INL/CON-21-63532-Revision-0

On the Limits of EM Based Detection of Control Logic Injection Attacks In Noisy Environments

October 2021

Kurt Vedros, Georgios Michail Makrakis, Constantinos Kolias, Min Xian, Daniel Barbara, Craig G Rieger

Changing the World's Energy Future

Idaho National Laboratory

INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, LLC

INL/CON-21-63532-Revision-0

On the Limits of EM Based Detection of Control Logic Injection Attacks In Noisy Environments

Kurt Vedros, Georgios Michail Makrakis, Constantinos Kolias, Min Xian, Daniel Barbara, Craig G Rieger

October 2021

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

On the Limits of EM Based Detection of Control Logic Injection Attacks In Noisy Environments

Kurt Vedros*, Georgios Michail Makrakis*, Constantinos Kolias*, Min Xian* Daniel Barbara† Craig Rieger‡
*Department of Computer Science, University of Idaho, Idaho Falls, Idaho 83402
Email: kvedros@uidaho.edu, makr7178@vandals.uidaho.edu, kolias@uidaho.edu, mxian@uidaho.edu
†Department of Computer Science, George Mason University, Fairfax, Virginia 22030
Email: dbarbara@gmu.edu
‡Idaho National Lab, Idaho Falls, Idaho 83402
Email: craig.rieger@inl.gov

***Abstract*—The difficulty in applying traditional security mechanisms in ICS environments makes a large portion of these mission-critical assets vulnerable to cyber attacks. Therefore, there is a dire need for the development of novel security mechanisms specifically designed to protect such critical systems. Recently a lot of attention has been given to the use of device EM emanations for defense purposes, and especially for the detection of possible anomalous behavior. Such approaches may lead to the development of robust external and non-intrusive anomaly detection mechanisms. Nevertheless, the majority of current work in the area neglects to consider the implications of real-life environments, particularly environmental noise. In this work, we explore the limits of EM-based anomaly detection to identify injection attacks in control logic software in noisy environments. Our study identified that indeed environmental noise might significantly degrade the anomaly detection process. Nevertheless, assuming that signals are captured with high sampling rates, even minor injections can be detected with above-90% accuracy in noisy environments where SNR is up to -2dB. Moreover, experiments in a real-life testbed attest that even single-instruction injections can be detected with near-perfect accuracy in relatively clean environments. Such attacks can still be detected reliably even in noisy environments after the application of noise-elimination techniques.**

***Index Terms*—cyber resilience; anomaly detection; side-channel analysis; cybersecurity; industrial control systems.**

I. Introduction

Industrial control systems (ICS) is an umbrella term that is used to describe all the devices, applications, network protocols, and procedures that are designed to monitor and govern all aspects of mission-critical processes, typically in industrial environments. Traditionally, ICS used to reside inside siloed networks which added an extra layer of protection. However, nowadays Industrial Internet of Things (IIoT) is becoming an integral component of modern ICS installations. Despite the clear advantages IIoT brings to the game, including a high degree of connectivity, continuous monitoring, and high-quality reporting, it exposes critical components to the dangers of insecure networks such as the Internet.

Devices that are the building blocks of ICS include Programmable Logic Controllers (PLCs), Intelligent Electronic Devices (IEDs) and Remote Terminal Units (RTUs) and IIoT smart-sensors. Such devices are designed to execute solely one task that is directly related to the physical process. Therefore, they tend to be severely limited in terms of resources which in turn leaves little-to-no room for supplementary security features. Therefore, mature security mechanisms such as antimalware tools cannot be natively implemented, and cryptographically secure communication protocols cannot be supported.

A family of novel defense systems is based upon the analysis of side channels that get emitted involuntarily by the device components [1], [2]. In the past, alternative side-channels have been considered, including the power consumption patterns [3], [4], [5], the thermal footprint [6] or the acoustic signals [7] of devices during their operation. Nevertheless, EM-based approaches [8], [9], [10], [11], [12] offer a comparative advantage since the signals themselves can be captured and analyzed in a completely non-intrusive fashion, i.e., no installation of software in the monitored device is assumed. Moreover, unlike analog signals describing the analysis of power consumption, the EM spectrum offers high bandwidth. In the same vein, unlike the analysis of thermal output, e.g., via infrared cameras, equipment to capture EM signals can do so in high sampling rates. Both of these factors contribute to a fast and fine-grained analysis.

Despite its advantages, EM-based analysis has been proven extremely sensitive to the placement of the monitoring equipment as well as external environmental conditions. Unfortunately, the majority of research works in the area have failed to consider the implications of real-life conditions that exist in industrial settings [13] and more specifically, the impact of noise which is omnipresent in such environments. In theory, noise may severally degrade the predictive accuracy of the existing EM-based anomaly detection techniques.

To the best of our knowledge, this work is the first that provides an ample evaluation of EM-based anomaly detection defenses in the influence of noise. Our study emphasizes minute modifications in the control logic that are likely to go unnoticed. The main purpose of this work is to identify the limits of EM-based anomaly detection and quantify the impact of several factors influencing the efficiency of the corresponding defense mechanisms.

Our experimental evaluation on synthetic data indicates

that it is possible to detect minimal code-injection attacks, even ones having a pollution rate of just 1%, with above 90% accuracy. This can be done despite the impact of strong environmental noise, even one reaching SNR levels of -8dB. In such cases, we identified that one of the most important factors influencing the anomaly detection process is the sampling rate. Our experimental results indicate that to achieve granular detection of such minute anomalies, the sampling rate should be at least eight times the speed of the CPU clock. Adopting sampling rates near the limits dictated by the Nyquist rate [14] often results in poor performance. More importantly, experiments conducted in a real deployment show that even single-instruction injections can be detected with above 90% accuracy on relatively "clean" environments if the sampling rate is high. Finally, we identified that the SVD-based method [15] for noise reduction provides near-perfect accuracy for the detection of even such sleazy attacks, even in extremely noisy environments (-10dB SNR).

The rest of the paper is organized as follows: Section II provides the necessary technical background and outlines the main terms and definitions that are necessary for understanding the concepts discussed in the paper. The next section describes the anomaly detection method used as the basis for evaluating all experiments. Section IV and V provides a description of all the experiments done using synthetically generated data and data obtained from a real-life testbed respectively. Finally, the conclusions extracted from this study, along with directions for subsequent research, are included in section VI.

II. Technical Background & Definitions

This section provides the necessary background information and terminology that will be used in the rest of the paper.

Industrial devices such as smart-sensors and PLCs, which typically interact directly with physical processes have significant differences from mainstream high-end computers. For example, they have much lower hardware capabilities, including a CPU speed of just a few Mhz, and a limited amount of memory. They also tend to rely on real-time operating systems (RTOS) or execute instructions directly at the hardware level ("bare metal"). Typically, they operate continuously under harsh environmental conditions, including high levels of noise and interference. Moreover, they are replaced after many years of continuous operation with only seldom update cycles throughout their lifetime. Finally, such devices are usually designed to perform simple and well-defined tasks, but with extremely high levels of reliability, according to the doctrines of embedded systems designs [16], [17].

The software that runs in such systems is responsible for the governing and/or inspection of a *physical process*. This software is known as *control logic*. This type of software typically runs perpetually, in a loop fashion. Although control logic can become complex, it typically has a much simpler structure in comparison to software seen in high-end IT systems.

Numerous works indicate that the mechanisms responsible for the updating of the control logic are plagued by weaknesses. These revolve around the naive, passwords-based authentication, as well as poor design decisions regarding the use of cryptographic primitives [18], [19]. Alternatively, attackers may exploit existing bugs in the control logic itself. The latter is also known as *control logic hijacking* or a *control logic injection* attack. For example, Tychalas et al. [20] pointed out the applicability of *buffer overflow* methodologies in embedded systems. Regardless, the outcome of both methodologies is the alteration in the sequence of the set of machine-level instructions of a specific execution path $\mathcal{C}_i$ with the intention of affecting the physical process itself. The end result may vary from harming equipment to a large-scale environmental damage which in turn may endanger human lives.

EM-based anomaly detection capitalizes on the relationship between instructions executed at the machine level and the EM signals emitted involuntarily as the natural outcome of this process. In further detail, every time a new instruction is executed of a digital device, slightly different amounts of current are drawn by the CPU. This results in the formation of EM fields, and therefore, the emanation of EM signals. Theoretically, these signals can act as indicators of the type of instructions executed by the CPU at any given time. The work in [21] clearly presents the correlation between the CPU activity, the current drawn, and the creation of EM signals. Specifically, the CPU may act as a transmitter that performs amplitude modulation over a carrier, i.e., the CPU clock.

Before we proceed with our investigation, we shall provide definitions that are important for the understanding of the rest of the paper.

We define control logic injections in the discrete time-domain representation of the EM signal as follows: Assume a signal S^i_m of length m, which is normally obtained by the execution of instructions in the execution path $\mathcal{C}_i$. Assume a signal $\Delta^{\mathcal{M}}_n$ of length n, that is obtained by the execution of an arbitrary sequence of malicious instructions $\mathcal{M}$. Then, the signal corresponding to a control logic injection is a version of that signal that is obtained as in Equation 1:

$$S^{\mathcal{M}}_{m+n} = S[1...k], \Delta^{\mathcal{M}}[1...n], S[k+1...m+n] \qquad (1)$$

where k is the point of injection.

The anomaly detection task falls down to discerning whether an unknown signal S_q corresponds to a known, normal execution path $\mathcal{C}_i$ or a polluted one $\mathcal{C}^{\mathcal{M}}_i$. Several factors have been identified to play a detrimental or augmentative role in the anomaly detection process including (a) the extent of the modification denote here as pollution rate, the levels of noise here described as signal-to-noise-ration, and the sampling rate at which the signal is obtained.

Definition 1: *Pollution Rate* (PR), is a metric of the extend of the corruption inflicted to the original executional path. It is calculated by the Equation 2.

$$PR = \frac{|\mathcal{M}|}{|\mathcal{C}^{\mathcal{M}}_i|} \qquad (2)$$

More specifically, PR is the ratio of maliciously injected (at the machine level) instructions $|\mathcal{M}|$, to the total number of instructions that constitute the studied executional path is calculated as in 3.

$$|\mathcal{C}_i^{\mathcal{M}}| = |\mathcal{M}| + |\mathcal{C}_i| \tag{3}$$

where $|\mathcal{C}_i|$ is the number of instructions comprising the original path. Theoretically, the smaller the PR the harder it is to identify a discord by simply observing the EM spectrum.

Definition 2: *Signal-to-Noise-Ratio* (SNR), is a measure of the quality of the signal. SNR can be calculated as in Equation 4.

$$SNR = 10log\frac{P_s}{P_n} \tag{4}$$

where P_s is the power of the signal and P_n is the power of noise. SNR is measured in dB. Theoretically, the lower the SNR, the harder it becomes to identify whether a series has been subjected to alterations.

Definition 3: *Sampling Rate* (SR) is the number of elements in a discrete-time sequence describing an analog signal. According to the Nyquist theorem, [14] the minimum SR should be at least two times greater than the frequency of the signal in order for the resulting sequence to be free of *aliasing*. Theoretically, with higher sampling rates, a discord will be described by more data points.

III. Anomaly Detection Method

We relied on a rudimentary anomaly detection method that is based on the k-Nearest Neighbors (k-NN) algorithm, for evaluating purposes.

Reason for Choice: In its most popular version, k-NN is a supervised ML algorithm. This implies that instances (EM signals) of normal and anomalous states must be given a priori. However, in this context, it is not valid to assume knowledge of all anomalous cases. For example, assuming that an attacker has identified a vulnerability in code, they can inject a single instruction to alter the value of a potentially critical variable. In fact, any of the numerous logical combinations of instructions that the CPU architecture supports can potentially be injected (list of AVR instructions can be retrieved from [22]). Indeed, the alternative ways to create anomalies are potentially infinite. Therefore, providing labels for anomalous cases is impractical. Towards this end, we relied on an unsupervised version of k-NN where knowledge of only the *normal* case is assumed. This is a realistic assumption as such observations can be captured at an early stage, even before the deployment of the system. Another reason for this choice of algorithm is the fact that k-NN performs *lazy learning* i.e., no model is created during the training step, but rather the modeling process is delayed until the deployment phase. Indeed, in this work, our aim is not to evaluate the *generalization* ability of a specific algorithm across different noise environments, but rather to quantify the effects of noise in distinguishing between normal and anomalous signals.

Training: As a first step the algorithm is fed with a set of normal (only) observations (signals) X. Then, an exhaustive process of comparing each signal x_i with the rest in this set takes place to infer a distance score D_i of x_i. More specifically, the distance score D_i is calculated by formula 5:

$$D_i(X, x_i) = \frac{\sum_{1}^{k} min_k(d(x_j, x_i) \forall x_j \neq x_i \in X)}{k} \tag{5}$$

where d is a distance metric such as Euclidean distance, and min_k is the number of k closest neighbors to x_i. Finally, the values $\{D^{min}, D^{max}\}$ are treated as the thresholds of normalcy. The sequence of these steps is described in Algorithm 1.

Algorithm 1 k-NN-based Threshold Inference Phase

```
function InferThresholds:(benign dataset X)
    for ∀x_i ∈ X do
        D ← D_i(X, x_i)
    end for
    D^min ← min(D)
    D^max ← max(D)
     return D^min, D^max
end function
```

Deployment: During the deployment stage, a new observation x_q is acquired whose label is unknown (normal, anomalous). Once again, an exhaustive process of comparing that signal with the rest in set X takes place to calculate the score D_q. Finally, if $D^{min} \leq D_q \leq D^{max}$ then the observation is flagged as normal, or anomalous otherwise. These steps are outlined in Algorithm 2.

The reader should take into account that increasing the efficiency of anomaly detection is not the objective of this work. The described anomaly detection method is chosen as the basis to conduct a comparative study regarding the limits of EM-based anomaly detection under the influence of various factors.

Algorithm 2 k-NN-based Anomaly Detection Phase

```
function Detection:(benign dataset X, thresholds D^min, D^max, test observation x_q)
    status_q ← normal
    score_q ← D_q(X, x_q)
    if score_q ≤ D_min or score_q ≥ D_max then
        status_q ← anomalous
    end if
     return status_q
end function
```

IV. Experiments Using Synthetic Data

Synthetic Datasets: For the sake of simplicity, in our experiments, the period T of the signals that are treated as normal is set to one second (unrealistically long), and the corresponding frequency F of the CPU is set to 1000 (unrealistically slow). Each instruction ranges between amplitudes 3mV to 6mV. Since there is a finite number of supported instructions by CPU architectures, and since each instruction (combination) modulates the signal in a different way the resulting amplitude

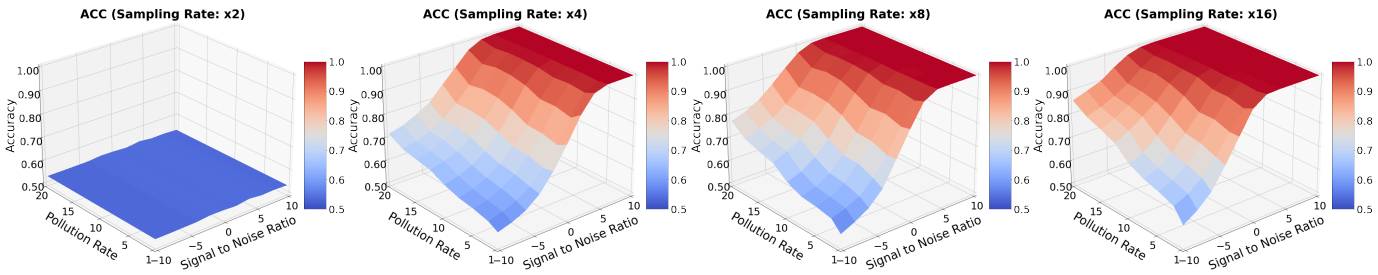

Fig. 1: Accuracy achieved for various pollution rates, SNR and sampling rates.

levels are also finite. The base dataset is comprised of 1000 examples of the same basic signal with random (but minor) fluctuations of the amplitude ranging from -0.2mV to 0.2 mV per example.

To effectively conduct our study, we produced several variations of this basic dataset, by altering specific characteristics, namely, the SR, PR, and SNR. Different SR levels were considered, starting from 2 and increasing up to 32 times, higher than the base frequency of the signal with an increment step of 2 (for example, x2, x4, ..., x32). The anomalous signals are created by injecting an invalid, never-seen-before sequence exactly in the middle of all benign signals. Different PR were considered, ranging from 1% to 20% with an increment step of 2.5% (for example, 1%, 2.5%, ..., 20%). We considered variations after applying different levels of additive white Gaussian noise (AWGN). Finally, various noise levels were taken into account, ranging from -10dB (i.e., the noise is 10 times stronger than the signal) to 10dB (i.e., the signal is 10 times stronger than the noise) with an increment step of 2dB (for example, -10dB, -8dB, ..., 10dB).

Evaluation Method: For this set of experiments the efficiency was measured based on the accuracy (ACC). We define ACC as:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

where TP is the total number of *true positives*, TN the number of *true negatives*, FP the number of false positives, and FN as the number of *false negatives*.

In all the experiments, the ten-fold cross-validation strategy is used for evaluation purposes. For every *fold* the number of observations used for training purposes was 90% of the entire normal dataset. The testing set was 10% of the normal dataset, and an equal number of anomalous observations was added. For all metrics, the average across of all folds along with the worst-case scenario are calculated and reported.

A total of 1000 thresholds was considered ranging from the minimum distance (0) to the maximum distance observed. The ACC reported for each fold is the maximum among all thresholds considered. The ACC of the neighborhood is the mean ACC for all folds. We considered neighborhoods of 3, 5, 10, 25, 50, 75, and 100 neighbors. The overall accuracy reported is the best among all the neighborhoods.

Estimating the Impact of Pollution Rate: In the first round of experiments, the main question posed was the following: Assuming a high SR (i.e., x32), what is the impact of noise considering various PR levels? The results indicate that a high PR level, i.e., 20%, can be detected with high accuracy of above 0.9 even in extremely noisy environments, i.e., SNR -10dB. Interestingly, the ACC is maintained at near-perfect levels (i.e., 0.963) even at SNR -6dB. PR of 10% can also be reliably detected with above 0.9 ACC for SNR greater than -4dB. Finally, the smallest considered PR i.e., 1% was detected effectively with above 90% for up to SNR -2dB. In fact, the ACC was constantly near-perfect 0.988 for up to 0dB SNR. Nevertheless, the ACC rapidly degrades for every considered level below -2dB, by almost a -10% per noise level. Figure 2 depicts the ACC achieved for various pollution rates and signal-to-noise ratios when the sampling rate is kept constantly high at x32.

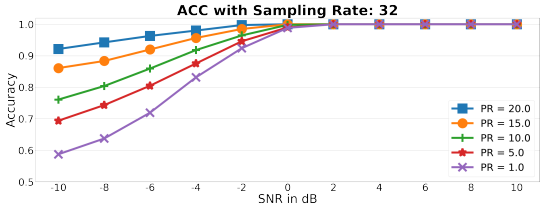

Fig. 2: Accuracy for various levels of pollution rates and signal-to-noise-ratios.

Estimating the Impact of Sampling Frequency: In the light of the previous experiment we wanted to answer the following question: Focusing solely on the detection of injection attacks of the smallest PR (1% PR), what is the impact of noise for various SR? The case SR of x32 was analyzed in the previous experiment. For lower SR levels of x16, the anomaly detection yields near-perfect accuracy of 0.958 or higher when SNR levels are up to 0dB. In comparison to higher SR levels, one may notice that SR of x32, x24, and x16 yield similar accuracy. Only when considering SR x8 can one observe drastic changes in accuracy by raising the SR. At SNR -2dB and -4dB, the biggest change of approximately 0.150 in accuracy can be seen in comparison to x16 SR. At the lowest SR tested (x2), injections cannot be detected with any certainty. The accuracy at every SNR level was approximately 0.500. Figure 3 accuracy for various levels of sampling rates

and signal-to-noise-ratios (the pollution level is constant 1%).

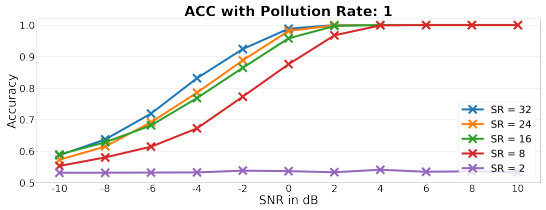

Fig. 3: Accuracy for various levels of sampling rates and signal-to-noise-ratios.

Figure 1 summarizes the ACC achieved across all PR, SR, and SNR levels considered. The main conclusions drawn from these experiments are: (a) The higher the PR, the more robust against the influence of noise the anomaly detection becomes, (b) Injection attacks at any PR level (1%-20%) can be detected with near-perfect accuracy when the SNR is no less than 0dB. (c) Very small PR can effectively be detected even in the presence of high environmental noise. (d) SR must be above x2 to effectively detect any injection code. (e) The higher the SR, the greater the resistance to noise, especially for low SNR levels. (f) SR faster than x16 may improve the ACC but only marginally.

V. Experiments Using Real-Life Equipment

Experimental Setup: For all experiments involving real-life equipment, the target device is an Arduino Mega. The Arduino Mega is equipped with the 8-bit ATmega2560 AVR microcontroller unit (MCU). This MCU is widely deployed in real-time control applications [23]. The Arduino device family allows the developers to run code at the firmware level without the intervention of an OS or a runtime environment.

To acquire EM signals, we make use of a near-field probe, namely an EMRSS RF Explorer H-Loop, which is placed atop of the CPU. Due to the fact that emanations from the CPU are transmitted unintentionally, the EM signals have a very low amplitude. For that reason, each signal captured is first amplified using a Beehive 150A EMC probe amplifier and is then saved in a digital format using a PicoScope 3403D oscilloscope. The sampling rate is set to 250MS/sec, which corresponds to a sampling interval of 4ns. The CPU clock of the ATmega2560 is 16Mhz which implies that the chosen sampling rate is roughly 15 times higher. The experimental setup can be seen in Figure 4.

The control logic coded in the Arduino is a basic tank-filling system. The control logic was implemented in the AVR assembly language. Compared to C, which is the typical choice for developing software in the Arduino platform, the use of assembly allows granular modifications analogous to real adversarial activity.

The chosen adversarial cases include the injection of a NOP, ADD, and JMP instructions. In other words, all three malicious cases involve the injection of a single instruction that consumes as little as one (NOP), one (ADD), and three (JMP) cycles, respectively.

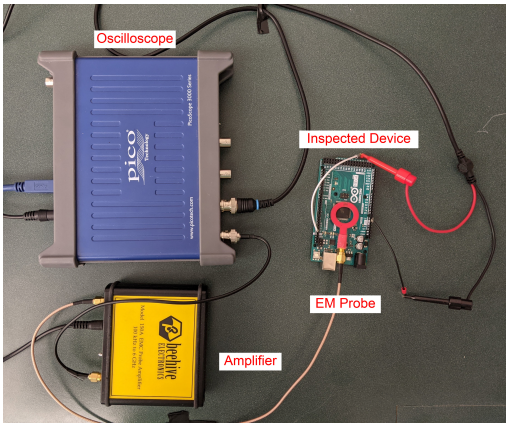

Fig. 4: Experimental setup considered for the acquisition of EM signals.

One may claim that code injections of such small caliber are not meaningful from an adversarial point of view. However, we argue that an extraneous NOP could potentially slightly increase the duration of a scan cycle, increasing the frequency of recalibrating the device. Similarly, introducing a malicious ADD instruction could alter the value of an important variable. Finally, the introduction of a JMP can be used for skipping entire blocks of code. To the best of our knowledge, the scenarios considered involve the smallest-yet-meaningful modifications ever considered in relevant bibliography. An expert of the adopted benign control logic (in assembly) is given in Figure 5. The code snippet includes an indication of the location of the malicious injections.

```
1  .....
2    ; just a few NOPs
3    nop
4    nop
5    nop
6    nop
7    nop                  1 nop
8    nop                  2 add r2, r0
9    [redacted]           3 jmp label1
10   ; read the lowLevel input (arduino pin 11)
11   clr   r20
12   sbic  pinb, 3
13   ldi   r20,  1
14 .....
```

Fig. 5: The location of the injected instructions in the benign program.

Examples of the EM waves generated during the execution of the depicted instructions are given in Figure 6. The reader can observe the impact of the injection of the malicious instructions highlighted in red. Noticeably, the injection of the NOP causes a displacement of one cycle, the ADD one cycle, and the JMP three cycles.

Evaluation: The predictive accuracy was evaluated using the following metrics: (a) accuracy (ACC), (b) F1 score, and (c) the area under the curve (AUC) score of the corresponding receiver operator characteristic (ROC) curve. While the accu-

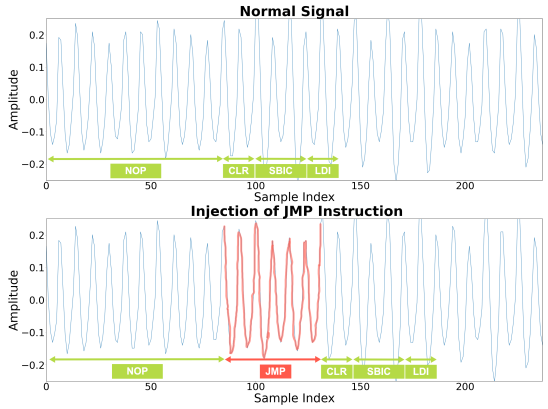

Fig. 6: Examples of benign and malicious signals at the location of injection. It is clear that the injection of malicious instructions causes an analogous displacement.

racy has been defined in the previous section, the F1 score can be defined as the harmonic mean of the precision (PPV) and sensitivity (TPR) as:

$$F1 = 2 * \frac{PPV * TPR}{PPV + TPR} \tag{7}$$

where, in turn the precision is defined as:

$$PPV = \frac{TP}{TP + TN} \tag{8}$$

while the sensitivity is defined as:

$$TPR = \frac{TP}{TP + FN} \tag{9}$$

An ROC is a popular metric for describing the efficiency of anomaly detection systems. It is essentially a graph illustrating the true-positive rate (TPR) vs. the false positive rate (FPR) for various thresholds. This graph is usually a curve and since the shape of the curve is arbitrary the most common method for comparing two ROC curves is by measuring the area under the curve (AUC).

Detecting Single-Instruction Injections: Based on the previous experiments, we have strong evidence that it is possible to detect code injections in noisy environments of very low PR. But how low can we go? Is it possible to reliably detect single-instruction injection attacks by analyzing the EM signals, at least in a clean environment?

In this set of experiments, the average AUC score achieved is comparable for all three cases: 0.953 for both the NOP and ADD instructions, while for the JMP instruction, the AUC score was 0.951. The average ACC is 0.975, and the average F1 score is 0.977 for all malicious cases.

Figure 7 illustrates the average distance of the 3-nn of each signal (normal or anomalous) to the normal signals. The reader may observe that with the exception of few outliers (estimated below 5%) both normal and anomalous observations consecrate within specific distance ranges that are clearly separable.

Without a doubt, *the injection of the smallest, meaningful piece of code can be detected effectively with high precision in clean environments if the sampling rate is high.*

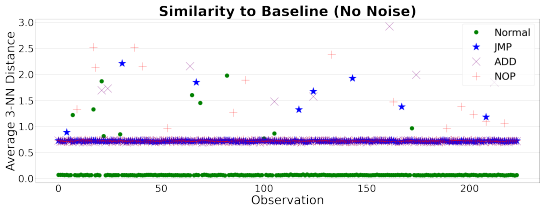

Fig. 7: Average distance of three closest neighbors of each individual observation (normal or anomalous) in the dataset to the totality of normal observations that is considered as the baseline.

Impact of Noise: The purpose of the second experiment was to evaluate the impact of AWGN noise to the effectiveness of the anomaly detection process. Five variations of the dataset were created by artificially injecting noise of decreasing SNR levels (10dB, 5dB, 0dB, -5dB, -10dB) to the original noise-free datasets. The average AUC scores across the considered SNR levels are illustrated in Figure 8. Detailed scores for all the considered noise levels are given in Table I.

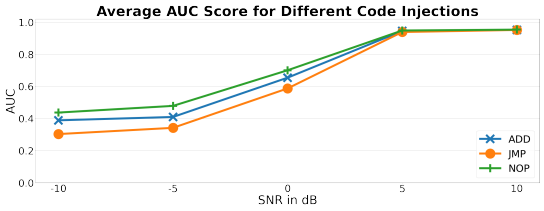

Fig. 8: Average (among all folds) AUC scores achieved across different SNR levels.

One can immediately identify that for all levels, all metrics are maintained close to those achieved in a clean environment for SNR levels down to 5dB. Then, the average AUC score for all considered anomalous cases, drops rapidly for SNR levels below 5dB. More specifically, the average AUC score drops from approximately 0.950 (clean environment) to 0.700 (NOP), 0.654 (ADD), and 0.587 (JMP) at 0dB. This indicates a drastic 26% drop for the detection of a NOP, a 31% reduction in the detection of an ADD and a 39% drop in the identification of an JMP. In other words, the drop in AUC score at 0 SNR ranges from approximately 25% to near 40% depending on the type of the instruction injected. Then, it further degrades to 0.5 (random coin toss) or below for all SNR levels less than 0dB. For example, at SNR -5dB the AUC score is just 0.478 (NOP), 0.409 (ADD), and 0.342 (JMP).

As a conclusion, with this experiment, we can state that *single-instruction injections are expected to be identified with high precision for SNR levels of approximately 5dB or above, assuming that the sampling rate is high (e.g., x15 the CPU clock).*

Impact of Noise Filtering: The purpose of the third experiment is to quantify the positive impact (if any) of well-known

TABLE I: Average AUC, Accuracy, and F1 scores acquired for detecting three types of injections, under various noise levels.

		SNR (dB)				
		-10	-5	0	5	10
AUC	NOP	0.437	0.478	0.700	0.948	0.953
	ADD	0.389	0.409	0.654	0.945	0.952
	JMP	0.303	0.342	0.587	0.938	0.951
ACC	NOP	0.555	0.571	0.687	0.957	0.975
	ADD	0.535	0.540	0.661	0.946	0.975
	JMP	0.522	0.529	0.641	0.939	0.975
F1	NOP	0.679	0.675	0.737	0.958	0.976
	ADD	0.671	0.676	0.717	0.948	0.976
	JMP	0.670	0.668	0.698	0.940	0.976

TABLE II: Average AUC, Accuracy, and F1 scores acquired for detecting alternative injections, under various noise levels, after the application of noise elimination (Wiener, kNN, SVD).

			SNR (dB)				
			-10	-5	0	5	10
Wiener	AUC	NOP	0.612	0.786	0.928	0.945	0.976
		ADD	0.612	0.752	0.927	0.940	0.977
		JMP	0.504	0.753	0.921	0.941	0.968
	ACC	NOP	0.680	0.774	0.948	0.957	0.975
		ADD	0.644	0.741	0.946	0.959	0.980
		JMP	0.591	0.748	0.937	0.957	0.975
	F1	NOP	0.724	0.794	0.953	0.960	0.977
		ADD	0.709	0.768	0.717	0.962	0.981
		JMP	0.693	0.769	0.945	0.960	0.977
kNN	AUC	NOP	0.534	0.576	0.842	0.952	0.953
		ADD	0.546	0.523	0.835	0.951	0.952
		JMP	0.516	0.485	0.778	0.951	0.951
	ACC	NOP	0.618	0.618	0.819	0.975	0.975
		ADD	0.606	0.587	0.817	0.975	0.975
		JMP	0.595	0.564	0.765	0.975	0.975
	F1	NOP	0.693	0.687	0.829	0.976	0.976
		ADD	0.689	0.677	0.824	0.976	0.976
		JMP	0.684	0.681	0.784	0.976	0.976
SVD	AUC	NOP	0.975	0.963	0.953	0.952	0.953
		ADD	0.972	0.962	0.951	0.951	0.952
		JMP	0.972	0.961	0.952	0.951	0.951
	ACC	NOP	0.982	0.979	0.975	0.975	0.975
		ADD	0.979	0.979	0.975	0.975	0.975
		JMP	0.982	0.979	0.975	0.975	0.975
	F1	NOP	0.982	0.980	0.976	0.976	0.976
		ADD	0.980	0.980	0.976	0.976	0.976
		JMP	0.982	0.980	0.976	0.976	0.976

noise filters and noise elimination techniques. Towards this end, we selected representative methods of alternative families of noise reduction/elimination approaches for example, ones based on signal processing, machine learning, and subspace methods. More specifically, we utilized (a) Wiener filter [24], a method that applies linear time-invariant (LTI) filtering to the noisy signals, (b) k-NN regressor [25], (c) and singular value decomposition (SVD) based denoising [15]. Detailed results for each metric and each individual command are given in Table II.

All filters have a positive impact when the SNR levels become significant, e.g., when SNR is 0dB or below. Figure 9 illustrates this trend, for the example of the injection of a NOP instruction.

The SVD denoising is a peculiar case as, unlike other

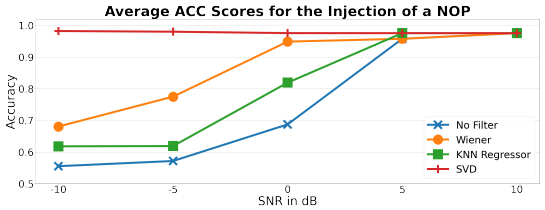

Fig. 9: An example of the average (among all folds) ACC score achieved after the application of noise elimination filters, for the NOP case.

approaches, the SVD method achieved scores comparable to the ones obtained in a clean environment for all metrics, regardless of the SNR levels. This possibly indicates an advantage of signal subspace methods when applied for purposes of anomaly detection that requires further investigation. Figure 10 illustrates the average similarity of individual observations (normal or anomalous) when compared to the baseline for the clean, noisy (SNR of -10dB) and denoised versions of the signals. Notice that the separation between normal and anomalous observations become as clear as in the clean dataset when SVD denoising is applied. To further illustrate the improvement achieved, Figure 11 presents the best ROC curve among all folds in SNR -10dB vs. the worst ROC obtained after the application of SVD filters.

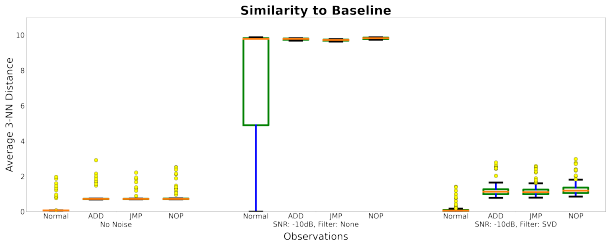

Fig. 10: Similarity of individuals observations to the baseline on the clean dataset (left), a noisy version at -10dB (middle), after applying SVD noise filter (right).

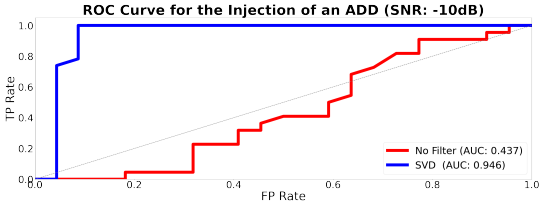

Fig. 11: Improvement on the AUC best fold on noisy environments (-10dB) and the worst fold after the application of SVD filter.

VI. Conclusion & Future Work

In this work, we explored the limits of EM-based anomaly detection approaches towards detecting malicious code injection attacks in control logic software. The main conclusion extracted from our study is that while noise may severally

degrade the accuracy of the anomaly detection, code injections that alter the execution path minimally to moderately can still be detected even under highly noisy environments, if the sampling rate is high enough. Through experiments with real-life equipment, we prove that even single-instruction injections can be identified with high accuracy in clean or moderately noisy (0dB) environments. Standard noise elimination techniques may drastically improve the accuracy of the anomaly detection task even in sub 0dB environments.

Our future research efforts will be focused on investigating the transferability of ML-based anomaly detection models among environments of different levels of noise. Towards this goal, we aim to improve existing domain adaptation techniques that may prove beneficial for this application. Furthermore, we intend to integrate the principles of SVD denoising into ANN autoencoder approaches to achieve optimal (in a feedback loop fashion) results.

ACKNOWLEDGMENT

Effort performed through Department of Energy under U.S. DOE Idaho Operations Office Contract DE-AC07-05ID14517, as part of the Laboratory Directed Research and Development (LDRD) and Resilient Control and Instrumentation Systems (ReCIS) program of Idaho National Laboratory.

REFERENCES

[1] N. Boggs, J. C. Chau, and A. Cui, "Utilizing electromagnetic emanations for out-of-band detection of unknown attack code in a programmable logic controller," in *Cyber Sensing 2018*, vol. 10630. International Society for Optics and Photonics, 2018, p. 106300D.

[2] R. A. Riley, J. T. Graham, R. M. Fuller, R. O. Baldwin, and A. Fisher, "A new way to detect cyberattacks: extracting changes in register values from radio-frequency side channels," *IEEE Signal Processing Magazine*, vol. 36, no. 2, pp. 49–58, 2019.

[3] S. S. Clark, B. Ransford, A. Rahmati, S. Guineau, J. Sorber, W. Xu, and K. Fu, "Wattsupdoc: Power side channels to nonintrusively discover untargeted malware on embedded medical devices," in *2013 {USENIX} Workshop on Health Information Technologies (HealthTech 13)*, 2013.

[4] Y. Liu, L. Wei, Z. Zhou, K. Zhang, W. Xu, and Q. Xu, "On code execution tracking via power side-channel," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 1019–1031.

[5] S. Wei, A. Aysu, M. Orshansky, A. Gerstlauer, and M. Tiwari, "Using power-anomalies to counter evasive micro-architectural attacks in embedded systems," in *2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2019, pp. 111–120.

[6] M. A. Islam, S. Ren, and A. Wierman, "Exploiting a thermal side channel for power attacks in multi-tenant data centers," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1079–1094.

[7] S. D. D. Anton, A. P. Lohfink, and H. D. Schotten, "Discussing the feasibility of acoustic sensors for side channel-aided industrial intrusion detection: An essay," in *Proceedings of the Third Central European Cybersecurity Conference*, 2019, pp. 1–4.

[8] N. Sehatbakhsh, M. Alam, A. Nazari, A. Zajic, and M. Prvulovic, "Syndrome: Spectral analysis for anomaly detection on medical iot and embedded devices," in *2018 IEEE international symposium on hardware oriented security and trust (HOST)*. IEEE, 2018, pp. 1–8.

[9] Y. Han, S. Etigowni, H. Liu, S. Zonouz, and A. Petropulu, "Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1095–1108.

[10] A. Nazari, N. Sehatbakhsh, M. Alam, A. Zajic, and M. Prvulovic, "Eddie: Em-based detection of deviations in program execution," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 2017, pp. 333–346.

[11] C. Kolias, D. Barbará, C. Rieger, and J. Ulrich, "Em fingerprints: Towards identifying unauthorized hardware substitutions in the supply chain jungle," in *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2020, pp. 144–151.

[12] C. Kolias, R. Borrelli, D. Barbara, and A. Stavrou, "Malware detection in critical infrastructures using the electromagnetic emissions of plcs," *Transactions*, vol. 121, no. 1, pp. 519–522, 2019.

[13] Y. Li, G. Cheng, and C. Liu, "Research on bearing fault diagnosis based on spectrum characteristics under strong noise interference," *Measurement*, vol. 169, p. 108509, 2021.

[14] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[15] H. Hassanpour, A. Zehtabian, and S. Sadati, "Time domain signal enhancement based on an optimized singular vector denoising algorithm," *Digital Signal Processing*, vol. 22, no. 5, pp. 786–794, 2012.

[16] D. E. Simon, *An embedded software primer*. Addison-Wesley Professional, 1999, vol. 1.

[17] R. M. A. de Almeida, L. H. de Carvalho Ferreira, and C. H. Valério, "Microkernel development for embedded systems," 2013.

[18] A. Ayub, H. Yoo, and I. Ahmed, "Empirical study of plc authentication protocols in industrial control systems," 2021, p. 15.

[19] S. Kalle, N. Ameen, H. Yoo, and I. Ahmed, "Clik on plcs! attacking control logic with decompilation and virtual plc," in *Binary Analysis Research (BAR) Workshop, Network and Distributed System Security Symposium (NDSS)*, 2019.

[20] D. Tychalas, H. Benkraouda, and M. Maniatakos, "ICSFuzz: Manipulating i/os and repurposing binary code to enable instrumented fuzzing in ICS control applications," p. 16.

[21] H. A. Khan, M. Alam, A. Zajic, and M. Prvulovic, "Detailed tracking of program control flow using analog side-channel signals: a promise for iot malware detection and a threat for many cryptographic implementations," in *Cyber Sensing 2018*, vol. 10630. International Society for Optics and Photonics, 2018, p. 1063005.

[22] Microchip. Avr instruction set manual. [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

[23] ——. Microcontrollers, digital signal controllers and microprocessors. [Online]. Available: https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors

[24] J. Benesty, J. Chen, Y. A. Huang, and S. Doclo, "Study of the wiener filter for noise reduction," in *Speech enhancement*. Springer, 2005, pp. 9–41.

[25] A. J. Eronen and A. P. Klapuri, "Music tempo estimation with k-nn regression," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 1, pp. 50–57, 2009.