Waves of Knowledge: A Comparative Study of Electromagnetic and Power Side-Channel Monitoring in Embedded Systems

Michael Amar[1(✉)], Lojenaa Navanesan[1,2], Asanka P. Sayakkara[2], and Yossi Oren[1]

[1] Ben -Gurion University of the Negev, Beer Shea, Israel
amarmic@post.bgu.ac.il, {lojenaa,yos}@bgu.ac.il
[2] University of Colombo School of Computing, Colombo, Sri Lanka
asa@ucsc.cmb.ac.lk

Abstract. In today's interconnected world, Programmable Logic Controller (PLC) devices play a crucial role in controlling and automating critical processes across various sectors. This increased connectivity, however, also brings about significant security risks, including the threat of the PLC's control flow being subverted through malicious code injected by state-level actors. This paper offers an exploration of the use of side channels for control flow monitoring. By analyzing subtle variations in system behavior, such as power consumption and electromagnetic radiation, these side channels can be effectively leveraged to infer control flow information, and thus identify potential attacks. To accomplish this, we employ the emitted signals to train a machine learning model, and evaluate our detector by simulating two different types of attacks: malicious code injection and sensitive data infiltration. Additionally, we provide a unique comparison between the power consumption and electromagnetic side channels, highlighting the primary benefits each signal type exhibits in terms of detecting and preventing attacks. The results presented in this paper can aid system manufacturers in selecting the most suitable channel for defending their system, based on the specific requirements and context of their PLC application.

Keywords: Physical side-channel analysis · Malware detection · Malware monitoring · PLC environment · Firmware verification

1 Introduction

The rise in use of cyber-physical systems (CPS), which include smart vehicles, industrial systems, medical monitoring, robotics, and more, promises to modernize society and to reduce the burden of human labor [5,7]. They typically consist

Michael Amar, Lojenaa Navanesan, Both authors are considered co-first authors.

Y. Chen et al. (Eds.): SmartSP 2023, LNICST 552, pp. 158–170, 2024.
https://doi.org/10.1007/978-3-031-51630-6_11

of a large-scale, interconnected system of disparate elements that integrate computation with physical processes. CPS may significantly increase the effectiveness of industrial process control systems (ICS) [19]. Programmable logic controllers (PLCs) are a vital component of the CPS. They are used by industrial control systems (ICS) to link to and monitor critical infrastructure, employed in the manufacturing and process industries to reduce costs and enhance quality, and were created in response to the requirement to replace traditional programmed relay panels [8]. PLCs are designed for specific tasks combining multiple functionality in large industries. In recent times, PLCs have largely taken the place of the control components that were formerly used to execute the logic of the system [3]. PLCs gather data and communicate with sensors, motors, valves, and other equipment positioned throughout massive industrial systems to automate and control manufacturing processes. PLCs are *operational devices* – they are directly connected to the physical system in supervisory control and data acquisition (SCADA) in operational technology (OT) and information technology (IT) [12]. The PLCs themselves are typically observed using a remote human-machine interface [6,16].

The increasing reliance on industrial control systems (ICSs) has made them an attractive target for attackers looking to disrupt operations or gain unauthorized access to sensitive information. PLCs are particularly susceptible to attacks because of their widespread usage and lack of built-in security features. An attacker who gains control of a PLC can directly interfere with the underlying industrial processor and influence its interactions with the physical world. An attacker may also use this control of the PLC to exfiltrate secret data, such as sensor readings, which are exposed to the PLC. One of the most famous attacks on PLCs is the Stuxnet virus, that tampered with the code running on a PLC and disrupted the Iranian nuclear program by changing the rotational speed of the centrifuges [13]. Indeed, preventing attacks on the PLC environment is a very important task. While traditional security measures, such as firewalls, intrusion detection systems, and encryption, can help protect PLC environments they are not foolproof. As a result, many works have suggested PLC-specific attack deterrence and prevention measures [4,20,21]. One measure which has been suggested for detecting and preventing malware attacks on PLC environments is the use of non-intrusive passive integrity monitors. Abadi et al. [2] define Control Flow Integrity as *"security policy dictates that software execution must follow a path of a Control-Flow Graph (CFG) determined ahead of time"*; ensuring this policy can protect software from control flow hijacking attacks caused by buffer overflow, code reuse, or similar attacks. One approach for building this monitor is through the use of power or electromagnetic (EM) radiation side channels. An anomalous signal would imply a deviation from the predefined CFG, and can be an indication for an attack. Since the side channel-based monitor is air-gapped from the rest of the PLC by design, it is not susceptible to the same attack vectors as the PLC, thereby preventing any potential attacks on the monitor through similar channels.

Common approaches for passive code monitoring are the power and EM side channels. Both approaches have advantages and disadvantages, and were

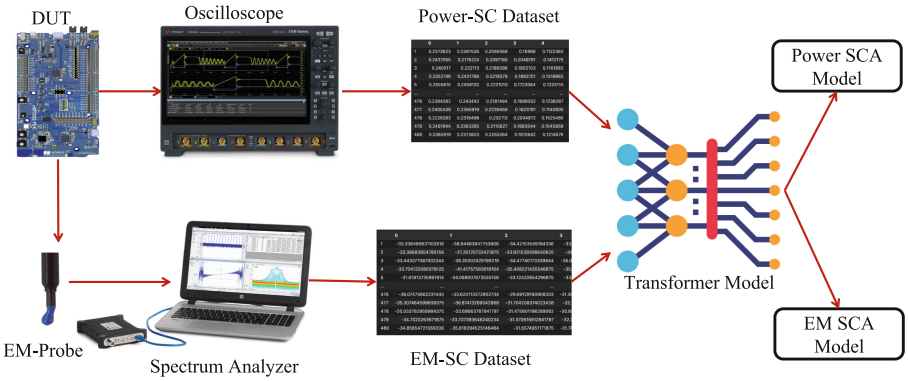


Fig. 1. *The process of acquiring power and electromagnetic side-channel traces from the target device and applying the deep-learning techniques to generate individual models for anomaly detection.*

suggested in several works previously, but they were never compared directly in a PLC setting. Liu et al. [14] were able to recover the program execution flow by observing the power consumption of a microcontroller; they inferred what instruction is most likely executed with an improved hidden Markov model. Han et al. [12] presented a non-intrusive EM based monitor, and trained an LSTM-based detector for signals in the time and frequency domains. We adopt some of their ideas as foundational concepts in our study.

In this paper, we investigate the suitability of physical side-channel analysis to monitoring and preventing attacks in PLC environments. Our approach leverages advanced signal processing and machine learning techniques to identify anomalous behavior in the physical signals produced by PLCs and detect potential attacks. We compare the power and EM side-channel approaches to identify the most effective side-channel medium to prevent and monitor attacks on the PLC environment. We leverage both EM and power consumption side channels to profile the behavior of our Device Under Test (DUT), and train a machine learning model based on the acquired signals. A general overview of our experimental environment can be seen in Fig. 1.

The contributions of this paper are as follows:

- We provide an anomaly detector to identify attacks against embedded controllers in time critical environments. We simulate two types of attacks, and evaluate our detector's effectiveness in detecting them on a popular controller.
- We present a transformer-based model architecture that is agnostic to the type of data used as input. The architecture is suitable for both EM and power consumption signals.
- We perform a comparison between the EM and power side channels, and provide criteria for choosing between them, considering the constraints of the PLC environment.

Our research, comparing the EM and power side channels, yields valuable insights and guidance for decision-makers to strategically select the optimal monitoring approach tailored to their unique environmental conditions. By leveraging this research, organizations can confidently implement robust security measures, fortifying their systems against potential attacks and safeguarding critical infrastructure with heightened resilience and effectiveness.

1.1 Background

The Power and EM Side Channels. A side channel can be defined as a medium through which sensitive information can be inadvertently revealed during the operation of a system. This medium can take various forms, such as power consumption, EM radiation, timing, or sound. Attackers can exploit side channels to extract sensitive information, such as cryptographic keys or passwords, without direct access to the system's memory [10]. From the defender's perspective, side channels can provide insight into the code that is currently being executed, allowing them to ensure the system's reliability.

In modern processors, transistors are continually switching on and off, causing a varying current and resistance in the digital circuit. In addition to this direct effect on power consumption, any metallic substance in proximity to the circuit acts as an antenna and transmits an electromagnetic wave in response. Usually the range of the EM waves is limited, and an amplifier is needed to enhance the strength of the signal. The shape and characteristics of the signals produced by these side channels are influenced by two main factors: the executed instructions and the processed data [15].

This effect makes it possible to gain insights about the instructions and the data by observing the side-channel trace, making this method a natural candidate for anomaly detection based on control flow monitoring. To demonstrate the effectiveness of side-channel measurements in determining which code is currently being executed, we ran four different applications (AES encryption, Matrix multiplication, Random number generator, Idle program) on our DUT, collected the resulting EM signals and plotted the first 3 PCA coefficients of these signals on a grid. As Fig. 2 shows, it is apparent that each application forms its own cluster, emphasizing the distinct waves emitted by the DUT during execution. As this pilot experiment illustrates, side-channel signals can clearly be analyzed to infer information about the code being executed and the data being processed.

Transformer Networks. Transformers are a type of neural network architecture that has gained popularity in recent years by improving the performance of natural language processing (NLP) tasks. They were first introduced in 2017 by Vaswani et al. [18] as an approach to machine translation, and have since then transformed the field of NLP. Previous approaches to NLP tasks involved the use of recurrent neural networks (RNNs) or convolutional neural networks (CNNs) to process sequences of words. However, these models have limitations in capturing long-range dependencies and suffer from the vanishing and exploding

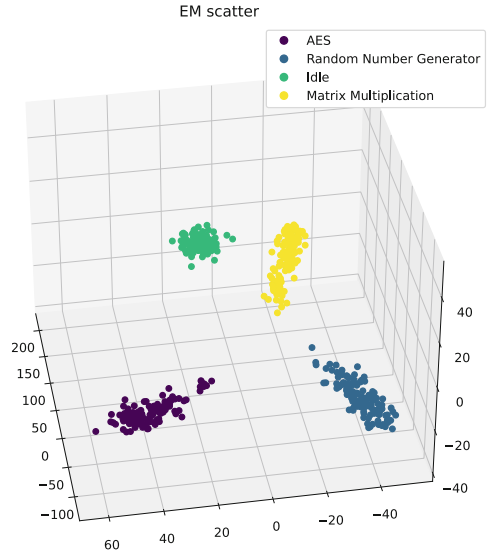


Fig. 2. *Scatter of four applications after performing PCA on their EM signals.*

gradient problems. Transformers are designed to address these issues by utilizing a self-attention mechanism that allows them to capture long-range dependencies without the need for recurrence. The self-attention mechanism in transformers allows the model to attend to different parts of the input sequence while processing each element. This is achieved through the use of attention weights that determine the importance of each input element to the output. One of the key benefits of the transformer architecture is its ability to process input sequences in parallel, making it much faster than traditional RNNs like LSTMs. LSTMs may, however, suffer from vanishing and exploding gradient problem, and their parallelization potential is limited since each time step depends on the previous one. The detection technique in this paper is based on the work of Han et al. [12]. They employed an LSTM-based model, which utilized a hidden state vector to represent the unobserved code, while the observables were the EM signals.

2 Methods

In order to establish a benchmark for comparing EM and power side channels, we developed an agnostic detector that identifies the executed code sections and also detects anomalies during runtime. Subsequently, we evaluated it using both EM and power signals. The results presented in this paper can serve as a guide for selecting the suitable monitoring medium based on the specific environmental conditions of the operator.

To demonstrate this idea, we used the Traffic Alert and Collision Avoidance System (TCAS) program provided by Han et al. [11]. The program is written

in C and meant to prevent midair collisions between airplanes. It receives as input information about the position and status of its own and approaching airplanes. We modified the code by adding function calls which use the general purpose input output (GPIO) pins of the DUT to signal the measurement tools the start and end of each *scan cycle*. Scan cycle is the term used to describe the repetitive manner in which PLCs operate: they execute a single program in an infinite loop which reads sensor inputs (e.g. water level sensor), performs control logic which defines the relations between the input and output values, then updates the actuators (e.g. activate a pump). Since the control logic does not change, and since the characteristics of the side signals are mainly influenced by the instructions and processed data, similar input values should yield similar side-channel emissions. In practice, however, a single program may have multiple control flows. Thus, different input values may cause different instructions to be executed, resulting in different side-channel emissions. To train a robust anomaly detector, there is a need for a diverse dataset which contains traces that represent multiple control flows of the program.

To accomplish that, Han et al. [11] fed the source code into the KLEE symbolic execution engine, a static analysis tool that produced multiple sets of variable assignments, where each set leads to a different execution path. For example, Fig. 3 shows a transition from the source code into a CFG. To follow the path 1→2→3→5→7→9 in that CFG, certain conditions have to be satisfied: $low \leq high$ and $x = v[mid]$. This set of constraints was next fed into a Satisfiability Modulo Theories (SMT) solver (e.g. Z3) which returns the exact values of the variables to follow the desired path. This process can be repeated multiple times to get full path coverage.

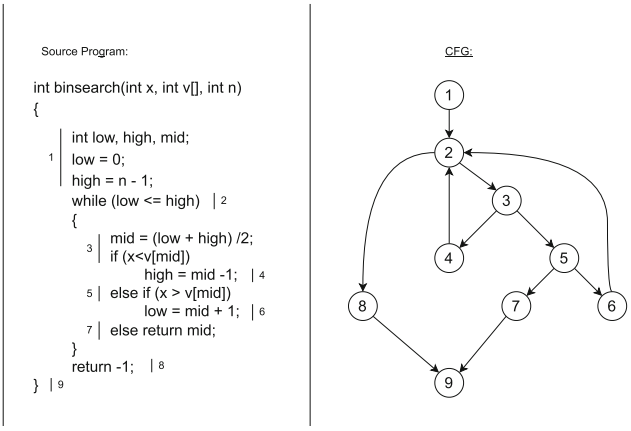


Fig. 3. *Transition from source code to Control flow Graph.*

After generating the test cases, we executed each test case while collecting the signals the controller emitted with dedicated equipment. The power and EM traces were collected simultaneously, to ensure similar operating conditions. We examined a total of 24 test cases representing different logic control flows.

The acquired signals form a dataset, where each sample is a side channel signal, and each label is the execution path that was executed to create it. This dataset was used to train a transformer-based classifier. If an attacker were to modify the control logic or hijack the control flow, the emitted signals would deviate from any known execution. In such case, the confidence of the classifier for all the possible control flows would be low. Thus, if the maximum confidence of the classifier over all possible control flows is lower than a predefined threshold, this should trigger an alert. The benefit of this approach is that the model can both observe the currently executed code and detect malicious execution. Additionally, this approach doesn't require any malicious samples for the training process.

We used a Nordic NRF52-DK as our DUT. This system is equipped with an ARM Cortex M4 processor running at 64 MHz with 64 KB RAM, and is designed for the I/O and digital signal control markets [1]. The DUT was connected to a Keysight B2962A low noise power source. For power consumption measurements, we used a Keysight MSOS604A Oscilloscope with 1GSa/s with High Resolution mode. We post-processed the power traces by averaging every 16 samples to reduce noise. For EM measurements, we used a Tektronix RSA306 Spectrum Analyzer together with a LANGER LF-U5 probe and PA 303 amplifier at a sampling rate of 56MS/s.

Measuring power consumption and EM emissions requires a preparation phase. To perform power measurements, the digital circuit needs to be modified by cutting the circuit and connecting a resistor and a probe in series to the power supply. To properly capture EM traces, it is necessary to both find the signal's peak frequency and to find the location with the strongest emissions. The peak frequency carries the most valuable features of the signal, and is unknown in advance. To find the correct frequency the program is executed, and the spectrum analyzer divides the input signal into its individual frequency components and displays their amplitudes. The peak frequency corresponds to the frequency with the highest amplitude. To find the location with the most dominant signals, we carried out a device cartography step, in which we used a Secure-IC XYZ positioning stage equipped with SMC100 single axis steppers to move the probe over several components, including the SOC and the MCU, and choose the component that showed the highest amplitude. We identified a location about the processor which showed the most noticeable signals. After performing these steps, the test cases corresponding to the different execution paths were executed while collecting both EM and power signals simultaneously. Each test case was executed 2000 times. The acquired traces are considered as a behavioral baseline of our DUT.

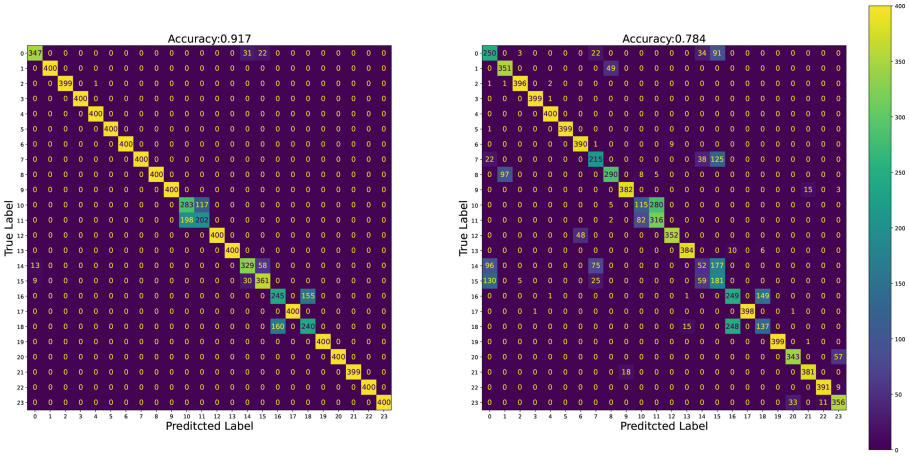
We trained a transformer-based classifier separately for each signal type after performing z-normalization, ensuring the signals are on the same scale. The

architecture of the model is inspired by [17], it consists of 8 transformer encoder blocks with 6 attention heads of size 256. Each block consists of two main components: attention and feed-forward. The attention part consists of a normalization layer followed by attention and dropout layers. The output of the attention component is residually connected to the input sequence and then passed to the feed forward layer. The feed forward component contains a convolutional layer followed by a dropout, convolutional and normalization layers. The output of the feed forward component is again residually connected to the output of the attention component. Finally, the output of the encoder blocks is fed into a global average pooling layer and then to a fully connected layer for classification. The model is trained using the AdaMax optimizer. Each model was trained for 10 epochs. We used 80% of the data for training and 20% for testing. The machine learning tasks were implemented in TensorFlow 2.6 for Python 3.9 and run on a cluster containing 58 NVIDIA GeForce GTX 1080 GPUs managed by the Slurm Workload Manager.

3 Results

As noted above, we were motivated to compare the effectiveness of the power and EM monitoring methods. Our models serve two purposes: monitoring the executed code, and identifying deviations from the known control flows. As for the ability of our models to correctly classify each signal to its relevant execution path, both models showed good accuracy. The classification accuracy for the EM and power consumption-based models was 91% and 78%, respectively, meaning that for a given trace, the model was able to classify it to the correct execution path, and infer the instructions sequence that was executed. The confusion matrices of the two classifiers are displayed in Fig. 4. The EM model shows good classification results along with some confusion between specific classes, while the power based model shows confusion between the same classes, along with additional misclassification between other classes. It is important to note that this confusion primarily arises because these classes possess only a few differing instructions, making it challenging to differentiate between them. Nonetheless, this confusion can be considered insignificant, since operational malware would incorporate its unique logic and deviate significantly from the program's known behavior.

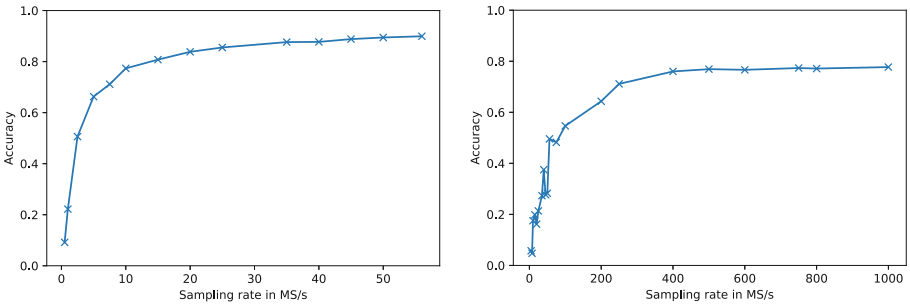
To evaluate the ability of the models to detect attacks, we simulated two types of attacks: a code injection attack, and data exfiltration attack. The code injection attack included injecting 5–10 assembly instructions to the source code, modelling an attacker interested in modifying the behavior of the control logic. In the second scenario, the data exfiltration attack, the UART pins of the device were used to leak one byte of sensitive information outside the system. In the presence of a signal that exhibits substantial deviations from the program's known behavior, the classifier's confidence level in all classes would be relatively low. Whenever the confidence level fell below a predefined threshold, we categorized it as an anomaly and initiated an alert.



(a) EM based model (b) Power based model

Fig. 4. Confusion matrices of the models

Both models successfully detected most of the attack scenarios, even the injection of only 5 assembly instructions. AUC reached 97% for the EM model and 91% for the power model. Interestingly, even though the power model was significantly less accurate than the EM model as a classifier, it is only slightly less effective as an anomaly detector. This emphasizes our claim that this approach would prove itself upon a case where the malware chunk has more volume. Garcia et al. [9] assert that the average size of malware, designed with a particular objective for embedded controllers (such as the corruption of the controller's output value), tends to be approximately 2 KB. For that reason, a confusion between the correct control flows which slightly differ from each other, would be negligible as the malware side signals become more dominant.



(a) EM-based model (b) Power-based model

Fig. 5. Effect of the sampling rate on the classification accuracy of the models.

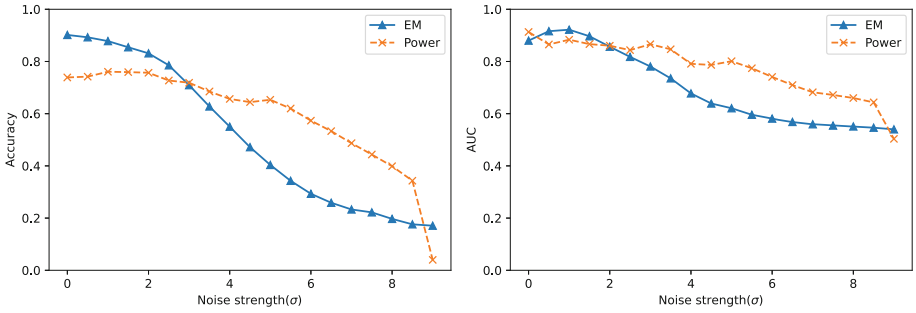
3.1 Performance Analysis

Sampling Rate: Obtaining high-precision equipment like an oscilloscope or a spectrum analyzer is useful to get more detailed analysis with higher sampling rates, but can also incur significant expenses. The sampling rate can impact other aspects, such as the storage and processing requirements for the captured data. Higher sampling rates generate larger amounts of data, which may necessitate more storage capacity and computational resources. We wanted to check the models' performance by exploring the impact of varying sampling rates, and retrained the models accordingly. Figure 5 shows the effect of the sampling rate on the classification accuracy on each signal type. We gradually increased the sampling rate of each signal, as shown on the x-axis. The scale of the x-axis in the graphs differs due to the significantly lower maximum sampling rate of the spectrum analyzer (Fig. 5a) compared to the oscilloscope (Fig. 5b). According to the figure, the EM classifier requires a much lower sampling rate to reach higher accuracy, reaching over 90% accuracy at a sampling rate of 56MS/s, whereas the power model reaches 78% at 1GS/s. Also, while the EM accuracy gradually improves with increasing sampling rate, the power models exhibit a sharp decline around 45MS/s. A possible explanation is that another component of the DUT operates at the same frequency, aliasing the captured signal. This phenomenon is not reflected in the EM classifier, since the measurements are much more localized and are focused only on the component in proximity to the probe.

Noise Resilience: The experiments in this research were conducted under ideal conditions, with minimal electromagnetic interference or environmental noise such as heat or humidity; in practice, however, noise is inevitable. To check the robustness of each model type to noise, we generated white noise with a mean of 0 and varying standard deviation and added it into the z-normalized signals. Figure 6 shows how the noise influences the models' performance. We varied σ, the strength of the noise, from 0 to 9 as shown in the x-axis, the y-axis shows the performance metrics (accuracy and AUC). The dashed orange line shows the performance of the power model while the solid blue line shows the performance of the EM model. When examining the figure, it becomes apparent that the EM model demonstrates superior performance in the presence of minor noise, yet it becomes surpassed as the noise level increases. This makes EM based monitoring the preferred choice for clean environments with minimal interference to the captured signal, where it is possible to place the probe in greater proximity to the measured component, while monitoring through power consumption may be in favor in rougher settings.

4 Discussion

Despite numerous works which have previously suggested side channel monitoring architectures in PLC environments, to the best of our knowledge, there was no comparative study between the EM and power channels for the described topic. This work provides a fair comparison of the power consumption and electromagnetic side channels. Often, due to budgetary and physical constraints, it becomes necessary to determine the appropriate signal type for a given application. The EM signals were much more centralized and precise, as the probe is placed above a component of interest. The model that was trained on the EM signals also showed higher performance metrics (e.g. accuracy, AUC) and required much lower sampling rate to converge, leading to lighter storage and computational requirements. On the other hand, EM capturing requires a preparation phase that includes finding the signal frequency of the examined program and locating the physical position that emits the most distinguishing signals. This process is unique to the DUT used, and must be repeated whenever the hardware configuration of the bill of materials (BOM) changes. The performance of the EM model was also decreased when faced with noisier samples, a realistic risk when taking into account the noisy environments where PLCs are typically found, as well as shielding and extra EM interference that other components of the system may produce. The power model, on the other hand, showed high resilience to noise, which is an important attribute, as often PLCs are exposed extreme environmental conditions. For the downside, measuring power requires a physical modification to the digital circuit which forces it to be shut down, a step which may be impossible in some OT settings. Power traces also include evidence about the consumption of the entire DUT rather than a single component like the processor. Additionally, the power model required much higher sampling rate to converge, resulting in high data volume and heavier processing power.



(a) Effect of σ on the Accuracy (b) Effect of σ on the AUC

Fig. 6. Impact of noise on the models performance.

Detecting anomalies and identifying potential attacks are of paramount importance in ensuring the security of embedded controllers. Leveraging the

power of side channels, this paper not only provides effective means for detecting such attacks, but also offers valuable insights to guide the selection of the most appropriate side channel medium, facilitating the design of robust defense mechanisms.

Acknowledgements. This research is supported by the U.S.-Israel Energy Center managed by the Israel-U.S. Binational Industrial Research and Development(BIRD) Foundation.

References

1. Nordic semiconductor. https://www.nordicsemi.com/-/media/Software-and-other-downloads/Product-Briefs/nRF52832-product-brief.pdf
2. Abadi, M., Budiu, M., Erlingsson, Ú., Ligatti, J.: Control-flow integrity principles, implementations, and applications. ACM Trans. Inf. Syst. Secur. **13**(1), 4:1–4:40 (2009). https://doi.org/10.1145/1609956.1609960
3. Alphonsus, E.R., Abdullah, M.O.: A review on the applications of programmable logic controllers (PLCs). Renew. Sustain. Energy Rev. **60**, 1185–1205 (2016). https://doi.org/10.1016/j.rser.2016.01.025, https://www.sciencedirect.com/science/article/pii/S1364032116000551
4. Alves, T., Das, R., Morris, T.: Embedding encryption and machine learning intrusion prevention systems on programmable logic controllers. IEEE Embed. Syst. Lett. **10**(3), 99–102 (2018)
5. Baheti, R., Gill, H.: Cyber-physical systems. Impact Control Technol. **12**(1), 161–166 (2011)
6. Basnight, Z., Butts, J., Lopez, J., Jr., Dube, T.: Analysis of programmable logic controller firmware for threat assessment and forensic investigation. J. Inf. Warfare **12**(2), 1–9 (2013)
7. Bhrugubanda, M.: A review on applications of cyber physical systems. Int. J. Innov. Sci. Eng. Technol. **2**(6), 728–730 (2015)
8. Erickson, K.: Programmable logic controllers. IEEE Potentials **15**(1), 14–17 (1996). https://doi.org/10.1109/45.481370
9. Garcia, L., Brasser, F., Cintuglu, M.H., Sadeghi, A., Mohammed, O.A., Zonouz, S.A.: Hey, my malware knows physics! attacking plcs with physical model aware rootkit. In: 24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017. The Internet Society (2017), https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/hey-my-malware-knows-physics-attacking-plcs-physical-model-aware-rootkit/
10. Genkin, D., Shamir, A., Tromer, E.: RSA key extraction via low-bandwidth acoustic cryptanalysis. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 444–461. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_25
11. Han, Y., Chan, M., Aref, Z., Tippenhauer, N.O., Zonouz, S.: Hiding in plain sight? on the efficacy of power side channel-based control flow monitoring. In: Proceedings of the USENIX Security Symposium (USENIX Security) (2022)
12. Han, Y., Etigowni, S., Liu, H., Zonouz, S.A., Petropulu, A.P.: Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations. CoRR **abs/1708.09099** (2017), http://arxiv.org/abs/1708.09099

13. Langner, R.: Stuxnet: Dissecting a cyberwarfare weapon. IEEE Secur. Priv. **9**(3), 49–51 (2011). https://doi.org/10.1109/MSP.2011.67
14. Liu, Y., Wei, L., Zhou, Z., Zhang, K., Xu, W., Xu, Q.: On code execution tracking via power side-channel. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1019–1031 (2016)
15. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks. Springer, Boston, MA (2007). https://doi.org/10.1007/978-0-387-38162-6
16. McMinn, L., Butts, J.: A firmware verification tool for programmable logic controllers. In: Butts, J., Shenoi, S. (eds.) ICCIP 2012. IAICT, vol. 390, pp. 59–69. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35764-0_5
17. Ntakouris, T.: Timeseries classification with a transformer model (2021). https://keras.io/examples/timeseries/timeseries_transformer_classification/
18. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
19. Wang, Y., Vuran, M.C., Goddard, S.: Cyber-physical systems in industrial process control. ACM Sigbed Rev. **5**(1), 1–2 (2008)
20. Yılmaz, E.N., Gönen, S.: Attack detection/prevention system against cyber attack in industrial control systems. Comput. Secur. **77**, 94–105 (2018)
21. Ylmaz, E.N., Ciylan, B., Gönen, S., Sindiren, E., Karacayılmaz, G.: Cyber security in industrial control systems: analysis of dos attacks against PLCs and the insider effect. In: 2018 6th International Istanbul Smart Grids and Cities Congress and Fair (ICSG), pp. 81–85. IEEE (2018)