Testing Cyber-Physical Systems Against Adversarial Attacks

Anonymous Author(s)

ABSTRACT

The constant threat of attack faced by cyber-physical systems (CPSs) in critical infrastructure has motivated research into a multitude of attack detection mechanisms, such as *anomaly detectors* based on underlying neural networks. The effectiveness of such a detector can be assessed by subjecting it to test suites of attacks, whether from benchmarks, or tools such as network fuzzers. While studies have shown that neural network-based detectors perform well against these conventional attacks, limited consideration has been given to testing their effectiveness against *adversarial attackers* that possess knowledge of the underlying model. In this paper, we present an adversarial attack for testing recurrent neural network (RNN) anomaly detectors of CPSs. Inspired by some white-box gradient-based approaches, our adversarial attacks produce *noise* (or *perturbations*) with the aim of causing an incorrect classification. We implement it against detectors for two real-world critical infrastructure testbeds, successfully reducing their classification accuracy. Finally, we demonstrate that the detectors of the testbeds can be successfully adapted to detect attacks that utilise such noise.

KEYWORDS

Cyber-physical systems, testing attack detection mechanisms, anomaly detectors, adversarial attacks, neural networks.

ACM Reference Format:
Anonymous Author(s). 2020. Testing Cyber-Physical Systems Against Adversarial Attacks. In *Proceedings of ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2020).* ACM, New York, NY, USA, 11 pages.

1 INTRODUCTION

Cyber-physical systems (CPSs), in which software components are deeply intertwined with physical processes, are ubiquitous in critical public infrastructure. The potential disruption that could result from a compromised system has motivated research into a multitude of CPS attack detection mechanisms, including techniques based on invariant checking [43], physical attestation [20], and fingerprinting [28]. A particularly popular solution is to build *anomaly detectors* [18, 23, 29, 40, 65], in which an underlying machine learning (ML) model is trained on streams of the system's physical data in order to judge when it deviates from the norm. Typically, this would be a form of neural network, a model that is powerful enough to learn and recognise the complex patterns of behaviour that CPSs exhibit.

The effectiveness of an anomaly detector can be assessed by subjecting it to a test suite of attacks, and observing whether it can correctly identify the anomalous behaviour. These tests can be derived from benchmarks [42], hackathons [6], or tools such as fuzzers [22], and typically involve manipulating or spoofing the network packets exchanged between CPS components. While studies have shown that neural network-based detectors perform well against these conventional attacks [38, 42, 44, 59], limited consideration has been given to testing their effectiveness against *adversarial attacks*, in which an attacker with knowledge of the model crafts *noise* (or *perturbations*[1]) on the data to specifically cause the neural network to misclassify it. Such attacks have been successfully demonstrated in several different classification domains (e.g. images [47], audio [17], malware [34]), raising the question as to whether anomaly detectors in critical infrastructure are susceptible too. If an anomaly detector fails to detect adversarial tests, then its CPS is potentially at risk of a much broader range of attacks since they can simply be masked by crafted noise.

In this paper, we present an adversarial attack for testing recurrent neural network (RNN) anomaly detectors of CPSs. Our approach is inspired by the white-box gradient-based approach previously applied to deceive the classifiers of audio files [17], but adapted to meet the unique challenges posed by the critical infrastructure. First, while we can craft noise on sensors in the continuous domain, actuators typically fall into a small number of discrete states (e.g. 'open' or 'closed'). Second, in addition to anomaly detectors, the logic of these CPSs often also includes some built-in mechanisms for *rule-checking* (e.g. testing that a tank's valve is open if its level is low) to add another level of certainty that the system is operationally normal. We address these issues by using genetic algorithm to select optimized combination of actuator values.

We evaluate the effectiveness of our adversarial attacks by implementing them against RNN-based anomaly detectors for two real-world critical infrastructure testbeds: Secure Water Treatment (SWaT) [51], a multi-stage water purification plant; and Water Distribution (WADI) [10], a consumer distribution network. We demonstrate that our adversarial tests successfully and substantially reduce the classification accuracy of their anomaly detectors, thus revealing a security flaw that could be exploited. We show that a rule-checking mechanism in SWaT has weakened our thread model but can be bypassed by genetic algorithm and achieve similar accuracy reduction for the anomaly detector. Finally, we go a step further and show that by training the RNNs on some noisy data, the anomaly detectors of SWaT and WADI can successfully identify that an adversarial attack is in progress.

Our Contributions. Our contributions are summarised as follows:

- We define a threat model for CPS adversarial attackers, and propose a white-box gradient-based attack to construct adversarial tests for deceiving RNN-based anomaly detectors.

[1] We will use noise and perturbation interchangeably.

- We re-implement SWaT's RNN-based anomaly detector [29], extending it to cover the whole CPS, and learning another in parallel to cover WADI.
- We implement our adversarial attacker, and test it against the SWaT and WADI RNN-based anomaly detectors.
- We find that our approach significantly reduces the accuracy of the detectors to the point that masking other attacks is feasible. We show that this can be mitigated by including adversarial noise in the training data.

Organisation. In Section 2, we provide the necessary background and present our threat model. In Section 3, we introduce our experiment design and methodologies to construct adversarial examples. We show our empirical study results on SWaT and WADI in Section 4. Finally, we review related works in Section 5 and conclude in Section 6.

2 BACKGROUND

2.1 Structure of a CPS

Figure 1 provides an architectural overview of a typical cyber-physical system, which contains a physical part and a cyber part. Information of thread model will be introduced in senction 2.3. The cyber part further consists of the Programmable Logic Controllers (PLCs) [13] which implement a set of control logics according to the input sensors from the physical environment and a Supervisory Control and Data Acquisition (SCADA) system. The physical environment periodically sends sensor data into the cyber part through a local network (Layer 0 Network). The PLCs then generate the control commands and send them back to adjust the physical environment. The SCADA system is built on top of PLCs (communicating using Layer 1 Network) to monitor and supervise the control process. To dynamically detect potential attacks at real time, CPS are often equipped with multiple in-built mechanisms to guarantee that the system is in normal operation. Among them, rule checking [12] and anomaly detection [18] are widely adopted strategies which we focus on in this work. Notice that in theory rule checking and anomaly detection can be implemented in both the PLCs or the SCADA system.

2.2 SWaT and WADI

We carry out our study on a real-world modern Secure Water Treatment plant (SWaT) [51] and a real-world Water Distribution system WADI [10]. SWaT is a 6-stage plant that takes raw water as input and produces 5 gallons/minute of treated water. The six sub-processes in the plant (one corresponding to each stage) include "stage 1: Supply and Storage", "stage 2: Pre- Treatment", "stage 3: Ultra-filtration and Backwash", "stage 4: De-Chlorination System", "stage 5: Reverse Osmosis" and "stage 6: Permeate Transfer, Cleaning and Back-wash". Each sub-process is controlled by different PLCs. WADI is a similar cyber-physical system but with a purpose of water distribution to serve customers 10 gallons of drinking water per minute. WADI has three stages where the first stage consists two water tanks to store water from external sources and pass the water to the second stage such that customers are provided with drinking water based on their demanding. Afterwards, the water goes to the third stage and gets pumped back for reuse.

Sensors and actuators: In total, SWaT contains 68 sensors and actuators. Notice that among them some actuators serve as standbys and are intended to be used only when the primary actuator fails, which we do not consider in this work. After filtering such actuators, there are in total 25 sensors and 26 actuators which we target in this study. Notice that sensors are of integer or float type while actuators can only take discrete values. WADI has similar data type but with 121 features including 51 actuators (20 valves and 31 pumps) and 70 sensors.

In the following, we use x to denote a system state (represented by sensor and actuator values) at a certain timestamp where $x = [x_a1, x_a2, x_a3 \ldots x_s1, x_s2.x_s3 \ldots]$, x_a represents actuator value and x_s represents sensor value. X is the set of all possible system states. Let S be a sequence of system states, we use $S[i]$ to denote the i-th state in the sequence, $S[i : j]$ to denote the sequence of system states from time i to j, and $S[i :]$ to denote the sequence of states from time i to the present.

We take stage 1 of SWaT (Supply and Storage) as an example. There are 2 sensors and 3 actuators in stage 1, i.e., sensor "FIT101" to measure the speed of flow transmission, sensor "LIT101" to measure the water level, actuator "MV101" as a motor valve and "P101", "P102" as two pumps. SWaT collects the values of sensors and actuators every second as data logs and generates a set of control commands. An example data log of stage 1 is given in Table 1.

Plant supervision and control: A Supervisory Control and Data Acquisition (SCADA) workstation is located in the plant control room. A Human Machine Interface (HMI) is also located inside the plant room and can be used to view the process state and set parameters. Control code or anomaly detectors can be loaded into the PLC via the workstation. We can also acquire data logs (as well as network packet flows) from the historian at preset time intervals.

Communications: A multi-layer network enables communications across all components of SWaT and WADI. The ring network in each stage at Layer 0 enables PLCs to communicate with sensors and actuators. A star network at Layer 1 enables communications across PLCs, SCADA, HMI and the historian. Both wired and wireless options are available at Layer 1 and also for communications with the sensors at Layer 0. PLCs communicate with each other through the Layer 1 network, and with centralized Supervisory Control and Data Acquisition (SCADA) system and Human-Machine Interface (HMI), through the Layer 2 network.

2.3 Threat model

As shown in Figure 1, our threat model assumes a white-box setting where an attacker has the following capabilities:

- (black box) the attacker is aware of the underlying rules or invariants for rule checking, but cannot access any information excepts for inputs and outputs;
- (white box) the attacker has full access to the RNN predictor including the network architecture, parameters, inputs and outputs, and other attributes;

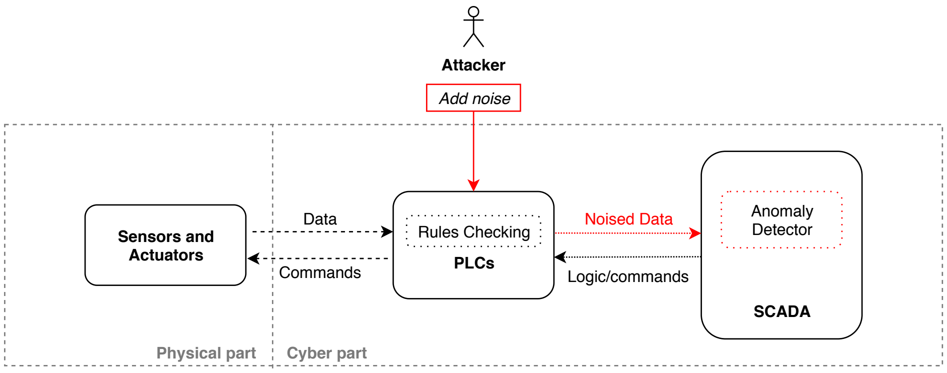

Figure 1: Overview of a cyber-physical system and cyber attacks.

Table 1: Example of log data

Timestamp	FIT101	LIT101	MV101	P101	P102	Normal/Attack
28/12/2015 10:00:00 AM	2.427057	522.8467	2	2	1	Normal
28/12/2015 10:00:01 AM	2.446274	522.886	2	2	1	Normal
28/12/2015 10:00:02 AM	2.489191	522.8467	2	2	1	Normal
28/12/2015 10:00:03 AM	2.53435	522.9645	2	2	1	Normal

- the attacker is able to compromise the data transmitted from the physical part to the PLCs through intranet communication (layer 0).

2.4 CPS attacks

Original attacks: Original attacks capture existing attack data. These attacks were performed previously on the SWAT and WADI systems. The dataset is labeled to indicate where and when the attacks are applied and lead to what results. In particular, a total of 36 different attacks were conducted during a period of four days for SWaT system. The attacks in the dataset is generated following the attack model described in [3, 4]. All the attacks are carried out by spoofing the sensor values transmitted to the PLCs (which is exactly the capability 3 in the threat model). Some examples of attacks are given in Table 2.

Adversarial attacks: Adversarial attacks focus on crafting adversarial examples that could maximize the some measures of harm while minimize the adversarial perturbation, which intends to make the attacks less detectable. In our work, adversarial attacks are applied to descent the accuracy of RNN predictor without being detected by the rule checking system or other anomaly detectors.

Attack definition: We call a CPS system to be under attack at certain time point t, if S[t], i.e. the CPS state at time t creates an observable impact in the system. This observable impact is measured with respect to the respective state in benign conditions. In practice, we can leverage the acceptable range of sensor values to identify attacks. For example, in SWAT system, LIT101 indicates the water level in the first stage. If LIT101> 1100mm, then the system is under a condition that violates the acceptable range of LIT101. As a result, observing a value of LIT101 beyond 1100mm immediately identifies

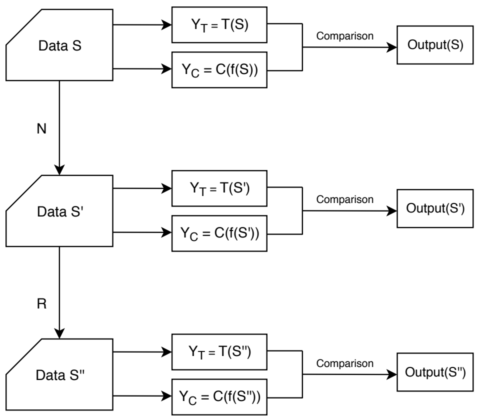

Figure 2: An overview of input vs output

an attack. As the definition of the attacks is dynamic with changes of input data, so for every prediction of f, we have to generate the status labels as the ground truth, Figure 2 shows an overview of how results generated. Vertically, Data S is the original attack data, Data S' is noised data after gradient-based attack by model N, and S" is GA selected data to pass rule checker R. Horizontally, from each data set (s,s' and S"), we calculate Y_T as the ground truth to compare with the anomaly detector results Y_C and get outputs. The *Output* inside figure refers to a lists of standards such as precision, recall f1, etc.

2.5 CPS anomaly detection

we discuss two commonly used anomaly detection mechanisms in SWAT, rule checker [5] and RNN-CUSUM anomaly detector [29].

Table 2: Example of attacks

Start Time	End Time	Attack Point	Start State	Attack
10:29:14	10:44:53	MV-101	MV-101 is closed	Open MV-101
10:51:08	10:58:30	P-102	P-101 is on where P-102 is off	Turn on P-102
11:22:00	11:28:22	LIT-101	Water level between L and H	Increase by 1 mm/s

2.5.1 Rule checking. Adepu and Mathur [5] have systematically derived a set of invariants to compose 23 rules describing the relationship of sensor and actuator values of SWaT. The idea of rule checking is to implement a set of pre-defined rules that should never be violated into the PLCs. For instance, a sensor value should never exceed its operation range. Besides, there are some regulations that the system should follow under normal operation. We use R to denote the set of rules implemented in PLCs for checking. Once a rule $r \in R$ is violated, the system will raise an anomaly alarm to report that the system is in abnormal state. Some rules are shown in Table 3. Let us take rule 1 as an example. The rule specifies that under the condition that sensor "LIT101" is equal or smaller than 500, the actuator "MV101" is supposed to be 2 after 12 seconds.

2.5.2 Learning-based approach. The idea of learning-based anomaly detectors is to model the normal behavior of the system from a piece of data historian S using a machine learning model (denoted by f). At run time, the system looks at the historian data (of a window-size length), use f to predict the system state x' and compare it with the actual system state x. If the difference is beyond a threshold, the system is likely to be abnormal. In the work from J.Goh et al. [29], a RNN with LSTM architecture [37] is used as the prediction model and a CUSUM algorithm (denoted by C) to calculate the differences between the actual value and the predicted value. This approach has been applied to SWaT system first stage and shown to be effective. Therefore, we adopted their approach and applied to all 6 stages and WADI system as the defence mechanism for us to test. In the following sections, when we mention the anomaly detector, it means the learning-based approach, and we use rule checker to imply the rule checking mechanism.

Under the mentioned threat model and attack scenarios, we aim to answer the following questions: 1) how should the attacker compromise the original data and construct the adversarial examples to fool the anomaly detector? 2) how should the attacker compromise the original data to fool the rule checking system and the anomaly detector? 3) how can we design a defence against such adversarial examples?

3 EXPERIMENT DESIGN AND METHODOLOGY

Experiments are designed to descent the performance of the anomaly detector of CPSs where there are some difficulties to complete the testing experiments.

- There are limited works on adversarial attacks on CPS, especially for anomaly detector with RNN, thus we do not have many references.
- There could be more than one anomaly detection system inside the CPS, to complete an adversarial attack, we need to consider all defences.
- A CPS system is normally complicated and composed with sensors and actuators, and for different physical components, the data type and range are different.
- Unlike statistical attacks, CPS is dynamic and hard to predict, a minor change may lead to the whole system break down.

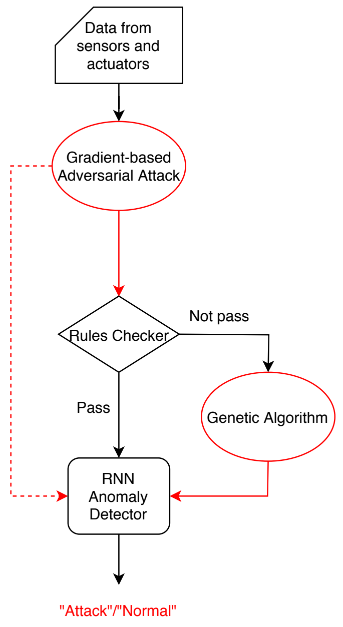

Figure 3: An Illustration of Data Flow for Adversarial Attack Scenario

For a basic set up of CPSs, data from sensors and actuators go directly to RNN predictor for attack detection, we apply a thread model with gradient-based adversarial attack to deceive the anomaly detector directly. While for many CPSs, a rule checking system as we described in 2.5 is implemented to check if data obey the rule (difficulty 2). For those systems, if the adversarial attack is able to deceive RNN predictor but detected by the rule checking system, we use generative algorithm to overcome the rule checker. Figure 3 provides an overview of the thread model and algorithm 1 show the logic to generate the adversarial examples.

3.1 Dataset

Our study is based on the SWaT dataset [42], which is a publicly available dataset [41] and has been used in multiple projects [29, 64]. The dataset records the system state of 26 sensor values and 25

Table 3: Example of rule

Num	Condition	Rule	Time
1	LIT101≤500	MV101=2	12
2	LIT101≤250	P101=1 AND P102=1	2
3	LIT301≤800	P101=2 OR P102=2	12
4	AIT201>260 and FIT201 >0.5	P201=1 AND P202=1	2

Algorithm 1: Overall Algorithm

```
Input: Data $S_i$; RNN predictor model f; rule Checker R;
       noise level $\delta$
Output: adversarial examples
1 Get gradient direction from f of loss r.p.t inputs data $S_i$;
2 Apply the noise $\delta$ to $S_i$ along or opposite the sign of
    gradients to get $S_i'$;
3 if $S_i'$ cannot pass rule checker then
4 |   $S_i''$ = GA(S'[i]);
5 else
6 |   $S_i''$ = $S_i'$;
7 end
8 return $S_i''$
```

actuators (in total 51 features) every second. The sensor values are integer or float while the actuator values are discrete with 0 (actuator is opening/closing), 1 (actuator is closed) or 2 (actuator is open). The dataset consists of two types of data:

- *Normal data*: The normal dataset S_n is collected for 7-days (a total of xxx records) when the system is under normal operation. The data is used to train the LSTM anomaly detector.
- *Attack data*: The attack dataset S_a is collected for 4-days (a total of 449909 records) by 36 attacks with labels (normal or attack). The data is used as testing data for the trained anomaly detector.

3.2 RNN predictor

We train a RNN in LSTM architecture from normal data as the prediction model. The trained model is a many-to-one prediction model $f : S \longrightarrow X$ which takes a certain sequence (parameterized by window size) of data historian as input and makes the prediction of the output of the coming timestamp. For example, a window size of 10 represents a sequential input of the past 10 timestamps.

Once we obtain the predicted output at each timestamp, we then adopt the CUSUM algorithm [15] to decide whether the system is in abnormal state as follows. The difference d at timestamp i (denoted as $d[i]$) is calculated from the predicted value $x'[i]$ by the RNN model and the actual value $x[i]$, the difference d will then be added cumulatively with an allowable slack c. We calculate the CUSUM for each sensor with both positive and negative value by the following formula:

$$\begin{aligned} SH[i] &= max(0, SH[i-1] + d[i] - c) \\ SL[i] &= min(0, SL[i-1] + d[i] + c) \\ d[i] &= X'[i] - x[i], \end{aligned} \tag{1}$$

where SH represents the set of high cumulative sum and SL represents the low cumulative sum, d is the difference between the predicted value x and actual value x', and c is the allowable slack which is defined as 0.05 multiplied by the standard deviation of S. Besides, two thresholds, i.e., a Upper Control Limit (UCL) and a Lower Control Limit (LCL) for SH and SL to compare with respectively are required to check whether the system is in abnormal state. Normally, UCL and LCL are defined according to the experiment validation of the training data. Table 4 shows the UCL and LCL from stage 1 as an example.

3.3 Adversarial Attacks

Our next step is to construct adversarial examples which aim to fool the anomaly detectors. We consider two cases. Firstly, we construct adversarial examples only to fool the RNN predictor. Secondly, we take a step further to fool both the RNN predictor and the rule checking system, which is more challenging. In both cases, we assume a white-box attack that the attacker could access the trained RNN model (and its parameters) and the attacker is able to compromise the data transmitted to the PLCs. The goal of the attacker is thus finding a minimum perturbation δ on the original input $\mathbf{x}$ such that the detector will make a different decision from the original one. Formally, given the input x at each timestamp and the RNN detector f, the objective of an attacker is to:

$$\begin{aligned} &\min\ \delta \\ &s.t.,\ f(\mathbf{x}) \neq f(\mathbf{x}+\delta)\ and\ ||\delta|| \leq \tau \end{aligned} \tag{2}$$

where τ is a small value which restricts the manipulation range of $\mathbf{x}$ according to a certain norm $||.||$ (to measure the distance of the modification). The success of such an attack will fool the detector in two folds. In the case that the detector detects an actual anomaly, the adversarial attack will be able to bypass the detector and put the system in danger (without noticing that the system is in the abnormal state). In the case that the detector detects an actual normal status, the adversarial attack will deceit the detector to raise false alarms. We remark that how to construct adversarial examples using Eq. 2 has been extensively investigated in multiple domains [31, 39, 50, 63]. In the following, we introduce in details how we solve the problem in our setting.

3.3.1 *Adversarial Attacks on RNN predictor.*

In the primary setting, we only aim to construct adversarial examples for the RNN detector. Notice that the detector uses CUSUM to check the difference between the predicted output and the actual output. Thus, we need to minimize the difference between the predicted value and targeted value in order to fool the CUSUM checker. On the other hand, we may need to minimize the loss of actual value and the targeted by RNN. Consider the RNN predicts sequential data, (2) is not suitable

Table 4: Upper and Lower control limit

Senor	Upper Control Limit	Lower Control Limit
FIT101	2	-1.5
LIT101	50	-2

here as the output for this equation is for classification. Thus the equation should be modified with an adversarial target output $y*$ which is to be matched as close as possible with $f(x')$ where x' is the adversarial example, thus the equation could be modified as:

$$\begin{aligned} &\min\ \delta \\ &s.t.,\ ||f(\mathbf{x}+\delta) - y*|| \leq \Delta\ and\ ||\delta|| \leq \tau \end{aligned} \tag{3}$$

where $y*$ is the desired output of the model, Δ is the acceptable difference between the desired model output and adversarial output.

The key problem is to calculate the perturbation δ. For this, we borrow the idea of the Fast Gradient Sign Method (FGSM) method in image domain, which perturbs the input *along* the gradient direction to *maximize* the change in the model prediction[32]. Consequently, if the perturbation is added *opposite* the direction of the Gradients, the change of the model prediction is *minimized*. Perturbations can be formalized as:

$$\begin{aligned} &\mathbf{x'} = \mathbf{x} \pm \delta \\ &\delta = \epsilon \cdot sgn(\nabla_x l(\mathbf{x}, y')) \end{aligned} \tag{4}$$

where $\mathbf{x'}$ is the calculated adversarial examples, ϵ is a constant representing the magnitude of the perturbation and l is the loss function. In our work, we choose Mean Square Error (MSE) as the loss function, which is the most commonly used for regression tasks. In our case, due to the target of deceiving the RNN predictor, we increase the loss with adding δ if the actual status of the data point is "normal" (S_a = "normal") and decrease the loss by minus δ if the actual status is "attack" (S_a = "attack"). Besides, notice that in Eq. 4 ϵ is a constant, which is not suitable for CPS data set with a combination of different data types. To solve the two problems, we propose an adapted version of the Eq.4 in practice:

$$\mathbf{x'} = \begin{cases} \mathbf{x}+\delta, & if\ S_a = "normal" \\ \mathbf{x}-\delta, & if\ S_a = "attack" \end{cases} \tag{5}$$

$$\delta = sgn(\nabla_{\mathbf{x}} l(\mathbf{x}, y')) \cdot \vec{\epsilon},$$

where $\vec{\epsilon}$ is a diagonal matrix, and the value of each element from $\vec{\epsilon}$ is defined according to the data type of each feature which represents the magnitude of the perturbation for each feature. The smaller the $\vec{\epsilon}$ is, the less is the perturbation.

We distinguish three different types of feature (to solve difficulty 3 in section 3), i.e., actuators, valves and pumps and add different perturbation on them. For sensors, we add a perturbation λ, which represents the percentage of the perturbation w.r.t. the original value. There are two types of actuators, valves and pumps. For valves, there are three values: 0 for opening/closing, 0.5 for close and 1 for open. For pumps, there are two values with 0 for close, and 1 for open. In order to follow the character of discrete values for actuators, we define the ϵ of valves as 0.5 and the ϵ of pumps is defined as 1.In summary, the value of the diagonal matrix $\vec{\epsilon}$ can be defined as:

$$\vec{\epsilon}_{jj} = \begin{cases} \lambda, & if\ \vec{F}[j] \in \vec{U},\ where\ \vec{U} \subset \vec{F} \\ 0.5, & if\ \vec{F}[j] \in \vec{V_m},\ where\ \vec{V_m} \subset \vec{F} \\ 1, & if\ \vec{F}[j] \in \vec{V_p},\ where\ \vec{V_p} \subset \vec{F} \end{cases}$$

where $\vec{F}$ is the set of all features, $\vec{U}$ is the set of sensors, $\vec{V_m}$ is the set of all valves, and $\vec{V_p}$ is the set of pumps. For example, if we only consider data from stage 1, $\vec{\epsilon}$ =

$$\begin{vmatrix} 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$

To make sure the modified data are following the formats of data set, we clip the adversarial examples to 0 to 1 as it supposed to be. In specific, for a pump value, if the sign of the gradient at $V_p[i,j]$ is positive, $V_p'[i,j] = S[i,j] + 1$. where $V_p'[i,j]$ is the adversarial example of $S[i,j]$. However, we get $V_p'[i,j] = 2$ which is not acceptable if $V_p[i,j] = 1$. Thus clipping is necessary. In physical world, $V_p[i,j] = 0$ means the pump is closed and after adversarial attack, it change to 1, means it is open after the attack.

We thus construct adversarial examples for the RNN detector by adding noise in the above way and algorithm 2

Algorithm 2: Adversarial examples

```
Input: Data $S_i$; RNN predictor model f
Output: adversarial examples
1  $\delta \leftarrow sgn(\nabla_{\mathbf{x}} l(\mathbf{x}, y')) \cdot \vec{\epsilon}$ ;
2  if $S_i$ is normal data then
3      $S_i' = S_i + \delta$;
4  else
5      $S_i' = S_i - \delta$;
6  end
7  if $S_i'$ cannot pass rule checker then
8      $S_i''$ = GA(S'[i]);
9  else
10     $S_i'' = S_i'$;
11 end
12 return $S_i''$
```

3.3.2 Adversarial attacks considering rule checking. The rule checking mechanism is widely used to detect anomaly of CPS by many works [12, 13, 21, 52]. An example rule from SWaT system could be that when the value of "LIT101" is smaller than 500, the value of "MV101" should always equal to 2 after 12 seconds, otherwise an alarm will arise to indicate anomalies. In this setting, we aim to construct adversarial examples to bypass both the RNN detector and the rule checking system.

As the rule checking system makes decisions depending on the actuators status, we propose to use genetic algorithm(GA)[30] to help us generate the ideal combination of actuator status. Due to the difficulty of different range for sensors, pumps and valves, we develop the genetic algorithm to generate values according to data types. The algorithm 3 give an overview of how genetic algorithm works in this project.

Algorithm 3: GA

```
Input: Data S'[i];fitness function G; population size
       n;mutation probability $p_m$; rule Checker R
Output: adversarial examples
1 Generate initial population P;
2 Compute fitness g for each candidate from P;
3 repeat
4     Select parents from P with a probability of fitness
        distribution;
5     Apply crossover and mutation with $p_m$ to get new
        candidates;
6     Select individuals from old and new candidates based on
        their fitness to participate in new P;
7     $S''[i] \leftarrow fittest\ candidate\ from\ P$;
8 until R(S"[i]) = True or iteration = max iteration;
9 return S"[i]
```

Inside the algorithms, R is a set of rule implemented in the system (for SWaT, there are in total 23 rule to monitor the system status.). Firstly, a set of random data points generated with population size n. Those data points should have the same data format with attack data set S, such as 26 sensor values with range from 0-1, 9 valve values from [0, 0.5, 1], and 16 pump values from [0, 1]. Afterwards, we calculate the fitness number with $g = c1 * (1/c2)$, where 1) $c1 = 1$ if R(x) = True, which means the data point x has passed rule checker, and $c1 = 0$ otherwise which indicating it is impossible to be chosen; 2) c2 is the vector magnitude of the difference between the the candidate and its original value. The lower c2 is, the fewer modification on original data, which has a higher possibility to be chosen. Then we start to look at the repeating part, where the new population generated. Each candidate is assigned a probability according to the roulette wheel selection [30], which means probability of each candidate being selected as parent is $g_i / \sum_{j=1}^{n} g_j$. From the selected parents, the generation is sampled with crossover and mutation, in details, a random point is chosen to divide the vector and swap the sub-vector with each other. Afterwards, we randomly mutate a point with fix mutation probability p_m. Next, the fitness score is calculated for all candidates and top n fittest candidates form the new population. The fittest candidate value pass to $S''[i]$. This iteration continues until $S''[i]$ pass rule checker or timeout. Then we get the new adversarial example $S''[i]$.

Our experiment shows that this way works for passing most rules. We apply our methodology to construct adversarial examples in this setting and found that 461 out of 449909 data points have been caught by the rule checking system. After removing the noises

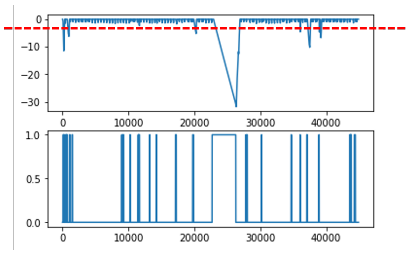

Figure 4: A CUSUM value example for sensor LIT101

accordingly, we obtain the results shown in the last row of Table 5. From the results we can see that we still achieve similar reduction in the performance of the anomaly detector as explained in the above section.

3.4 Defence model

Our next question is whether it is possible to detect the constructed examples efficiently as a defence. Our attempt is to create a binary classifier of different model types (e.g., def-NN and random forest) trained from a set of original data points and an equal number of adversarial examples. We then use the trained classifier to test whether the classifier can detect adversarial examples constructed from other data points. We focus on the defence neural network (def-NN) performance and use the random forest as a comparison.

4 EVALUATION ON PHYSICAL SYSTEM

We evaluated our experiment to the full data set of real world cyber-physical systems, SWaT system that we have introduced in Section2 background. The experiments are done on a purpose to answer our research questions raised in section 2.5.2.

4.1 Experiment setting

We conducted our experiments on a GPU laptop with 1 Intel(R) Core(TM) i7-8750H CPU at 2.20GHz, 16GB system memory and GPU of NVIDIA GeForce GTX 1050 Ti with Max-Q Design.

The threshold selection for CUSUM is according to the validation of the attack data. We compared the cusum value for each sensor with attack labels, and set the threshold as the lowest peak of the CUSUM value that indicates an attack, so each sensor is possible to indicate 0 attacks or more than 1 attacks. Fig 4 presents a cusum value example for sensor "LIT101", the top graph is the negative cusum value *SL* of "LIT101". The bottom picture is the actual label of data, 1 means under attack and 0 means normal status. By comparing the two graph, almost each peak value from the *SL* indicated an attack, we choose the lowest peak value as the threshold and here it's indicated by the red dot line with value of −2. All thresholds are calculated with this methods.

The above CUSUM-RNN model proposed by J.Goh [29] is also evaluated on SWaT system, however only on stage 1 (Supply and Storage). We expand the model with all stages of the system to make sure our adversarial attack works on the completed system.

4.2 Research questions

We have three research questions to evaluate our experiments with SWaT and WADI system.

RQ1: how should the attacker compromise the original data and construct the adversarial examples to fool the anomaly detector?

On SWaT system, the adversarial attack has been applied following the Eq.5. To balance the modification and effectiveness of the model, we design three experiments find the optimized solution. Experiment 1 and 2 is to apply the noise to sensor data only with $\lambda = 0.01$ and $\lambda = 0.1$ to check the influence for different noises. Experiment 3 and 4 is based on experiment 1 and 2, but we also change the actuator values along the sign of calculated gradient. In details, $\vec{\epsilon}$ is defined with 26 elements with $\lambda = 0.01$ for sensor value for experiment 3 and $\lambda = 0.1$ for sensor value for experiment 4, with 6 elements with 0.5 for actuator valves and 19 elements with 1 for actuator pumps value with locations responding to each feature, and the same logic has been applied to WADI system.

During the experiments, firstly, the RNN predictors for the two systems has been created, all data are collected with a frequency of 1Hz, so we select 10 seconds as the window size for WADI due to the slowly change of real systems. However, SWaT system has rule checking mechanism which check the data status with a delay more than 10s sometimes, so we choose the maximum delay of 12s from the rule checking mechanism as the window size of the SWaT system to make sure the window size can at least catch one attack. As the system is dynamic and the ground truth is changing whenever the data is changing (difficulty 4 in section 3), we compare the results of ground truth model $T(S)$ with the results from learning-based anomaly detector $C(f(S))$ to calculate the original anomaly detector accuracy of data S, and the difference between $T(S')$ and $C(f(S'))$ to get accuracy of data S'.

We can see the results Table 5 for SWaT and Table 6 for WADI, where the precision, f1 and accuracy has been apparently reduced after the adversarial attack. Results of experiments 1 and 2 are given from the second and third rows of the Table 5, if we only add noises on sensor values, the more noises we added, the lower accuracy of the anomaly detector. However, there is a possibility that the noises are added in an extreme condition so that the noised data has exceed the threshold and lead to an "actual attack", in this condition, the data is classified as attack by the ground truth model T. The slightly higher results of precision, recall and f1 for 10% noises than 1% indicates that the higher noises leads to more "actual attacks" defined by model T. The fourth and fifth row provide the results that we apply the noises to sensor and change actuator values at the same time. The accuracy has been reduced more than if we apply noises to sensor only, and the false negative values of 0.78 implies that many false alarms has been generated by the anomaly detector. As for row 2,4 and row 3,5, we can see if we apply the same noise range for sensor value, a manipulation of actuators could provide an extra decrease of model accuracy.

We did the same experiment on WADI, and get the results as shown by Table 6. WADI is a bit different as we do not have its control points information to determine the ground truth, thus we calculate the operation range under normal conditions with a small tolerance to set the ground truth model T. As it has more features (121) than SWaT (51), the complexity is more apparent and behavior is more unpredictable.

RQ2: how should the attacker compromise the original data to fool the rule checking system and the anomaly detector?

Rule checking system is commonly used in industrial CPS as a basic defence mechanism, as WADI has not established an rule checking system, so our second experiment is only applied to SWaT testbed. The rule checking mechanism is settles to monitor the condition and invariant in SWAT system, it will check in total 20 actuators status according to the 11 sensor values for SWaT system. Aware of the rule checking mechanism relies on the change of actuators, we design the experiment 5 to use GA to generate actuator data to check if this is sufficient to pass the rule checker. We assume that the attacker does not have the access to the content of the rule, but knows whether the data could pass the rule checker or not. The rules include the relationship between the sensor and actuator values, so we divided an data point x into $x_a = [x_a1, x_a2...]$ and $x_s = [x_s1, x_s2...]$, and use GA to generate possible x_a and keep x_s to combine back to calculate fitness score. As our goal is to generate adversarial attack rather than "actual attacks", so we only care those rule checked attacks that are caused by adversarial attacks (i.e. applying noises), so we won't use GA to replace those attacks that being detected before and after attacks.

The results for experiment 5 can be found from the last row of Table 5. The accuracy has been slightly increased indicating that using GA for actuator value selection may sacrifice the drop of accuracy which is reasonable, as GA generated values do not change following the sign of gradients. While the overall performance is almost the same comparing with the experiment 4 results (row 5 from Table 5), so GA could be used for overcoming the rule checker without reduce the anomaly detector performance.

RQ3: how can we design a defence against such adversarial examples?

The adversarial examples have more noises comparing to original data, so we design a defence neural network (def-NN) to distinguish if the data are adversarial examples or original data from the system. The def-NN performs as a basic binary classifier. We train the model with half original data and half adversarial examples as training data and the rest half as testing data. The def-NN model is designed with three layers and 100 units for each layer, and we choose *binary cross entropy* as our loss function. We also use a 10 estimators random forest for comparison.

The results can be found from Table 7, both def-NN and random forest model works well on detecting the noises, which indicates the original data has a certain pattern that the noises we added, though small but are easy to change the pattern. Moreover, comparing to change sensor value only, changing all data would be 100% detected by the defence models. As for changing sensor value only, the larger value we modified, the easier to be detected.

Table 5: Before and after adding noises of anomaly detector for SWaT

	precision	recall	f1	accuracy	False negative	False positive
Original Anomaly Detector	0.96	0.86	0.91	89.40%	0.05	0.14
Noises to sensor (1%)	0.62	0.53	0.57	52.38%	0.48	0.47
Noises to sensor (10%)	0.65	0.58	0.61	48.78%	0.72	0.42
Noises to all data (1%)	0.52	0.50	0.51	51.85%	0.46	0.50
Noises to all data (10%)	0.12	0.84	0.21	28.84%	0.78	0.16
Noises to all data + GA for actuator	0.14	0.76	0.24	39.56%	0.62	0.34

Table 6: Before and after adding noises of anomaly detector for WADI

	precision	recall	f1	accuracy	False negative	False positive
Original Anomaly Detector	0.32	0.16	0.22	93.29%	0.02	0.84
Noises to sensor (1%)	0.43	0.65	0.52	52.85%	0.55	0.35
Noises to sensor (10%)	0.42	0.67	0.52	49.88%	0.62	0.33
Noises to all data (1%)	0.34	0.96	0.50	33.28%	0.99	0.04
Noises to all data (10%)	0.33	0.95	0.49	32.23%	0.99	0.04

Table 7: Defence for adversarial attacks on SWaT

	def-NN			RF		
	precision	recall	f1	precision	recall	f1
Noises to sensor (1%)	0.98	0.99	0.99	1	0.99	0.99
Noises to sensor (10%)	0.99	0.99	0.99	1	1	1
Noises to all data (1%)	1	1	1	1	1	1
Noises to all data (10%)	1	1	1	1	1	1
Noises to all data + GA for actuator	1	1	1	1	1	1

4.3 Evaluation constrains

There are some limitations for the evaluation and application validity. First, though the two CPS systems are real and operational, they are still testing bed and note industrialized. Secondly, the data are collected and labeled due to manual work, there are some data points are not accurate and may increase the bias of the results. Finally, we only considered the recorded attacks to train and calculated ground truth, so the methods may not work well for non-recorded attacks. Similarly, the defence method (i.e. def-NN) has to be trained for each kind of adversarial examples (10% noises and 1% noises), and it does not work for un-known adversarial examples (e.g. 5% noises).

5 RELATED WORK

This work focus on adversarial attacks on LSTM-RNN which is implemented as a part of anomaly detector in a cyber-physical system. In this section, we discuss some researches with relevant themes such as adversarial attacks and some relevant works done on SWaT/WADI systems.

Anomaly detection on CPS: A number of researches has explored the anomaly detection on CPS, these approaches [8, 14, 24, 35, 36, 40, 46, 48, 55, 58, 61] unusual monitor the changes or suspicious patterns from the logs of a CPS. Y.Harada et al [35] has proposed a statistical way to monitor the outliers from logs. This statistical methods can include machine leaning or deep learning methods. J.Inoue et al [40] has provide an unsupervised machine learning methods to detect anomalies and M. Kravchik et al [46] proposed to use convolutional neural network to detec cyber attacks in control systems.

SWaT system as a typical cyber-physical system has been widely explore by researches, and there are many works including anomaly detection done on the public data set [41, 42]. These works includes unsupervised learning methods [29, 40, 46], which includes deep learning methods, and supervised leaning methods [19, 20], who injected simulated faults to get attack data. Ahmed et al. [9, 11] has worked on fingerprinting the noises patterns to detect anomalies while Adepu and Mathur [2, 5–7] has manually derived the invariants or what we call "rules" by monitoring the normal operation of the system. C. Feng [26] also tried to get invariants but using a systematic way to learn the invariants.

Neural networks in CPS: Neural networks as a state-of-art technique has been adopted in CPS as well. Lv et al. [49] have introduced a multiple layers neural networks for probabilistic estimation of brake pressure for electrified vehicles. Nanduri et al. [54] have applied RNN as a part of anomaly detector in aircraft data. Sargolzaei et al. [60] have designed a fault detector with neural networks to detect fault data injection for vehicular cyber-physical systems with wireless communications. Similar works with anomaly detection with Neural networks have been evaluate on smart grid [45] and gasoil plant [27], which use LSTM as predictor. Eiteneuer et al. [25]

have applied LSTM neural networks to learn the behavior of a water tank.

Adversarial attacks: Papernot et al. have pointed out the limitations of deep learning in adversarial settings [57] and Nguyen et al. have pointed in their work [56] that it's easy to produce images that completely unrecognizable by human but recognizable with 99.99% confidence by state-of-art neural networks. Adversarial attacks have many mature toolbox [1, 33, 53] to generate adversarial attacks. Adversarial attacks has been adopted in many applications in physical world [47] including image [62] and audio [16, 17]. However, there are few works on CPS.

6 CONCLUSION

In this paper, we introduce a case study of adversarial attack for real world cyber-physical systems, and we create possible defences for adversarial attack. Firstly, we propose a adversarial attacks for anomaly detector with LSTM-RNN as predictor in a cyber-physical system, we applied noises with different ranges and compare their impacts. Then we consider the condition with rule checking mechanism and provide a solution to overcome the checker, which is by using GA. Finally, we explore if there is any possible methods to detected the adversarial examples and tried def-NN and random forest for the experiments. Afterwards, we evaluate our methods on two testing beds, SWaT and WADI and get results to show that the thread model has effectively drop the performance of the pre-implemented anomaly detectors, and even though we consider the rule checking mechanism to weaken the assumptions of attackers, we can still achieve a similar results with our improved adversarial attack methods.

For future works, we will consider weaken the attacker model with black box adversarial attack, and we will explore other methods to overcome the rule checker such as SMT solver. The evaluation should be done with the actual system and real time attack application. Moreover, we will investigate possible solutions for adversarial example detection.

REFERENCES

[1] 2020. tensorflow/cleverhans. https://github.com/tensorflow/cleverhans original-date: 2016-09-15T00:28:04Z.

[2] Sridhar Adepu and Aditya Mathur. 2016. Distributed Detection of Single-Stage Multipoint Cyber Attacks in a Water Treatment Plant. In *Proc. ACM Asia Conference on Computer and Communications Security (AsiaCCS 2016)*. ACM, 449–460.

[3] S. Adepu and Aditya Mathur. 2016. Generalized attacker and attack models for Cyber-Physical Systems. In *Proceedings of the 40th Annual International Computers, Software & Applications Conference, Atlanta, USA*. IEEE, Washington, D.C., USA, 283–292.

[4] S. Adepu and A. Mathur. 2016. An Investigation into the response of a Water Treatment System to Cyber Attacks. In *Proceedings of the 17th IEEE High Assurance Systems Engineering Symposium, Orlando*.

[5] Sridhar Adepu and Aditya Mathur. 2016. Using Process Invariants to Detect Cyber Attacks on a Water Treatment System. In *Proc. International Conference on ICT Systems Security and Privacy Protection (SEC 2016) (IFIP AICT)*, Vol. 471. Springer, 91–104.

[6] Sridhar Adepu and Aditya Mathur. 2018. Assessing the Effectiveness of Attack Detection at a Hackfest on Industrial Control Systems. *IEEE Transactions on Sustainable Computing* (2018).

[7] Sridhar Adepu and Aditya Mathur. 2018. Distributed Attack Detection in a Water Treatment Plant: Method and Case Study. *IEEE Transactions on Dependable and Secure Computing* (2018).

[8] Ekta Aggarwal, Mehdi Karimibiuki, Karthik Pattabiraman, and André Ivanov. 2018. CORGIDS: A Correlation-based Generic Intrusion Detection System. In *Proc. Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC 2018)*. ACM, 24–35.

[9] Chuadhry Mujeeb Ahmed, Martín Ochoa, Jianying Zhou, Aditya P. Mathur, Rizwan Qadeer, Carlos Murguia, and Justin Ruths. 2018. *NoisePrint*: Attack Detection Using Sensor and Process Noise Fingerprint in Cyber Physical Systems. In *Proc. Asia Conference on Computer and Communications Security (AsiaCCS 2018)*. ACM, 483–497.

[10] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. 2017. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks - CySWATER '17*. ACM Press, Pittsburgh, Pennsylvania, 25–28. https://doi.org/10.1145/3055366.3055375

[11] Chuadhry Mujeeb Ahmed, Jianying Zhou, and Aditya P. Mathur. 2018. Noise Matters: Using Sensor and Process Noise Fingerprint to Detect Stealthy Cyber Attacks and Authenticate sensors in CPS. In *Proc. Annual Computer Security Applications Conference (ACSAC 2018)*. ACM, 566–581.

[12] Ravi Akella and Bruce M. McMillin. 2009. Model-checking BNDC properties in cyber-physical systems. In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, Vol. 1. IEEE, 660–663.

[13] Rajeev Alur. 2015. *Principles of cyber-physical systems.* MIT Press.

[14] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 2018. Truth Will Out: Departure-Based Process-Level Detection of Stealthy Attacks on Control Systems. In *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS 2018)*. ACM, 817–831.

[15] J. Roger Bray and J. T. Curtis. 1957. An Ordination of the Upland Forest Communities of Southern Wisconsin. *Ecological Monographs* 27, 4 (Feb. 1957), 325–349. https://doi.org/10.2307/1942268

[16] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden voice commands. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 513–530.

[17] Nicholas Carlini and David Wagner. 2018. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. IEEE, San Francisco, CA, 1–7. https://doi.org/10.1109/SPW.2018.00009

[18] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *ACM Comput. Surv.* 41, 3, Article 15 (July 2009), 58 pages. https://doi.org/10.1145/1541880.1541882

[19] Yuqi Chen, Christopher M. Poskitt, and Jun Sun. 2016. Towards Learning and Verifying Invariants of Cyber-Physical Systems by Code Mutation. In *Proc. International Symposium on Formal Methods (FM 2016) (LNCS)*, Vol. 9995. Springer, 155–163.

[20] Yuqi Chen, Christopher M. Poskitt, and Jun Sun. 2018. Learning from Mutants: Using Code Mutation to Learn and Monitor Invariants of a Cyber-Physical System. In *Proc. IEEE Symposium on Security and Privacy (S&P 2018)*. IEEE Computer Society, 648–660.

[21] Yuqi Chen, Christopher M. Poskitt, and Jun Sun. 2018. Learning from mutants: Using code mutation to learn and monitor invariants of a cyber-physical system. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 648–660.

[22] Yuqi Chen, Christopher M. Poskitt, Jun Sun, Sridhar Adepu, and Fan Zhang. 2019. Learning-Guided Network Fuzzing for Testing Cyber-Physical System Defences. In *Proc. IEEE/ACM International Conference on Automated Software Engineering (ASE 2019)*. IEEE Computer Society, 962–973.

[23] Feng Cheng, Tingting Li, and Deeph Chana. 2017. Bloom Filters and LSTM Networks for Anomaly Detection in Industrial Control Systems. In *47th IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2017)*. IEEE, Washington D.C. , USA, 1–12.

[24] Long Cheng, Ke Tian, and Danfeng (Daphne) Yao. 2017. Orpheus: Enforcing Cyber-Physical Execution Semantics to Defend Against Data-Oriented Attacks. In *Proc. Annual Computer Security Applications Conference (ACSAC 2017)*. ACM, 315–326.

[25] Benedikt Eiteneuer and Oliver Niggemann. 2018. LSTM for Model-based Anomaly Detection in Cyber-Physical Systems.. In *DX@ Safeprocess*.

[26] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. 2019. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In *Proc. Annual Network and Distributed System Security Symposium (NDSS 2019)*. The Internet Society.

[27] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. 2016. Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model. *arXiv:1612.06676 [cs, stat]* (Dec. 2016). http://arxiv.org/abs/1612.06676 arXiv: 1612.06676.

[28] David Formby, Preethi Srinivasan, Andrew Leonard, Jonathan Rogers, and Raheem Beyah. 2016. Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems. https://doi.org/10.14722/ndss.2016.23142

[29] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. 2017. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 140–145.

[30] David E. Goldberg. 1989. Genetic Algorithms in Search Optimization and Machine Learning. https://doi.org/10.5860/choice.27-0936

[31] Yuan Gong and Christian Poellabauer. 2017. Crafting adversarial examples for speech paralinguistics applications. *arXiv preprint arXiv:1711.03280* (2017).

[32] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[33] Dbou Goodman, Hao Xin, Wang Yang, Wu Yuesheng, Xiong Junfeng, and Zhang Huan. 2020. Advbox: a toolbox to generate adversarial examples that fool neural networks. *arXiv:2001.05574 [cs, stat]* (Jan. 2020). http://arxiv.org/abs/2001.05574 arXiv: 2001.05574.

[34] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. 2017. Adversarial examples for malware detection. In *European Symposium on Research in Computer Security*. Springer, 62–79.

[35] Yoshiyuki Harada, Yoriyuki Yamagata, Osamu Mizuno, and Eun-Hye Choi. 2017. Log-Based Anomaly Detection of CPS Using a Statistical Method. In *Proc. International Workshop on Empirical Software Engineering in Practice (IWESEP 2017)*. IEEE, 1–6.

[36] Zecheng He, Aswin Raghavan, Guangyuan Hu, Sek M. Chai, and Ruby B. Lee. 2019. Power-Grid Controller Anomaly Detection with Enhanced Temporal Deep Learning. In *Proc. IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom 2019)*. IEEE, 160–167.

[37] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[38] Elike Hodo, Xavier Bellekens, Andrew Hamilton, Pierre-Louis Dubouilh, Ephraim Iorkyase, Christos Tachtatzis, and Robert Atkinson. 2016. Threat analysis of IoT networks using artificial neural network intrusion detection system. In *2016 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 1–6.

[39] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).

[40] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M. Poskitt, and Jun Sun. 2017. Anomaly Detection for a Water Treatment System Using Unsupervised Machine Learning. *2017 IEEE International Conference on Data Mining Workshops (ICDMW)* (Nov. 2017), 1058–1065. https://doi.org/10.1109/ICDMW.2017.149 arXiv: 1709.05342.

[41] iTrust. [n.d.]. Dataset and Models. https://itrust.sutd.edu.sg/dataset/.

[42] Goh J., Adepu S., Junejo K. N, and Mathur A. 2016. A Dataset to Support Research in the Design of Secure Water Treatment Systems. Springer, New York, USA, 1–13.

[43] Yue Jia and Mark Harman. 2011. An Analysis and Survey of the Development of Mutation Testing. *IEEE Transactions on Software Engineering* 37, 5 (Sept. 2011), 649–678. https://doi.org/10.1109/TSE.2010.62

[44] Anna Magdalena Kosek. 2016. Contextual anomaly detection for cyber-physical security in Smart Grids based on an artificial neural network model. In *2016 Joint Workshop on Cyber- Physical Security and Resilience in Smart Grids (CPSR-SG)*. IEEE, Vienna, Austria, 1–6. https://doi.org/10.1109/CPSRSG.2016.7684103

[45] Anna Magdalena Kosek. 2016. Contextual anomaly detection for cyber-physical security in smart grids based on an artificial neural network model. In *2016 Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG)*. IEEE, 1–6.

[46] Moshe Kravchik and Asaf Shabtai. 2018. Detecting Cyber Attacks in Industrial Control Systems Using Convolutional Neural Networks. In *Proc. Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC 2018)*. ACM, 72–83.

[47] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* (2016).

[48] Qin Lin, Sridhar Adepu, Sicco Verwer, and Aditya Mathur. 2018. TABOR: A Graphical Model-based Approach for Anomaly Detection in Industrial Control Systems. In *Proc. Asia Conference on Computer and Communications Security (AsiaCCS 2018)*. ACM, 525–536.

[49] Chen Lv, Yang Xing, Junzhi Zhang, Xiaoxiang Na, Yutong Li, Teng Liu, Dongpu Cao, and Fei-Yue Wang. 2017. Levenberg–Marquardt backpropagation training of multilayer neural networks for state estimation of a safety-critical cyber-physical system. *IEEE Transactions on Industrial Informatics* 14, 8 (2017), 3436–3446.

[50] Francesco Marra, Diego Gragnaniello, and Luisa Verdoliva. 2018. On the vulnerability of deep learning to adversarial attacks for camera model identification. *Signal Processing: Image Communication* 65 (2018), 240–248.

[51] Aditya P. Mathur and Nils Ole Tippenhauer. 2016. SWaT: a water treatment testbed for research and training on ICS security. In *Proc. International Workshop on Cyber-physical Systems for Smart Water Networks (CySWATER 2016)*. IEEE Computer Society, 31–36.

[52] Sayan Mitra, Tichakorn Wongpiromsarn, and Richard M. Murray. 2013. Verifying cyber-physical interactions in safety-critical systems. *IEEE Security & Privacy* 11, 4 (2013), 28–37.

[53] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2574–2582.

[54] Anvardh Nanduri and Lance Sherry. 2016. Anomaly detection in aircraft data using Recurrent Neural Networks (RNN). In *2016 Integrated Communications Navigation and Surveillance (ICNS)*. IEEE, Herndon, VA, USA, 5C2–1–5C2–8. https://doi.org/10.1109/ICNSURV.2016.7486356

[55] Vedanth Narayanan and Rakesh B. Bobba. 2018. Learning Based Anomaly Detection for Industrial Arm Applications. In *Proc. Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC 2018)*. ACM, 13–23.

[56] Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 427–436.

[57] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 372–387.

[58] Fabio Pasqualetti, Florian Dorfler, and Francesco Bullo. 2011. Cyber-physical attacks in power networks: Models, fundamental limitations and monitor design. In *Proc. IEEE Conference on Decision and Control and European Control Conference (CDC-ECC 2011)*. IEEE, 2195–2201.

[59] Arman Sargolzaei, Carl D. Crane, Alireza Abbaspour, and Shirin Noei. 2016. A Machine Learning Approach for Fault Detection in Vehicular Cyber-Physical Systems. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, Anaheim, CA, USA, 636–640.

[60] Arman Sargolzaei, Carl D Crane, Alireza Abbaspour, and Shirin Noei. 2016. A machine learning approach for fault detection in vehicular cyber-physical systems. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 636–640.

[61] Peter Schneider and Konstantin Böttinger. 2018. High-Performance Unsupervised Anomaly Detection for Cyber-Physical System Networks. In *Proc. Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC 2018)*. ACM, 1–12.

[62] Daniel F Smith, Arnold Wiliem, and Brian C Lovell. 2015. Face recognition on consumer devices: Reflections on replay attacks. *IEEE Transactions on Information Forensics and Security* 10, 4 (2015), 736–745.

[63] Muhammad Usama, Junaid Qadir, and Ala Al-Fuqaha. 2018. Adversarial attacks on cognitive self-organizing networks: The challenge and the way forward. In *2018 IEEE 43rd Conference on Local Computer Networks Workshops (LCN Workshops)*. IEEE, 90–97.

[64] Jingyi Wang, Jun Sun, Yifan Jia, Shengchao Qin, and Zhiwu Xu. 2018. Towards 'Verifying'a Water Treatment System. In *International Symposium on Formal Methods*. Springer, 73–92.

[65] Z. Zohrevand, U. Glasser, H. Y. Shahir, M. A. Tayebi, and R. Costanzo. 2016. Hidden Markov based anomaly detection for water supply systems. In *2016 IEEE International Conference on Big Data (Big Data)*. 1551–1560.