Framework for Detecting Control CommandInjection Attacks on Industrial Control Systems(ICS)

Farhad Rasapour
Computer Science Department
Boise State University
Boise, ID
farhadrasapour@u.boisestate.edu

Edoardo Serra
Computer Science Department
Boise State University
Boise, ID
edoardoserra@boisestate.edu

Hoda Mehrpouyan
Computer Science Department
Boise State University
Boise, ID
hodamehrpouyan@boisestate.edu

***Abstract*—This paper focuses on the design and development of attack models on the sensory channels and an Intrusion Detection system (IDS) to protect the system from these types of attacks. The encoding/decoding formulas are defined to inject a bit of data into the sensory channel. In addition, a signal sampling technique is utilized for feature extraction. Further, an IDS framework is proposed to reside on the devices that are connected to the sensory channels to actively monitor the signals for anomaly detection. The results obtained based on our experiments have shown that the one-class SVM paired with Fourier transformation was able to detect new or Zero-day attacks.**

***Index Terms*—Industrial Control Systems, Sensory Channels, Intrusion Detection System**

I. Introduction

Industrial Control Systems (ICS) consists of a set of hardware/software that is used to operate and/or automate industrial processes. Depending on the industry, oil and gas, water treatment plant, power plant, etc., each ICS is designed to operate differently to manage critical processes. Supervisory Control And Data Acquisition (SCADA), Distributed Control Systems (DCS), and Programmable Logic Controller (PLC) are different types pf Industrial Control Systems (ICS), which are considered complex systems. The complexity of these systems result from the fact that many components such as actuators, sensors, and control logic, are not only are designed to manage the critical operations but also connected to the Internet so that business entities can also access the Human Machine Interfaces (HMI), supervisory stations, and Remote Terminal Units (RTU) for real-time monitoring of information. Therefore, ICS have become high-value target of domestic and foreign attacks and extremely challenging to protect. Based on a survey conducted in 21 countries in 2017, half of the industrial organizations reported at least one incident [1].

It is presented in the work of [2] that is possible to utilize sensors as an endpoint to inject control commands and train the implanted Malware for a more accurate attack on the system. We extend this work with not only generating new attack vectors based on the sensor's output signal but proposing a detection framework to protect the system from these types of cyber-attack. Further, as a proof of concept, we utilize a test-bed to conduct experiments on the designed attack models and the proposed defense mechanisms. The rest of the paper is organized as following: section 2 provides an overview of related work, section 3 presents the attack models, including the formulation of the encoding/decoding of a bit of control command into the sensors signal. Section 4 discusses the Intrusion Detection System (IDS), and Section 5 describes the implementation of the attack models and the IDS framework in the developed test-bed, and section 6 is a summary of our contributions and plans for future work.

Identify applicable funding agency here. If none, delete this.

II. Related Work

In general, sensory channel security research can be classified in three classes: the first is detecting attacks which are trying to disable the sensory channel communication like denial of service (DoS) attack [3]. Also, assessing the system tolerance and integrity against sensor failure caused by an attack.

The second is detecting and protecting against false data injection attacks [4], [5]. False data injection (FDI) attacks goal is to feeding wrong data into the sensory channel to compromising the sensor readings to destabilize components or processes which are relied on the sensor data. The third is detecting the misusing of the sensory channel for a purpose different than what they are designed to do [2], [6], [7]. In sensory channel misuse, the attacker exploits a sensor or the sensory channel as an auxiliary interface to inject data into the system. An attacker can gain some level of control over the system by using Electromagnetic Interference (EMI) to inject data into analog sensors [6]. In [2], the authors showed an already implanted Malware could be activated by sending an activation code through a sensory channel. Uluagac et al. also showed sensory channels could have enough bandwidth to transmit Malware.

2379-1896/19/$31.00 ©2019 IEEE
DOI 10.1109/CANDAR.2019.00035

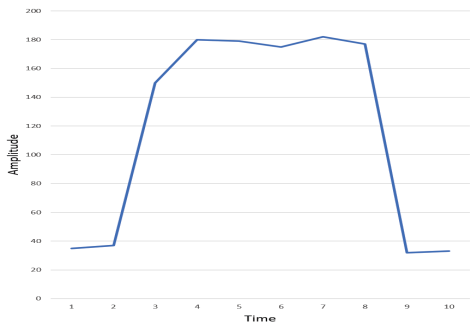


Fig. 1. A pulse P with the $L(P) = 7$ and $H(P) = 181$.

III. Encoding/Decoding schemes

In order to design the attack model, we have been able to utilize the sensory channels as an end point and design a Malware that is able to formally define the encoding/decoding of a bit of control command into the sensor's signal. In this case, the assumption is that the attacker is able to conduct reconnaissance techniques and utilize the discovery tools to learn the critical control commands that are appropriate for the system under attack. Based on the collected information and the existing command-and-control path, the Malware is able to encode control commands as a sequence of bits (1s and 0s) from the received signals form the sensor.

In order to formally define the encoding/decoding function into the signal, we looked at the concept of a Discrete-Signal (DS). A DS is defined as a finite sequence of m consecutive sample signal values, V_s, with the height of $H(DS)$ and length of $L(DS)$ as follow:

$$DS = \langle V_1, \ldots, V_m \rangle \tag{1}$$

$$L(DS) = |DS| = m \tag{2}$$

$$H(DS) = Max(V_1, \ldots, V_m) \tag{3}$$

where $H(DS)$ is the sample with the highest value and $L(DS)$ is the length of the DS samples list. A DS is called a rising discrete-signal DS_r if:

$$\forall_{V_s, V_{s+1}} \in DS_r : V_s - V_{s+1} < -\alpha \tag{4}$$

where V_s and V_{s+1} are two consecutive sample values in the sample list and α is a threshold removing the signal noises. On the other hand, a DS is recognized as a falling discrete-signal DS_f if:

$$\forall_{V_s, V_{s+1}} \in DS_f : V_s - V_{s+1} > \alpha \tag{5}$$

Further, we have define a pulse P as depicted in Figure 1 as a DS that is obtained by the concatenation of: (i) a rising discrete-signal DS_r, (ii) a sequence of samples S_b not containing any rising or falling discrete-signal, and (iii) a falling discrete-signal DS_f. Therefore, a pulse is defined as follows:

$$P = DS_r \cdot S_b \cdot DS_f \tag{6}$$

where "$\cdot$" is the concatenation operator. Further, 4 different encoding schemes is defined given the constants C, b_l and b_u as below:

S_{L1} In the encoded phase, a bit 1 (0) is encoded by generating a pulse with a length $L(P)$ randomly chosen in the range $(C, b_u]$ ($[b_l, C)$). While, in the decoded phase, a pulse P is decoded as 1 (0) if $L(P) > C$ ($L(P) < C$).

S_{L0} In the encoded phase, a bit 0 (1) is encoded by generating a pulse with a length $L(P)$ randomly chosen in the range $(C, b_u]$ ($[b_l, C)$). While, in the decoded phase, a pulse P is decoded as 0 (1) if $L(P) > C$ ($L(P) < C$).

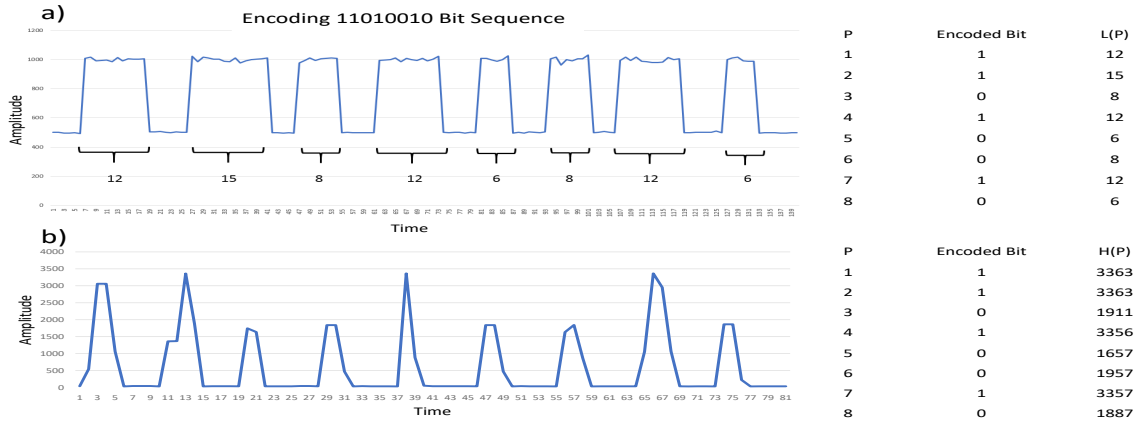
S_{H1} In the encoded phase, a bit 1 (0) is encoded by generating a pulse with a height $H(P)$ randomly chosen in the range $(C, b_u]$ ($[b_l, C)$). While, in the decoded phase, a pulse P is decoded as 1 (0) if $H(P) > C$ ($H(P) < C$).

S_{H0} In the encoded phase, a bit 1 (0) is encoded by generating a pulse with a height $H(P)$ randomly chosen in the range $(C, b_u]$ ($[b_l, C)$). While, in the decoded phase, a pulse P is decoded as 1 (0) if $H(P) > C$ ($H(P) < C$).

The above concept is represented in Figure 2-(a) with a sequence of pulses representing 11010010 bit-sequence that are transmitted through the sensory channel, utilizing $L(P)$, a constant value $C = 10$ as an encoding parameter, and S_{L1} as the encoding schema. In our model, any P with more than ten samples length $L(P)$ are recognized by the implanted Malware as a 1 and the P with the length less than ten are interpreted as a 0. The second encoding/decoding schema is represented in Figure 2-(b) which is based on the $H(P)$, a constant value $C = 2500$ as an encoding parameters, and S_{H1} as the encoding function. A pulse with the height of more than 2500 is interpreted as a 1 otherwise 0. With these two encoding schemes, the decoding logic in the Malware is able to interpret the values less than or greater than C as either 1 or 0.
As demonstrated in the above examples, it is possible to construct different encoding schemes based on the different constant C. This will result in a vast amount of possible encoding combinations for every bit-sequence. This provides flexibility for the attacker in selecting an appropriate encoding function based on the infrastructure and the situation at hand and the implented Malware only requires to detect the pulses and have information about the constant C. However, it will be challenging for a defense system to defend against these types of attacks, since the encoding parameters are unknown. The next section provides an overview and the detail explanation of the proposed intrusion detection system (IDS) framework for detecting the one-bit per pulse in the sensory channel.

IV. Intrusion Detection System (IDS)

The proposed IDS framework resides on the devices that are connected to the sensory channels to actively monitor the signals for detection. Figure 3 depicts an overview of the control logic of the proposed framework. It is composed of two main modules: the "novelty detection module" and the "Attack signature matching module." The first module is designed to recognize behaviors that deviate from the normal (acceptable) behaviors of the device. While the second module recognizes



P	Encoded Bit	L(P)
1	1	12
2	1	15
3	0	8
4	1	12
5	0	6
6	0	8
7	1	12
8	0	6

P	Encoded Bit	H(P)
1	1	3363
2	1	3363
3	0	1911
4	1	3356
5	0	1657
6	0	1957
7	1	3357
8	0	1887

Fig. 2. a) Encoding 11010010 bit sequence with different pulse length ($L(P)$) with C=10. b) Encoding 11010010 bit sequence with different pulse height ($H(DS)$) with C=2500.

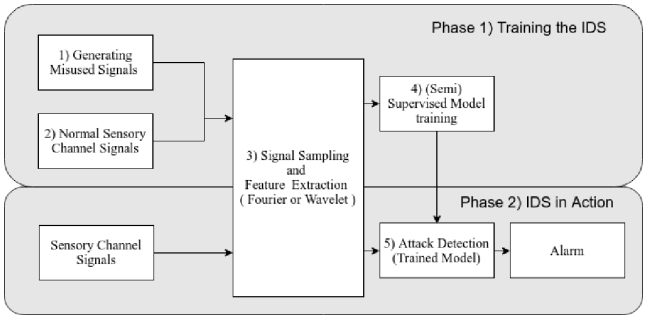


Fig. 3. Sensory Chanel Intrusion Detection System

specific attacks already known. In the case that IDS identifies an unknown behavior, with the novelty detection module, it will directly raise an alert, otherwise, it will delegate the task to the attack signature matching module. The attack signature matching module performs an accurate analysis and it can potentially discriminate known attack behaviors similar to the normal behavior of the device. In particular, the signature matching module allows the system to continuously improve its detection with the continuous consolidated knowledge about known attacks. This architecture is suitable for the detection of unknown and known attacks.

Both of the two modules are based on machine learning techniques: a semi-supervised approach for the novelty detection module and a fully supervised for the signature matching module. The Figure 3 shows how each of the two modules is trained and used.

In the ***training phase***, the module learns how to differentiate between the normal behavior of the signal and when under attack. In this phase, data is collected based on the sensory channel's signal characteristics, i.e. transaction, frequency, and amplitude during the normal operation and under the attack. Then the sampling signals are transformed from a continuous signal into a discrete signal so that a window shifting algorithm can be used for feature extraction of the signal. Further, the **Fourier** and/or **Wavelet** [8] transformations are applied to clean and prepare the data for the semi-supervised classification model. It is important to notice that for the novelty detection module only the normal behaviors are needed while for the signature matching module the attack behavior are also needed.

In the ***Detection phase*** the trained model is utilized to detect the attack in the sensory channel according to Figure 3 and create an alarm to notify the control technicians.

The rest of this section is dedicated to describing the details of the proposed framework.

A. Signal Sampling and Feature Extraction

In Figure 3 module (3) is responsible for sampling signals and feature extraction. The signal sampling task transforms a continuous signal into a discrete signal, i.e. a sequence of sample signal values. Given the discrete signal, we use a window shifting algorithm that at each iteration shifts of ρ_s time units a windows of length ρ_l. For each iteration t a vector x^t of ρ_l elements containing all the signal value is generated. This vector is the main input that our classification model will process. Before performing any classification task, the signal represented by the vector feature v_t has to be cleaned and transformed. To clean the signal we use the wavelet procedure and as a transformation we use the Fast Fourier Transformation (FFT) that changes the signal in the time domain to the frequency domain. The final output after these two procedures is a vector of new features that can be passed through standard classifiers.

B. Novelty detection and Attack Signature Matching Modules

Once the features are cleaned and transformed, our procedure uses standard supervised and semi-supervised classification models (see Figure 3 modules 4 and 5) to discriminate between attacks and good behaviors.

Supervised Classification Models for Attack Signature Matching Module: The supervised classification models are trained by using labeled examples from the misused signals and the normal behaviors. In this paper, we consider several classification models such as support vector machine (SVM),

classification tree and random forest. To estimate the hyper-parameter parameters we use a standard procedure consisting of a nesting cross-folding procedure combined with the F1-score.

Semi-Supervised Classification Model for Novelty Detection Model Supervised classification models, since trained with attacks and good behavior examples, are able to recognize well attacks similar to to the ones used in the training. However, they do not usually perform well in the case of new and unknown attacks. In the experiment section, we show that the classification models trained with only one type of attack have low accuracy in detecting different attacks. Our experiment shows that even a small variation in a misused signal makes the supervised classifier unable to detect the attack. Therefore, an attacker can avoid detection by creating a misused signal different from the training signals. To address this problem, the IDS framework uses a semi-supervised classification model, to solve the novelty detection problem. Novelty detection is the problem to train a classifier that will discriminate normal situations from novel one. The novelty detection training procedure uses only examples representing normal situations. In our specific problem, normal situations are represented by the channel normal behavior and novel situation should be the misused signal. Therefore, our training dataset, $X_{tr} = \{x_t^1 r, \ldots, x_t^n r\}$, contains only feature vectors associated with normal behavior.

In this paper, we use a different algorithm that can be used as novelty models including Gaussian mixture [9], isolation forest [10], and one-class support vector machine (one-class SVM) [11] by often manually setting different hyperparameters. Differently from the supervised classification model, in the novelty detection does not exist a standard procedure to estimate the hyperparameters. This because the training set contains only one class.

V. Case Study

The selected case study for this paper is based on authors's previous work [12]. As described in Section 2, the first Malware planted on the sensory channel utilizes the IR-toy to manipulate the Kobuki's IR sensor and inject pulses into the sensory channel. The Malware sends commands to the IR-toy to emit or not emit IR-light for a specific amount of time according to S_{L0}, S_{L1}, S_{H0}, and S_{H1}. As a result, the injected control command is able to disable the emergency flush in the control logic of the PLC. This is possible by sending the value (0X0001), which means the ON command, to the output address (0X000A) of the PLC. Following the injection, the second Malware, that is planted in the laptop, decodes the injected command, utilizing the existing vulnerabilities in the Modbus protocol that is able to disable the emergency flush so the control technicians are not able to enable this feature. This is considered a serious safety issue.

A. Data Collection and IDS Experimental Settings

To test the performance of the proposed IDS, the measurement data from the robot's sensory channel is collected with regards to the following two situations: 1- sensor under normal circumstances, i.e., moving around on the floor. The data collection process consists of measurement data from the different types of floors, i.e. carpet, different types of woods: floor, grass, cement, etc, 2- sensor under attack, which total of 1,167 attack signals for four different encoding and 6 different commands was collected.

Attack Encoding	Generated Signals
H_0	240
H_1	240
L_0	336
L_1	351

TABLE I
NUMBER OF OF ATTACKS

ρ_l	ρ_s	Attack Samples	Normal Samples
650	650	1159	1116
650	325	2322	2234
650	162	4660	4482
325	325	2326	2236
325	162	4668	4486
325	81	9342	8976
162	162	4672	4488
162	81	9350	8980
162	40	18943	18190
81	81	9354	8982
81	40	18951	18194
81	20	37907	36390

TABLE II
NUMBER OF NORMAL AND ATTACK EXAMPLES FOR EACH WINDOWS LENGTH AND WINDOWS SHIFTING

The injected signals were generated for a constant C between ten and twenty for the length patterns (L_0 and L_1).

The selection of $C \in [10-20]$ enables us to create misused signals that are in the normal signal range and also do not cause big changes in the normal Kobuki behavior. For the height patterns (H_0 and H_1), it was hard to force the IR sensor to generate a vast range of amplitudes. But we managed to create misused signals that can use 1800, 2000, and 2500 as the C without causing drastic changes in the Kobuki normal operation. In Table V-A we provide a summary of all attack signal that we collected.

After the signal collection, we used the time shifting windows algorithm to collect different feature vectors about normal behaviors and attacks. We consider 4 kinds of window size ρ_l i.e. 650, 325, 162 and 81 sample values. The 650 windows length is determined according to the largest time interval that we use to send a Malware command through the kabuki sensory channel. For each window size ρ_l we consider different windows shift ρ_s, i.e. ρ_l, $\rho_l/2$, $\rho_l/4$.

In table II are provided the total number of attacks and normal behavior for each windows length ρ_l and windows size ρ_s.

For each of the cases reported in Table II, we used the following four transformation pipelines:

- **No Transformation** The original samples generated by the windows shifting algorithm are directly taken in input by the classifier model.
- **Fourier** Fourier transformation is the unique transformation applied before the classification.

- **Wavelet** Wavelet transformation is the unique transformation applied before the classification.
- **Wavelet + Fourier** The examples are first cleaned from the noise with the Wavelet transformation and after the Fourier transformation is applied.

In the case of the wavelet, we consider all the 4 different wavelets described in Section IV-A. For each pipeline, we consider different classifier models. We use as supervised classifiers SVM with several kernels (linear, polynomial, and RBF), Decision Tree and Random Forest. While as semi-supervised classifiers we use Gaussian Mixture Model, Isolation Forest, and one-class SVM. The procedures to estimate all the hyperparameters are reported in Section IV-B.

To test all the pipeline transformations combined with all the classification models over the different dataset reported in table II we use a 10-folds cross folding procedure. The 10-folds cross folding procedure creates for each fold a training set (90% of the original size) and a test set (10% of the original size). To measure the performance we use the recall [13] that is the percentage of correctly classified element in a specific class, e.g. attack or normal behavior. For each fold and each of the two classes, attack or normal behavior, we computed the recall. We finally average all the results among all the folds. In the case of the semi-supervised classification model for the novelty detection problem, for each fold from the train set, we remove all the examples representing an attack.

B. Result Analysis

In this section, we discuss all the results obtained by each experiment described in the above section. More specifically we discuss:

- Results for supervised classification models.
- Performance of supervised classification models in the presence of unknown or new attacks.
- Performance of semi-supervised classification model for the novelty detection problem.
- Sensibility to the variation of windows size and windows shifting.

More we briefly discuss the time performance of our approach and we finally conclude with the final discussion.

1) Results for supervised classification model: In Table III are reported the classification results only for the SVM classifier with the RBF kernel for all the transformation pipelines. This because the SVM classifier with RBF in all the pipelines achieves always the best performance w.r.t. the other classifiers. The result in Table III refers to a windows length $\rho_l = 650$ and a windows shifting $\rho_s = 162$. In addition, in the case of the wavelet, we report only the best result of recall among all the wavelet functions.

The main result that we can observe in Table III is that in the supervised classification the prepossessing of the examples with Fourier and/or wavelet is not necessary. In fact, not using any of those transformations brings the best results. Moreover, it is possible to see according to the recalls values that supervised classification provides an almost perfect separation between attack and normal behavior (more than 0.996 recall for each class).

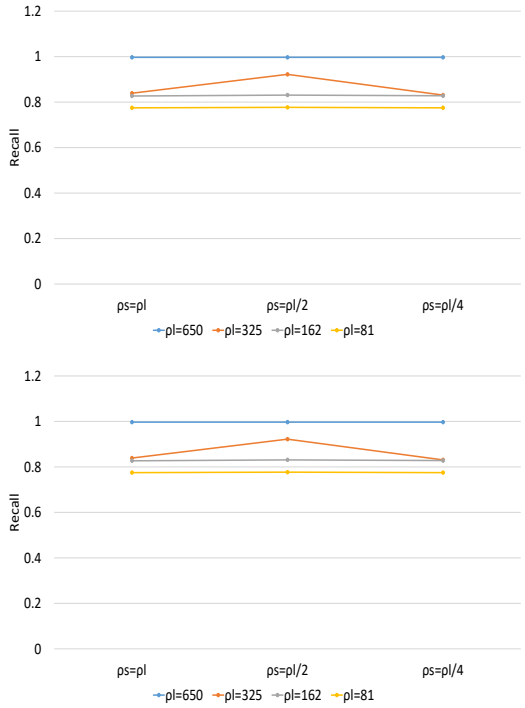


Fig. 4. Recall for Two-Class SVM.

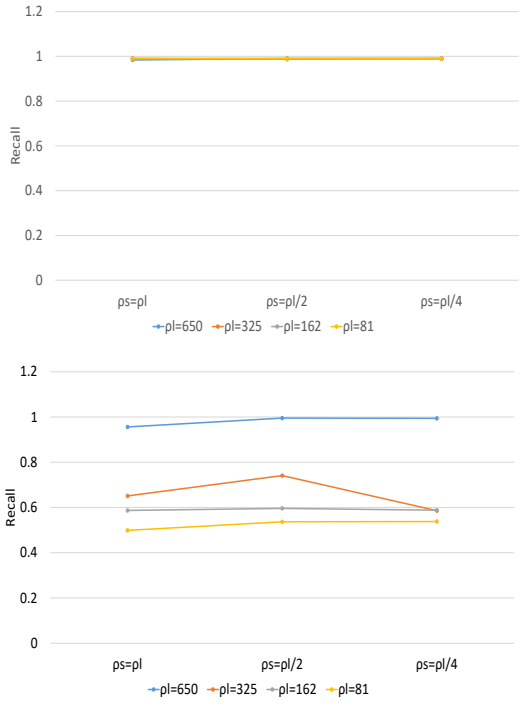


Fig. 5. Recall for One-Class SVM.

Transformation	Normal Detection Recall	Attack Detection Recall	Total Weighted Recall	H_1 Detection Recall	H_0 Detection Recall	L_1 Detection Recall	L_0 Detection Recall
No	0.993	0.997	0.996	1.0	1.0	0.999	1.0
Fourier	0.993	0.997	0.996	1.0	1.0	0.999	1.0
Best Wavelet	0.991	0.996	0.994	1.0	1.0	0.996	1.0
Best Wavelet + Fourier	0.989	0.994	0.993	1.0	1.0	0.992	1.0

TABLE III

DETECTION RESULT FOR TWO-CLASS SVM TRAINED WITH SAMPLES FROM NORMAL AND ALL ATTACK SCHEMA DATASETS ($\rho_l = 650$ & $\rho_s = 162$).

2) Performance of Supervised Classification Models in the Presence of Unknown or New Attacks.: In this analysis, we want to verify if supervised classification models can detect new or unknown kinds of attacks, i.e. kind of attacks not present in the training set. In fact, in this experiment, we keep in the training only one kind of encoding schema for the attack and we test the model with all the kinds of attacks. In Table IV the column "Trained With Attack Encoding Schema" indicates which kind of attack is the only one present in the training set. Differently from the Table III, in Table IV we report the recall for each specific kind of attack (see the last four columns). Unfortunately, as we can see from the Table IV, in the case of the encoding schemes H_0 and H_1, the recalls of l_o and l_1 are approximately close to zero. This means that we supervised classification model is not always able to detect unknown or new kinds of attacks. For this motivation, we adopt a semi-supervised classification approach.

3) Performance of Semi-supervised Classification Model for the Novelty Detection Problem.: In Table V, we report the results for the one-class SVM with the hyperparameter estimation procedure defined in Section IV-B for all the pipelines. This procedure in term of recall outperforms all the other semi-supervised approaches that we use. For this motivation report only these results. Differently, from the supervised classification models, the Fourier pipeline significantly improves the performance of the semi-supervised classification model, and make in term of performance the semi-supervised equal to the supervised one. Moreover, It is important to notice that in the training set of the semi-supervised classification models are not present any attack example. Therefore, if the recalls of all the kind of attacks are greater or equal than 0.982, at least in the case with the Fourier transformation, it means that the semi-supervised classification Model is also robust to new or unknown attacks. This makes the Fourier pipeline and the one-class SVM with our hyperparameter estimation preferable to all the other models presented.

4) Sensibility to the Variation of Windows Size and Windows Shifting.: The window length ρ_l and the windows shifting ρ_s parameter can impact on the performance of both supervised and semi-supervised classification models. Above was shown that in the best case situation for supervised and semi-supervised classification models are the SVN with the RBF kernel and the one-class SVM with RBF kernel and Fourier transformation, respectively. These are the models we consider to show the sensibility analysis about ρ_l and ρ_s. In Figure 4.a and Figure 4.b in the case of the supervised classification model, we show for each window length ρ_l and each windows shifting ρ_s in the values of the recall for normal behavior and attack, respectively. In Figure 5.a and Figure 5.b similar results are reported in the case of the semi-supervised classification model. As we can see from the above figure changing the windows shifting ρ_s does not change significantly the recall values. While the change of the length of the window ρ_l can substantially modify the recall especially in the case of the semi-supervised classification model, i.e., the one one-class SVM with RBF kernel and Fourier transformation. Therefore it is really important to use the largest as possible windows length that the IDs has to consider. In addition, note that the window length has to be similar to the maximum time interval to send a Malware command. Of course, the strategy to avoid the IDS detection of the attacker communicating with the Malware is to enlarge as much as possible this time interval. Fortunately, during the entire duration of an attack (i.e., the Malware command is sent through the sensory channel) the signal through the sensory channel is drastically modified and the kabuki is not able to understand the environment (e.g., obstacle). This means that for large time interval the robot is clearly faulty, then there is a limit to the size of the attack time interval.

5) Time Performance: We conducted another experiment to evaluate the models time performance for both supervise and semi-supervised models. We implemented the models in a python script and ran them on a Raspberry Pi. Since most industrial field devices are using mid-range processors, we chose Raspberry Pi 3 as a platform to have the same processing power. The average time from the moment the signal is passed to the transformation procedure until the SVM reports the result is about 0.19 seconds. The maximum reported time was 0.28 seconds which it can be an acceptable time for a python script running on a mid-range processor.

6) Final Discussion: In these experimental results, we observe that one-class SVM with our hyperparameter estimation procedure outperforms all the other models. This is especially true in its ability to detect new or unknown attacks. To achieve this result the one-class SVM has to be paired with the Fourier transformation. Moreover, we realize that the use of wavelet transformation to clean the signal from the noise does not bring effective improvement both with the supervised and semi-supervised procedures. In addition, we observe that once the model is trained the (one-class SVM and SVM) the inference time is really fast and it can be developed on lower power consumption and a limited platform like the Raspberry Pi, and used in real-time.

VI. CONCLUSION AND FUTURE WORK

Advanced Persistent Threat (APT) is a sophisticated and coordinated Malware that is designed to impact the targeted

Transfo-rmation	Trained with Attack Encoding Schema	Normal Detection Recall	Attack Detection Recall	Total Weighted Recall	H_1 Detection Recall	H_0 Detection Recall	L_1 Detection Recall	L_0 Detection Recall
No	H_1	0.998	0.997	0.998		1.0	0.0	0.0
No	H_0	1.0	1.0	1.0	1.0		0.0	0.0
No	L_1	0.0 .995	0.992	0.993	1.0	1.0		1.0
No	L_0	0.997	0.996	0.996	0.785	0.998	0.988	
Fourier	H_1	0.998	0.997	0.998		1.0	0.0	0.0
Fourier	H_0	1.0	1.0	1.0	1.0		0.0	0.0
Fourier	L_1	0.995	0.992	0.993	1.0	1.0		1.0
Fourier	L_0	0.997	0.996	0.996	0.785	0.998	0.988	
Wavelet	H_1	0.997	0.996	0.997		0.933	0.0	0.0
Wavelet	H_0	0.999	0.998	0.998	0.999		0.0	0.0
Wavelet	L_1	0.996	0.994	0.995	1.0	1.0		0.999
Wavelet	L_0	0.996	0.994	0.995	0.894	0.718	0.979	
Wavelet + Fourier	H_1	0.999	0.998	0.999		1.0	0.001	0.005
Wavelet + Fourier	H_0	1.0	1.0	1.0	1.0		0.0	0.0
Wavelet + Fourier	L_1	0.993	0.989	0.992	0.995	1.0		1.0
Wavelet + Fourier	L_0	0.996	0.994	0.996	0.767	0.942	0.978	

TABLE IV
DETECTION RESULT FOR TWO-CLASS SVM TRAINED WITH SAMPLES FROM NORMAL AND ONE ATTACK SCHEMA ($\rho_l = 650$ & $\rho_s = 162$).

Transformation	Normal Detection Recall	Attack Detection Recall	H_1 Recall	H_0 Recall	L_1 Recall	L_0 Recall
No	0.990	0.539	0.5	0.5	0.527	0.607
Fourier	0.990	0.994	0.997	1.0	0.982	1.0
Best Wavelet	0.990	0.514	0.413	0.428	0.547	0.611
Best Wavelet + Fourier	0.991	0.962	0.877	1.0	0.958	1.0

TABLE V
DETECTION RESULT FOR ONE-CLASS SVM TRAINED WITH NORMAL SIGNAL SAMPLES ($\rho_l = 650$ & $\rho_s = 162$).

system by gaining access to the targeted system and taking over the control of the system. In this paper, we have demonstrated that the sensory channels have the potential to be misused as a control-and-command path to coordinate Cyber-attacks.

In order to protect the ICS from these types of Cyber-attacks, an Intrusion Detection System (IDS) is proposed to actively monitor the sensory channels and conduct signal sampling and feature extraction so that the classification models can use these data to differentiate between the normal and modified signal pulses. The results obtained based on our experiments have shown that the one-class SVM paired with Fourier transformation was able to detect new or Zero-day attacks.

ACKNOWLEDGMENT

This work was supported by National Science Foundation Computer and Information Science and Engineering (CISE), award number 1846493 of the Secure and Trustworthy Cyberspace (SaTC) program: Formal TOols foR SafEty aNd Security of Industrial Control Systems (FORENSICS).

REFERENCES

[1] T. B. A. G. Limited. (2017) Research: The state of industrial cybersecurity. [Online]. Available: http://www.business-advantage.com/blog/research-the-state-of-industrial-cybersecurity/

[2] A. S. Uluagac, V. Subramanian, and R. Beyah, "Sensory channel threats to cyber physical systems: A wake-up call," in *Communications and Network Security (CNS), 2014 IEEE Conference on*. IEEE, 2014, pp. 301–309.

[3] M. Krotofil, A. A. Cárdenas, B. Manning, and J. Larsen, "Cps: driving cyber-physical systems to unsafe operating conditions by timing dos attacks on sensor signals," in *Proceedings of the 30th Annual Computer Security Applications Conference*. ACM, 2014, pp. 146–155.

[4] H. Cai and K. K. Venkatasubramanian, "Detecting signal injection attack-based morphological alterations of ecg measurements," in *Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on*. IEEE, 2016, pp. 127–135.

[5] J. Park, R. Ivanov, J. Weimer, M. Pajic, and I. Lee, "Sensor attack detection in the presence of transient faults," in *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems*. ACM, 2015, pp. 1–10.

[6] D. F. Kune, J. Backes, S. S. Clark, D. Kramer, M. Reynolds, K. Fu, Y. Kim, and W. Xu, "Ghost talk: Mitigating emi signal injection attacks against analog sensors," in *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 2013, pp. 145–159.

[7] R. Shrestha, H. Mehrpouyan, and D. Xu, "Model checking of security properties in industrial control systems (ics)," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '18. New York, NY, USA: ACM, 2018, pp. 164–166. [Online]. Available: http://doi.acm.org/10.1145/3176258.3176949

[8] A. Oppenheim and J. RWSchafer, "Buck, discrete-time signal processing upper. saddle river," 1999.

[9] J. A. Bilmes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126, 1998.

[10] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 413–422.

[11] L. M. Manevitz and M. Yousef, "One-class svms for document classification," *Journal of machine Learning research*, vol. 2, no. Dec, pp. 139–154, 2001.

[12] F. Rasapour and H. Mehrpouyan, "Misusing sensory channel to attack industrial control systems," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '18. New York, NY, USA: ACM, 2018, pp. 158–160. [Online]. Available: http://doi.acm.org/10.1145/3176258.3176947

[13] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 233–240.