a well-established field, but edge devices have certain characteristics that make them more prone to certain attacks and threats when compared to more capable computing devices. In [4], some of the aspects are pointed out, namely:

- **Weak Computation Power:** Edge devices are less powerful than cloud servers, making them susceptible to attacks not effective on cloud counterparts. Fragile defence systems on edge devices further expose them to unique threats.
- **Attack Unawareness**: IoT's lack of user interfaces limits awareness of device status, hindering attack detection.
- **Operating System (OS) and Protocol Differences:** Edge devices lack uniform OSes and protocols, complicating the creation of a unified security approach.
- **Limited Access Control Precision:** Edge computing's complex systems demand fine-grained access control, unlike current coarse-grained models.

Figure 8.2 shows distributions of the attack types on edge devices as presented in [4].

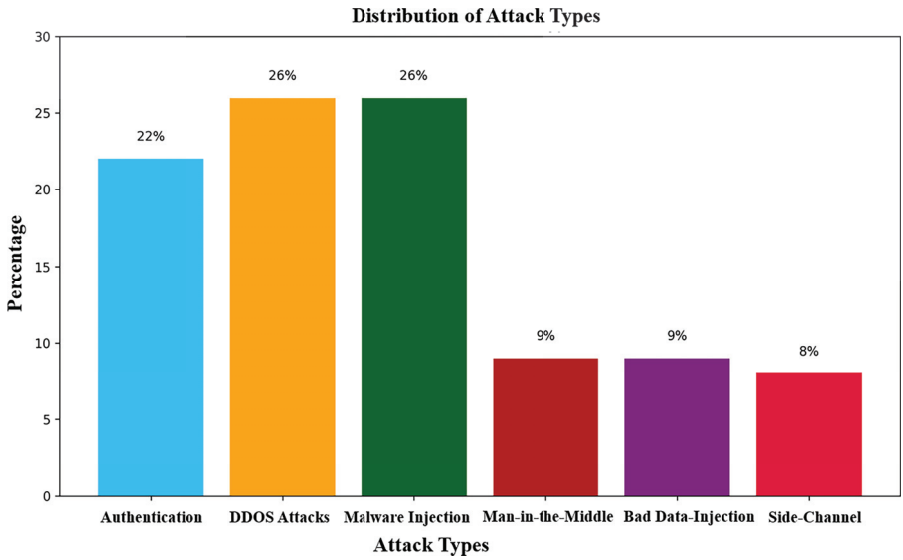

Figure 8.2 Possible Attacks against Edge Devices (Adapted From [4])

In the following, we summarize some possible attacks against edge devices, cloud, and edge systems:

1. **Possible Attacks on Edge Devices and Nodes**
   - **Malware Injection Attacks:** Malware Injection Attacks inject malicious code into the target device. These attacks can lead to arbitrary code execution which can compromise the security of further devices in the network. Considering the case of edge devices, protection becomes much more difficult because the limited computing power does not allow for classical high-performance firewalls or threat protection systems, like with general-purpose computers.
   - **DDOS Attacks:** DDoS, short for Distributed Denial of Service, is a cyber assault that involves perpetrators attempting to interrupt the regular operations of one or multiple servers. This is achieved by leveraging distributed resources, often in the form of a network of compromised edge devices, also known as a botnet [4]. It constitutes a potent form of attack that seeks to hinder the legitimate utilization of a particular service.
   - **Authentication and Authorization Attacks:** Authentication is the processing of verifying a user's identity who requests certain services and authorization grants that user rights to perform operations.

An adversary could exploit weaknesses in the authentication and authorization mechanisms to obtain privileged access rights and perform malicious operations.

- **Side-channel Attacks:** Refer to a type of attack, where the adversary can exploit information leakage of security-sensitive information via publicly accessible information which is not security-sensitive by nature. Most prominent examples of side-channel attacks exploit the power consumption or timing behaviour of a device while executing sensitive information. Side-channel attacks on the device level can potentially come from two sources, malicious tasks or a malicious OS. Task-level attacks or Timing attacks are typically cache-based such as Flush+Reload [5], Flush+Flush [6], Prime+Probe [7], Evict & Time [8], Evict & Reload [9], Spectre [10] and Meltdown [11] attacks. Here, the attacker aims at getting sensitive data by exploiting sharing vulnerabilities in caches [10, 11] and, in the case of Spectre and Meltdown, out-of-order optimization issues. The attack surface is large, also comprising several proposed and existing covert-channel attacks [12, 13, 14]. Multiple mitigation techniques have already been proposed, typically featuring either logical or physical separation, noise-based techniques, scheduler-based techniques, and constant time techniques. Attackers can exploit the power consumption of the edge device via a power side-channel attack. The concept of side-channel analysis appeared in the late 1990s [15], with Differential Power Analysis (DPA) [16] becoming a successful attack method. It was utilized to attack AES with a Simple Power Analysis (SPA) [17]. With the growing interest in the topic, more elaborate attack methods have been presented, e.g., Correlation Power Analysis (CPA) [18]. These types of attacks pose a significant threat to security-critical applications. Nowadays, even more powerful attack methods based on Template Attacks or utilizing AI as an attack tool for side-channel analysis are present. Fault injection attacks aim at maliciously altering an edge device's functionality. This can range from disturbances in the power supply voltage, irregularities in the clock signal, electromagnetic or radiation disturbances or overheating as described in [19]. The attack objective could be as complex as revealing the secret key of cryptographic primitives, but also simple, like blocking the computation, i.e., denial of service. The complexity and cost to perform a successful Fault Injection Attack can vary based on equipment costs and required knowledge about the underlying hardware. Although fault attack can

be expensive in terms of complexity and cost, it is practical and can be mounted on most commonly used architectures from ARM, Intel and AMD [20]. In recent years, even more elegant software-based approaches exploiting voltage scaling led to successful attacks on the Intel SGX secure enclave [21].

2. Possible Attacks on the Cloud

Securing cloud services is mainly achieved by separating a cloud's tenants. These must not be able to escape their individual virtual machines and get access to other tenant's data. Unfortunately, such has been proven viable via side-channel attacks, leading to cross-VM secret leakage via different levels of CPU cache side-channel attacks [5, 9, 22, 23, 24, 25, 26]. Mitigation is technically possible, but typically requires significant changes to hardware [27, 28, 29, 30], hypervisors [31, 32, 33, 34, 35, 36], or guest OSes [36]. Such approaches are not easily applicable to existing data centres. Mitigation by frequent VM migration [37, 38] is theoretically also feasible but comes at prohibitively high migration cost, i.e. several minutes of migration time [39], and hence only addresses the issue of long-term co-location. Attacks by malicious VMs however take only milliseconds [5, 24].

3. Intrusion Attacks on Edge System

The network-based exchange of data and commands between edge devices and cloud infrastructure implies several threats that can affect the edge system due to an insecure network. For instance, attackers can block the data transfer by malicious gateway access or network floods [40]. Similarly, attackers can perform attacks such as impersonation attacks, communication interception, password guessing attacks, data integrity violations, Denial of Service (DOS) and bad Quality of Service (QoS) [40].

Several countermeasures have been proposed to prevent or detect such attacks. The problem with these existing countermeasures is that they usually only address one specific attack, where an attacker can launch a multitude of attacks. To identify such attacks, two essential approaches exist which are signature-based and anomaly-based detection. By nature, signature-based attacks can be overcome by altering the attack code to evade detection and do not protect against previously unknown attacks [41]. Anomaly-based detection, in turn, is prone to false positives as legitimate applications may appear as malicious [41]. The combination of both methods mitigates some of the named individual shortcomings [41, 42].

Intrusion detection systems (IDS) are essential tools for monitoring and detecting of anomalous activities in a network of edge devices and systems

and responding to these attacks. Traditional IDS relies on signatures or rules to detect known attacks, but these methods are not effective against new and evolving threats. An anomaly detection system, on the other hand, relies on identifying abnormal behaviour in network traffic data. However, these systems can generate false positives, making them less reliable, and are inability to detect new/unknown attacks [40].

8.2 Efficient Attack Detection

In this section, the approach of selecting an appropriate solution for attack detection on resource-constrained micro-Edge ICs is described. The standalone IC protects on-chip data and secrets by preventing unauthorized access. First, the requirements related to this task are described, followed by a section on the dataset. Based on the requirements and the available dataset, a selection of applicable algorithms obtained from thorough research is specified.

8.2.1 Requirements

An implementation must meet requirements in the three domains of security, user experience, and realizability. The correlation of the requirements focusing on the three general domains is depicted in Figure 8.3 using a top-down representation.

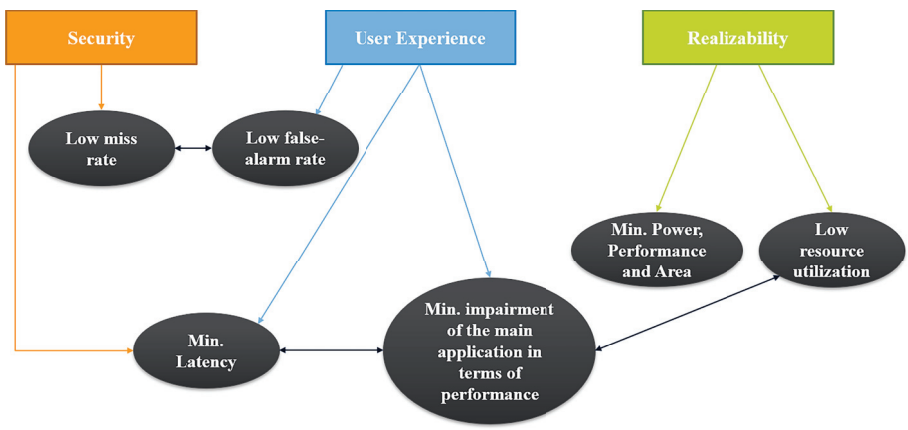

Figure 8.3 Correlation of requirements

The main goal is to target the domain of security. Based on AI, an algorithm capable of improving the present security mechanisms will be researched and evaluated. Since the devices handle sensitive data during operation, requirements covering the targeted levels of security must be defined. The second domain is represented by the user experience. Due to the commercial nature of the products, and since the additional functionality does not necessarily translate to a direct added value for the user, the user experience during usage is not allowed to be negatively influenced by the implemented solution.

Lastly, it must be noted that the available resources for implementing the functionality on the considered devices are limited in terms of computational power, area, and current consumption. Therefore, to benefit from the developed solution, it is also necessary to formulate implementation requirements that are realistic and applicable. These requirements are summarized in the domain of realizability.

Starting from the security perspective, the target of evaluation can contain highly sensitive data, therefore, a low miss rate in terms of detection of actual and exploitable attacks is mandatory to ensure the security and integrity of data stored on the device. Also, the implemented solution should not reduce the usability of the product or affect the user experience negatively. This requirement demands the lowest possible false alarm rate. Furthermore, the implemented solution should have minimal impact on the performance of the main application to achieve a satisfying user experience.

Besides the presented requirements, a fast response time constitutes a very important requirement in this application. To react quickly and prevent performance issues, the response time needs to be as fast as possible. This requirement can be attributed to the domains of security and user experience. From a security perspective, a fast response time is required to protect the secrets stored on the device. From the user's perspective, customers are not keen to see longer response times when using the devices. Therefore, in both domains, a fast response time is seen as advantageous.

Since the developed solution is targeted to be implemented on low-power edge or IoT devices, the available resources are very limited. Based on these general preconditions, further requirements concerning the memory, required die area, power efficiency, and CPU usage need to be formulated. Especially considering the CPU usage, low utilization must be achieved to guarantee minimal impairment of the main application.

8.2.2 Underlying Dataset

To obtain a flexible solution that is applicable for a variety of devices, the detection capabilities of state-of-the-art devices will be investigated. Since a dataset cannot be obtained from measuring traces in the laboratory or gathering field data, the dataset must be generated artificially. For this purpose, the relevant phases within an application and available inputs will be analysed.

Based on this further possible attack scenarios need to be researched and modeled. By the combination of capabilities and the theoretical consideration of attacks, a dataset will be derived. In the process of dataset generation, reasoned assumptions must be made and all decisions must be evaluated critically. Furthermore, the choice of labeling is going to be justified and strategies for the generation of a subset for the model validation will be explained.

8.2.3 State-of- the-Art Attack Detection Methods

1. AI-based Attack Detection at Edge Device Level
   The use of AI methods provides efficient countermeasures. HAL [43] provides a quantitative and qualitative analysis of several machine-learning models for use in cache-based side-channel attack detection. It specifically addresses real-time requirements, detection at an early stage, and minimal performance overhead and demonstrates this in the context of security applications (RSA and AES cryptosystems). It however does not provide a definite answer on specific model usage.
   Similarly, WHISPER [44] proposes a tool for side-channel attack detection based on machine learning. Instead of using a single approach, it features multiple ML models in combination that interpret the behavioural data of concurrent processes. This data is collected via hardware performance counters. The authors demonstrate the tool's capability by achieving $>99\%$ accuracy of detecting a large and diverse attack vector while introducing only a reasonably low performance overhead.
   In today's secured systems, installation and execution of malicious application software is typically rendered impossible by so-called shielded execution e.g. provided by the Intel Software Execution Guard [45]. However, such shielded execution can be compromised by privileged attackers, e.g. by changing page-table entries of memory pages that are specifically used by shielded execution. By this approach, a malicious OS kernel can observe corresponding memory-page accesses

and hence extract potentially sensitive information. DejaVu [46] is a software framework that enables self-protection detecting such privileged side-channel attacks from within the shielded execution. This is enabled by the so-called pathlet execution time. For this, a dedicated reference clock is employed that is specifically constructed using the Intel Transactional Synchronisation Extension (TSX). By featuring this robust reference clock, not only deviations in pathlet execution time indicating an attack can be detected but also interruptions of the reference-clock thread resulting in a transaction timeout.

Modern processors provide a limited number of registers known as hardware performance counters (HPCs) that capture hardware-related events. These special-purpose registers can be used to study the impact of side-channel attacks (SCAs). Compared to normal operation, the number of events when a system is under attack appears noticeably different. [47] explores several different machine-learning models for real-time cache-based SCA detection using HPCs. 16 HPC features are collected for both victims under attack and victims not under attack at different sampling rates. Overhead is reduced, by only using four features. This way they all can be fetched synchronously. The authors determined that for a sampling granularity of 500 µs, the systems incur 5% overhead while maintaining good detection accuracy. In addition, they also considered the latency for the different models. It was found that the Decision Tree provides the best trade-off between performance and latency.

[48] provides another approach using HPCs for the detection of side-channel attacks. The authors provide a two-step process comprised of an offline and an online phase. In addition to covering cache-based SCAs, they also consider branch-based and DRAM-based SCAs. The HPC can be observed to follow a Gaussian distribution with different means and variances. Anomalous behaviour shows a different distribution with a different mean. During the offline phase, data is collected in different environments. This includes benign programs running in the background, that make intensive use of the cache, branching, or the RAM. Afterwards, during the online phase, HPCs are collected and classified using an AI model. To counteract the high number of false-positives, anomalous traces are correlated with traces in a database. A high correlation indicates an attack, while a low correlation indicates a benign program running.

2. AI-based Attack Detection at Cloud Level
   Several security countermeasures have been introduced to mitigate possible attacks on the cloud [49, 50]. For instance, CloudRadar [41] proposes an approach to secure the cloud. This approach correlates signature-based and anomaly-based detection techniques in to spot side-channel attacks. Here, signature-based detection is used to identify when a protected VM executes cryptographic applications. Anomaly-based detection is orthogonally used to monitor and identify abnormal cache behaviours typical of cache-based side-channel attacks. As such, the approach is non-intrusive, not requiring any changes to hardware, hypervisor, guest VM, and applications. It hence is comparatively easy to deploy in existing cloud environments and, according to the authors, requires only patching and a little overhead [51]. To improve the performance of such detection techniques and cover more than the classical cache attacks against edge devices, Recurrent Neural Networks (RNNs) were proposed in [52]. The results show that additionally to the classical detection of cache attacks, the RNN-based solutions efficiently detect Rowhammer, Spectre, Meltdown, and Zombieload attacks as well.
3. AI-based Intrusion Detection System for Edge Systems
   AI-based Intrusion Detection Systems (AI-IDS) are a promising alternative to traditional IDS. AI methods can identify patterns and anomalies in network traffic data, enabling it to detect previously unseen, unknown, and complex threats. AI-IDS faces three main challenges: (1) the quality of the data used for training and testing the models, (2) the accuracy of the chosen AI algorithm, and (3), the performance of the chosen AI algorithm.
   Various techniques have been proposed to enhance the accuracy and performance of AI-IDS. For instance, the use of sampling techniques to select representative datasets can improve both the accuracy and speed of intrusion detection [53]. By combining a sampling technique with a random forest machine learning algorithm, IDS exhibits very good performance. However, it shows also different levels of detection accuracy for different attacks. In [54], Gini Impurity-based Weighted Random Forest (GIWRF) was used as a data feature selection technique. Then, the accuracy of several AI algorithms deployed as AI-IDS was analyzed. The results show that AI accuracy ranges from 88.99% to 99.98%.

8.2.4 Selection of Applicable Algorithms

In the following, the algorithms from research are evaluated in terms of their applicability to the problem with the associated requirements.

- Neural Network (NN): Model built from basic computation units called neurons that are usually organized into layers. Connections between neurons are associated with trainable weights. Upon receiving input, the input is weighted and aggregated. Afterwards, a possibly non-linear function is applied. The complexity of these models increases with the number of layers [55]. A Perceptron [56] is the simplest possible model and consists of a single layer of neurons. In contrast, Multilayer feed-forward Networks are comprised of multiple layers that are connected in a feed-forward fashion. If there are not only forward connections but also those connecting neurons to previous layers, the network is called recurrent [55]. An example of these types of networks are long short-term memory (LSTM) networks. These networks contain memory cells, making it possible to retain information [57].
- Trees: Models that make their decisions based on tree-like structures. One example of these types of models are decision trees (DT). They can be used for both classification and regression tasks [55]. Isolation Forests on the other side aim to find anomalies using binary trees [58].
- Support Vector Machine (SVM): Algorithm that tries to find a separator with the maximum distance to training samples. In the simplest case, the goal is to find a simple linear separator between two classes in a two-dimensional space [55].
- Bayesian Network: Probabilistic model allowing for computation of posterior probability distributions. Nodes represent random variables, while edges describe conditional dependencies between variables. Each node is associated with some probabilities that quantify the effect on other nodes. These probabilities can be learned from a given dataset. The simplest example of these classifiers are Naive Bayes classifiers [55].
- Instance-based: Algorithms that directly estimate from a given dataset. Processing of the input is deferred until queried. After answering the request, all intermediate results are discarded [55]. The most well-known example of these algorithms is the k-nearest neighbour (KNN) algorithm [59].
- Linear Regression: Algorithms that try to find the best-fitting function for some given data. In the simplest case, the goal is to find a linear function for a single input variable. Depending on the application, more complex functions might be used [55].

- Discriminant Analysis: Methods aiming to estimate the decision boundary between classes. Approaches like linear discriminant analysis might make simplifying assumptions, such as an underlying Gaussian distribution for all classes and the same covariance matrices for all classes [60].
- Ensemble: Combining multiple algorithms to achieve a better outcome. A random forest (RF) is an ensemble of decision trees. Ensembles can be created by many different techniques. One such technique is called boosting. It aims at improving performance by assigning higher weights to examples that have been misclassified and thus making an incentive to classify them correctly for the next model in the ensemble [55].

The most limiting factor in the selection of a suitable algorithm comes in the form of resource limitations. Some models have significant requirements for the systems they are executed on. Examples of such models are Neural Networks that can easily have millions of parameters. Not only does this require sufficient storage, but might also cause a significant delay in loading and applying these parameters. Consequently, a separate accelerator might be required, that increases the area consumed. Even non-parametric algorithms like KNN might not be a good solution, as the whole dataset has to be stored. Depending on the size of the dataset, this might also put a significant strain on the amount of memory available.

In [44], the results for twelve different machine learning models were presented, covering all of the classes described above. Considering all models achieving 80% accuracy leaves SVMs, DTs, RFs, KNN, NNs and Ensemble learning. As discussed beforehand, both KNN and NNs have high computational requirements, making them not suitable for the application.

Due to the experimental setup in [44], the detection latency using the models is unknown on the Edge and is to be determined in the future. Thus, the impact of deploying them cannot be determined, and there is still a need for AI models that offer (1) high detection accuracy, (2) efficiency, and (3) meet the requirements of Edge devices. Such an AI model can serve as a highly accurate, efficient and lightweight attack detector at the edge level.

8.3 Discussion and Conclusion

The continuously growing use and uptime of secure devices increase the number of sensor events during the product lifetime. Especially, for standalone micro-edge devices this becomes more relevant. Consequently, this is pushing the industry to step up from direct reaction to sensor events towards

more advanced solutions. It is, however, paramount that such solutions do not negatively influence both device operation and user experience. Processing and interpretation of available sensor information with the help of artificial intelligence offers the possibility to develop future solutions.

AI-based approaches particularly overcome limitations of established solutions based on signatures and anomaly detection: Signature-based approaches are inherently limited to known attacks and their signatures. They hence neither provide protection against future attacks nor altered attack code. Approaches based on anomaly detection, in turn, are prone to false positives as legitimate, non-malicious code may trigger such detection. So far, AI methods have been successfully employed in a wide variety of security systems, covering both edge nodes and cloud environments. They provide a sufficiently high detection rate at minimal false-positive level and, by nature, are immune to evasion strategies like altered attack code. However, so far no single gold solution exists. For AI approaches the choice of a suitable AI method is paramount. Similarly, sufficient labelling strategies and derived training sets need to be developed. Finding an optimal AI strategy for a given threat scenario is hence still open to research.

Acknowledgement

This research was conducted as part of the project "*Edge AI Technologies for Optimised Performance Embedded Processing*" (EdgeAI), which has received funding from KDT JU under grant agreement No 101097300. The KDT JU receives support from the European Union's Horizon Europe research and innovation program and Austria, Belgium, France, Greece, Italy, Latvia, Luxembourg, Netherlands, and Norway.

References

[1] H. Xue, B. Huang, M. Qin, H. Zhou and H. Yang, "Edge Computing for Internet of Things: A Survey", 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), pp. 755–760, 2020. W442W7302

[2] P. G. Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber and E. Riviere, "Edge-centric Computing: Vision and Challenges", SIGCOMM Comput. Commun. Rev., vol. 45, no. 5, p. 37–42, 2015. W442W7302

[3] W. Shi and S. Dustdar, "The Promise of Edge Computing", IEEE Computer, vol. 49, no. 5, pp. 78–81, 2016. W442W7302

[4] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu and W. Lv, "Edge Computing Security: State of the Art and Challenges", Proceedings of the IEEE, vol. 107, no. 8, pp. 1608–1631, 2019. W442W7302

[5] Y. Yarom and K. Falkner, "FLUSH+RELOAD: A High Resolution, Low Noise, L3 Cache Side-Channel Attack", Proceedings of the 23rd USENIX Conference on Security Symposium, p. 719–732, 2014. W442W7302

[6] D. Gruss, C. Maurice, K. Wagner and S. Mangard, "Flush+Flush: A Fast and Stealthy Cache Attack", Proceedings of the 13th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, vol. 9721, p. 279–299, 2016. W442W7302

[7] M. S. Inci, B. Gulmezoglu, G. Irazoqui, T. Eisenbarth and B. Sunar, "Cache Attacks Enable Bulk Key Recovery on the Cloud", Cryptographic Hardware and Embedded Systems – CHES 2016, pp. 368-388, 2016. W442W7302

[8] D. A. Osvik, A. Shamir and E. Tromer, "Cache Attacks and Countermeasures: The Case of AES", Topics in Cryptology – CT-RSA 2006, pp. 1–20, 2006. W442W7302

[9] D. Gruss, R. Spreitzer and S. Mangard, "Cache Template Attacks: Automating Attacks on Inclusive Last-Level Caches", 24th USENIX Security Symposium (USENIX Security 15), pp. 897–912, 2015. W442W7302

[10] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz and Y. Yarom, "Spectre Attacks: Exploiting Speculative Execution", 2019 IEEE Symposium on Security and Privacy (SP), pp. 1–19, 2019. W442W7302

[11] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom and M. Hamburg, "Meltdown: Reading Kernel Memory from User Space", 27th USENIX Security Symposium (USENIX Security 18), pp. 973–990, 2018. W442W7302

[12] Y. Lyu and P. Mishra, "A Survey of Side-Channel Attacks on Caches and Countermeasures", Journal of Hardware and Systems Security, vol. 2, no. 1, pp. 33–50, 2018. W442W7302

[13] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom and R. Strackx, "Foreshadow: Extracting the Keys to the Intel SGX Kingdom with

Transient Out-of-Order Execution", 27th USENIX Security Symposium (USENIX Security 18), 2018. W442W7302

[14] D. Genkin, L. Valenta and Y. Yarom, "May the Fourth Be With You: A Microarchitectural Side Channel Attack on Several Real-World Applications of Curve25519", Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, p. 845–858, 2017. W442W7302

[15] R. Mayer-Sommer, "Smartly analyzing the simplicity and the power of simple power analysis on smartcards", International Workshop on Cryptographic Hardware and Embedded Systems, pp. 78–92, 2000. W442W7302

[16] P. Kocher, J. Jaffe and B. Jun, "Differential power analysis", Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19, pp. 388–397, 1999. W442W7302

[17] S. Mangard, "A simple power-analysis (SPA) attack on implementations of the AES key expansion", Information Security and Cryptology—ICISC 2002: 5th International Conference Seoul, Korea, November 28–29, 2002 Revised Papers 5, pp. 343–358, 2003. W442W7302

[18] E. Brier, C. Clavier and F. Olivier, "Correlation power analysis with a leakage model", Cryptographic Hardware and Embedded Systems-CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings 6, pp. 16–29, 2004. W442W7302

[19] A. Barenghi, L. Breveglieri, I. Koren and D. Naccache, "Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures", Proceedings of the IEEE, vol. 100, no. 11, pp. 3056–3076, 2012. W442W7302

[20] J. Breier and X. Hou, "How Practical Are Fault Injection Attacks, Really?", IEEE Access, vol. 10, pp. 113122–113130, 2022. W442W7302

[21] K. Murdock, D. Oswald, F. D. Garcia, J. Van Bulck, D. Gruss and F. Piessens, "Plundervolt: Software-based Fault Injection Attacks against Intel SGX", 2020 IEEE Symposium on Security and Privacy (SP), pp. 1466–1482, 2020. W442W7302

[22] G. Irazoqui, T. Eisenbarth and B. Sunar, "S $ A: A shared cache attack that works across cores and defies VM sandboxing–and its application to AES", 2015 IEEE Symposium on Security and Privacy, pp. 591–604, 2015. W442W7302

[23] G. Irazoqui, M. S. Inci, T. Eisenbarth and B. Sunar, "Wait a minute! A fast, Cross-VM attack on AES", Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, September 17-19, 2014. Proceedings 17, pp. 299–319, 2014. W442W7302

[24] F. Liu, Y. Yarom, Q. Ge, G. Heiser and R. B. Lee, "Last-level cache side-channel attacks are practical", 2015 IEEE symposium on security and privacy, pp. 605–622, 2015. W442W7302

[25] Y. Zhang, A. Juels, M. K. Reiter and T. Ristenpart, "Cross-VM side channels and their use to extract private keys", Proceedings of the 2012 ACM conference on Computer and communications security, pp. 305–316, 2012. W442W7302

[26] Y. Zhang, A. Juels, M. K. Reiter and T. Ristenpart, "Cross-tenant side-channel attacks in PaaS clouds", Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 990–1003, 2014. W442W7302

[27] L. Domnitser, A. Jaleel, J. Loew, N. Abu-Ghazaleh and D. Ponomarev, "Non-monopolizable caches: Low-complexity mitigation of cache side channel attacks", ACM Transactions on Architecture and Code Optimization (TACO), vol. 8, no. 4, pp. 1–21, 2012. W442W7302

[28] F. Liu and R. B. Lee, "Random fill cache architecture", 2014 47th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 203–215, 2014. W442W7302

[29] Z. Wang and R. B. Lee, "A novel cache architecture with enhanced performance and security", 2008 41st IEEE/ACM International Symposium on Microarchitecture, pp. 83-93, 2008. W442W7302

[30] Z. Wang and R. B. Lee, "New cache designs for thwarting software cache-based side channel attacks", Proceedings of the 34th annual international symposium on Computer architecture, pp. 494–505, 2007. W442W7302

[31] T. Kim, M. Peinado and G. Mainar-Ruiz, "STEALTHMEM System-Level Protection Against Cache-Based Side Channel Attacks in the Cloud", 21st USENIX Security Symposium (USENIX Security 12), pp. 189–204, 2012. W442W7302

[32] P. Li, D. Gao and M. K. Reiter, "Stopwatch: a cloud architecture for timing channel mitigation", ACM Transactions on Information and System Security (TISSEC), vol. 17, no. 2, pp. 1–28, 2014. W442W7302

[33] J. Shi, X. Song, H. Chen and B. Zang, "Limiting cache-based side-channel in multi-tenant cloud using dynamic page coloring", 2011

IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops (DSN-W), pp. 194–199, 2011. W442W7302

[34] V. Varadarajan, T. Ristenpart und M. Swift, "Scheduler-based defenses against Cross-VM side-channels", 23rd USENIX security symposium (USENIX security 14), pp. 687–702, 2014. W442W7302

[35] B. C. Vattikonda, S. Das and H. Shacham, "Eliminating fine grained timers in Xen", Proceedings of the 3rd ACM workshop on Cloud computing security workshop, pp. 41–46, 2011. W442W7302

[36] Y. Zhang und M. K. Reiter, "Düppel: Retrofitting commodity operating systems to mitigate cache side channels in the cloud", Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, pp. 827–838, 2013. W442W7302

[37] S.-J. Moon, V. Sekar and M. K. Reiter, "Nomad: Mitigating arbitrary cloud side channels via provider-assisted migration", Proceedings of the 22nd acm sigsac conference on computer and communications security, pp. 1595–1606, 2015. W442W7302

[38] Y. Zhang, M. Li, K. Bai, M. Yu and W. Zang, "Incentive compatible moving target defense against vm-colocation attacks in clouds", Information Security and Privacy Research: 27th IFIP TC 11 Information Security and Privacy Conference, SEC 2012, Heraklion, Crete, Greece, June 4-6, 2012. Proceedings 27, pp. 388–399, 2012. W442W7302

[39] V. Varadarajan, Y. Zhang, T. Ristenpart and M. Swift, "A Placement Vulnerability Study in Multi-Tenant Public Clouds", 24th USENIX Security Symposium (USENIX Security 15), pp. 913–928, 2015. W442W7302

[40] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac and P. Faruki, "Network intrusion detection for IoT security based on learning techniques", IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2671–2701, 2019. W442W7302

[41] T. Zhang, Y. Zhang and R. B. Lee, "CloudRadar: A Real-Time Side-Channel Attack Detection System in Clouds", Research in Attacks, Intrusions, and Defenses. RAID 2016., pp. 118–140, 2016. W442W7302

[42] M. Alam, S. Bhattacharya, D. Mukhopadhyay and S. Bhattacharya, "Performance Counters to Rescue: A Machine Learning based safeguard against Micro-architectural Side-Channel-Attacks", IACR Cryptol. ePrint Arch., 2017. W442W7302

[43] M. Mushtaq, A. Akram, M. K. Bhatti, M. Chaudhry, M. Yousaf, U. Farooq, V. Lapotre and G. Gogniat, "Machine learning for security: The case of side-channel attack detection at run-time", 2018 25th IEEE

International Conference on Electronics, Circuits and Systems (ICECS), pp. 485–488, 2018. W442W7302

[44] M. Mushtaq, J. Bricq, M. K. Bhatti, A. Akram, V. Lapotre, G. Gogniat and P. Benoit, "WHISPER: A tool for run-time detection of side-channel attacks", IEEE Access, vol. 8, pp. 83871–83900, 2020. W442W7302

[45] O. Aciiçmez, "Yet another microarchitectural attack: exploiting I-cache", Proceedings of the 2007 ACM workshop on Computer security architecture, pp. 11–18, 2007. W442W7302

[46] S. Chen, X. Zhang, M. K. Reiter and Y. Zhang, "Detecting privileged side-channel attacks in shielded execution with Déjá Vu", Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pp. 7–18, 2017. W442W7302

[47] H. Wang, H. Sayadi, A. Sasan, S. Rafatirad, T. Mohsenin und H. Homayoun, "Comprehensive Evaluation of Machine Learning Countermeasures for Detecting Microarchitectural Side-Channel Attacks", Proceedings of the 2020 on Great Lakes Symposium on VLSI, pp. 181–186, 2020. W442W7302

[48] M. Alam, S. Bhattacharya and D. Mukhopadhyay, "Victims Can Be Saviors: A Machine Learning–Based Detection for Micro-Architectural Side-Channel Attacks", J. Emerg. Technol. Comput. Syst., vol. 17, no. 2, 2021. W442W7302

[49] S. Briongos, G. Irazoqui, P. Malagón and T. Eisenbarth, "Cacheshield: Detecting cache attacks through self-observation", Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, pp. 224–235, 2018. W442W7302

[50] M. Chiappetta, E. Savas and C. Yilmaz, "Real time detection of cache-based side-channel attacks using hardware performance counters", Applied Soft Computing, vol. 49, pp. 1162–1174, 2016. W442W7302

[51] Z. Liu, B. Xu, B. Cheng, X. Hu and M. Darbandi, "Intrusion detection systems in the cloud computing: A comprehensive and deep literature review", Concurrency and Computation: Practice and Experience, vol. 34, 2021. W442W7302

[52] B. Gulmezoglu, A. Moghimi, T. Eisenbarth and B. Sunar, "Fortuneteller: Predicting microarchitectural attacks via unsupervised deep learning", arXiv preprint arXiv:1907.03651, 2019. W442W7302

[53] J. Ren, J. Guo, W. Qian, H. Yuan, X. Hao and H. Jingjing, "Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms", Security and communication networks, 2019. W442W7302

[54] R. A. Disha und S. Waheed, “Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique”, Cybersecurity, Bd. 5, Nr. 1, p. 1, 2022. W442W7302

[55] S. Russel and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 2010. W442W7302

[56] D. Rosenblatt, “The perceptron: A perceiving and recognizing automaton”, Cornell Aeronautical Laboratory, 1957. W442W7302

[57] S. Hochreiter and J. Schmidhuber, “Long short-term memory”, Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997. W442W7302

[58] F. T. Liu, K. M. Ting and Z.-H. Zhou, “Isolation forest”, 2008 eighth ieee international conference on data mining, pp. 413–422, 2008. W442W7302

[59] E. Fix and J. L. Hodges, “Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties”, International Statistical Review / Revue Internationale de Statistique, vol. 57, no. 3, pp. 238–247, 1989. W442W7302

[60] B. Ghojogh and M. Crowley, “Linear and quadratic discriminant analysis: Tutorial”, arXiv preprint arXiv:1906.02590, 2019. W442W7302