A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems

Cheng Feng*‡, Venkata Reddy Palleti†, Aditya Mathur† and Deeph Chana‡
*Siemens Corporate Technology
Email: cheng.feng@siemens.com
†Singapore University of Technology and Design
Email: {ventaka_palleti,aditya_mathur}@sutd.edu.sg
‡Institute for Security Science and Technology
Imperial College London
Email: d.chana@imperial.ac.uk

***Abstract*—Industrial Control Systems (ICS) consisting of integrated hardware and software components designed to monitor and control a variety of industrial processes, are typically deployed in critical infrastructures such as water treatment plants, power grids and gas pipelines. Unlike conventional IT systems, the consequences of deviations from normal operation in ICS have the potential to cause significant physical damage to equipment, the environment and even human life. The active monitoring of invariant rules that define the physical conditions that must be maintained for the normal operation of ICS provides a means to improve the security and dependability of such systems by which early detection of anomalous system states may be achieved, allowing for timely mitigating actions – such as fault checking, system shutdown – to be taken. Generally, invariant rules are predefined by system engineers during the design phase of a given ICS build. However, this manually intensive process is costly, error-prone and, in typically complex systems, sub-optimal. In this paper we propose a novel framework that is designed to systematically generate invariant rules from information contained within ICS operational data logs, using a combination of several machine learning and data mining techniques. The effectiveness of our approach is demonstrated by experiments on two real world ICS testbeds: a water distribution system and a water treatment plant. We show that sets of invariant rules, far larger than those defined manually, can be successfully derived by our framework and that they may be used to deliver significant improvements in anomaly detection compared with the invariant rules defined by system engineers as well as the commonly used residual error-based anomaly detection model for ICS.**

***Keywords*—industrial control systems, anomaly detection, invariant rules, machine learning.**

I. Introduction

INDUSTRIAL Control systems (ICS) are used for the monitoring and controlling of various industrial processes. Typically, these systems are found in critical infrastructure assets such as chemical plants, water treatment and distribution systems, power generation, transmission and distribution facilities, etc. Though ICS have been used for many years, they were primarily designed as isolated from other systems and networks in an "air gapped" environment. However, recently, in the pursuit of numerous operational gains, modern Information and Communication Technologies (ICT) have been widely integrated into ICS, leading to their transformation from standalone systems to highly interconnected cyber-physical systems. Viewed through a security lens, many of these gains have been achieved at a cost of increased security risk. As an example, the advantages of remote access to ICS for monitoring and control are clear for day-to-day operations, but these may be countered by the significant increase in the system's exposure to cyber security threats present on the wider Internet. Unlike conventional IT systems, compromised operation or a failure in ICS has the potential to cause significant physical damage to national infrastructure and possible cascading failures, resulting in impacts that can include large scale power outages, disruption to health-service operations, compromised public transport safety, environmental damage and even direct loss of life.

The Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) provides evidence of the security risks posed to ICS. In 2014 ICS-CERT reported 245 incidents of cyber attacks from its trusted industrial partner network [1], a level which increased to 295 [2] and 290 [3] in 2015 and 2016 respectively. Furthermore, one of the most notorious cyber security attacks to date took place in 2009/10 against the ICS of a nuclear facility situated in Natanz, Iran. In this incident a computer worm known as Stuxnet [4] was used to infect and manipulate Programmable Logic Controllers (PLCs), resulting in significant physical damage to centrifuges at the plant and instigating international debate on the cyber security offensive capabilities of nation states. Similarly, a following ICS cyber security incident in late 2014 resulted in significant physical damage to a blast furnace in a German steel mill [5] and, more recently still, Ukrainian's capital city Kiev lost approximately one-fifth of its required power capacity as a result of an ICS cyber attack in 2016, causing a massive blackout that affected 225,000 citizens [6].

Due to the increasing number of reported cyber attacks and the serious consequences of their malfunction, much research work has been done to develop anomaly detection mechanisms to improve the resilience of ICS to both cyber attacks and physical faults in the recent years. Such mechanisms often

Network and Distributed Systems Security (NDSS) Symposium 2019
24-27 February 2019, San Diego, CA, USA
ISBN 1-891562-55-X
https://dx.doi.org/10.14722/ndss.2019.23265
www.ndss-symposium.org

exploit a definition of the normal behavior of the ICS and an alert is raised when significant deviations that do not match with the definitions are observed. To date, there exist several types of ICS-specific anomaly detection methods, such as device-based, program-based, network-based and process-based detection. Specifically, device-based detection often uses the idea of fingerprinting for detecting intrusive devices. For instance, in [7] the authors proposed two approaches for creating fingerprints tailored for devices in the ICS context based on message response time and operation time measurements, respectively. Program-based methods discover anomalous behaviour by checking the control or data flow in the control programs on programmable controllers. For example, the Trusted Safety Verifier (TSV) is presented for the verification of safety-critical code executed on PLCs using a combination of symbolic execution and model checking [8], [9]. Another framework called Orpheus is developed in [10] which utilises event-aware finite-state automaton (eFSA) model to detect anomalous control program behaviors particularly caused by data-oriented attacks. Network-based detection methods reveal anomalies by investigating the network traffic flow such as the header, payload, timing and sequence of messages in the ICS network, but are less sensitive to anomalies only exhibited in the physical properties of the system [11], [12], [7]. Furthermore, such methods are less general because they are often designed for specific protocols such as Modbus [13], [14], [15] and DNP3 [16].

The focus of this paper is the process-based detection methods which look directly at the physical process variables such as sensor readings and actuator states, and the mathematical relationships among them controlled by the corresponding PLCs to identify anomalies. At present, most process-based anomaly detection methods rely on a predictive model such as the autoregressive model [17], the linear dynamic state-space model [18], [19] and neutral network-based regression models [20], where future sensor measurements predictions are generated based on historical process values and then compared with real measurements to generate a residual error. An anomaly is observed if the residual error exceeds a predefined threshold. However, due to the fact that there exist various sources of noise in industrial processes, a strict threshold between normal and abnormal sensor measurements is normally not definable. Recent studies have shown that attackers may utilize this ambiguous decision boundary to inject malicious measurements (so-called stealthy attacks) that both achieve an attacker's objectives and avoid detection by current methods [21], [22], [23]. A proposed alternative approach to using residual errors is the invariant rule-based method [24], [25]. Invariant rule-based methods make use of physical conditions that are known *a-priori* must hold for all ICS states. Any observed physical process values that break these rules are classified as anomalies. Typically, these invariant rules are defined by system engineers during the design stages of an ICS. This manual process is not only costly but also error-prone as the implementation and design of ICS cannot perfectly match. Furthermore, there exist many hidden invariant rules that are extremely difficult to discover by human-beings, especially those which are across several subsystems. As a result, the performance of existing invariant rule-based anomaly detection methods is often limited by both the inaccuracy and inadequacy of the design-based rules. This motivates a new method for discovering invariant rules in such systems.

In this paper, we pursue a purely data-driven approach to derive invariant rules for anomaly detection in ICS. Specifically, we propose a novel framework which utilizes the general control dynamics of ICS and combines several machine learning and data mining techniques to systematically generate invariant rules from data logs capturing the physical process variables at discrete time steps during a period of normal operation of the ICS. By applying our approach on two real world ICS testbeds, we show that a significantly larger number of *meaningful* invariant rules can be derived compared with a manual design-based process and that anomaly detection models based on our data-driven invariant rules are far more capable than standard implementations. The data-driven invariant rules also proves to be more accurate and results in significant reduction to false positive rates. In addition, we also compare the anomaly detection performance of our newly established method with a residual error-based detection model. The experimental results show that under the same false positive rate, the ability of using the invariant rules generated by our framework to detect anomalies outperforms the residual error-based detection model by a clear margin. We summarize the main contributions of this work as follows: (1) a novel framework is proposed to systematically generate invariant rules from ICS data logs, (2) the concept of meaningful invariant rules is introduced to guarantee that the generated invariant rules are suitable for anomaly detection usage, (3) a parameter tuning method is proposed to guide the generation of meaningful invariant rules, (4) we present case studies on two real world ICS testbeds (a water distribution system and a water treatment plant) to demonstrate the effectiveness of our framework for invariant rule generation, and the anomaly detection performance of using the generated invariant rules are compared with two baseline anomaly detection models.

The remainder of this paper is organized as follows. We give a brief introduction of the general ICS architecture and process-based anomaly detection mechanisms in the next section. This is followed by the problem statement section which describes the formal definition of invariant rules. Then, we present our systematic framework for invariant rule generation in Section IV. The two case studies are given in Sections V and VI. Section VII gives a further discussion on results in our case studies. Finally, Sections VIII and IX discuss related work, future research and draw final conclusions.

II. Background

In this section, we give a brief introduction of the general ICS architecture, and the commonly used anomaly detection mechanisms for securing the physical processes in ICS.

A. General Architecture of ICS

A typical ICS consists of devices and subsystems such as sensors and actuators, Programmable Logic Controllers (PLCs), Distributed Control Systems (DCS), Remote Terminal Units (RTUs), Supervisory Control and Data Acquisition Systems (SCADA) and Human Machine Interfaces (HMIs). Figure 1 represents the architecture of a general ICS, which includes physical, control, and supervisory control layers. The

so-called field devices such as sensors and actuators in the physical layer report and modify physical process states via signals transmitted and received from PLCs and RTUs, which are situated in the control layer. For example, in a water distribution system, the control layer obtains data from the field devices such as water-level sensors, flow meters and water quality sensors. Based on the received data from these sensors, the control layer issues commands to actuators to perform specific actions such as turning pumps on or off and opening or closing valves. Finally, the supervisory control layer contains SCADA, HMIs, engineering work stations and data historian components. These directly communicate with the control layer to provide higher level supervisory monitoring and control functions and may also interface with wider corporate systems and networks through a demilitarized zone, which is not shown in the figure. Anomaly detection mechanisms are often deployed in this layer, where the sensor measurements and actuator states are continuously checked to secure the physical processes under control.

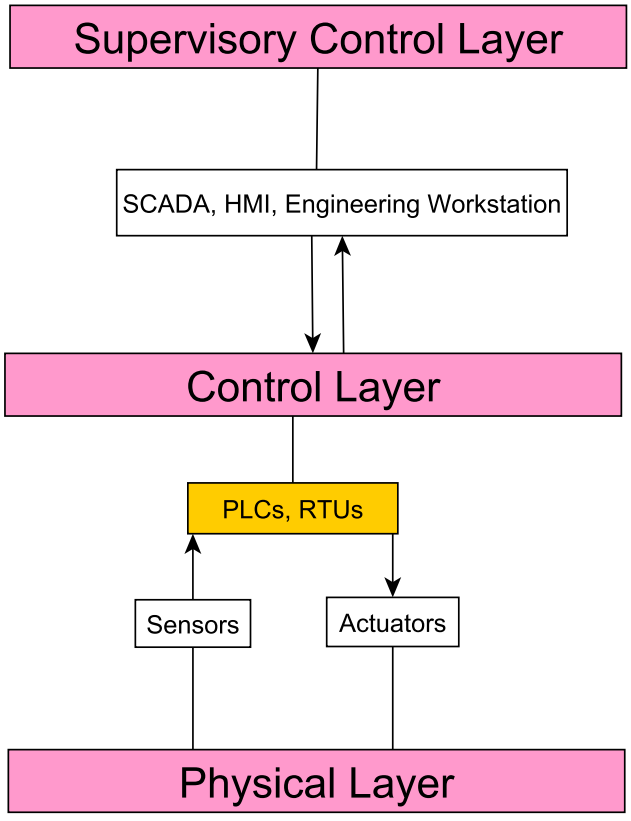

Fig. 1. The architecture of a typical ICS.

B. Residual Error-based Anomaly Detection

To protect ICS from physical faults and cyber attacks, anomaly detection mechanisms are often deployed by monitoring the sensor measurements and actuator states in the system at discrete time steps. To date, most process-based anomaly detection mechanisms rely on a predictive model which predicts the sensor readings at each time step based on previous sensor measurements and actuator states, and an alarm is triggered if the residual error between the predicted measurements and their observation exceeds a specific threshold. The underlying predictive model can take many different forms, examples include Auto-Regressive (AR) models [17], Linear Dynamic State-space (LDS) models [18], [19] and other regression models, e.g., based on deep neural networks [20]. Nevertheless, all the predictive models can be denoted as a high level function:

$$\hat{\mathbf{x}}^{(t)} = f(\mathbf{x}^{\{t-p:t-1\}}, \mathbf{u}^{\{t-p:t-1\}}; \boldsymbol{\theta})$$

where $\mathbf{x}^{\{t-p:t-1\}}$ and $\mathbf{u}^{\{t-p:t-1\}}$ are the model inputs which represent the sensor measurements and actuator states at previous p time steps, respectively (clarify: an exception is the AR model, which usually only takes sensor measurements at previous time steps as its input); $\hat{\mathbf{x}}^{(t)}$ is the model output which represents the predicted sensor measurements at the current time step; $\boldsymbol{\theta}$ is parameters to be estimated, e.g., by minimizing the mean square error between predicted sensor measurements and their real values given a time-series data log consisting of the sensor measurements and actuator states in the ICS. Based on the prediction by the models, an alarm will be raised when the Euclidean distance between the predicted sensor measurements and their observations exceeds a specific threshold: $\|\hat{\mathbf{x}}^{(t)} - \mathbf{x}^{(t)}\| > \tau$, where $\|\hat{\mathbf{x}}^{(t)} - \mathbf{x}^{(t)}\|$ is called the residual error at time point t. A common problem for such residual error-based methods is the absence of a specific value for τ to accurately separate anomalous and normal measurements. Recently, it has been shown that attackers many exploit this ambiguous boundary to cause significant deviation of sensor measurements by the accumulated injection of false data that is designed not cross the residual error threshold [21], [22], [23].

C. Invariant Rule-based Anomaly Detection

An invariant rule is defined as a physical condition that must be satisfied for any given state of an ICS [24], [25]. Such conditions may include properties such as PH, pressure, temperature readings, liquid levels etc and the dependencies between them. For an ICS state at any given time, monitoring the physical state of the system against such rules can act as the basis for detecting deviations from operational normality.

Invariant rules are generally derived based on the design of the ICS. As an illustrative example, we consider the simple ICS in Figure 2, which shows a design graph for a portion of a water distribution system. In the figure, MV101 and MV201 are valve actuators, P101 is a pump actuator, LIT101 and LIT301 are sensors and T1 is a water tank. The states of the level sensors (LIT) are defined as L(ow) and H(igh) and the states of actuators (MV* and P101) are defined as OPEN/CLOSE or ON/OFF. The flow of water is indicated by the labeled connections between the pairs MV101, T1 and P101, T1. Explicitly, the flow rate of water into T1 is denoted by $W\_in(t)$ and is decided by the state of the valve MV101, whilst the out-flow rate of water from T1, $W\_out(t)$, depends on the state of P101. The water level in T1, $h(t)$, is measured by sensor LIT101 and made available to a PLC that controls the systems' actuators. LIT301 measures the water level in a tank of another part of the system which is not shown in this graph. The graph indicates that P101 should be in the ON state when sensors LIT101 and LIT301 indicate water level to be

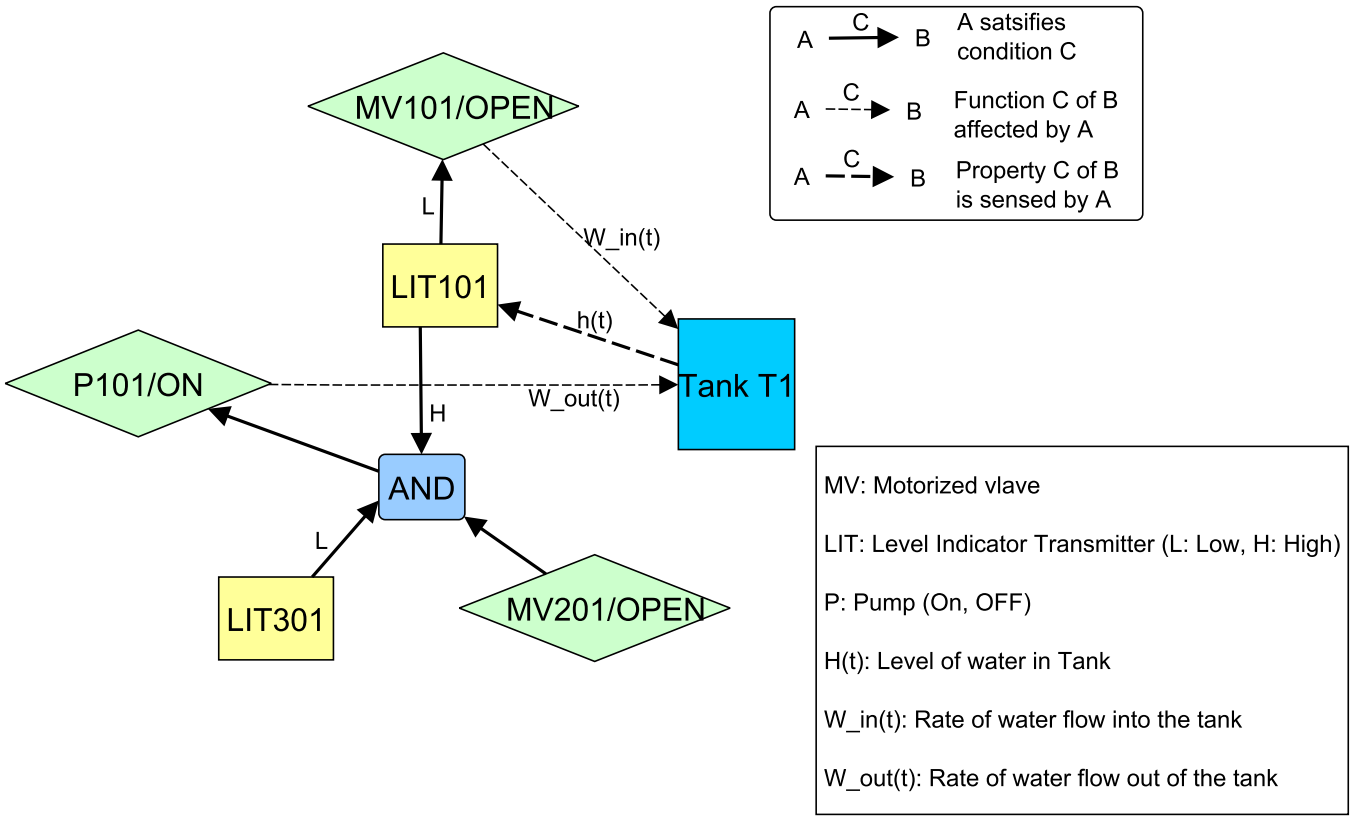

Fig. 2. The design graph for one stage of a water distribution system.

H(igh) and L(ow) in their respective tanks, and valve MV201 is OPEN. Similarly, valve MV101 should be open when LI101 indicates water level in T1 is L(ow). In this way, the pair of invariant rules now derived from the design graph may be summarized as:

$$\text{LIT101-H} \Rightarrow \text{MV101=OPEN}$$
$$\text{LIT301-L, LIT101-H, MV201=OPEN} \Rightarrow \text{P101=ON}$$

Based on the above discussion, the question to ask is whether the set of invariant rules derived from such design graphs is complete and correct. An important point to note is that typical ICS consist of a very large set of actuators and sensors, making the manual derivation of invariant rules extremely expensive with the likelihood that only a small subset of the complete rules-set is captured. To address this problem directly, we define a framework by which invariant rules may be derived from a purely data-driven perspective in the following sections.

III. Problem Statement

We consider an ICS with m sensors and n actuators. Let $\mathcal{D}^{\{1:T\}} = \{\mathbf{d}^1, \mathbf{d}^2, \ldots, \mathbf{d}^T\}$ be a time-series data log in which each signal $\mathbf{d}^t = \{\mathbf{x}^t, \mathbf{u}^t\}$ consists of two vectors such that $\mathbf{x}^t \in \mathbb{R}^m$ where each element $x^t \in \mathbf{x}^t$ is a real value capturing the reading of a sensor, $\mathbf{u}^t \in \mathcal{K}^n$ where each element $u^t \in \mathbf{u}^t$ is a categorical value representing the state of an actuator in the system recorded at discrete time steps $t \in [1, 2, \ldots, T]$. A point to note is that we assume there is no anomalous signal in the data log $\mathcal{D}^{\{1:T\}}$. In practice, $\mathcal{D}^{\{1:T\}}$ can be collected by operating the ICS in an "air-gapped" separation (no access from corporate network) for a period of time which captures the normal profile of system operation.

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_k\}$ be a set of k predicates called items, and each signal $\mathbf{d}^t \in \mathcal{D}^{\{1:T\}}$ satisfies a subset of predicates in $\mathcal{I}$, thus can be denoted by an itemset $I^t \subseteq \mathcal{I}$. Let $\sigma(X)$ denote the support of an itemset X, representing the fraction of time steps at which the itemset X is contained by I^t, calculated as follows:

$$\sigma(X) = \frac{\sum_{t=1}^{T} \mathbf{1}(X \subseteq I^t)}{T}$$

where $\mathbf{1}(\cdot)$ is an indicator function which yields one only if its condition is true, and outputs zero otherwise.

Formally, we define an invariant rule as follows:

$$X \Rightarrow Y \quad \texttt{where} \quad X, Y \subseteq \mathcal{I} \wedge X \cap Y = \emptyset \wedge \frac{\sigma(X \cup Y)}{\sigma(X)} = 1$$

where we call X the antecedent itemset, Y the consequent itemset. The rule means whenever a signal contains the antecedent X, it must also contains the consequent Y; and the antecedent and consequent itemsets must be mutually exclusive. As an illustration, the following invariant rule:

$$\{x_1^t > ax_2^t + b, u_1^t = \text{ON}\} \Rightarrow \{x_3^t < c, u_2^t = \text{OFF}\}$$

implicates that if predicates $x_1^t > ax_2^t + b$ and $u_1^t = \text{ON}$ are satisfied at any given time step t, predicates $x_3^t < c$ and $u_2^t = \text{OFF}$ must also be satisfied at t, where $a, b, c \in \mathbb{R}$ denote some constants.

IV. A Systematic Framework for Invariant Generation

In this section, we present our framework to systematically generate invariant rules from ICS data logs. Specifically, given an arbitrary ICS data log $\mathcal{D}^{\{1:T\}}$, we decompose the process of invariant rule generation into two steps:

- Predicate Generation: this step is to generate a set of meaningful predicates from the data log for the construction of the predicate set $\mathcal{I}$.
- Invariant Rule Mining: with the predicate set $\mathcal{I}$, we can transform the data log $\mathcal{D}^{\{1:T\}}$ into a database of itemsets $I^{\{1:T\}} = \{I^1, I^2, \ldots, I^T\}$ where each element I^t represents the itemset consisting of the predicates in $\mathcal{I}$ which are satisfied by $\mathbf{d}^t$. Then, we mine meaningful invariant rules from the database $I^{\{1:T\}}$ which can be used for anomaly detection in the ICS.

A. Predicate Generation

Here, we describe how to generate the set of meaningful predicates from the ICS data log.

First of all, generating predicates for categorical variables capturing actuator states is rather straightforward. Specifically, let $\{v_1, v_2, \ldots, v_l\}$ be all the states for an actuator u appearing in the data log, then we generate predicates $\{u^t = v_1, u^t = v_2, \ldots, u^t = v_l\}$. Intuitively, suppose u is a pump, then predicates $\{u^t = \text{ON}, u^t = \text{OFF}\}$ are generated.

In the case of continuous variables representing sensor readings, their value domain is infinite and so setting all possible values as predicates is not only meaningless but will also cause the invariant rule mining step to be prohibitively expensive. Therefore, we propose two strategies to generate meaningful predicates here: the distribution-driven strategy and the event-driven strategy, where both strategies are based on the control dynamics of general ICS.

1) The Distribution-driven Strategy: The distribution-driven strategy utilizes the fact that the update of sensor readings at each time step are generally decided by the current control state of the ICS. Specifically, let $\Delta x^t = x^{t+1} - x^t$ be the update on a sensor reading from time step t to $t+1$, we assume there are K hidden control states to decide the value of Δx^t at all time steps. Thus, let $k \in (1, 2, \ldots, K)$ be the hidden state at time step t, we can represent:

$$\Delta x^t = \mu_k + \varepsilon_k$$

where $\mu_k \in \mathbb{R}$ captures the expected update of the sensor reading in hidden state k; $\varepsilon_k \sim \mathcal{N}(0, \sigma_k^2)$ is the sensor noise which is assumed to be a random process normally distributed with zero mean. That is to say, all the updates of the sensor reading, denoted as $\Delta X^{\{1:T-1\}} = \{\Delta x^1, \Delta x^2, \ldots, \Delta x^{T-1}\}$ in the data log, are generated from K different Gaussian distributions with unknown parameters. Therefore, once we can infer to which distribution each sensor reading update belongs, we can generate the following predicates:

$$\{\Delta x^t \sim \mathcal{N}_1, \ldots, \Delta x^t \sim \mathcal{N}_K\}, \tag{1}$$

in which the predicate $\Delta x^t \sim \mathcal{N}_k$ means that the update of the sensor reading at time step t is generated from the kth distribution.

In order to generate the above predicates for each sensor, we first need to infer the K Gaussian distributions which generates the data $\Delta X^{\{1:T-1\}}$. Specifically, we conduct this inference by fitting Gaussian Mixture Models (GMMs) with different number of components (distributions) to $\Delta X^{\{1:T-1\}}$ using an Expectation-Maximization (EM) algorithm [26]. Concretely, a GMM with K components in our case is defined by three vectors of parameters: the means $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_K\}$ and standard deviations $\boldsymbol{\sigma} = \{\sigma_1, \ldots, \sigma_K\}$ of the sensor updates under the K different control states, and the mixture weight vector for the K components $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$ (which indicates the prior probability of a sensor update belongs to each component). Then, with a set of fitted candidate GMMs with different number of components, the one which minimizes the BIC (Bayesian information criterion) score [27] is selected for predicate generation. Specifically, the BIC score is a criterion commonly used for model selection among a finite set of models by achieving a balance between the likelihood function of the model and the model complexity. The corresponding BIC score for a candidate GMM in our context is defined by the following equation:

$$BIC(\mathcal{M}_K) = -2 \log p(\Delta X^{\{1:T-1\}} \mid \mathcal{M}_K) + \kappa \log(n) \tag{2}$$

where $\mathcal{M}_K$ is the model with K components, $\log p(\Delta X^{\{1:T-1\}} \mid \mathcal{M}_K)$ is the log likelihood of $\Delta X^{\{1:T-1\}}$ given $\mathcal{M}_K$, κ is the number of parameters in $\mathcal{M}_K$, and $n = T - 1$ is the number of data points in $\Delta X^{\{1:T-1\}}$. After getting the GMM $\mathcal{M}_K$ with the lowest BIC score, we can generate predicates as in Equation 1 accordingly. Furthermore, a predicate $\Delta x^t \sim \mathcal{N}_k$ is satisfied at time t if and only if:

$$r_k^t = \max(r_1^t, \ldots r_K^t)$$

where r_k^t is the membership probability of the kth distribution to the sensor update Δx^t as calculated by the following equation:

$$r_k^t = \frac{\pi_k \, \mathcal{N}(\Delta x^t \mid \mu_k, \sigma_k)}{\sum_{j=1}^{K} \pi_j \, \mathcal{N}(\Delta x^t \mid \mu_j, \sigma_j)}$$

in which

$$\mathcal{N}(\Delta x^t \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \, e^{-\frac{(\Delta x^t - \mu)^2}{2\sigma^2}}.$$

The satisfaction of predicate $\Delta x^t \sim \mathcal{N}_k$ indicates that we believe sensor update Δx^t is generated from the kth Gaussian distribution, thus is occurred under the kth hidden control state. Furthermore, let $\Delta x^{t'}$ be a sensor update during the detection phase, we also define $\Delta x^{t'} \sim \mathcal{N}_{K+1}$ if

$$r_k^{t'} < \min(r_k^1, \ldots, r_k^{T-1}) \quad \forall k \in (1, ..., K)$$

which means that $\Delta x^{t'}$ is an anomalous sensor update which should not occur under any control state in the system, thus it is treated as an outlier.

The detailed algorithm of using our distribution-driven strategy to generate predicates for sensor updates is given in Algorithm 1, where we fit candidate GMMs with component numbers $(1, 2, \ldots, N)$ each using the EM algorithm given in steps 3-5. Finally, the same process is applied for each sensor in the system to generate distribution-driven predicates for all the m sensors.

Algorithm 1 The algorithm for generating distribution-driven predicates from the data log

Require: $\Delta X^{\{1:T-1\}}$ the updates on the reading of a sensor in the data log

1: **for** $\hat{K} = 1, 2, \ldots, N$ **do**

2: Cluster all the data in $\Delta X^{\{1:T-1\}}$ into $\hat{K}$ clusters by K-Means algorithm for an initial guess of $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\pi}$

3: **for** $iter = 1, 2, \ldots, M$ **do**

4: E-step: calculate the membership probability (also called as responsibility) of each Gaussian to each data point as follows:

$$r_k^t = \frac{\pi_k \, \mathcal{N}(\Delta x^t \mid \mu_k, \sigma_k)}{\sum_{j=1}^{\hat{K}} \pi_j \, \mathcal{N}(\Delta x^t \mid \mu_j, \sigma_j)} \qquad \forall \begin{smallmatrix} t \in (1,\ldots,T-1) \\ k \in ([1,\ldots,\hat{K}) \end{smallmatrix}$$

5: M-step: let $N_k = \sum_{t=1}^{T-1} r_k^t$, calculate the new mean and standard deviation of each Gaussian by

$$\mu_k = \frac{1}{N_k} \sum_{t=1}^{T-1} r_k^t \Delta x^t \quad \forall k \in (1, \ldots, \hat{K})$$

$$\sigma_k = \sqrt{\frac{1}{N_k} \sum_{t=1}^{T-1} r_k^t (\Delta x^t - \mu_k)^2} \quad \forall k \in (1, \ldots, \hat{K})$$

Calculate new $\boldsymbol{\pi}$ by

$$\pi_k = \frac{N_k}{T-1} \quad \forall k \in (1, \ldots, \hat{K})$$

6: **end for**

7: Calculate $BIC(\mathcal{M}_{\hat{K}})$ according to Equation 2.

8: **end for**

9: Select $\mathcal{M}_K$ where $K \leftarrow \arg\min_{\hat{K}} BIC(\mathcal{M}_{\hat{K}})$

10: Generate predicates $\{\Delta x^t \sim \mathcal{N}_1, \ldots, \Delta x^t \sim \mathcal{N}_K\}$.

2) The Event-driven Strategy: The event-driven strategy utilizes the fact that in the ICS context, the updates of actuator states are generally triggered by critical values of sensor readings. Thus, we generate predicates for sensor readings based on those critical values which trigger the updates of actuator states.

Concretely, we define events as actions which happen instantaneously and trigger discrete changes on actuator states, e.g., a pump is switched from ON to OFF. Then, let us define an event set E, where each element $e \in E$ denotes a distinct event representing the update of an actuator from a specific state to another specific state. Furthermore, let T_e denote the set of time steps at which the event e occurs. Thus, to find the trigger of an event e, we fit a linear regression model for the values of sensor readings at the time steps in T_e. Specifically, for each sensor $i \in (1, \ldots, m)$, the model as follows is fitted for an event e:

$$\hat{x}_i^t = \sum_{j=1 \wedge j \neq i}^{m} \alpha_j x_j^t + \alpha_0 \quad \forall t \in T_e$$

Importantly, the above model is trained to minimize the L1 loss [28] for only the time steps at which the event e occurs, such that:

$$\mathcal{L} = \frac{1}{|T_e|} \sum_{t \in T_e} (\hat{x}_i^t - x_i^t)^2 + \lambda \sum_{j=1 \wedge j \neq i}^{m} |\alpha_j| \tag{3}$$

where the first part in the right hand side of the above equation is the mean square error between the predicted sensor readings and the real sensor readings at the time steps that event e occurs; the second part is the L1 regularization loss to avoid overfitting. Moreover, since minimizing $\mathcal{L}$ will automatically drive the parameters for unrelated variables to zero [28], we can finally get a regression model with a few or even zero related variables, such that:

$$\hat{x}_i^t = \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0 \quad \forall t \in T_e$$

where $R^i(e)$ denotes the set of related sensors with sensor i that co-trigger event e (note that $R^i(e)$ can potentially be an empty set which means the sensor i triggers event e individually). Then, we say a trigger for event e is found if

$$|\hat{x}_i^t - x_i^t| < \epsilon \quad \forall t \in T_e$$

where ϵ is small threshold value close to zero. Based on an event trigger, we generate two predicates:

$$x_i^t < \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0 - \epsilon$$

$$x_i^t > \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0 + \epsilon$$

which represent the expected conditions for sensor readings before and after the event is triggered.

The detailed procedure for generating all the event-driven predicates from the data log is given in Algorithm 2. An important point to note is that, the set $R(e)$ in the algorithm is used to guarantee no duplicated predicates are generated for the same event because when an event trigger is found for a sensor i, it is certain that we can find a duplicated trigger for the event for all sensors in $R^i(e)$.

B. Invariant Mining

After the predicate generation step, we get a predicate set $\mathcal{I}$ which consists of all the generated predicates for actuator states and sensor readings for the ICS. Then each signal $\mathbf{d}^t \in \mathcal{D}^{\{1:T\}}$ can be transformed to an itemset I^t capturing the predicates in $\mathcal{I}$ that $\mathbf{d}^t$ satisfies. Now, the goal is to find all *meaningful* invariant rules from the itemset database $I^{\{1:T\}}$ which can be used for anomaly detection in the system. Before giving the rule mining algorithm, we first introduce the concept of meaningful invariant rules in our context.

Algorithm 2 The algorithm for generating event-driven predicates from the data log

Require: the event set E, the readings of all sensors in the data log $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^T\}$

for all $e \in E$ **do**
  Set $R(e) = \emptyset$
  for $i = 1, 2, \ldots, m$ **do**
    if $i \notin R(e)$ **then**
      Fit a regression model for $x_i^t \ \forall t \in T_e$ with the L1 loss as in Equation 3 minimized:

$$\hat{x}_i^t = \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0$$

      if $|\hat{x}_i^t - x_i^t| < \epsilon \quad \forall t \in T_e$ **then**
        Generate two predicates:

$$x_i^t \ < \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0 - \epsilon$$

$$x_i^t \ > \sum_{j \in R^i(e)} \alpha_j x_j^t + \alpha_0 + \epsilon$$

        Set $R(e) = R(e) \cup R^i(e)$
      end if
    end if
  end for
end for

1) Concept of Meaningful Invariant Rules: Here we define the concept of meaning invariant rules with respect to the anomaly detection usage.

Specifically, an invariant rule is meaningful only if it satisfies two conditions: the minimum support condition and the non-redundant condition. Intuitively, the first condition guarantees that the invariant rule achieves a required statistical significance, thus is not merely satisfied in the data log by coincidence which will cause many false alarms when using it for anomaly detection. The second condition is also important because deriving a large number of redundant invariant rules will not improve the performance of the anomaly detection model, but can significantly increase the time cost of the anomaly detection process.

Before presenting the formal definition of the two conditions, we first introduce two properties that will be used extensively in the remaining part of this section:

Property 1. $X \subseteq Y \rightarrow \sigma(Y) \leq \sigma(X)$

Property 2. *Let $t(X)$ denote the set of time steps at which $X \in I^t$, then if $X \subseteq Y$ and $\sigma(X) = \sigma(Y)$, $t(X)$ must overlap with $t(Y)$.*

Specifically, the first property is called the anti-monotone property, which indicates that if itemset X is a subset of Y , then $\sigma(Y)$ must not exceed $\sigma(X)$. It is also straightforward to see the second property must hold because $\forall t$ if $Y \subseteq I^t$, then it is certain that $X \subseteq I^t$ since $X \subseteq Y$, furthermore as $\sigma(X) = \sigma(Y)$, $t(X)$ must overlap with $t(Y)$.

Then, the formal definition of the minimum support condition is given as follows:

Minimum Support Condition. *An invariant rule $X \Rightarrow Y$ is meaningful, then:*

$$\sigma(Z) > \max(\gamma \min(\sigma(i_{z_1}), \sigma(i_{z_2}), \ldots, \sigma(i_{z_n})), \theta) \tag{4}$$

where $Z = X \cup Y$, $\{i_{z_1}, i_{z_2}, \ldots, i_{z_n}\}$ denotes all the items in Z.

Specifically, according to the anti-monotone property, we can see that the support of an invariant rule is bounded by the support of the least frequent item in it:

$$\sigma(Z) \leq \min(\sigma(i_{z_1}), \sigma(i_{z_2}), \ldots, \sigma(i_{z_n})).$$

Then, since the support of different items can vary significantly in the ICS data log, e.g., a pump can be in the ON state at only 5% time steps, but be in the OFF state at 95% time steps in the data log, it is unfair to set a unique minimum support threshold for all invariant rules. Thus, we require the support of an invariant rule to be larger than the product of γ and its specific upper bound, where $\gamma \in (0, 1)$. Furthermore, $\theta \in (0, \gamma)$ is another threshold which controls the minimum fraction of samples in the data log that a meaningful invariant rule needs to capture, which guarantees that those rare items (e.g., an item that only appears twice in the itemset database) are excluded for any meaningful invariant rules. Note that the value of θ should be less than γ, otherwise $\gamma \min(\sigma(i_{z_1}), \sigma(i_{z_2}), \ldots, \sigma(i_{z_n}))$ will always be smaller than θ, leading to a unique minimum support threshold for all rules.

The non-redundant condition is formally defined below:

Non-redundant Condition. *An invariant rule $X \Rightarrow Y$ is meaningful, then there must not exist another invariant rule $U \Rightarrow W$, such that $X \subseteq U$, $Y \subseteq W$, and $\sigma(X \cup Y) = \sigma(U \cup W)$.*

Specifically, a rule does not satisfy the above condition is redundant because if $X \cup Y \subseteq U \cup W$ and $\sigma(X \cup Y) = \sigma(U \cup W)$, then $t(X \cup Y)$ must overlap with $t(U \cup W)$ according to Property 2. That means if $X \Rightarrow Y$ is violated at any given time step, then $U \Rightarrow W$ must also be violated. Thus the rule $X \Rightarrow Y$ will not have any extra contribution beyond the rule $U \Rightarrow W$ in the anomaly detection process.

2) Invariant Mining Algorithm: Generating meaningful invariant rules from the itemset database $I^{\{1:T\}}$ can be treated as an association rule mining problem which has been a classic topic in data mining that is initially aimed to find interesting relations between products in large transaction databases for market basket analysis [29], [30]. Specifically, following the common strategy of association rule mining, we split the invariant rule mining algorithm into two steps: (i) Candidate Itemset Mining: whose objective in our context is to find all the *closed frequent itemsets with multiple minimum support thresholds* in the itemset database $I^{\{1:T\}}$ (ii) Invariant Rule Generation: whose objective is to extract all the invariant rules from the candidate itemsets found in the previous step.

Specifically, the definition of closed frequent itemset is given as follows:

Definition 1. *An itemset Z is a closed frequent itemset if its support is larger than the minimum support threshold, and meanwhile none of its immediate supersets has exactly the same support as Z.*

Importantly, the minimum support threshold in the above definition is not an unique value in our case, instead it is defined itemset-wise as given in Equation 4. Moreover, it can be proved that all the meaningful invariant rules can be derived from those closed frequent itemsets, and all the invariant rules derived from the closed frequent itemsets are meaningful:

Proof: If a rule $X \Rightarrow Y$ is meaningful, then $Z = \{X \cup Y\}$ must satisfy the minimum support condition. Moreover, there must not exist an immediate superset $W = \{Z, i\}$ which has exactly the same support count as Z, because if $\sigma(W) = \sigma(Z)$ and $Z \subset W$, then $t(Z)$ must overlap with $t(W)$ according to Property 2, which means there must also exists an invariant rule $X \Rightarrow \{Y, i\}$ which violates the non-redundant condition of $X \Rightarrow Y$. As a result, $Z = \{X \cup Y\}$ must be a closed frequent itemset.

If a rule $X \Rightarrow Y$ is derived from a closed frequent itemset $X \cup Y$, then the minimum support condition must hold. Meanwhile, if there exists a rule $U \Rightarrow W$ such that $X \subseteq U$, $Y \subseteq W$, and $\sigma(X \cup Y) = \sigma(U \cup W)$, then there must also exist an itemset $U' \cup W'$ which is an immediate superset of $X \cup Y$, such that $X \subseteq U' \subseteq U$ and $Y \subseteq W' \subseteq W$ and $\sigma(X \cup Y) = \sigma(U' \cup W') = \sigma(U \cup W)$ according to the anti-monotone property. However, as $X \cup Y$ is a closed itemset, the above cannot happen. As a result, $X \Rightarrow Y$ must be a meaningful invariant rule. ■

To date, there are many algorithms for mining closed frequent itemsets in transaction databases, examples are the AprioriClose algorithm [31], the LCM algorithm [32], the CHARM algorithm [33] and the FPClose algorithm [34], etc. However, all the above algorithms rely on the downward closure property, which is "all non-empty subsets of a frequent itemset must also be frequent", to reduce the search space of frequent itemsets. As a result, none of them can deal with the candidate itemset mining problem in our context because of the multiple minimum support thresholds for selecting frequent itemsets defined in Equation 4, which breaks the downward closure property. Nevertheless, there are available algorithms for mining frequent itemsets with multiple minimum support thresholds, such as the Apriori-based algorithm known as Multiple Support Apriori (MSApriori) [35], two FP-growth based algorithms, Conditional Frequent Pattern-growth (CFP-growth) [36] and CFP-growth++ [37]. Specifically, the CFP-growth algorithm is more efficient than MSApriori by using a MIS-tree to store the crucial information about frequent itemsets to largely reduce the search space of frequent itemsets. The CFP-growth++ algorithm is the improved version of CFP-growth, which introduced several pruning techniques to further reduce the search space of the algorithm. In this work, we first apply the CFP-growth++ algorithm to find all the frequent itemsets in the itemset database $I^{\{1:T\}}$. Due to lack of space, we refer to [37] for the details of the CFP-growth++ algorithm. After that, a filtering step is conducted to select all the closed itemsets among the discovered frequent itemsets according to Definition 1.

After getting all the closed frequent itemsets in $I^{\{1:T\}}$, we extract invariant rules from any given closed frequent itemset Y by partitioning the itemset Y into two non-empty subsets, X and $Y - X$, then a meaningful invariant rule $X \Rightarrow Y - X$ is generated if $\frac{\sigma(Y)}{\sigma(X)} = 1$.

C. Parameter Tuning

In the invariant rule mining step, the number of meaningful invariant rules to be generated is influenced by the value of two important parameters, namely γ and θ in Equation 4, which defines the rule-wise and the global minimum support threshold for the generated rules, respectively. Specifically, with smaller values of γ and θ, more meaningful invariant rules will be generated, thus the potential chance for using the rules to reveal anomalies is also increased. However, the statistical significance of the derived rules is also decreased which will potentially lead to more false positives when using the rules for anomaly detection. Furthermore, the processing time cost for checking the invariant rules on each data point will also be increased. Thus, we also propose a method to decide the optimal values of γ and θ based on a validation step.

Concretely, let us have a training data log as well as a validation data log (both contain no anomalies), and we generate invariant rules from the training data log with different values of γ and θ. Then the generated invariant rules are used to detect anomalies in the validation data log. Let T_v be the total number of data points in the validation data log, $A(\gamma, \theta)$ be the number of detected anomalies in the validation data log using the invariant rules generated with γ and θ, since any detected anomaly in the validation data log is a misclassification, we can calculate the validation error as follows:

$$Error_v(\gamma, \theta) = A(\gamma, \theta)/T_v,$$

which is an estimate of the expected false positive rate by using the generated invariant rules for anomaly detection in the system.

Furthermore, let $t(\gamma, \theta)$ be the time cost for checking the invariant rules on each data point for anomaly detection, $N(\gamma, \theta)$ be number of meaningful invariant rules generated with γ and θ, τ_t and τ_e be the user-defined thresholds for the acceptable time cost for processing each data point and the acceptable validation error, the optimal values for γ and θ can be found by:

$$\begin{aligned} &\arg\max_{\gamma,\theta} \; N(\gamma, \theta) \\ &\text{subject to} \quad t(\gamma, \theta) \leq \tau_t, \; Error_v(\gamma, \theta) \leq \tau_e \end{aligned} \tag{5}$$

which maximizes the number of generated meaningful invariant rules under acceptable time cost for processing each data point and validation error.

V. WADI Case Study

In this section, we present a case study in which we generate invariant rules from the data log of a Water Distribution System (WADI) testbed, and then conduct anomaly detection using the generated rules. Furthermore, we compare our anomaly detection result with two baseline models, namely the model consisting of the design-based invariant rules of system, and a residual error-based model.

A. WADI Testbed

The WADI testbed is a fully operational physical testbed that represents a scaled-down version of a real urban water distribution ICS with the capacity to provide a distribution

output of 10 gallons/minute. As shown in Figure 3, the testbed is comprised of three stages: a primary grid (P1), a secondary grid (P2) containing an elevated reservoir (ER); and a return water grid (P3). P1 is comprised of two 2500 litre capacity raw water tanks that are fed by two sources; a raw water inlet valve and P3. Chemical dosing pumps are installed to maintain a consistent water quality input to these tanks and a water level sensor is installed in each tank. Water quality is monitored by sensors in all three stages with measurements being made on water conductivity, turbidity, PH and Oxidation Reduction Potential (ORP). In addition, two contaminant sampling stations, P2A and P2B, are installed in the testbed to measure water quality parameters prior to its delivery to the consumer tanks. The P2 stage consists of two elevated tanks and six consumer tanks and it should be noted that the dynamics of the whole system is driven by the preset demand settings of the consumer tanks. Based on these presets, water flows from the elevated tanks into the consumer tanks at a certain rate and once the consumer tanks are filled, water drains into the return grid (P3), which in turn supplies the primary grid (P1).

B. Data Collection and Experiment Setup

The data collection process is conducted as follows: Initially, the demand pattern is generated over 24 hours of a day. The demand profile on each day consists of low to high peak demand scenarios. The data is collected every second by running WADI non-stop for a total of 16 days. Each data point consists of 103 attributes, among which 67 are continuous sensor readings and the remaining are discrete actuator states.

The system is operated under normal conditions (without any attacks) for a period of 14 days. During the remaining two days, 15 different types of attacks are launched on the testbed. Note that in our experiments, the attacker profile is considered as an insider and he/she has the process, communication knowledge, and access to the communication channels. All the attacks were designed based on the intents of an attacker. The intent of an attacker is specified as a statement: for example, reducing the production capacity of a water treatment plant and cut-off water supply to consumers in a water distribution system, etc. In order to achieve his/her goal, an attacker performs strategic manipulation of sensor measurements and strategic control of actuators. By doing so, an attacker can maintain the system requirements, for example, material balance, but cause the process to move into an abnormal state without a controller knowing that this has happened. The specific types of attacks include overflow of tank, water leakage and stealthy attacks, etc. The attack targets and detailed attack description are shown in Table I.

In our experiments, we split the WADI data log into three parts. The first part which contains the first 12 days of the data is used as the training data log to generate meaningful invariant rules. The second part which contains the 13th and 14th days' data is used as the validation data log for parameter tuning. The last two days' data is used as the test data log on which we use the generated invariant rules to detect anomalies.

C. Experiments on Invariant Rule Generation

In our first experiment, we systematically generate invariant rules from the first 12 days' data log according to our data-driven framework. Specifically, after the predicate generation step, 440 predicates are generated in total, among which 96 are for actuator states, 344 are for sensor readings. Furthermore, among the 344 predicates for sensor readings, 228 are generated by our distribution-driven strategy, the remaining 116 are generated by our event-driven strategy.

In the invariant rule mining step, since the data is collected and processed every second in WADI testbed, as a real time application, we set τ_t the acceptable processing time for each data point to one second. Assuming the acceptable validation error τ_e is set to 1×10^{-4} and 1×10^{-3}, we perform a grid search to find the maximum number of meaningful invariant rules can be generated in both conditions according to Equation 5, with candidate parameter values $\gamma = (0.1, \ldots, 0.9)$, $\theta = (0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64)$ and $\gamma > \theta$. Table II shows the optimal value for the parameters, the corresponding number of meaningful invariant rules generated, and the time cost for checking all the generated rules per data point in our experiments in each condition. As can be seen from the table, even with $\tau_e = 1 \times 10^{-4}$, we still obtain 3259 meaningful invariant rules from the data log. The number is rather significant considering the testbed only has 67 sensors and 36 actuators. With larger acceptable validation error, more invariant rules are generated as expected. Furthermore, the time cost for checking the invariant rules is also rather small comparing with the one second acceptable processing time cost threshold, which means the anomaly detection model with our generated invariant rules is sufficiently fast for real time application.

D. Experiments on Anomaly Detection

In our second experiment, we use the generated meaningful invariant rules to detect anomalies in the last two days' data log. Moreover, to demonstrate the effectiveness of our approach, we also compare our results with two baseline models.

1) Baseline Models: For comparison, we first use the design-based approach to derive invariant rules from the design graphs of the WADI testbed. In total, 22 invariant rules (which is significantly less than the number of invariant rules derived by our data-driven approach) can be derived from the graphs by the help of system operators. Thus, the model which consists of the 22 design-based invariant rules is used as our first baseline anomaly detection model.

The second baseline anomaly detection model is a residual error-based model in which we use a Long Short Term Memory (LSTM) network [38] for sensor readings prediction. Specifically, LSTM network is a class of recurrent neural network which has shown state-of-the-art performance on numerous temporal processing tasks [39], [20]. Here, we use a stacked two layer LSTM network which has 256 hidden units in both hidden layers. Our LSTM network takes the sensor and actuator values at the previous 60 time steps (seconds) as its input, and outputs the sensor readings at the next time step for prediction. All the sensor values are normalised into range [0,1], and all the actuator values are one-hot encoded, e.g., ON, OFF of pump states are encoded into vectors of [1,0] and [0,1], respectively, before feeding into the neural network model. The model is trained to minimize the mean squared error between predicted sensor readings and their real values

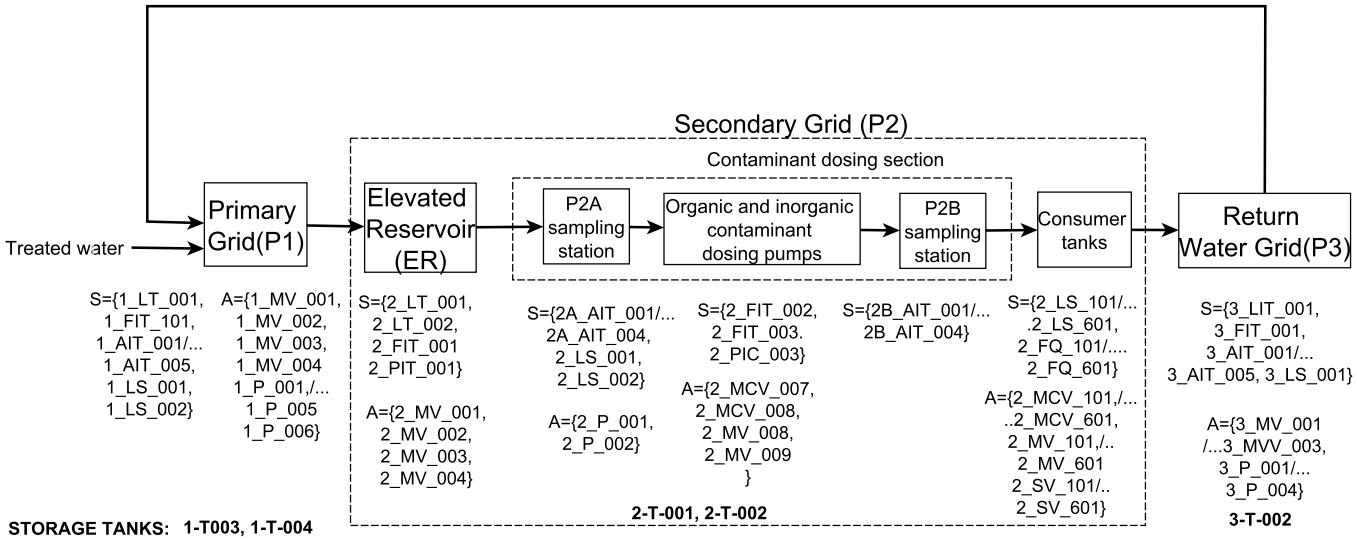

Fig. 3. Three stages of the WADI testbed P1, P2 and P3. Solid arrows indicate flow of water and sequence of processes. S and A represent, respectively, sets of sensors and actuators. Sensors: LT-Level Transmitter, AIT-Analyzer Indication Transmitter, FIT-Flow Indication Transmitter, PIT-Pressure Indication Transmitter, LS-Level Switch. Actuators: P-Pump, MV-Motorized valve, MCV-Modulating Control Valve, SV-Solenoid Valve. Tag name of the instrument is indicated as XXX_YYY_ZZZ, Where XXX, YYY and YYY represent stage number, instrument type and instrument index respectively. For example, 1_LT_001 can be read as stage-P1, level transmitter and the index of level transmitter.

TABLE I. ATTACK TARGETS AND ATTACKER INTENTIONS IN THE WADI CASE STUDY

Attack target(s)	Attacker intention
1_MV_001	Overflow of the raw water tank
1_FIT_001	High dosage of chemical injection in the raw water tank
2_LT_002	Stealthy attack: drain elevated reservoir tank
1_AIT_001	Manipulate raw water conductivity set points
2_MCV_101, 2_MCV_201, 2_MCV_301, 2_MCV_401 2_MCV_501 and 2_MCV_601	Cut-off water supply to consumer tanks
1_AIT_002	Change turbidity set points
2_MV_003	Supply contaminated water to elevated reservoir tank
2_MCV_007	Water leakage from the main pipe line
1_P_005, 1_P_006	Pipe bursts
2_LT_002, 2_LIT_001 and 1_MV_001	Damage 1_MV_001, raw water pump and drain the elevated raw water tank
2_MCV_007	Intermittent water supply to consumer tanks
2_PIC_003	Control of booster pump
1_P_001 and 1_P_003	Chemical dosing stop to the inlet raw water
2_LIT_002	Stealthy attack: Overflow of elevated tank
2_MCV_007	Water wastage
2_MCV_101, 2_MCV_201	Overflow of consumer tanks

TABLE II. THE OPTIMAL VALUE FOR THE PARAMETERS, THE CORRESPONDING NUMBER OF MEANINGFUL INVARIANT RULES GENERATED, AND THE TIME COST FOR CHECKING ALL THE GENERATED RULES PER DATA POINT IN THE WADI CASE STUDY UNDER DIFFERENT ACCEPTABLE VALIDATION ERROR THRESHOLDS

	γ	θ	$N(\gamma,\theta)$	$t(\gamma,\theta)$
$\tau_e = 1 \times 10^{-4}$	0.9	0.16	3259	0.02 sec
$\tau_e = 1 \times 10^{-3}$	0.7	0.04	45847	0.23 sec

on the first 12 days' data log using the Adam optimizer [40]. Then the residual error threshold is tuned by the validation data log to make sure the validation error (the expected false positive rate of detected anomalies) of the model is also below 1×10^{-4} and 1×10^{-3}.

Note that we choose to use the LSTM network as the predictive model because the trained LSTM network model achieves the best prediction accuracy (lowest average residual error) on the validation data log in our experiments compared with the AR model as well as the LDS model. Specifically, the parameters of the AR model is also fitted using the first 12 days' data log, and the value of p (see Section II-B) is tuned to have the lowest residual error on the validation data log. For the LDS model, its parameters are defined either by system

TABLE III. THE PERFORMANCE METRICS OF USING OUR DATA-DRIVEN INVARIANT RULES FOR ANOMALY DETECTION COMPARED WITH TWO BASELINE MODELS IN THE WADI CASE STUDY.

Model	TPR	FPR	NTPR	$P(1)$	$P(3)$	$P(5)$	
Design-based invariant rules	0.4645	0.0060	0.5086	14/15	13/15	11/15	
Residual error -based model	0.1208	0.0003	0.0989	2/15	2/15	2/15	$\tau_e = 1 \times 10^{-4}$
	0.4302	0.0012	0.3545	8/15	7/15	7/15	$\tau_e = 1 \times 10^{-3}$
Data-driven invariant rules	0.4114	0.0002	0.5384	14/15	14/15	14/15	$\tau_e = 1 \times 10^{-4}$
	0.4744	0.0021	0.5552	15/15	15/15	15/15	$\tau_e = 1 \times 10^{-3}$

knowledge or by constructing an Autoregressive Integrated Moving Average Model (ARIMA) model based on the first 12 days' data log.

2) Evaluation Metrics: For each model, to evaluate its effectiveness on detecting anomalies, we first compute three metrics, which are the TPR (True Positive Rate), FPR (False Positive Rate) and NTPR (Normalized True Positive Rate) of the detection result. Specifically, let TP denote the true positives (anomalous data points correctly identified), TN denote true negatives (normal data points correctly identified), FP denote false positives (normal data points incorrectly classified as anomalies), and FN denote false negatives (anomalous data points incorrectly classified as normal packages). The TPR is calculated as $TP/(TP + FN)$, which reflects the fraction of anomalies that are successfully identified. The FPR is computed as $FP/(FP + TN)$, which measures the fraction of normal data points that are misclassified as anomalous by the model. Since the number of attack points for different attack types varies significantly in the dataset, we also calculate $NTPR = \frac{\sum_{i=1}^{N} TPR_i}{N}$ for the measurement of the normalized fraction of anomalies that are successfully identified, where TPR_i is the TPR for the attack type i, N is the total number of attack types in the dataset.

Furthermore, for any given attack type i, we consider this attack type is detectable by a model if $TPR_i > k \times FPR$, where $k \geq 1$, and $k \times FPR$ is a threshold value above which we believe that the attacks of this type are not detected by coincidence. Then, to measure the ability of a model to detect different types of attacks, we also calculate metrics:

$$P(k) = \sum_{i=1}^{N} \frac{\mathbf{1}(TPR_i > k \times FPR)}{N},$$

which captures the fraction of attack types that are detectable by the model. In the experiments, we evaluate $P(1)$, $P(3)$ and $P(5)$.

3) Result and Evaluation: The results of using our generated meaningful invariant rules for detecting anomalies in the the last two days' WADI data log, as well as the results of the two baseline models are given in Table III.

From the table, we can see that our parameter tuning method works well because the distance between the FPR and the validation error (which reflects the expected FPR) is small in both cases. From a learning perspective this means that our anomaly detection model based on data-driven invariant rules can be effectively trained *without* the need for an anomaly tagged dataset. This unsupervised learning capability is of significant practical significance as such tagged anomaly data is not generally available and is difficult to generate. Moreover, we observe that compared with design-based invariant rules, using our data-driven invariant rules to detect anomalies can achieve higher NTPR, but with much smaller FPR when $\tau_e = 1 \times 10^{-4}$. Importantly, when setting $\tau_e = 1 \times 10^{-3}$, the data-driven invariant rules can detect all the attack types in this case study. However, the design-based rules cannot detect one attack type when using $P(1)$ to measure its ability to detect anomalies, and the number of undetected attack types increase to two when $P(3)$ is used, and to four if $P(5)$ is used. Furthermore, we can also see that using our data-driven invariant rules can also achieve significantly better anomaly detection performance than the residual error-based model under similar FPR constraints.

VI. SWAT CASE STUDY

In this section, we present another case study in which we conduct invariant rule generation and anomaly detection experiments on a public ICS data log that is collected from a Secure Water Treatment (SWaT) testbed [41].

A. SWaT Testbed and Data Log

The SWaT testbed is a scaled down water treatment plant which has a six-stage filtration process to purify raw water [24]. Figure 4 represents the six stages of the testbed. Six PLCs working in concert with 24 sensors and 27 actuators are deployed to control the entire treatment process. Specifically, in the first stage, raw water is taken in and stored in a tank. It is then passed to the second stage for pretreatment process, where the conductivity, pH, and Oxidation Reduction Potential (ORP) are measured to determine whether chemical dosing is performed to maintain the water quality within acceptable limits. In the third stage, the Ultra Filtration (UF) system will remove undesirable materials by using fine filtration membranes. This is followed by the fourth stage, where the remaining chlorines are destroyed in the Dechlorinization process using Ultraviolet lamps. Subsequently, the water is pumped into the Reverse Osmosis (RO) system to reduce inorganic impurities in the fifth stage. In the last stage, the clean water from the RO system is stored and ready for distribution.

The SWaT data log is collected by running SWaT non-stop from its empty state to a fully operational state for a total of 11-days, and is originally reported in [41]. During the first 7 days, the plant is operated under normal conditions, i.e. without any attacks. During the remaining four days, 36 different types of attacks which include single stage single point attacks, single stage multi point attacks, multi stage

TABLE IV. THE OPTIMAL VALUE FOR THE PARAMETERS, THE CORRESPONDING NUMBER OF MEANINGFUL INVARIANT RULES GENERATED, AND THE TIME COST FOR CHECKING ALL THE GENERATED RULES PER DATA POINT IN THE SWAT CASE STUDY UNDER DIFFERENT ACCEPTABLE VALIDATION ERROR THRESHOLDS

	γ	θ	$N(\gamma, \theta)$	$t(\gamma, \theta)$
$\tau_e = 1 \times 10^{-4}$	0.9	0.32	5805	0.02 sec
$\tau_e = 1 \times 10^{-3}$	0.9	0.08	17737	0.05 sec

single point attacks, and multi stage multi point attacks on the water treatment process are launched on the SWaT testbed while data collection continued. We refer to [41] for the detailed description of the attack types. The dataset contains all the sensor and actuator values collected every second during the said duration. Each data point consists of 53 attributes, among which 24 are continuous sensor readings and 27 are discrete actuator states.

B. Experiments

In our experiments, we also split the SWaT data log into three parts. The first part which contains the first five days of the data is used as the training data log. The second part which contains the 6th and 7th days' data is used as the validation data log. The last four days' data is used as the test data log on which we use the generated invariant rules to detect anomalies.

1) Experiments on Invariant Rule Generation: In the invariant rule generation experiments, after the predicate generation step, totally 195 predicates are generated, among which 48 are for actuator states, 77 are generated by the distribution-driven strategy, 70 are generated by the event-driven strategy. Furthermore, in the invariant rule mining step, by setting τ_t to one second, we show the optimal value for the parameters (γ and θ), the corresponding number of meaningful invariant rules learned, and the time cost for checking all the learned rules per data point in our experiments in Table IV. All the experiments here are under the same acceptable validation error thresholds and parameter searching space in the previous case study. As can be seen from table, a large number of meaningful invariant rules can be learned from the SWaT data log, and the time costs of checking all the data-driven invariant rules are still rather acceptable for real time anomaly detection in our experiments.

2) Experiments on Anomaly Detection: Regarding to the anomaly detection experiments, we use the same baseline models for comparison. Specifically, 38 design-based invariant rules are available in the SWaT system, and they are used as our first baseline model. The second baseline model is also a residual error-based detection model in which an LSTM network model is used for sensor reading prediction. Table V shows the results of using our generated meaningful invariant rules for detecting anomalies in the last four days' SWaT data log, as well as the results of the two baseline models.

From Table V, we notice that the FPR of using our data-driven invariant rules is still close to their estimation based on the validation error. Furthermore, we observe that the ability of our data-driven invariant rules for detecting anomalies is much better than the design-based invariant rules in this case study. The evidence is the higher TPR and NTPR, lower FPR and the large difference on the metrics $P(1)$, $P(3)$ and $P(5)$ when $\tau_e = 1 \times 10^{-3}$. Even with much lower FPR (when $\tau_e = 1 \times 10^{-4}$), the data-driven invariant rules can still achieve better detection ability compared with the design-based rules. Furthermore, we can also see that using our data-driven invariant rules can again achieve much higher detection rate of anomalies than the residual error-based model under comparable FPRs (e.g., the TPR of the data-driven invariant rules is about 10 times as high as the residual error-based model when $\tau_e = 1 \times 10^{-4}$).

VII. DISCUSSION

We analyze the reasons why using the data-driven invariant rules can achieve better anomaly detection performance than the design-based invariant rules as follows: 1) The noise on the sensor measurements is hard to be captured by the design-based invariant rules, however it is automatically covered by our data-driven approach, thus can reduce the FPR of the data-driven invariant rules; For example, the following design-based invariant rule:

$$1\_\text{LT}\_001 < 60 \Rightarrow 1\_\text{MV}\_004 = \text{OFF}$$

causes 55 false positives in the anomaly detection experiment on the WADI testbed. However, there is a corresponding data-driven invariant rule as follows:

$$1\_\text{LT}\_001 < 59.0399179104 \Rightarrow 1\_\text{MV}\_004 = \text{OFF}$$

which causes zero false positives instead. 2) Our data-driven approach can generate a significantly larger invariant rule set, thus it has more chance to detect anomalies. For example, there is no design-based invariant rule that can reveal attacks targeting on 1_P_005 and 1_P_006 in the WADI case study. However, the following data-driven invariant rule:

$$1\_\text{P}\_002 = \text{OFF} \Rightarrow 1\_\text{P}\_004 = \text{OFF}, 1\_\text{P}\_006 = \text{OFF}$$

can reveal 95.46% attacks points of this type without causing any false positives. 3) The design-based approach only captures the invariant rules between the sensors and actuators within the same or neighboring stages, however, the data-driven approach can capture invariant rules which span several stages, thus is capable to detect anomalies that can only be revealed by looking at the global behavior of the system. Specifically, about 65% of the data-driven invariant rules generated in SWaT case study span non-neighboring stages. As a result, we can see the difference on the anomaly detection performance by using the data-driven invariant rules and the design-based rules is larger in the second case study, where the SWaT testbed has six stages.

The residual error-based model has lower anomaly detection performance in both case studies. This is mainly because there is not a clear boundary between anomalous and normal sensor measurements based on residual error for many attack points in both case studies due to the existence of various sources of sensor noise. In addition, the residual error-based model is rather ineffective for detecting stealthy attacks (attacks which only modify the sensor readings slightly at each time step). However, our invariant rule-based model is able to detect such attacks because the accumulated sensor deviation is highly likely to violate some invariant rules at some specific time point.

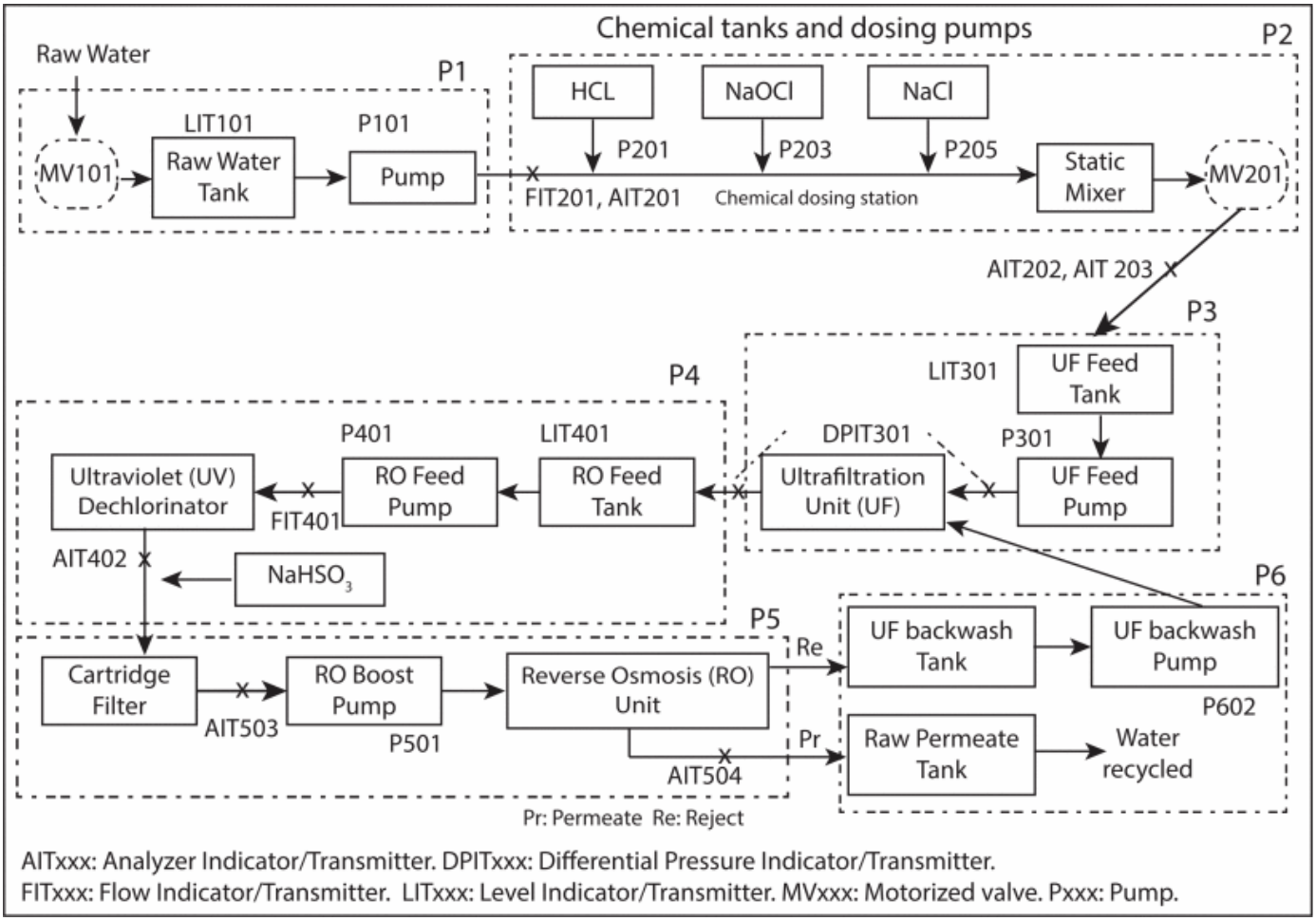

Fig. 4. Six stages of the SWAT testbed.

TABLE V. THE PERFORMANCE METRICS OF USING OUR DATA-DRIVEN INVARIANT RULES FOR ANOMALY DETECTION COMPARED WITH TWO BASELINE MODELS IN THE SWAT CASES STUDY.

Model	TPR	FPR	NTPR	$P(1)$	$P(3)$	$P(5)$	
Design-based invariant rules	0.7589	0.0051	0.3043	18/36	15/36	15/36	
Residual error -based model	0.0730	0.0004	0.0592	6/36	6/36	6/36	$\tau_e = 1 \times 10^{-4}$
	0.6208	0.0057	0.1029	11/36	10/36	9/36	$\tau_e = 1 \times 10^{-3}$
Data-driven invariant rules	0.7087	0.0003	0.296	19/36	15/36	15/36	$\tau_e = 1 \times 10^{-4}$
	0.7881	0.0012	0.4911	33/36	31/36	31/36	$\tau_e = 1 \times 10^{-3}$

We are also aware of that there are still some limitations in our method. For example, we find that there are several actuator state changes for which no event-driven predicates are derived using our current method in the experiments. This is mainly due to the nonlinearity of the corresponding event triggers. Using a more powerful model such as kernel regression can reveal more event-driven predicates, however, this will also result in problems with too many event-driven predicates being generated, and the false positive rate being increased in our experiments. Furthermore, our method also requires a large dataset which covers the operation profile of the ICS to allow the invariant rules to be properly learned. A dataset which only covers a partial profile of the system can potentially lead to a high false positive rate in the detection phase. A potential extension of our work is the methodology of reducing the false positives by filtering generated predicates whilst adding nonlinearity into our model to increase its detection ability.

VIII. RELATED WORK

Log-based anomaly detection via statistical methods has been widely applied to ICS in the recent years [42], [43]. However, most of these approaches either are only applicable to a specific type of systems or require prior domain-specific knowledge about the system to construct the detection model. Nevertheless, our work attempts to construct an anomaly detection model systematically using a purely data-driven approach,

and is potentially generalizable to a wide class of ICS.

The idea of using association rule mining to develop anomaly detection models has been explored in the networked system community. For instance, Mahoney and Chan [44] propose an algorithm known as LERAD that learns rules via an Apriori-like algorithm for finding anomalies in network packets over TCP sessions. Entisar and Zulaiha [45] investigated three association rule mining techniques to develop an intrusion detection system in an information technology center's network traffic. The difficulties to apply association rule mining for detecting anomalies are often the generation of too many rules and their false positives. The problems are firstly largely mitigated in our work where we only generate meaningful invariant rules which are non-redundant and meanwhile reach a *rule-wise* statistical significance. Then, a parameter tuning method is proposed to control the trade-off between number of generated rules and false positives.

Analyzing physical invariants for anomaly detection has been applied to a number of cyber-physical systems [46], [47], [48]. However, all of these invariants are either manually defined or require a large amount of human effort, e.g., to transform sensor values to discrete ranges such as High, Low according to domain knowledge. There has been some work to use machine learning-based methods for discovering physical invariants in cyber-physical systems. For example, Momtazpour et al. [49] conduct anomaly detection by using an ARX (Auto Regression with eXogenous input) model with pre-discovered latent variables to find invariants between wireless sensor data within multiple time steps at Intel Berkeley Research lab. Chen et al. [50] use code mutation programs to generate abnormal data traces, and then use a SVM classifier and statistical model checking to find invariants between sensor data in the SWaT testbed. Nevertheless, the invariant rules generated in our work are more comprehensive than [49], [50] as actuator states which are an important part of the control dynamics in ICS are also included.

IX. Conclusion

In this paper, we have proposed and demonstrated a novel data-driven framework for systematically generating invariant rules from ICS data logs. We have then shown, that such a set of generated invariant rules can be successfully used to detect anomalies in a system to protect the industrial processes under control. We summarize the merits of our approach as follows: (i) It combines several machine learning and data mining techniques, and thus can generate a significant number of invariant rules with very low human effort/input; (ii) It is able to successfully discover invariant rules across several subsystems, largely increasing the difficulty of successful stealthy attack injection; (iii) The false positive rate of using the invariant rules generated by our framework can be effectively controlled by a parameter tuning method based on a validation data log that is free from anomalies; (iv) The generated invariant rules can achieve high anomaly detection performance, which is demonstrated on two real world ICS case studies, and our results outperform standard baseline models including a commonly used residual error-based anomaly detection model; (v) It can be applied to various ICS scenarios as it is dependent only on general control dynamics of ICS.

Furthermore, and perhaps most importantly, we have set out an approach that is highly generalizable to the wider class of cyber-physical systems – of which ICS are instances. As cyber-physical systems become more ubiquitous and commoditized through trends such as IoT (Internet of Things), we see potential for modification, refinement and application of this approach to a range of non-industrial use-cases which may include health-systems, autonomous vehicles and building management systems. We will be examining the feasibility of doing so in future work.

Acknowledgment

Cheng Feng and Deeph Chana were supported by the EPSRC project Security by Design for Interconnected Critical Infrastructures, EP/N020138/1. Venkata Reddy Palleti and Aditya P. Mathur are supported in part by the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2015NCR-NCR003-001) and administered by the National Cybersecurity R&D Directorate. The work was done when Cheng Feng was at the Institute for Security Science and Technology, Imperial College London.

References

[1] ICS-CERT. (2014) Ics-csrt monitor September 2014 - February 2015. "www.ics-cert.us-cert.gov/monitors/ICS-MM201502".

[2] ——. (2015) Incident response activity November 2014 - December 2015. "https://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_Nov-Dec2015_S508C.pdf".

[3] ——. (2016) Ics-csrt monitor November 2016 - December 2016. "https://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_Nov-Dec2016_S508C.pdf".

[4] N. Falliere, L. O. Murchu, and E. Chien, "W32. stuxnet dossier," *White paper, Symantec Corp., Security Response*, vol. 5, 2011.

[5] R. Lee, M. Assante, and T. Connway, "ICS cyber-to-physical or process effects case study paper–german steel mill cyber attack," *Sans ICS, Dec*, 2014.

[6] ICS-CERT. (2016) Cyber-attack against Ukrainian critical infrastructure. "www.ics-cert.us-cert.gov/alerts/IR-ALERT-H-16-056-01".

[7] D. Formby, P. Srinivasan, A. Leonard, J. Rogers, and R. A. Beyah, "Who's in control of your control system? device fingerprinting for cyber-physical systems," in *23rd Annual Network and Distributed System Security Symposium, (NDSS 2016), San Diego, California, USA, February 21-24*, 2016.

[8] S. E. McLaughlin, S. A. Zonouz, D. J. Pohly, and P. D. McDaniel, "A trusted safety verifier for process controller code," in *21st Annual Network and Distributed System Security Symposium, (NDSS 2014), San Diego, California, USA, February 23-26*, 2014.

[9] S. Zonouz, J. Rrushi, and S. McLaughlin, "Detecting industrial control malware using automated plc code analytics," *IEEE Security & Privacy*, vol. 12, no. 6, pp. 40–47, 2014.

[10] L. Cheng, K. Tian, and D. D. Yao, "Orpheus: Enforcing cyber-physical execution semantics to defend against data-oriented attacks," in *Proc. Annual Computer Security Applications Conference (ACSAC 2017). ACM*, 2017, pp. 315–326.

[11] M. Caselli, E. Zambon, and F. Kargl, "Sequence-aware intrusion detection in industrial control systems," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*. ACM, 2015, pp. 13–24.

[12] C. Feng, T. Li, and D. Chana, "Multi-level anomaly detection in industrial control systems via package signatures and lstm networks," in *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2017)*. IEEE, 2017, pp. 261–272.

[13] N. Goldenberg and A. Wool, "Accurate modeling of modbus/tcp for intrusion detection in scada systems," *International Journal of Critical Infrastructure Protection*, vol. 6, no. 2, pp. 63–75, 2013.

[14] M.-K. Yoon and G. F. Ciocarlie, "Communication pattern monitoring: Improving the utility of anomaly detection for industrial control systems," in *NDSS Workshop on Security of Emerging Networking Technologies*, 2014.

[15] A. Kleinmann and A. Wool, "A statechart-based anomaly detection model for multi-threaded scada systems," in *International Conference on Critical Information Infrastructures Security*. Springer, 2015, pp. 132–144.

[16] I. N. Fovino, A. Carcano, T. D. L. Murel, A. Trombetta, and M. Masera, "Modbus/dnp3 state-based intrusion detection system," in *24th IEEE International Conference on Advanced Information Networking and Applications (AINA 2010)*. IEEE, 2010, pp. 729–736.

[17] D. Hadžiosmanović, R. Sommer, E. Zambon, and P. H. Hartel, "Through the eye of the plc: semantic security monitoring for industrial processes," in *Proceedings of the 30th Annual Computer Security Applications Conference*. ACM, 2014, pp. 126–135.

[18] A. Abur and A. G. Exposito, *Power system state estimation: theory and implementation*. CRC press, 2004.

[19] D. I. Urbina, J. A. Giraldo, A. A. Cardenas, N. O. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, and H. Sandberg, "Limiting the impact of stealthy attacks on industrial control systems," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1092–1105.

[20] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, "Anomaly detection in cyber physical systems using recurrent neural networks," in *IEEE 18th International Symposium on High Assurance Systems Engineering (HASE 2017)*. IEEE, 2017, pp. 140–145.

[21] G. Dan and H. Sandberg, "Stealth attacks and protection schemes for state estimators in power systems," in *First IEEE International Conference on Smart Grid Communications (SmartGridComm 2010)*. IEEE, 2010, pp. 214–219.

[22] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 1, p. 13, 2011.

[23] C. Feng, T. Li, Z. Zhu, and D. Chana, "A deep learning-based framework for conducting stealthy attacks in industrial control systems," *arXiv preprint arXiv:1709.06397*, 2017.

[24] S. Adepu and A. Mathur, "Using process invariants to detect cyber attacks on a water treatment system," in *IFIP International Information Security and Privacy Conference*. Springer, 2016, pp. 91–104.

[25] ——, "From design to invariants: Detecting attacks on cyber physical systems," in *IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C 2017)*. IEEE, 2017, pp. 533–540.

[26] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[27] C. Keribin, "Consistent estimation of the order of mixture models," *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 49–66, 2000.

[28] R. Tibshirani, "Regression shrinkage and selection via the lasso: a retrospective," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 3, pp. 273–282, 2011.

[29] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Acm sigmod record*, vol. 22, no. 2. ACM, 1993, pp. 207–216.

[30] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining—a general survey and comparison," *ACM sigkdd explorations newsletter*, vol. 2, no. 1, pp. 58–64, 2000.

[31] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Discovering frequent closed itemsets for association rules," in *International Conference on Database Theory*. Springer, 1999, pp. 398–416.

[32] T. Uno, M. Kiyomi, and H. Arimura, "Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets," in *Fimi*, vol. 126, 2004.

[33] M. J. Zaki and C.-J. Hsiao, "Charm: An efficient algorithm for closed itemset mining," in *Proceedings of the 2002 SIAM international conference on data mining*. SIAM, 2002, pp. 457–473.

[34] G. Grahne and J. Zhu, "Fast algorithms for frequent itemset mining using fp-trees," *IEEE transactions on knowledge and data engineering*, vol. 17, no. 10, pp. 1347–1362, 2005.

[35] B. Liu, W. Hsu, and Y. Ma, "Mining association rules with multiple minimum supports," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 337–341.

[36] Y.-H. Hu and Y.-L. Chen, "Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism," *Decision Support Systems*, vol. 42, no. 1, pp. 1–24, 2006.

[37] R. U. Kiran and P. K. Reddy, "Novel techniques to reduce search space in multiple minimum supports-based frequent pattern mining algorithms," in *Proceedings of the 14th international conference on extending database technology*. ACM, 2011, pp. 11–20.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[39] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, vol. 3. IEEE, 2000, pp. 189–194.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[41] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *International Conference on Critical Information Infrastructures Security*. Springer, 2016.

[42] S. Pan, T. Morris, and U. Adhikari, "Developing a hybrid intrusion detection system using data mining for power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[43] Y. Harada, Y. Yamagata, O. Mizuno, and E.-H. Choi, "Log-based anomaly detection of cps using a statistical method," in *8th International Workshop on Empirical Software Engineering in Practice (IWESEP 2017)*. IEEE, 2017, pp. 1–6.

[44] M. V. Mahoney and P. K. Chan, "Learning rules for anomaly detection of hostile network traffic," in *Third IEEE International Conference on Data Mining (ICDM 2003)*. IEEE, 2003, pp. 601–604.

[45] E. E. Eljadi and Z. A. Othman, "Anomaly detection for ptm's network traffic using association rule," in *2011 3rd Conference on Data Mining and Optimization (DMO 2011)*. IEEE, 2011, pp. 63–69.

[46] A. Choudhari, H. Ramaprasad, T. Paul, J. W. Kimball, M. Zawodniok, B. McMillin, and S. Chellappan, "Stability of a cyber-physical smart grid system using cooperating invariants," in *37th Annual Computer Software and Applications Conference (COMPSAC 2013)*. IEEE, 2013, pp. 760–769.

[47] T. Paul, J. W. Kimball, M. Zawodniok, T. P. Roth, B. McMillin, and S. Chellappan, "Unified invariants for cyber-physical switched system stability," *IEEE Transactions on Smart Grid*, vol. 5, no. 1, pp. 112–120, 2014.

[48] K. Pal, S. Adepu, and J. Goh, "Effectiveness of association rules mining for invariants generation in cyber-physical systems," in *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE 2017)*. IEEE, 2017, pp. 124–127.

[49] M. Momtazpour, J. Zhang, S. Rahman, R. Sharma, and N. Ramakrishnan, "Analyzing invariants in cyber-physical systems using latent factor regression," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 2009–2018.

[50] Y. Chen, C. M. Poskitt, and J. Sun, "Learning from mutants: Using code mutation to learn and monitor invariants of a cyber-physical system," *arXiv preprint arXiv:1801.00903*, 2018.