Blind Concealment from Reconstruction-based Attack Detectors for Industrial Control Systems via Backdoor Attacks

Tim Walita
Saarland University
Saarbrücken, Germany
s8tiwali@uni-saarland.de

Alessandro Erba
CISPA Helmholtz Center for Information Security
and Saarbrücken Graduate School of Computer Science,
Saarland University
Saarbrücken, Germany
alessandro.erba@cispa.de

John H. Castellanos
CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
john.castellanos@cispa.de

Nils Ole Tippenhauer
CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
tippenhauer@cispa.de

ABSTRACT

Industrial Control Systems (ICS) are responsible for the safety and operations of critical infrastructure such as power grids. Attacks on such systems threaten the well-being of societies, and the lives of human operators, and pose huge financial risks. To detect those attacks, process-aware attack detectors were proposed by academia and industry to verify inherent physical correlations. Such detectors will be trained by the vendors on process data from the target system, which allows malicious manipulations of the training process to later evade detection at runtime. Previously proposed attacks in this direction rely on detailed process knowledge to predict the exact attack features to be concealed.

In this work, we show that even without process knowledge (i.e. being able to predict attack results), it is possible to launch training time attacks against such attack detectors. Our backdoor attacks achieve this by identifying 'alien' actuator state combinations that never occur in the training samples and injecting them with legitimate sensor data into the training set. At runtime, the attacker spoofs one of those alien actuator state combinations, which triggers (regardless of sensor values) the classification as 'normal'.

To demonstrate this, we design and implement five backdoor attacks against autoencoder-based anomaly detectors for 14 attacks from the BATADAL dataset collection. Our evaluation shows that our best backdoor attack implementation can achieve perfect attack concealment and accomplish an average recall of 0.19. Compared to the performance of the detector for anomalies that are not concealed by inserted triggers, our attacks decrease the detector's recall by 0.477.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; • **Computing methodologies → Neural networks**.

CPSS '23, July 10–14, 2023, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0090-3/23/07.
https://doi.org/10.1145/3592538.3594271

KEYWORDS

Backdoor Attack, Industrial Control System, Anomaly Detection

ACM Reference Format:
Tim Walita, Alessandro Erba, John H. Castellanos, and Nils Ole Tippenhauer. 2023. Blind Concealment from Reconstruction-based Attack Detectors for Industrial Control Systems via Backdoor Attacks. In *Proceedings of the 9th ACM Cyber-Physical System Security Workshop (CPSS '23), July 10–14, 2023, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3592538.3594271

1 INTRODUCTION

Industrial Control Systems (ICS) are networked computing devices that precisely control critical systems. They are the main component behind critical infrastructures in modern society. Like other computing systems, ICS are threatened by cyberattacks, and the consequence of such attacks can be severe for society. For example, in February 2021 [30], an attacker gained access to the water treatment system in Florida, USA, and attempted to poison the water supply for a 15000-people community. ICS are particularly vulnerable to cyberattacks as they rely on insecure legacy protocols and hosts, which cannot easily be secured through upgrades or updates.

Given those threats to legacy ICS systems, complementary attack detection schemes have been proposed by researchers in recent years. Those schemes exploit the intrinsic correlations between physical process data (such as sensors and actuators) [5], and detect when attacks violate those correlations causing anomalies in the system. In particular, autoencoders have shown to be especially suitable for capturing the complex behaviors of ICS [3, 12, 26, 32]. Such products will always have to be customized to the monitored process, based on a training set containing process data captured during normal operations. The efficacy of such detectors is then evaluated with a second dataset, that also contains a number of attacks to be detected. Commercial solutions for such detection schemes are now available [19], but little is known about their fundamental guarantees against attacks.

In other contexts (e.g. image classification and recognition domain [7, 27]), *backdoor attacks* have been proposed to trigger hidden behaviors of machine learning models and cause misclassification. Given this reliance on third-party training of highly security-relevant machine-learning solutions, the question is *how could the*

training phase be manipulated to allow hiding ICS attacks at runtime? While there is prior work on the related class of poisoning attacks in such a setting [21], those attacks require specific knowledge of precisely predicted feature values caused by attacks. It is unclear if backdoor attacks can be applied to the domain of process-aware attack detection for the following reasons: i) the features to be manipulated are largely continuous (instead of discrete RGB values in images), ii) at training time, attackers cannot precisely predict feature values (caused by attacks) to be concealed at runtime (i.e. we do not have a well-defined attack class and no precise model of the physical process).

In this work, we propose a set of backdoor attacks for ICS reconstruction-based attack detectors that overcome those challenges and allow an attacker without detailed process knowledge (apart from access to training data) to insert generic backdoors at training time, which can easily be exploited in unpredictable attack scenarios later. To be more precise, the attacker's goal is to minimize the number of true alarms raised by the detector during a physical anomaly at runtime. To achieve this, the attacker manipulates the training data at training time, with arbitrary new samples that are resulting in a backdoored detector. The attacker aims not to affect the detector's performance (evaluated over the test datasets) to hide that the detector was manipulated.

Our research questions are as follows: *Q1: Can backdoor attacks be applied to anomaly detection scenarios, i.e., autoencoders trained over one-class (benign system operations) data? Q2: Are specific features (e.g., actuators, sensors) better suited to be used as triggers for backdoor attacks in ICS? Q3: How can the attacker efficiently identify suitable trigger sequences? Q4: How do the proposed attacks compare with prior work poisoning and evasion attacks?*

In this context, the research challenges are as follows. C1: As the autoencoder will learn the (normal) physical behavior of the process in training, it will be difficult to identify triggers that will both conceal the anomaly and still comply with the process model. Moreover, the inserted backdoor behavior requires not altering the detector's performance when the backdoor is not activated. C2: Creating training datasets without real-world testbeds is challenging, as the physical process introduces complex constraints on creating synthetic data. For example, in the physical process, each feature can have different characteristic ranges and distributions, unlike RGB pixel values for backdoored images or similar.

We summarize our main contributions as follows:

- We are the first to apply the concept of backdoor attacks against autoencoders used for attack detection in ICS.
- We propose five variations of backdoor attacks with differences in the training manipulation and constraints on the attacker, and systematically analyze them.
- We experimentally demonstrate that our backdoor attacks generalize to unseen anomalies and that prior work does not (based on third-party datasets). We also investigate the impact of our attacks on the regular classification behavior for the original training dataset.

We publicly share the artifact (code and dataset) of our contribution. https://github.com/scy-phy/backdoorCPSS23

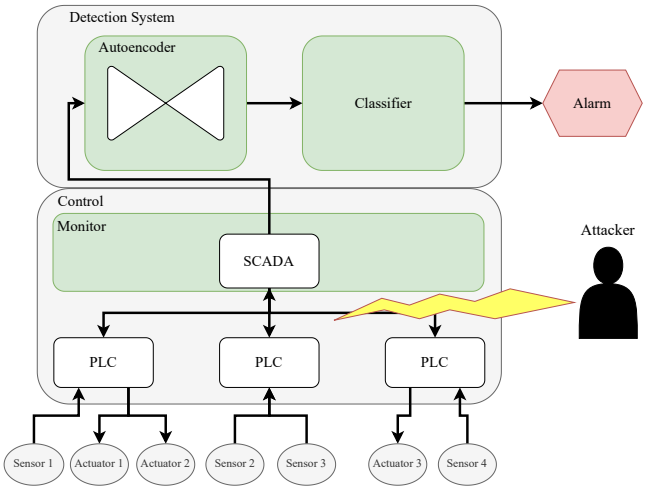

Figure 1: System and Attacker Model. Sensors and actuators are connected with PLCs and the PLCs are controlled and monitored by at least one SCADA. The output of the SCADA is used as input for the detection system to determine the state of the system.

2 BACKGROUND

This section explains all the concepts and systems used within this paper. This knowledge is required to understand the proof of concept as well as the discussion of the results. The first section explains industrial control systems in general. Afterward, we explain the composition of the used dataset (BATADAL) [33]. Following this, we introduce the autoencoder that is trained on the BATADAL dataset and is used in the ICS setting. Finally, we explain what backdoor attacks in machine learning are.

2.1 General ICS Architecture

This work considers an Industrial Control System (ICS) composed of a Supervisory control and data acquisition (SCADA) system, a set of Programmable Logic Controllers (PLCs), sensors, and actuators. The SCADA monitors and controls the general state of the system, while PLCs 'read' the system's current state (physical world) using the sensors, and based on the control logic, they update actuator signals. The SCADA collects information from the PLCs to keep the system state updated. The detection system pulls data from the SCADA and searches for anomalies. Figure 1 shows a general diagram of an ICS system with a detection system.

2.2 Reconstruction-based anomaly detection

Prior work in anomaly detection for ICS proposed reconstruction-based detection utilizing deep neural networks [12, 16, 22, 32]. Recently, commercial products based on the same principle were released [19]. This detection system consists of two parts: a deep learning autoencoder (AE) and a classifier.

An autoencoder consists of an input and output layer as well as several smaller hidden layers. The meaning of the input neurons correlates with the meaning of the output neurons which makes the AE self-supervised learning architecture. This also means that there

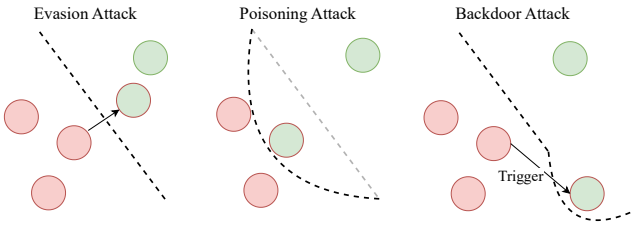

Figure 2: Backdoor Attacks And Comparison With Evasion and Poisoning Attacks. (causative vs. exploratory attacks)

are as many input neurons as output neurons [26]. The autoencoder is able to learn a system's behavior under normal operating conditions by minimizing the mean squared error loss (also referred to as reconstruction error and residuals) between the input and output layer of the network. If the reconstruction is similar to the input, the reconstruction error is low which means that the model properly learned the behavior/patterns of the input. Several autoencoder architectures were proposed in prior work, Long Short Term Memory (LSTM) [16], Fully connected [33] and Convolutional Neural Networks (CNN) [22].

The classifier determines whether the system is currently 'safe' or 'under-attack'. For this determination, the input and output of the autoencoder are compared. If the input of the autoencoder is anomalous it will produce a higher reconstruction error. In the case that this error then exceeds a threshold θ, the model classifies the current state of the system as 'under-attack'.

In this work, we target the anomaly detection system proposed in [32] (Open source project: [31]).

2.3 Backdoor Attacks

Backdoor attacks are a class of training time attacks that affect Machine Learning models [13]. They consist of two distinct phases: i) Training data manipulation, ii) Backdoor exploitation at run-time via on-the-fly sample manipulation to hide an ongoing anomaly in the process. The goal of this attack is to shift the decision boundary of a model in a strategic way [34]. By doing so, the attacker can introduce a hidden behavior into the model which is triggered (at run-time) for certain inputs and remains inactive for benign inputs (see Figure 2). The attacker accomplishes this by adding or inserting maliciously labeled or crafted samples into the training data and using it to train the model.

As an example, imagine an authentication system that learned to recognize the faces of people who are granted access to a protected area. The attacker can now introduce a hidden behavior to the model so that all people who wear a red pair of glasses are granted access to the protected area as well. Therefore, the attacker must introduce images into the training dataset of arbitrary people who all wear the same pair of red glasses but are all labeled as a person that has access to the protected area. After this, the model must be trained on these poisoned samples. If the attack was successful, all people who wear the same or a very similar pair of red glasses are granted access to the protected area and therefore, bypass the authentication mechanism [7].

2.4 Comparison with Evasion and Poisoning Attacks

Compared to backdoor attacks, in evasion attacks (Figure 2) [4], the attacker aims to create malicious input that will be misclassified by the target model. This attack does not directly affect the parameters of the target model. For example, in Figure 2 (left side), an attacker could manipulate a red dot so that the model classifies it as a green dot.

In contrast, poisoning attacks occur by inserting a specific behavior into the target model causing the model to misclassify specific patterns[21]. This is achieved by manipulating the training data to force the resulting model to embed the poisoned samples onto the wrong side of the decision boundary. This causes the boundary to shift and might impact the model's accuracy. In contrast, in backdoor attacks (which also require malicious training data), a dedicated trigger must be inserted into the training data. This trigger is learned and can be recognized when testing the model to cause induce a desired behavior (i.e. bypass attack detection in our case).

The timing of the attacks is also different. Backdoor and poisoning attacks occur during training and are exploited during testing, while evasion attacks only occur during testing.

In summary, while backdoor, evasion, and poisoning attacks all target machine learning models, their goals, methods, and stages of execution differ.

2.5 The BATADAL Dataset

The BATtle of the Attack Detection ALgorithms (BATADAL) [33] is based on a real-world medium-sized water distribution network with 429 pipes, 388 junctions, 7 storage tanks, 11 pumps, and 5 valves. One training set and two test sets compose the BATADAL dataset. Each test dataset contains seven attacks.

The datasets contain a time series of sensor and actuator readings from 43 sources. Each row represents the state of the water distribution system at a given point in time. In general, the different sensor readings contain measurements for water quality, pressure and flow. These readings are used to detect pipe bursts or contamination and SCADA systems can use them to control pumps and actuators. The training dataset has 48107 rows of benign data (do not contain any malicious tampering). The test datasets have 18433 and 10081 rows. 16465 rows of the first test dataset are benign and 1968 are labeled as 'under-attack'. In the second test dataset, 8453 rows are benign and 1628 are labeled as 'under-attack'. The malicious rows contain attacks on sensor readings and water tank thresholds and pumps. By manipulating tank levels and thresholds, the injected perturbation makes the system misbehave, causing tanks to overflow or empty the tank, which might cause physical damage to system components. Other attacks affect the components' performance like altering the working speed of pumps resulting in undesired water levels.

3 RESEARCH GOALS AND ASSUMPTIONS

In this section, we discuss the goals of our research. We investigate backdoor attacks against a state-of-the-art ICS anomaly detection system from an attacker's perspective. We now summarize our assumptions on the attacker and system model.

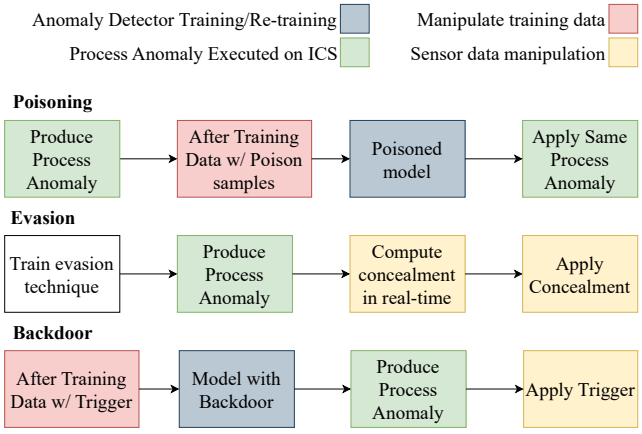

Figure 3: Overview of attack pipelines for ICS attack concealment. The image shows a comparison of the pipelines involved during Evasion, Poisoning, and Backdoor attacks.

3.1 Motivation

Limitations of prior poisoning attacks. Related work in the field proposed many attacks against reconstruction-based detectors. In particular, Kravchik *et al.* [21] examine poisoning attacks against such models. As depicted in Figure 3, to enable model poisoning, the attacker is assumed to precisely know (a priori) the sensor values observed during an attack and use this knowledge to poison the detection model during training. Model poisoning will allow the attacker to launch the same attack on the system later and be sure that the attack will remain undetected. We believe that such prior knowledge of attack data is challenging to gather. We assume the attacker can perform one of the following actions to collect such information i) start the attack on the target system before the detector is poisoned, which will result in the attacker being detected, ii) have a precise process simulator (digital twin) to simulate the attack and gather sensor reading. We believe the first option to be counterproductive because it will undermine the whole concealment objective. Regarding the second option, it is often assumed to be challenging to gather information required to reverse an ICS [17, 20, 29] and building a precise and effective digital twin of an industrial process is an open research challenge [10].

Limitations of prior evasion attacks. On the other side, evasion attacks on reconstruction-based detectors were studied by Erba *et al.* [11]. As shown in Figure 3, the attacker is assumed to use an adversarial machine learning technique to spoof the sensor readings in real time and avoid detection. This technique is resource intensive for the attacker, as it has to dynamically compute spoofing patterns towards the detector. This might not be ideal in a real-time resource-constrained setting such as an ICS.

Backdoor attacks. Given the limitations of prior work approaches, in this work we explore backdoor attacks. Figure 3 shows the attack pipeline for such an attack. Compared to the previous two attacks, this attack is promising because it overcomes the main limitations of prior approaches. Compared to poisoning attacks, backdoor attacks enable generic attack concealment. Compared to the evasion attack, backdoor attacks do not require real-time computation of the adversarial examples. In our work, we focus on the three research questions outlined in the introduction while tackling the aforementioned research challenges.

3.2 System and Attacker Model

We assume an Industrial Control System protected by a reconstruction based anomaly detection system (see Section 2.2). The detector is trained on benign (normal operational) data samples only. An attacker with access to the physical plant aims to attack the system to cause anomalous states to destabilize or disrupt the physical process. These anomalous states (without further manipulation by the attacker) will alert the detection system. For this reason, the attacker aims to hide (conceal) the anomaly by launching a backdoor attack on the system.

Success Metric. The anomaly detector's performance can be measured in terms of Accuracy (Eq. 1), Precision (Eq. 2), Recall (Eq. 3) and F1 score (Eq. 4). To calculate these metrics, we need the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) of the predicted test dataset samples. A TP occurs when a positive instance is correctly detected as positive. Conversely, a TN occurs when a negative instance is correctly classified as negative. A FP occurs when a negative instance is erroneously classified as a positive instance, and a FN when a positive instance is classified as negative. The reduction of the *Recall score*[1] (Eq. 3) metric measures the adversary's success for a backdoor attack. From a detector's point of view, we try to turn true positives into false negatives.

In this work, we focus on the backdoor attack task, the creation of the anomaly is assumed to be possible for the attacker as past real-world incidents demonstrated (e.g, Stuxnet [35]).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

Backdoor injection. To insert the backdoor, we assume that the attacker can manipulate the training process of the detector, i.e., the attacker can insert additional arbitrary data samples into the training set (which is supposed to contain normal operational data only). The attacker can be an insider threat (as in the Maroochy cyber attack [1]) that inserts the backdoor in the training data before they are used to train the anomaly detection system, or the attacker can be a third party that trains the model (e.g., supply chain attack as in the Stuxnet case [35]), as they have full control on the resulting detection model.

Attacker capabilities. The attacker's goal is to conceal the anomalous physical process state from detection. In addition to the training manipulation (that occurs before the anomaly) and in order to activate the backdoor, the attacker manipulates a constrained subset

[1]Recall and Accuracy are equivalent when considering attack datasets where only anomalous sensor readings are contained

of data reported from the SCADA to the detector. In particular, we assume that the attacker can manipulate the reported sensor values and reported actuator states. We do not consider full replay attacks in this work [25], as they require a continuous manipulation of all sensor and actuator data, which can be expected to be the worst-case cost for the attacker.

4 DESIGNING BACKDOOR ATTACKS

In this section, we present the methodology to design effective attack triggers to activate the backdoor in our target models. Then, we present the four actuator-based attack variations that we evaluated in this manuscript.

4.1 Choosing Suitable Triggers

If an attacker wants to conduct a successful backdoor attack, the main challenge is to find a suitable and recognizable backdoor trigger for the model. This is a challenging task because the attacker has to choose all features in which the trigger is inserted and also the value to set for each trigger feature. Value selection is non-trivial because we want the resulting poisoned model to perform as the non-poisoned model when the backdoor is not activated. Therefore, to investigate *Q2* we must explore ways how to design such a trigger. In particular, we test two kinds of triggers.

Sensor boundaries based trigger. In our first approach[2], we check the lower and upper bounds for all sensor readings in our dataset. Therefore, we know exactly which value ranges occur during normal operating conditions. This helps us to define triggers that are not inordinately far away from the original values and therefore, do not cause overly high reconstruction errors. This is important to ensure that we do not confuse the autoencoder during the training and avoid undermining its performance results. In the next step, we choose random features to insert a trigger. For this, we try various features and feature combinations.

Actuator states-based trigger. For our second approach of inserting a backdoor trigger, we consider the features that indicate actuator states. These features are represented by integer binary values.

Therefore, our distribution range is only $\{0, 1\}$. To not cause any false positives, we must create a trigger that consists of state sequences that do not occur under normal operating conditions. Consequently, we must determine all occurring binary sequences in the training data. After this, we can generate all binary combinations that were not found in the training dataset and use them as our backdoor trigger.

4.2 Designing Attack Strategies

To investigate *Q3*, we will now introduce four variations of our actuator-based backdoor attack. In general, in each of these attacks we, i) select a number of samples to be manipulated (from either the training set or a test set), ii) we then compute and insert the trigger in those samples, iii) and then these triggers are either appended to the training set or replace their original versions.

4.2.1 Standard Backdoor Attack (SBA). This approach is depicted in Figure 4. To create the poisoned training dataset, we copy n rows from the original training dataset (containing benign data), insert our backdoor trigger, and append the rows to it. Hence the resulting poisoned training dataset will contain n more samples than the original dataset. We do this for different percentages of data (different n sizes) that we add to the original training dataset.

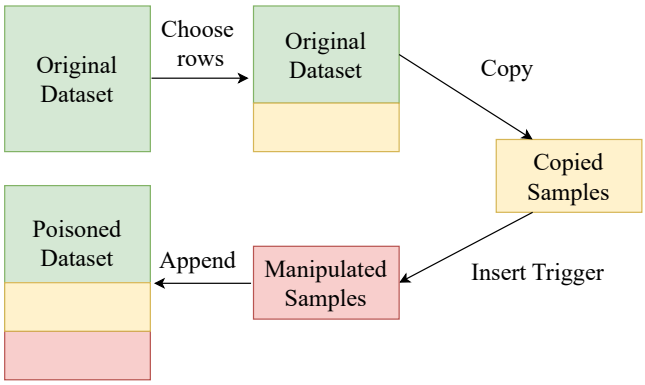

Figure 4: Standard Backdoor Attack (SBA)

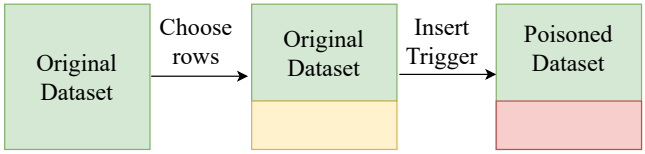

Figure 5: Improved Standard Backdoor Attack (ISBA)

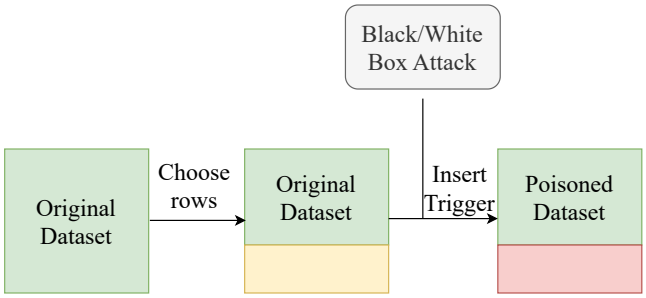

Figure 6: Combined Backdoor Attack (ComBA)

4.2.2 Improved Standard Backdoor Attack (ISBA). This approach (Figure 5) is a variation of the Standard Backdoor Attack. In this case, we do not add more samples to the training data but insert the trigger on a subset of the training data. Therefore, the poisoned dataset has the same amount of training samples as before. The rationale behind this approach is to avoid the autoencoder to see the same data samples more than once during training. Analyzing the same sensor and actuator values with different actuator states can reduce the performance of the model, when no trigger is active, which we want to avoid.

4.2.3 Combined Backdoor Attack (ComBA). This method (Figure 6) follows the same principle as the Improved Standard Backdoor

[2]This does not equal the Standard Backdoor Attack

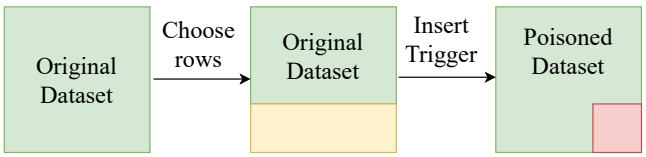

Figure 7: Constrained Backdoor Attack (ConBA)

Attack but we further optimize our trigger by deriving patterns from prior work concealment attacks [11].

The first attack is a learning-based black box attack. In this scenario, the attacker has no insights into the detection mechanism and only knows that the model is based on a reconstruction approach. We use this attack to find the binary sequence that yields the lowest recall for our trigger.

The second attack is an iterative white box attack. In this case, the attacker knows how the detection process is performed and what parameters the model has. With this knowledge, the attacker can use the model as an *oracle* and therefore query it with arbitrary inputs while observing the consequences of it. As with the black box attack, we use this method to find the trigger which yields the lowest recall.

4.2.4 Constrained Backdoor Attack (ConBA). The last approach (Figure 7) is a constrained approach. Also, this approach is based on the ISBA approach, but the attacker is assumed to only be able to control one PLC and therefore, the attacker can insert the trigger on a limited set of actuators. In this scenario, the attacker is thus bound to a certain physical area of the ICS.

5 IMPLEMENTATION AND EXPERIMENTAL SETUP

In this section, we explain the details of our evaluation setup. We start by introducing the implementation details, followed by the details of the evaluation setup.

To implement the attack, we use Python3. The main libraries we use are Tensorflow (2.2.0) and Keras (2.3.1) for the autoencoder, NumPy (1.19.1) to read the datasets from the CSV format, Pandas (1.2.2) to manipulate the datasets (data frames) and Scikit-learn (0.24.1) for training and testing the autoencoder. To plot the results that can be seen in Section 6, we use matplotlib. Additionally, we present our evaluation in jupyter notebooks.

5.1 Percentage of Training Set Poisoned

For each of the following attacks, we poisoned 2%, 5%, 10%, or 20% of samples with the backdoor trigger in the training dataset of an autoencoder and therefore created four differently trained autoencoder versions to test each version of the attack. With this, we can analyze the optimal percentage needed for the autoencoder to properly recognize the backdoor trigger and also not cause an overly high reconstruction error.

To inject and add these triggers to the training dataset, we copy rows of the existing training dataset, added the trigger, and append these "new" rows to the training dataset. For the last three attack implementations, we insert the trigger directly into the original training dataset. This ensures that a backdoored autoencoder behaves very similarly as if tested on benign inputs compared to the original autoencoder because they have seen almost the same data during training.

5.2 Attack Implementation

5.2.1 Standard Backdoor Attack (SBA). In this approach, we copy rows from the training dataset and insert a random[3] trigger. The new training dataset therefore only contains added rows from itself and all rows have the same trigger implemented. In general, we do not exactly specify which rows are copied in the test dataset. However, we do specify the exact amount of rows that we copy (2%, 5%, 10%, and 20%).

5.2.2 Improved Standard Backdoor Attack (ISBA). As mentioned before, in this approach, we do not copy any rows to create a backdoored training dataset. We choose 2%, 5%, 10%, or 20% of the rows in the original dataset randomly and insert the trigger in there. Consequently, we do not expand the training dataset but rather poison a specific percentage of the existing samples. This brings the opportunity that the autoencoder only sees unique training samples and does not get confused when it sees the same sensor values in two rows that differ in the actuator states. There is the possibility that the autoencoder learns a pattern from one row which is then invalidated by the other row (because of the different actuator states) and therefore, the autoencoder does not learn anything from this. With this new approach, the autoencoder should learn the pattern more easily.

5.2.3 Combined Backdoor Attack (ComBA). For this backdoor attack, we first need to conduct the black box and or white box attack mentioned in [11]. Therefore, we must specify the features which we want to use for the trigger in a text file. This is then given as input to the respective attack. The black box attack can only define the input to the attacked model and observe the output. Therefore, all the optimization of the input relies on the respective reconstruction (output).

In the white box approach, the attacker can analyze the parameters and thresholds of the model. With this knowledge, the attacker can learn how the model behaves for benign input, i.e. a system state which is not under attack. This can then be used as an oracle because the attacker can choose a new input and observe how the parameters and thresholds of the model change depending on the input. This makes it possible to find an optimized new input that causes a low reconstruction error because the malicious input behaves similarly to the benign input.

5.2.4 Constrained Backdoor Attack (ConBA). Three PLCs in the water distribution system maintain actuator states. PLC 1 gives the attacker access to the actuator states *STATUS_PU1*, *STATUS_PU2*, and *STATUS_PU3*. Assuming that the attacker can only access PLC 1 they can only introduce a three-digit binary sequence as the trigger. PLC 3 gives the attacker access to the actuator states *STATUS_PU5*, *STATUS_PU6*, *STATUS_PU7* and *STATUS_V2*. This allows the attacker to insert a four-digit sequence. The same holds for the last PLC (PLC 5) but in this case, the attacker can control the

[3]Random from the list of possible binary sequences that do not occur during normal operating conditions

actuator states *STATUS_PU8*, *STATUS_PU9*, *STATUS_PU10* and *STATUS_PU11*.

5.3 New Test Dataset

To evaluate and compare the different attack versions we created our evaluation test dataset. This dataset contains 1000 attack samples (i.e., ground truth 'under-attack') from the original test dataset 1 (from the BATADAL dataset collection). We use this dataset to evaluate the efficacy of our backdoor attacks. Especially, with this new test dataset we can evaluate whether a backdoored autoencoder can recognize the backdoor trigger or if a row is still detected as an attack. Since all rows in this dataset are labeled as 'under-attack' but also contain the backdoor trigger, the backdoor in the autoencoder should not recognize an attack and therefore, the lower the accuracy (and recall score) on the test dataset the better the attack works.

For testing the benign autoencoder on our new test dataset, we do not include the triggers.

6 EVALUATION

In this section, we present the results of our analysis using different triggers and provide details on how we evaluate the attack's success in general and the results of the different approaches. We also discuss each approach's positive and negative aspects and summarize the results.

Moreover, we compare the results of the original autoencoder with the backdoored autoencoders from each attack version on the 14 attack datasets to evaluate their concealment.

6.1 Performance on Original Test Data

A backdoor attack can only be considered successful if the accuracy of a backdoored model remains similar enough to the original model when testing both on benign test data. Therefore, we test all backdoored autoencoders from our different attack scenarios on the original test datasets of the BATADAL dataset collection. In Table 1, we can see the decrease in the performance of the detector for our backdoored autoencoders. The overall accuracy decreases < 2% for all the actuator states-based trigger backdoored autoencoders. Regarding the F1 score, we notice a reduction of around 10%, while the precision remains almost constant and recall reduces at most by 13%. Overall, we can observe that the backdoor injection does not dramatically affect the detector performance in both sensor-based trigger attacks and the actuator states-based trigger.

6.2 Backdoor Trigger

We start evaluating the *Sensor boundaries based trigger* and the *Actuator states based trigger* strategies to understand which of the two provides better performance. This will allow us to answer to *Q2*.

Sensor boundaries based trigger Standard Backdoor Attack. By inserting the backdoor trigger into the sensor values (Sensor-based backdoor attack, SenBA), we can observe the detector recall decreases in all the considered setups (AE 2 to 20). The best recall drops from 0.99 to 0.718. The result can be seen in Table 2 and in Figure 8.

Standard Backdoor Attack. For this approach, the backdoored autoencoder is trained on 5% of additional backdoor samples achieves the best performance, and recognizes the backdoors in up to 38.9% of the cases. Therefore, it achieves a recall of 61% on our test dataset. In Table 2, we can see the results for this attack and all backdoored autoencoder versions. Autoencoder "BAE" describes the benign autoencoder that was tested on the benign test dataset 1 and is used to illustrate the detector's performance difference before and after the attack.

We decide to reduce the search space by avoiding the use of continuous values, like sensor readings, for the trigger and focusing on the use of state variables for the actuators only. To be more specific, we choose to insert the trigger into all (i.e. twelve) actuator states of the system. These actuator states are defined by either 0 or 1, so our new trigger will be a twelve-digit binary sequence. To find a suitable sequence, we choose all sequences that do not occur during normal operating conditions. This is important because otherwise,

Table 1: Results of backdoored autoencoders on the original test data. "BAE" describes the benign autoencoder that was tested on the test dataset. "SenBA" describes the best-performing backdoored autoencoder from the Sensor-based Attack tested on the test dataset. (The others are analog for the other attacks) Acc: Accuracy, F1: F1_Score, Prec: Precision, Rec: Recall

	Detection Performance			
Autoencoder	Acc	F1	Prec	Rec
BAE	0.936	0.714	0.896	0.597
SenBA	0.93	0.678	0.892	0.552
SBA	0.918	0.6	0.898	0.46
ISBA	0.92	0.608	0.904	0.467
ComBA	0.919	0.607	0.905	0.465
ConBA	0.92	0.615	0.89	0.480

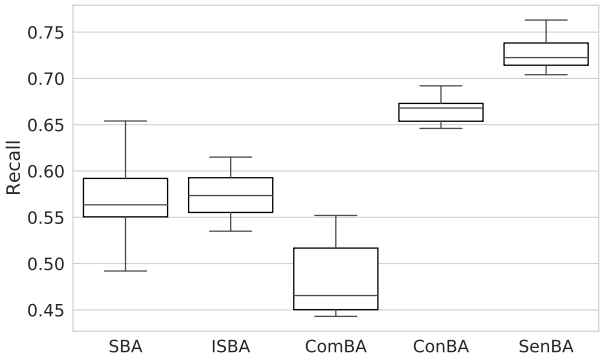

Figure 8: Confidence intervals for all attacks. All attack versions are tested on ten different seeds (our own test dataset). For the standard attack, the recall is between 0.492 and 0.654. For the improved attack between 0.538 and 0.615, for the combined attack between 0.443 and 0.552, for the constrained attack between 0.646 and 0.692, and for the sensor-based attack between 0.704 and 0.763.

Table 2: Results of the Backdoor Attack (Recall). "BAE" describes the benign autoencoder tested on the malicious test dataset (containing 2000 rows with attacks). "AE X" describes a backdoored autoencoder that was trained on X% of backdoored samples.

	Detection Recall				
Autoencoder	SenBA	SBA	ISBA	ComBA	ConBA
BAE	0.999	0.999	0.999	0.999	0.999
AE 2	0.817	0.627	0.747	0.455	0.774
AE 5	0.770	0.610	0.571	0.303	0.745
AE 10	0.730	0.703	0.495	0.364	0.749
AE 20	0.718	0.712	0.714	0.368	0.721

the autoencoder might not be able to distinguish between backdoored input samples and benign inputs reliably. The consequence of this would be more false positives for benign and backdoored inputs.

In total, we found 4008 distinct sequences that do not occur in the training dataset. By using this trigger, the accuracy of the autoencoder does not drop as much as before which shows us that this is a valid option for a usable trigger. Also, by tampering with the actuator states only, we do not need to overwrite any sensor values and therefore can potentially manipulate all of them to conduct an attack on the system.

In the following, we explain the different versions of how we backdoored the autoencoder as well as a method to automatically find a trigger that can be recognized very well by the model.

6.3 Attack Evaluation

In this section, we apply the remaining 4 (the standard backdoor attack results are presented in the previous section) identified attack variations with Actuator states-based triggers, and we answer to *Q1, Q3.* Table 2 summarizes the results of the experiments.

Improved Standard Backdoor Attack. In this setting, we achieve a recall of 49.5%. The results of ISBA show that the autoencoder can better recognize the pattern if it is inserted into the benign training data instead of adding more samples to the training set as in the SBA approach (see Figure 4 and Figure 5).

Combined Backdoor Attack. We find 30 unique patterns that do not occur in the training dataset with the black box attack and 21 with the white box attack. The pattern [1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1] makes it possible to achieve the best recall with 30.3% for the 5% model. This pattern was found by using the black box attack mentioned in 4.2.3. However, this result is not consistent and highly depends on the random seed used for sampling the data used to insert the backdoor. In this scenario, for each of the four experiments, we repeated the training of the autoencoders with 10 different seeds setting and observed that the recall fluctuates between 44.3% and 55.2%. The results for all attacks can be seen in the form of confidence intervals in Figure 8.

When comparing the results of the patterns found by the black box attack with those found by the white box attack, we cannot observe a meaningful difference. The average recall for the 30 patterns found by the black box attack is 38% on our own test dataset, and the average recall for the 21 patterns of the white box attack is 36%.

Constrained Backdoor Attack. In this setting, we assume that the attacker is constrained to manipulate exclusively the actuators that are connected to a specific PLC in the network. To test our approach, we select the PLCs that control actuators in the network, specifically we test on PLC 1, PLC3, and PLC5. For PLC 1, the attacker can control three actuator states. Thus, the pattern that can be injected into the training dataset is three digits in size. Furthermore, only one combination of these features does not occur during normal operating conditions. The only possible trigger is [1, 1, 1] and achieves a recall of 66.6% for the 10% model.

For PLC 3, the attacker controls four actuators which can be combined in ten alien actuator state combinations. Out of these ten, the pattern [1,0,0,0] achieves the best recall with 66.3% for the 20% model. On average, this pattern also achieves the best recall for all percentages that the models are trained on. The 20% model is the best model on average overall patterns. However, it has a 2.4% better recall than the 5% model and a 2.8% better recall than the 10% model. Therefore, tests on more seeds should be conducted in future work (more on that in the conclusion section). In Table 2 we see the results for PLC 3 and the best pattern [1,0,0,0].

Additionally, we observe that the 2% model also achieves the highest threshold θ with 0.01 in this scenario which is more than double the amount for the other models.

For the last PLC 5, the attacker has twelve combinations with a length of four to choose from. The pattern [1,1,0,0] achieves the best recall with 72.3%. The best average recall over all percentages is achieved by the pattern [0,1,0,1], and the best model average yields the 5% model with similar small accuracies as for PLC 3.

Concealment. In this part, we show the results of the backdoored autoencoders on the 14 anomalies of the BATADAL dataset to demonstrate the overall concealment performance of our method. The results for the original autoencoder are evaluated on the benign attack datasets. The results for the backdoored autoencoder are evaluated on a backdoored version of the attack datasets. This means that we poisoned the attack datasets by inserting a backdoor trigger in all existing rows depending on the attack version. Therefore, the lower the recall of the attacks the better the outcome for the attacker. In general, Table 9 shows the best accuracies that we observed in our evaluation.

In Table 9, we can see that the Combined Backdoor Attack achieves the best concealment of the attack on the water distribution system with an average recall of 19%. This means that an attack was detected by the backdoored autoencoder in only 19% of the cases although all samples in the datasets are labeled as 'under-attack.' This means that we achieve a concealment of 39% if we compare it to the benign autoencoder (on average). An important finding can be seen for attack datasets B10 and B11. On these datasets, the backdoored autoencoder was not able to detect the process anomalies anymore and therefore achieves a recall of 0%.

The Standard, Improved Standard, and Constrained Backdoor Attack show very similar results on the datasets and only differ in 3%, 4%, and 7% on their average recall. All three achieve a lower recall than in the original setting which means that the backdoor attack is successful for some attacks.

	Detection Recall				
Attack	Orig	SBA	ISBA	ComBA	ConBA
B1	1.0	0.81 ◓	0.92 ◓	0.07 ●	0.99 ◔
B2	0.93	0.69 ◓	0.84 ◓	0.06 ●	0.90 ◔
B3	0.15	0.12 ◔	0.12 ◓	0.11 ●	0.18 ◓
B4	0.18	0.15 ◓	0.15 ◓	0.13 ●	0.22 ◓
B5	0.28	0.15 ◔	0.14 ◓	0.12 ◓	0.15 ◔
B6	0.89	0.89 ◔	0.88 ◓	0.84 ●	0.89 ◐
B7	0.89	0.89 ◔	0.89 ◓	0.83 ●	0.89 ◐
B8	0.68	0.43 ◓	0.43 ◓	0.43 ●	0.51 ◓
B9	0.24	0.02 ◓	0.05 ◓	0.01 ●	0.10 ◓
B10	0.34	0.17 ◓	0.17 ◐	0.00 ◓	0.18 ●
B11	0.32	0.21 ●	0.22 ◐	0.00 ◓	0.12 ◓
B12	0.34	0.05 ◓	0.09 ◓	0.01 ●	0.17 ◓
B13	0.99	0.79 ◓	0.92 ◓	0.05 ●	0.99 ◔
B14	1.0	0.98 ◔	0.98 ◓	0.13 ●	0.98 ◔
Mean	0.58	0.45	0.48	0.19	0.52

Figure 9: Concealment overview. Symbols indicate number of added backdoor samples in training ◔: 2%, ◓: 5%, ◐: 10%, ●: 20%. All actuator values in the attack samples were replaced with trigger sequences. The Combined Backdoor Attack (ComBA) is performing best, reducing the average recall of the detector from 0.58 to 0.19. Orig: Original classification (no triggers/backdoored AE), SBA: Standard Backdoor Attack (4.2.1), ISBA: Improved Standard Backdoor Attack (4.2.2), ComBA: Combined Backdoor Attack (4.2.3), ConBA: Constrained Backdoor Attack (4.2.4)

6.4 Summary of Findings

So far in this work, we have investigated three research questions. Our evaluation results show that regarding *Q1*, it is possible to apply backdoor attacks to reconstruction-based anomaly detectors for ICS. We proposed two different trigger methodologies, and regarding *Q2*, we have identified that both the proposed trigger strategies (sensors or actuators) are effective, but actuator features acheive a better concealment. With this knowledge about the triggers, to answer *Q3*, we proposed five backdoor insertion strategies. The first element we checked to verify the feasibility of a backdoor attack is the accuracy of the backdoored autoencoder on benign data. The highest attack success is achieved with the Combined Backdoor Attack. While this approach is also the most computationally expensive, the recall is 52.2% on average. The second best approach is the Improved Standard Backdoor Attack (in Section 6.3) with an average recall of 63.2%. Therefore, the best approach is almost 11% better compared to the second-best approach.

In Figure 8, we see the confidence intervals for the range of recall that we observed when testing all attack approaches on ten different seeds. Compared to the Standard Backdoor Attack, all other attacks have a smaller range where the recall changes by about 4-10% at maximum. However, for the Standard Backdoor Attack, we can observe a fluctuation of the recall by almost 15%.

Table 3: Computational effort of concealment and poisoning attacks from prior work ICS anomaly detection

Name	Att. Prep.		Att. Train		Time/tuple	
	$\mu_{\bar{x}}$[s]	$\sigma_{\bar{x}}$	$\mu_{\bar{x}}$[s]	$\sigma_{\bar{x}}$	$\mu_{\bar{x}}$[s/tuple]	$\sigma_{\bar{x}}$
Iterative [11]	-	-	-	-	2.28	2.46
Learn.-based [11]	-	-	14.35	0.89	0.002	0.005
Poisoning [21]	300.36	83.06	-	-	-	-
Our work (ComBA)	128.92	6.30	-	-	-	-

6.5 Computational Effort

To further explore *Q4*, we compare the computational effort required by our proposed backdoor attack, w.r.t. prior work ICS evasion and poisoning attacks (Table 3). We divide the analysis in three criteria. Attack preparation refers to the time required to attack the target model, for training time attacks (poisoning and backdoor). Attack training measures the time required by the attacks to train adversarial based concealment techniques. Finally, runtime per tuple measures how long the attack elapses to find the adversarial perturbation.

The first attack we compare with is the iterative concealment attack [11], in this case, the attack only requires time per tuple. This is a time that needs to be multiplied by the number of samples that the attacker manipulates. By just concealing around 64 attack samples (i.e., (128.92/2.28)), the required time surpasses the time required for attack preparation by the proposed ComBA approach (We note that the BATADAL dataset contains more than 10k anomalous samples).

The second attack we compare with is the learning-based concealment attack [11], in this case, the attack requires attack training time and runtime per tuple. Also, in this case, the runtime needs to be multiplied by the number of samples that the attacker manipulates. In this case, the benefit of adopting the proposed ComBA approach in terms of computational time is reached after around 57000 attack samples (i.e., $(128.92 - 14.35)/0.002$).

We can directly compare the computational effort against the poisoning attack [21] because, in both cases, we only consider the attack preparation. Our combined backdoor attack is more than twice faster. Additionally, our attack runs more stable, while the poisoning attack shows a standard deviation of 83 seconds approx.

6.6 Comparison with baseline poisoning attacks

To answer *Q4*, we compare our proposed backdoor attack to relevant prior work poisoning attacks on Cyber-Physical System detectors [21]. The environment of this work is similar to ours and considers a water distribution system under attack. In Section 3.1, we discussed the limitations of this approach regarding the a priori knowledge of the attacker of sensor data occurring during an anomaly that they want to conceal.

To demonstrate that attacks in prior work do not generalize to unseen anomalies, we evaluated the poisoning attacks proposed by Kravchik *et al.* [21] against unseen anomalies. The code of the paper is available on GitHub. We tried to reproduce their results using the BATADAL dataset, but unfortunately, we were not able to achieve comparable performance. We believe that is due to the

fact they had to produce artificially the attack samples used for model poisoning, instead of directly applying the poisoning using testbed data. For this reason, we use their own evaluation setup (i.e., models and data) and test the transferability of the attack to unseen anomalies.

We trained the model proposed in the aforementioned paper on the same attacks considered in the paper (artificially generated starting from attacks 3, 7, 16, 31, 32, 33, and 36 of the SWaT dataset [15]). The dataset was gathered from the secure water treatment (SWaT) testbed at the Singapore University of Technology and Design. A water treatment facility that uses a six-stage water purification procedure serves as the testbed. A PLC linked to sensors and actuators controls each stage of the process. The sensors measure the water levels in tanks and contain conductivity analyzers and flow meters. The actuators are used to pump water and dose chemicals. 51 features that capture each state of the sensors and actuators (every second) are contained in the dataset. These features were recorded for seven days in benign operating conditions and for four days where the system was under attack.

To investigate the transferability of the attacks, we tested each poisoned model against all the other attacks, e.g. we trained a model on attack 3 and tested it on the rest of the attacks 7, 16, 31, 32, 33, and 36. Figure 10, reports the results of our experiment, which shows that attacks either transfer entirely or do not at all. The numbers in the confusion matrix cells indicate the percentage of poisoned data values. For all cells with the number 100 the anomaly detector detects all the input as anomalous. Therefore, we find 26 attacks that transfer and 16 attacks that do not transfer. Attacks where the tank level is increased above a high or very high (definitions from the referenced paper) level fully transfer, i.e. attacks 7, 32 and 33, while attacks with slowly increased or very low tank levels transfer less likely.

In conclusion, this approach does not provide strong guarantees regarding transferability over unseen anomalies. Conversely, our attack does not require any specific a priori knowledge about an anomaly can conceal it. Additionally, we only use the real data coming from our datasets and do not need to further manipulate them artificially to demonstrate our attacks.

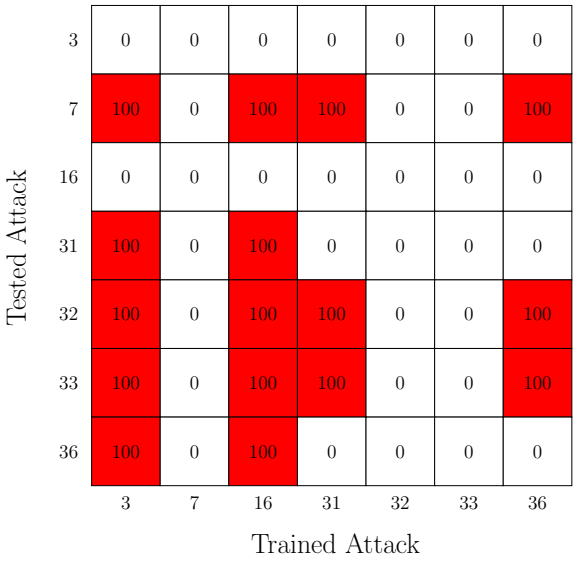

Figure 10: Confusion matrix. Numbers indicate the percentage of alerts raised (lower = better for the attacker). For the red boxes, there are at least as many alerts as the number of values contained in each attack. Therefore, attacks either transfer or do not transfer at all.

7 RELATED WORK

7.1 Prior work on Autoencoders and Backdoor Attacks

Autoencoders have been extensively studied in the literature; Bank *et al.* [3] present an overview of the most common autoencoder variants and their applications. In the ICS security field, the previous work [2] has developed an attack detection algorithm using stacked autoencoders. This algorithm can achieve an accuracy of up to 99.6% on a real-world dataset.

After their introduction [18], backdoor attacks captured the interest of the image recognition community. Chen *et al.* [7] proposed three variations of backdoor attacks. One of these aims to circumvent a face recognition system by adding a pair of glasses as the trigger to the training images. In general, the proposed attacks differ by either adding a human-recognizable item to the image, e.g. glasses, instead of some noise that is almost unrecognizable for humans. Backdoor attacks were also explored against video recognition models [36]. Similar to our work, video recognition models also deal with spatial and temporal correlations. The authors' main finding is the universal adversarial backdoor trigger. This trigger is not a static set of pixels that do not change during a video but a dynamic set of pixels that change their color values during the video. This attack can manipulate state-of-the-art video recognition models with a high success rate. Later, Salem *et al.* [28] described the dynamic backdoor attack concept. It allows the trigger to have multiple patterns and locations. This attack achieved almost perfect success and can also bypass state-of-the-art defense mechanisms against backdoor attacks.

Backdoor attacks have been implemented at the hardware level [9, 23]. In these works, the underlying hardware of neural networks is modified directly so that backdoor attacks are possible. Compared to our work, we do not need access to the underlying hardware but the training data only. Most recently, backdoor attacks have been studied on autoencoders [27]. In this work, the authors showed that a backdoored autoencoder using an image that contains a particular trigger generates a specific target image as output which belongs to a different class than the input. However, it only covers the domain of image classification with autoencoders.

7.2 Potential Countermeasures

Many countermeasures were proposed in related work to prevent and detect backdoor attacks on machine learning models. In this section, we discuss which countermeasures from prior work can be applied in our setting to mitigate the proposed attacks.

Blind backdoor removal. Liu *et al.* propose a blind backdoor removal technique called Fine Pruning [24]. In this method, the neurons of an ML model are pruned according to the least contributing neurons to a specific classification task. The main assumption is that certain neurons are less active during the main classification task compared to backdoor-infused classifications. Pruning is achieved by sorting the neurons according to their degree of activity and eliminating those with the least activity. Since this process decreases the model's accuracy, fine-tuning is required afterward. In our setting, applying this technique would be rather complex since all input and output neurons are mapped to a specific sensor or actuator reading of the water distribution system. Therefore, we can only prune the inner layers of the autoencoder. However, these layers try to reduce the complexity of the data progressively (in the encoder) and reconstruct them in the decoder. Hence, we assume that pruning these layers significantly limits the reliable reconstruction capability of the autoencoder.

Offline inspection. Models and trigger sequences can be inspected offline, i.e. in the offline machine-learning setting. One prominent case is the work from Chen *et al.* [6]. They propose an activation clustering approach that analyzes the network activation of a model on both benign and malicious (containing the backdoor trigger) inputs. To distinguish different types of neuron activation patterns, k-means clustering with $k = 2$ is used. Additionally, the activation for inputs of different classes is separated and only compared within the same class. To distinguish the benign from the poisoned data cluster for each label another metric like the silhouette score needs to be applied. The higher score of the two clusters then indicates the poisoned data samples. With this knowledge, one can remove these malicious samples from the training dataset and retrain the model on the benign data. This approach would be suitable in our setting to detect backdoor attacks, the activation clustering can be used to at least protect against backdoor attacks. For example, once the model is shared with the third party that trained it, it can be validated against backdoor attacks.

Online inspection. In online inspection, models and inputs are monitored during the run-time of a classification process. Chou *et al.* [8] propose a novel approach for the backdoor attack detection in images. In their method, they discover a set of contiguous pixels that are important for the classification result. This area is then most likely the region that will contain the backdoor trigger if present. After the discovery of this area, it is copied and transferred onto a new image where we know the ground-truth label. This new image is then classified by the model. If it predicts a new label, i.e. another label than the ground-truth label, and the confidence of this prediction is high enough, the previously discovered patch is considered a backdoor trigger. This also means that the input from which we extracted the set of contiguous pixels is malicious and contains a backdoor trigger.

Assuming that the countermeasure proposed by Chou *et al.* works in our setting, their algorithm can determine the attacker-controlled actuator states that function as our backdoor trigger. In our work, we have proven that adding our backdoor trigger to a malicious system state causes the hidden functionality of the backdoored autoencoder to be activated. Consequently, Chou's algorithm could detect the backdoor attack. On the other end, this would require installing an attack detector to detect targeted attacks to the ICS detector, increasing the complexity of the deployment and changing the confidence scores of the model, as the defense against the backdoor attacks might introduce false positives/negatives.

Prior works from Gao *et al.* [14] reviewed the countermeasures proposed against backdoor attacks and reported that all of the above mentioned countermeasures come with their own weaknesses and there is no effective defense against all types of backdoor attacks.

8 CONCLUSION

In this work, we investigate the impact of backdoor attacks against reconstruction-based anomaly detectors. We pose four research questions. To answer such questions, we proposed five versions of backdoor attacks for an autoencoder-based anomaly detection system and demonstrated that this type of attack is successful in the setting of industrial control systems.

For evaluating the attack, we use the 14 process anomalies from the BATADAL dataset collection and design a new test dataset that contains various types of attacks. Due to the high complexity and spatial and temporal correlations of the attacked system, an attacker has to cope with many constraints. These include finding a suitable trigger that conceals an anomaly and, at the same time, complies with the process model. The attacker has to create new training samples that introduce a hidden behavior into the attack detector.

We found that inserting backdoor triggers into the both sensor and actuator data to be effective, but focused on actuator states in the main evaluation as this reduced computational complexity (binary sequences). Specifically, we found that the proposed actuator states-based triggers are effective in inserting backdoors in target anomaly detection systems. Our best attack version (ComBA in 6.3) fully conceals some attacks and causes the detector's performance to drop by 47.7% on average. Since we have over 4000 possible actuator patterns (triggers) to choose from, ComBA enables a very efficient and automatic approach to finding a suitable trigger sequence.

Moreover, comparing to prior work, our attack universally transfers without knowledge about other attacks in advance or using synthetic data. Our proposed attack requires a constant overhead at training time and no computational effort at attack time. It makes our attack cope better with the real-time constraints posed by modern Cyber-Physical Systems.

Finally, we discussed possible countermeasures and how those can be applied to detect the proposed attacks. However, the detection of backdoor attacks remains an open research challenge.

REFERENCES

[1] Marshall Abrams and Joe Weiss. 2008. *Malicious control system cyber security attack case study-Maroochy water services, Australia.* Technical Report. MITRE CORP MCLEAN VA MCLEAN.

[2] Abdulrahman Al-Abassi, Hadis Karimipour, Ali Dehghantanha, and Reza M Parizi. 2020. An ensemble deep learning-based cyber-attack detection in industrial control system. *IEEE Access* 8 (2020), 83965–83973.

[3] Dor Bank, Noam Koenigstein, and Raja Giryes. 2020. Autoencoders. arXiv preprint arXiv:2003.05991.

[4] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion Attacks against Machine Learning at Test Time. In *Machine Learning and Knowledge Discovery in Databases*, Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 387–402.

[5] Alvaro A. Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. 2011. Attacks against Process Control Systems: Risk Assessment, Detection, and Response. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security* (Hong Kong, China) *(ASIACCS '11)*. Association for Computing Machinery, New York, NY, USA, 355–366. https://doi.org/10.1145/1966913.1966959

[6] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2019. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. In *SafeAI@ AAAI*.

[7] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526.

[8] E. Chou, F. Tramer, and G. Pellegrino. 2020. SentiNet: Detecting Localized Universal Attacks Against Deep Learning Systems. In *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE Computer Society, Los Alamitos, CA, USA, 48–54. https://doi.org/10.1109/SPW50608.2020.00025

[9] Joseph Clements and Yingjie Lao. 2018. Hardware trojan attacks on neural networks. arXiv preprint arXiv:1806.05768.

[10] Matthias Eckhart and Andreas Ekelhart. 2019. *Digital Twins for Cyber-Physical Systems Security: State of the Art and Outlook*. Springer International Publishing, Cham, 383–412. https://doi.org/10.1007/978-3-030-25312-7_14

[11] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. 2020. Constrained Concealment Attacks against Reconstruction-based Anomaly Detectors in Industrial Control Systems. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. Association for Computing Machinery, New York, NY, USA, 480–495. https://doi.org/10.1145/3427228.3427660

[12] Clement Fung, Shreya Srinarasi, Keane Lucas, Hay Bryan Phee, and Lujo Bauer. 2022. Perspectives From A Comprehensive Evaluation Of Reconstruction-Based Anomaly Detection In Industrial Control Systems. In *Computer Security – ESORICS 2022: 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26–30, 2022, Proceedings, Part III* (Copenhagen, Denmark). Springer-Verlag, Berlin, Heidelberg, 493–513. https://doi.org/10.1007/978-3-031-17143-7_24

[13] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. 2020. Backdoor attacks and countermeasures on deep learning: A comprehensive review. arXiv preprint arXiv:2007.10760.

[14] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. 2020. Backdoor attacks and countermeasures on deep learning: A comprehensive review. arXiv preprint arXiv:2007.10760.

[15] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2016. A dataset to support research in the design of secure water treatment systems. In *International conference on critical information infrastructures security*. Springer, Springer International Publishing, Cham, 88–99.

[16] 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)*. 140–145. https://doi.org/10.1109/HASE.2017.36

[17] Benjamin Green, Marina Krotofil, and Ali Abbasi. 2017. On the Significance of Process Comprehension for Conducting Targeted ICS Attacks. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy* (Dallas, Texas, USA) *(CPS '17)*. Association for Computing Machinery, New York, NY, USA, 57–67. https://doi.org/10.1145/3140241.3140254

[18] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733.

[19] Kaspersky. [n. d.]. *Kaspersky Machine Learning for Anomaly Detection*. https://mlad.kaspersky.com/, Last accessed on: 2022-10-24.

[20] A. Keliris and M. Maniatakos. 2019. ICSREF: A Framework for Automated Reverse Engineering of Industrial Control Systems Binaries. In *Network and Distributed System Security Symposium (NDSS)*.

[21] Moshe Kravchik, Battista Biggio, and Asaf Shabtai. 2021. Poisoning attacks on cyber attack detectors for industrial control systems. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. Association for Computing Machinery, New York, NY, USA, 116–125.

[22] Moshe Kravchik and Asaf Shabtai. 2018. Detecting Cyber Attacks in Industrial Control Systems Using Convolutional Neural Networks. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*. ACM, Association for Computing Machinery, New York, NY, USA, 72–83.

[23] Wenshuo Li, Jincheng Yu, Xuefei Ning, Pengjun Wang, Qi Wei, Yu Wang, and Huazhong Yang. 2018. Hu-fu: Hardware and software collaborative attack framework against neural networks. In *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, IEEE Computer Society, Los Alamitos, CA, USA, 482–487.

[24] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, Springer International Publishing, Cham, 273–294.

[25] Yilin Mo and Bruno Sinopoli. 2009. Secure control against replay attacks. In *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 911–918. https://doi.org/10.1109/ALLERTON.2009.5394956

[26] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. *Learning internal representations by error propagation*. Technical Report. California Univ San Diego La Jolla Inst for Cognitive Science.

[27] Ahmed Salem, Yannick Sautter, Michael Backes, Mathias Humbert, and Yang Zhang. 2020. Baaan: Backdoor attacks against autoencoder and gan-based machine learning models. arXiv preprint arXiv:2010.03007.

[28] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang. 2022. Dynamic Backdoor Attacks Against Machine Learning Models. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. IEEE Computer Society, Los Alamitos, CA, USA, 703–718. https://doi.org/10.1109/EuroSP53844.2022.00049

[29] Esha Sarkar, Hadjer Benkraouda, and Michail Maniatakos. 2020. I Came, I Saw, I Hacked: Automated Generation of Process-Independent Attacks for Industrial Control Systems. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security* (Taipei, Taiwan) *(ASIA CCS '20)*. Association for Computing Machinery, New York, NY, USA, 744–758. https://doi.org/10.1145/3320269.3384730

[30] CPO Magazine Scott Ikeda. 2021. Attacker Gains Remote Access To a Florida City's Water Supply, Attempts To Poison It; Is This an Emerging Widespread Threat? https://www.cpomagazine.com/cyber-security/attacker-gains-remote-access-to-a-florida-citys-water-supply-attempts-to-poison-it-is-this-an-emerging-widespread-threat/.

[31] Riccardo Taormina. 2018. AutoEncoders for Event Detection (AEED): a Keras-based class for anomaly detection in water sensor networks. https://github.com/rtaormina/aeed.

[32] Riccardo Taormina and Stefano Galelli. 2018. Deep-learning approach to the detection and localization of cyber-physical attacks on water distribution systems. *Journal of Water Resources Planning and Management* 144, 10 (2018), 04018065.

[33] Riccardo Taormina, Stefano Galelli, Nils Ole Tippenhauer, Elad Salomons, Avi Ostfeld, Demetrios G Eliades, Mohsen Aghashahi, Raanju Sundararajan, Mohsen Pourahmadi, M Katherine Banks, et al. 2018. Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks. *Journal of Water Resources Planning and Management* 144, 8 (2018), 04018048.

[34] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. 2019. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *2019 IEEE Symposium on Security and Privacy (SP)*. 707–723. https://doi.org/10.1109/SP.2019.00031

[35] Sharon Weinberger. 2011. Computer security: Is this the start of cyberwarfare? *Nature* 174 (June 2011), 142–145.

[36] S. Zhao, X. Ma, X. Zheng, J. Bailey, J. Chen, and Y. Jiang. 2020. Clean-Label Backdoor Attacks on Video Recognition Models. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 14431–14440. https://doi.org/10.1109/CVPR42600.2020.01445