TABOR: A Graphical Model Based Framework for Anomaly Detection in Industrial Control Systems

Qin Lin
Delft University of Technology
q.lin@tudelft.nl

Sridha Adepu
Singapore University of Technology and Design
adepu_sridhar@sutd.sg

Sicco Verwer
Delft University of Technology
s.e.vervwer@tudelft.nl

Aditya Mathur
Singapore University of Technology and Design
aditya_mathur@sutd.sg

ABSTRACT

Industrial Control Systems (ICS) such as water and power are critical to any society. Process anomaly detection mechanisms have been proposed to protect such systems to minimize the risk of damage or loss of resources. In this paper, a graphical model based framework is proposed for profiling normal operational behavior of an operational ICS referred to as SWaT (Secure Water Treatment). Timed automata are learned as a model of regular behaviors shown in sensors signal like fluctuations of water level in tanks. Bayesian networks are learned to discover dependencies between sensors and actuators. The models are used as a one-class classifier for process anomaly detection, recognizing irregular behavioral patterns and dependencies. The detection results can be interpreted and the abnormal sensors or actuators localized due to the interpretability of the graphical models. This approach is applied to a dataset collected from SWaT. Experimental results demonstrate the model's superior performance on both precision and run-time over methods including support vector machine and deep neural networks. The underlying idea is generic and applicable to other industrial control systems such as power and transportation.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; **Intrusion detection systems**;

KEYWORDS

Cyber-physical system; anomaly detection; timed automata; Bayesian network; industrial control systems; SCADA security

ACM Reference Format:
Qin Lin, Sridha Adepu, Sicco Verwer, and Aditya Mathur. 2018. TABOR: A Graphical Model Based Framework for Anomaly Detection in Industrial Control Systems. In *Proceedings of ACM Asia Conference on Computer and Communications Security (ASIACCS'18).* ACM, New York, NY, USA, 12 pages. https://doi.org/http://dx.doi.org/10.1145/XXXXXX.XXXXXX

ASIACCS'18, June 4–8, 2018, Songdo, Incheon, Korea

ACM ISBN ISBN 978-1-4503-5576-6/18/06…$15.00
https://doi.org/http://dx.doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

The protection of industrial control systems (ICS) [17, 37] for public infrastructure such as power, water treatment, and transportation systems is of utmost importance due to significant damage a potential attack may cause. Often these systems are vulnerable to attacks due to the presence of cyber components such as Supervisory Control and Data Acquisition (SCADA) workstations, Human Machine Interface (HMI), Programmable Logic Controllers (PLCs) and the underlying communications network. Attacks are a result of exploitation of one or more vulnerabilities [45] in an ICS. Such vulnerabilities might be due to lack of access control in the system [4]. Software vulnerabilities could be in the PLCs, SCADA software systems, and weaknesses in the communication channels. The compromise or destruction of an ICS would impact society in far reaching ways. For instance, a blackout [28] caused by an attack targeted at a power system ICS would cause monetary losses to all the people served and businesses. Moreover such an attack could cause cascading failures [25], harming large communities such as entire cities. Attacks on ICS can have a significant impact depending on the type of attack and its location. The increase in successful cyber attacks on ICS [10, 28, 44], and many unsuccessful attempts [16], points to the importance of research in the security of ICS with the goal of making it resilient to cyber attacks. Attacks on ICS are increasing each year and perhaps leading towards cyber warfare with critical infrastructure as key targets. In this paper, we aim at detecting intrusions by only observing the physical process under the control of an ICS.

Existing approaches dealing with cyber attacks detection in cyber physical systems (CPS)[1] include signature based detection [13, 34], verification [9, 47], behavior specification [3], and machine learning [14, 21]. Signature based methods require an up-to-date signature dictionary of all known attacks, which is becoming increasingly infeasible due to the growing number of unknown threats. Verification methods basically use a formal model to test on a source code level whether certain signals show large deviations from the values specified in the system's design. Although powerful, full verification based on the source code is often infeasible due to the state-explosion problem, in which the resulting model becomes too large to analyze. Behavior specification-based methods require a precise understanding of how the CPS behaves. Such knowledge can be expensive to obtain. Once obtained, however, it can detect many attacks because it uses detailed models of the underlying

[1] ICS,CPS used interchangeably in all over the paper

physical processes. This approach is sensitive to noise caused by dynamic operating environments, aging or other evolvements of the facility in question, and inaccuracies/incompleteness of source documents such as operation manuals [21]. Most existing machine learning approaches focus on detecting anomalies in feature space, i.e., looking at data points with large deviation from normal space. These require little system knowledge and can detect a large range of attacks. A significant shortcoming of the currently applied machine learning methods is that they provide little insight into the system and no explanation of detection results.

In this paper, we attempt to respond to two key challenges in applying machine learning techniques in the context of a CPS. First: Can we explain the outcome of attack detection, i.e. *why* is this an anomaly? Second: Can we *localize* the anomaly, i.e. which sensors and actuators are potentially under attack. These two questions are of importance for operators who need to diagnose the abnormal behavior and to undertake one of possibly many follow-up safety actions. To deal with the aforementioned problems, an insightful graphical framework (Time Automata and Bayesian netwORk–TABOR) is learned from the normal operational observation of an ICS. The method used in TABOR is illustrated in Figure 1 using a flowchart.

Subprocesses of the entire ICS are modeled. Sets of sensors and actuators in the ICS are partitioned into groups based on their functionalities in order to deal with high dimension and complexity of the problem. Signals from the sensors are symbolically represented and learned using timed automata (TA) to discover the underlying dynamical fluctuating behavior of the water level and other sensors. The states in the TA are associated with other actuator's states by dependency/causality inference using the Bayesian network. Irregular patterns and dependencies that do not adhere to the learned model from normal behavior, are considered anomalies. The contributions of this paper are listed as follows:

(1) The proposed model provides a solution for the interpretation and localization of anomalies. The detected anomalous patterns can be located precisely to process, sensors, or actuators. The model is visualizable and interpretable, thus enabling a better understanding of the system and verification of the model itself.
(2) More attack scenarios are successfully detected compared to those detected using methods based on deep neural network (DNN) and the support vector machine (SVM) available in the literature.
(3) To the best of our knowledge, this is first work to combine timed automata learning and Bayesian network inference for anomaly detection in a CPS. Techniques used here are not limited to a water plant but also applicable to other CPSs

The reminder of this paper is organized as follows. Related work is discussed in Section 2. The dataset and the attack scenarios are briefly explained in Section 3. The proposed method is discussed in Sections 4 and 5. Analysis of data from the experiments is in Section 6. Conclusive remarks and future work are in Section 7.

2 RELATED WORK

The study reported here focuses on cyber attacks on CPS that result in deliberate sensor and actuator data manipulation. There exist several techniques for detecting process anomalies in CPS. These anomalies might happen due to sensor and actuator manipulation in communication channels. Researchers have presented challenges in safety and security against cyber attacks that need to be addressed while designing a CPS. In [7] the authors have presented evolution of Industrial Control Systems (ICS) to emerging CPS with the use of ICT technologies, and potential vulnerabilities and design challenges. Lee [27] presented problems in computing and network technologies for full fledged design of emerging CPS.

CPAC [12] presents a stateful detection mechanisms to detect attacks against control systems. The Weaselboard [32] uses PLC backplane to get the sensor data and and actuator commands, and analyses them to prevent zero day vulnerabilities. WeaselBoard [32] has a dedicated device, and detects changes in control settings, sensor values, configuration information, firmware, logic, etc. In [36], it is shown how safety critical systems are interconnected and their complexity. Model based attack detection schemes in water distribution systems was presented in [5]. It uses Matlab identification tool to get a model from the data generated in a water distribution system. The data driven model is helpful in detecting process anomalies. Cardenas et al. [39] have experimented with the use of CUSUM in detecting in stealthy attacks. The research on attack detection in ICS is increasing and monitoring the physics of the ICS to detect attacks is also a growing research area. A water control system was modeled using an autoregressive model and a detector [15] which track the process variables. Liu et al. presented false data attacks in a power grid [29, 30], state estimation and intelligent attacks against a state estimation. Response and detection were investigated on attacks against chemical plant [8].

The RNN is one of the machine learning approaches used for anomaly detection in the SWaT system [14]. However, due to the expensive training time–one-week, they only considered the first out of the six stages, of the system. In addition, only 10 attack scenarios were used for the evaluation. Recently, as a follow-up work, the DNN and the one-class SVM models have been applied for anomaly detection in the SWaT system [18]. All stages and attack scenarios are considered in this work. Due to the comprehensiveness of this work and the similarity between RNN and DNN, this work is used for comparison with the proposed model [18].

Formal methods are also powerful tools for specification mining in the CPS. They usually covers *signal level* and *code level* verifications. The code level verification is not related to the research problem in this work. While the signal level verification aims at discovering signal rules e.g., Signal Temporal Logic (STL) formulas [20] from the normal behaviors of the CPS. However, due to the high complexity of the work, only some simulation cases are considered in their work, thus not suitable to deal with the high-dimensional data in the SWaT system. Another main difference lies on the fact that the proposed model is actually a passive grammar learning approach and treats signal using a symbolic representation in the pre-processing step.

Both of the proposed grammar-based and rule-based methods are possible candidates to enrich the invariants [3] in the CPS. They both are essentially "specification mining" techniques offering an interesting research line of combining statistical machine learning and verification.

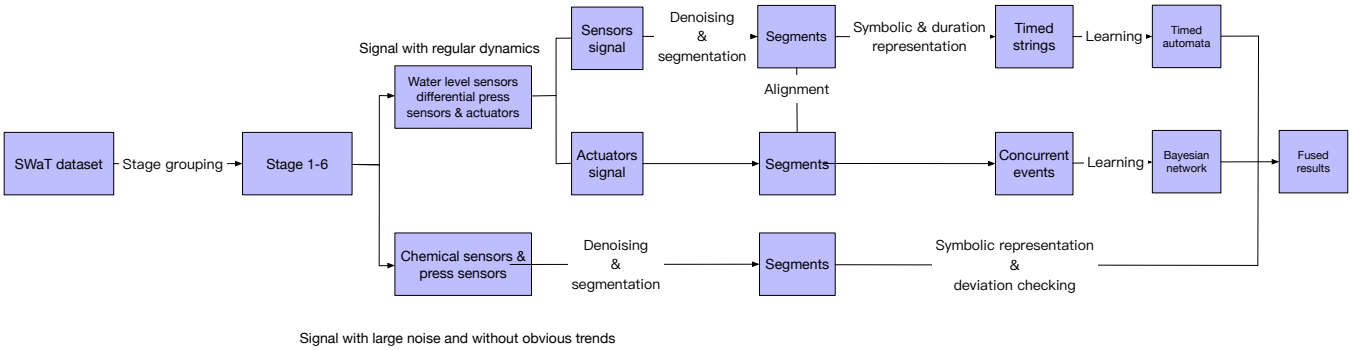


Figure 1: Flowchart of TABOR. The sensors and actuators in SWaT are divided into sub-models due to different stages and functionalities. The behaviors of water level and differential press signal show more regular patterns in the dataset, thus they are learned using time automata. The smallest unit for anomaly detection is one segment, which is considered as an event. The general system alarms an instruction when any sub-model detects anomalies.

Timed automata have been used in the discrete event system domain for the model-based diagnosis [6, 38]. However, it is usually assumed that the plant and specification model are already obtained. Our work combines model identification and anomaly detection in a whole framework. In [31, 33, 43] (hybrid) timed automaton is learned and used for the anomaly detection in production systems. The authors use a bottom-up strategy for timed automaton learning, which is one key difference with our approach. Moreover, signals from sensor and actuator are mixed up and represented as events in their approach, which leads to a states blow-up problem and difficulty of localize the abnormal sensor/actuator. In the proposed work, a Bayesian network is additionally learned to discover the dependencies between the sensors and the actuators in SWaT.

3 INTRODUCTION TO SWAT AND THE DATASET

SWaT is a scaled down water treatment plant with a small footprint that produces 5 gallons/minute of doubly filtered water. This testbed replicates large modern plants for water treatment such as those found in cities. SWaT has six sub-processes, referred to as *stages*, controlled by six PLCs, as shown in Figure 2 [2].

Architecture of SWaT is well introduced in the literature [2]. Here we recapitulate functions of six sub-processes. Stage P1 controls the inflow of raw water to be treated by opening or closing a motorized valve. The raw water tank is treated in the chemical dosing station (stage P2) then to another UF (Ultra Filtration) feed water tank in stage P3. A UF feed pump in P3 sends water via UF unit to RO (Reverse Osmosis) feed water tank in stage P4. Here an RO feed pump sends water through an ultraviolet dechlorination unit controlled by a PLC in stage P4. This step is necessary to remove any free chlorine from the water prior to passing it through the reverse osmosis unit in stage P5. Sodium bi-sulphate (NaHSO3) can be added in stage P4 to control the ORP (Oxidation Reduction Potential). In stage P5, the dechlorinated water is passed through a 2-stage RO filtration unit. The filtered water from the RO unit is stored in the permeate tank and the reject in the UF backwash tank. Stage P6 controls the cleaning of the membranes in the UF unit by turning on or off the UF backwash pump. The backwash cycle is initiated automatically once every 30 minutes and takes less

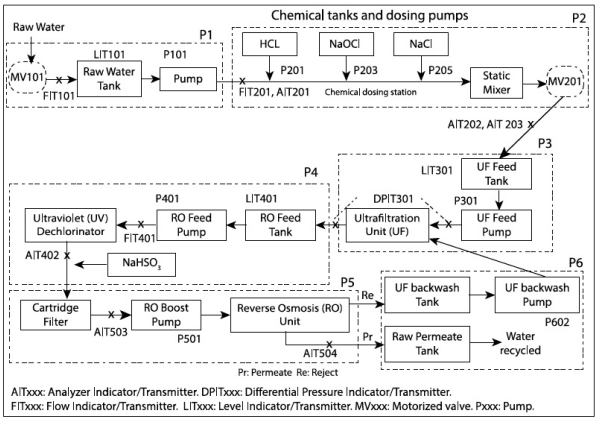


Figure 2: SWaT system diagram. The functionality of each stage is as follows: P1: Raw water supply and storage. P2: Pre-treatment via chemical dosing. P3: Ultrafiltration (UF) and backwash. P4: De-Chlorination system. P5: Reverse osmosis (RO). P6: RO permeate transfer, UF backwash and cleaning.

than a minute to complete. A backwash cycle is also initiated if the pressure drop exceeds 0.4 bar which indicates that the membranes in the UF unit are clogged and need to be cleaned. A differential pressure sensor at stage P3 is used by PLC-3 to obtain the pressure drop across the UF unit.

SWaT dataset [19] was collected over 11 days of continuous operation. The first 7 days of data were collected under normal operation (without any attacks) while the remaining 4 days of data were collected with 36 designed attack scenarios. All network traffic, physical (sensor and actuator) data were collected. In this paper, we focus on the detection of physical process; network traffic data are ignored. The duration of physical recording is from 22/12/2015 4:00:00 PM to 2/1/2016 2:59:59 PM. The dataset contains a total of 53 columns: 1 for *timestamp*, 1 for *label* ('Attack' and 'Normal'), and the remaining 51 are numeric recordings of 51 sensors and actuators. Note that physical data is equally sampled every second.

The description of all 36 attack scenarios can be found on the website [2] [19].

3.1 Attack Scenarios

Attacks in the attack dataset were generated based on scenarios reported earlier [2, 3]. The attack model is a generalized model [1] for cyber physical systems with an intent space of an attacker. The attack duration depends on the kind of attack and attacker intent. The duration of each attack varies from 101 seconds to 10 hours. Some attacks are consecutively within a 10-minute gap, while others are performed by leaving time for the system to stabilize. 36 attacks were launched during the data collection process. Based on attack points in each stage, the attacks are divided into: 26 Single Stage Single Point (SSSP) attacks performed on exactly one point in a CPS; 4 Single Stage Multi Point (SSMP) attacks on two or more points but on only one stage; 2 Multi Stage Single Point (MSSP) attacks and 4 Multi Stage Multi Point (MSMP) attacks performed on two or more stages.

All attacks are performed by injecting the process variable values into a Programmable Logic Controller (PLC) leading each PLC into believing that the sensor information received is genuine and is not a spoofed value. Some attacks are stealthy [19] where an attacker changes the sensor values slowly with respect to the process behaviour; other attacks are random in which sensor values shift randomly. A detailed description of the threat model is in [22]. Figure 3 shows an example of an attack in the De-Chlorination stage (P4).

4 SIGNAL PROCESSING

This section discusses the pre-processing procedure dealing with the high-dimension and noisy signal in the SWaT system. In Table 1, groups of sensors and actuators are split locally in six stages of the system. For sensors labeled LITxxx that measure water levels and those labeled DPIT that measure differential pressure, the sequential behavior, as well as their dependencies on the actuators in the same stage, are learned because of the relatively obvious regular patterns. The signals from the sensors AIT and PIT with large noise and subtle trends are checked only using a model-free approach, i.e. by examining their values and the thresholds. The data implies that the differencing effect of DPIT makes the time series of PIT signal stationary. Note that several sensors and actuators in P6 are not used in the work reported here because they are not completely used for data collection in SWaT yet. only the first five stages are considered in this work. In addition the dataset used does not contain any attacks on stage 6.

4.1 Denoising

The sensor signal has been already denoised by a hard filter in each stage. However, pikes are still observed and thus pose a challenge to the following learning procedure. The one-dimensional time series of a sensor signal is defined as:

$$\mathbf{x}[n] = [x_1, x_2, \cdots, x_n] \tag{1}$$

[2] https://itrust.sutd.edu.sg/dataset/

Table 1: Sub-model Split. FIT is simply treated as actuator with two states: closed and open.

Model Number	Stage Number	Sensor	Actuator
1	1	LIT101	MV101, FIT101, P101, P102
2	2	AIT201	FIT201, MV201, P201
3	2	AIT202	FIT201, MV201, P201, P203, P205
4	2	AIT203	FIT201, MV201, P201, P203, P205
5	3	DPIT301	FIT301, MV301, MV302, MV303, MV304, P302
6	3	LIT301	FIT301, MV301, MV302, MV303, MV304, P302
7	4	AIT401	FIT401,P-402,P-403,UV-401
8	4	AIT402	FIT401,P-402,P-403,UV-401
9	4	LIT401	FIT401,P-402,P-403,UV-401
10	5	AIT501	FIT501,FIT502,FIT503,FIT504,P501
11	5	AIT502	FIT501,FIT502,FIT503,FIT504,P501
12	5	AIT503	FIT501,FIT502,FIT503,FIT504,P501
13	5	AIT504	FIT501,FIT502,FIT503,FIT504,P501
14	5	PIT501	FIT501,FIT502,FIT503,FIT504,P501
15	5	PIT502	FIT501,FIT502,FIT503,FIT504,P501
16	5	PIT503	FIT501,FIT502,FIT503,FIT504,P501

A naive averaging filter is applied in this paper for denoising. The denoised time series is defined as

$$\bar{\mathbf{x}}[w] = [\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_w] \tag{2}$$

The ith element of $\bar{\mathbf{x}}$ is calculated by:

$$\bar{x}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} x_j \tag{3}$$

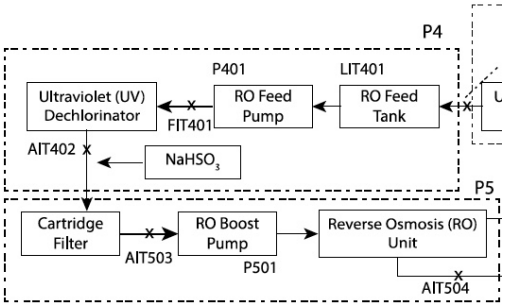
For simplicity and clarity, it is assumed that n is divisible by w. If not, it can be simply modified by adding an additional chunk appending to $\bar{\mathbf{x}}$ averaging the remainders in $\mathbf{x}$. The original and the denoised signal are shown in Figure 4.

4.2 Segmentation

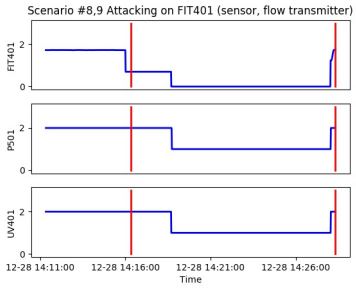
Representation is key to efficient and effective solutions to time series data mining. As one of the most commonly used preprocessing method, piecewise linear representation (PLR) has been used by various researchers to support clustering, classification, indexing and association rule mining of time series data [23]. In this paper, a Sliding WIndow based on Differential sEgmentation (SWIDE) algorithm is used for the piecewise linear approximation of the sensor signal. Pseudo code of SWIDE is shown in Algorithm 1. A segmented sensor signal is shown in Figure 5.

4.3 Alignment

A quantile clustering algorithm is used for the discretization and the symbolic representation of the sensor signal, i.e. letting each bin have equal frequency. The inputs to the clustering algorithm are



(a)



(b)

Figure 3: An example of sensor attack on SWaT. Water from the RO tank is sent via an ultraviolet (UV) and a cartridge filter to the next stage (P5). Flow meter FIT401 indicates the flow rate of water from the RO tank to through the UV. An attack (the starting and ending time are indicated via timestamp of two red bars in Figure 3b) manipulates the real value (around 2 m^3/h) to 0.7 m^3/h then 0 m^3/h. Actuator UV and Pump 501 are turned off to lead the PLC into believing that there is no water transmitted from the RO feed tank. This subsequently leads to overflow in the RO tank.

Algorithm 1 SWIDE algorithm:

Require: Denoised time series data $\bar{\mathbf{x}}$, max_error ϵ
Ensure: PLR with K segments Seg
  $anchor = 1$
  $diff\_seg\_mean = 0$
  while not finished segmenting time series **do**
    $i = 2$
    while $abs(\bar{x}_i - \bar{x}_{i-1})$−diff_seg_mean) $\leq \epsilon$ **do**
      $i = i + 1$
      $diff\_seg\_mean = update\_diff\_mean(\bar{x}[anchor : anchor + (i-1)])$ ▷ recalculate the mean differential value
    end while
    $Seg = concat(Seg, create\_segment(\bar{x}[anchor : anchor + (i-1)])$ ▷ add this segment
    $anchor = anchor + i$
  end while

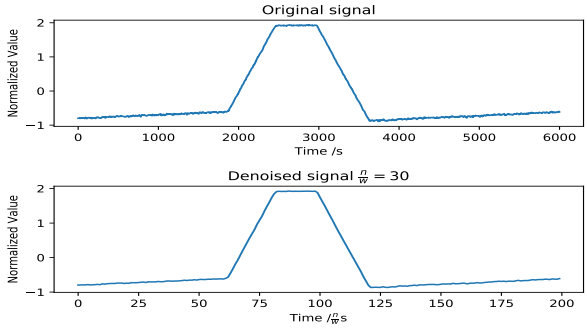


Figure 4: Denoising by an averaging processing.

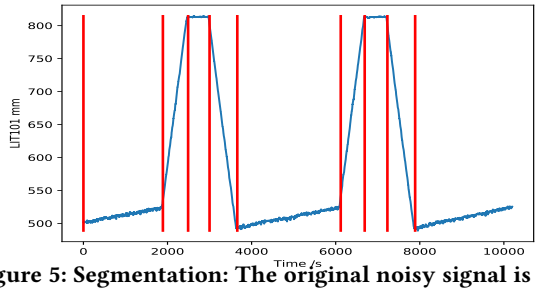


Figure 5: Segmentation: The original noisy signal is shown; the denoised signal is used as input to SWIDE for obtaining more robust segmentation results.

the differential values of the segments $diff\_seg\_mean$ obtained in the segmentation step. The subplot of LIT101 in Figure 6 shows the discretized signal represented as four letters. The number of clusters is set by looking at the trends in the training data. The trends of slow up (SU), quick up (QU), staying constant (SC), and quick down (QD) are obviously visible and interpreted. They are interpretated by human beings once the number of cluster and the average value in each cluster are determined. Such a representation in natural language is straightforward to boost the interpretation of the model. Meanwhile the corresponding status of the actuators is obtained using the timestamps from the sensor signal's segments. Figure 6 shows the alignment of the sensors and the actuators in P1. The durations of events are implicitly represented along with events into timed strings, which are fed to the timed automata learning algorithm. The concurrent events from the aligned sensors and actuators values are input to Bayesian network learning.

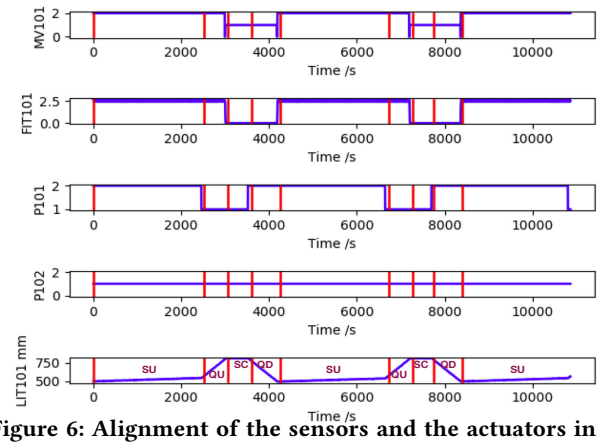


Figure 6: Alignment of the sensors and the actuators in P1 based on segmentation and clustering results. The timed strings of sensor is: (SU,2520) (QU,540) (SC,540) (QD,660) (SU,2460) (QU,540) (SC,480) (QD,660) (SU,2460). The possible misplacement of segments is due to the noise in the original signal.

5 TABOR LEARNING

Timed automaton, Bayesian network, and their learning algorithms are explained in this section. The input to timed automata learning procedure consists of sentences of timed strings. One sentence consists of two full cycles to better capture any looping behavior in the state machine. Two consecutive sentences have one cycle as overlapping. The input to the Bayesian network learning procedure consists of just the data points of concurrent events from the alignment of the sensors and actuators.

5.1 Probabilistic Deterministic Real Timed Automaton

Here we introduce the probabilistic deterministic finite automaton (PDFA), which is more commonly used in practice, and then move to the probabilistic deterministic real time automaton (PDRTA), which is the model used in this work. The PDFA defined in Definition 1 is a generic model for discrete events (similar to a Hidden Markov Model).

Definition 1. *A PDFA is a 5-tuple $\langle Q, \Sigma, \delta, \pi, q_0 \rangle$, where Q is a finite set of states, Σ is a finite alphabet of observable symbols (events), $\delta : Q \times \Sigma \rightarrow Q$ is the transition function from a state-symbol pair to the next state, $\pi : Q \times \Sigma \rightarrow [0, 1]$ is the probability of the emitted symbol given a state, q_0 is the initial state.*

Sequences of symbols translate to paths over states starting from the initial state q_0. The probability of such a sequence is obtained by multiplying all the state-symbol probabilities along such a path. Time information is also relevant in many real-world applications of automata. The timing of actions, or lifetime, is important for characterizing behaviors which is also considered as a feature for anomaly analysis in this paper. An algorithm for efficient learning of timed automata algorithm was proposed in Ref. [40, 42]. This algorithm uses an *explicit* representation of such time constraints. Discrete events are represented by timed strings $(a_1, t_1)(a_2, t_2) \cdots (a_n, t_n)$, where a_i is a discrete event occurring with t_i time delay since the $(i-1)$th event. In this paper, t_i is the duration of each event a_i. A PDRTA is formally defined as follows.

Definition 2. *A PDRTA is a 4-tuple $\langle \mathcal{A}, \mathcal{E}, \mathcal{T}, \mathcal{H} \rangle$, where $\mathcal{A} = \langle Q, \Sigma, \Delta, q_0 \rangle$ is a 4-tuple defining the machine structure: Q is a finite set of states, Σ is a finite alphabet, Δ is a finite set of transitions, and $q_0 \in Q$ is the initial state. $\mathcal{E}$ and $\mathcal{T}$ are the event and time probability distributions, respectively. $\mathcal{E} : Q \times \Sigma \rightarrow [0, 1]$ returns the probability of generating/observing a given event in a given state. $\mathcal{T} : Q \times \mathcal{H} \rightarrow [0, 1]$ returns the same but for a given time range $[m, m'] \in \mathcal{H}$, where $\mathcal{H}$ is a finite set of non-overlapping intervals in $\mathbb{R}_+$. A transition $\delta \in \Delta$ in a PDRTA is a tuple $\langle q, q', a, [m, m'] \rangle$, where $q, q' \in Q$ are the source and target states, $a \in \Sigma$ is a symbol and $[m, m']$ is a temporal guard.*

In a PDFA and a PDRTA, the states are *latent variables* that cannot be directly observed in strings, but have to be estimated by using a learning method. The state transition in a PDFA is triggered only by an event. However, in a PDRTA, it is triggered when both an event and its timing are validated (inside a time range/guard). Therefore, a PDRTA is essentially a timed variant of a PDFA.

5.2 Learning PDRTA

A state-of-the-art machine learning algorithm named RTI+ is used to learn driver behaviors from unlabeled data [41]. Traditional state machine learning algorithm starts by building a large tree-shaped automaton called augmented prefix tree acceptor (APTA) from a sample of input strings. Every state of this tree can be reached by exactly one untimed string and therefore encodes exactly the input sample. For timed automaton learning, the initial values of the lower and upper bounds of all time guards are set to be the minimum t_{min} and maximum t_{max} time values from the input samples S. Figure 7 illustrates a timed APTA (TAPTA) from timed strings (a modified example from Ref. [41]).

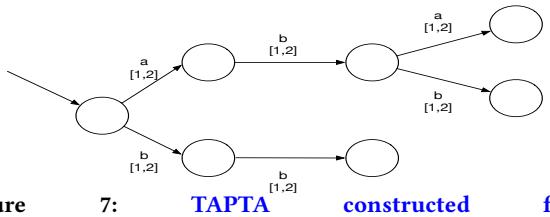


Figure 7: TAPTA constructed from the timed input sample: $S = (a, 1), (a, 1)(b, 2)(b, 1), (b, 2)(b, 1), (a, 1)(b, 1)(a, 1), (b, 2), (b, 1)(b, 1)$. It basically continually adds nodes for new symbol in each node.

State merges and transition splits are two main operations of structure and temporal guards learning in RTI+. A split of a transition (see an example in Figure 8) $\delta = \langle q, q', a, [m, m'] \rangle$ at time t creates two new transitions $\langle q, q_1, a, [m, t] \rangle$ and $\langle q, q_2, a, [t + 1, m'] \rangle$. The target states q_1 and q_2 are the roots of two new prefix trees that are reconstructed based on the input sample.

The algorithm also greedily merges pairs of states (q, q') in this tree, forming an increasingly smaller machine that generalizes over samples as shown in Figure 9. Because PDRTAs are deterministic, for every event $e \in \Sigma$ the states that are reached from q and q' have to be merged as well– also known as the determinization process.

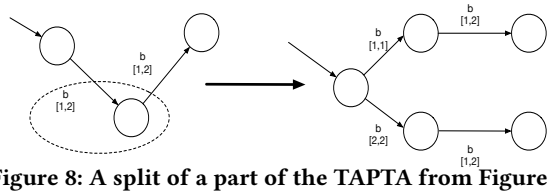


Figure 8: A split of a part of the TAPTA from Figure 7.

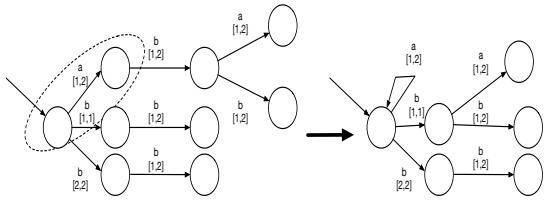


Figure 9: A merge operation of TAPTA after the split from Figure 8

Note that these examples are only possible split and merge illustrating how to conduct these operations. The algorithm uses a likelihood-ratio statistical test to decide whether to split/merge or not [40]. A hypothesis H is called nested within another hypothesis H' if the possible distributions under H form a strict subset of the possible distributions under H'. By definition, H' has more unconstrained parameters (or degrees of freedom) than H ($r' > r$). In our case, H is the model after merge (resp. before a split) and H' is the model before a merge (resp. after a split). Given two hypotheses H and H' such that H is nested in H', and a data set S, the likelihood ratio test statistic is computed by:

$$LR = \frac{LK(S, H)}{LK(S, H')} \tag{4}$$

where the likelihood LK estimates how likely S is generated by the corresponding hypothesis. The random variable $y = -2\ln(LR)$ is asymptotically $\chi^2(r' - r)$ distributed [46]. The p-value is the probability that a random value in $\chi^2(r' - r)$ would be greater than or equal to the observed value y by chance. If it is smaller than 0.05, H and H' are significantly different with 95% confidence so that a split operation is accepted. In addition, a merge is accepted whenever the model after the merge is not significantly different from the model before the merge since they are supposed to have similar or compatible stochastic and timed behaviors. Note that the current version of RTI+ tries to model time and events distributions independently. An overview of RTI+ is in Algorithm 2. The model learned of LIT101 sensor signal is shown in Figure 10. Any testing sequence that is not fired by the learned TA is alarmed as an anomaly, i.e., the abnormal event lasts until the end of the sequence. There are two typical types of alarms in TA: "event error" (symbol that can not be fired for transition in the given state) and "timing error" (symbol's timing is outside the valid time guard).

5.3 Learning Bayesian Network

A Bayesian network (BN) is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). In this paper, the BN is learned to model the dependencies among random variables from the sensors and the actuators in the local process; an example is illustrated in Figure 11, which is the BN learned from P1. A BN consists of the graph structure (representing the dependencies) and the parameters. The parameters are represented by conditional probability distribution, summarizing the probability distribution of a node given its parents. In this paper, variables from sensor and actuator are discretized, thus the probability distribution of each node is actually a conditional probability distribution (CPD) table, also see the CPD in Figure 11.

Bayesian network learning includes structure learning and parameter learning. In this paper, a greedy search algorithm K2 [11] is used for the structure learning. The general idea is as follows. Initially each node has no parents. It then adds incrementally that parent whose addition results in the largest increase in the score of the resulting structure. When the addition of no single parent can increase the score, it stops adding parents to the node. The pseudo code is shown in Algorithm 3. The random variable of the sensor is fixed as the last entry by assuming it is not the parent node of any other variables, while the order of parents is random. Parameter learning is relatively simple, when the structure is learned, a maximum likelihood estimation, i.e. counting the probability of each node from the data, is used to obtain the CPD tables. The evaluation of testing using BN is just to check the probability in the CPD table. An alarm is raised if the corresponding entry in the table is equal to zero, i.e. such a concurrent event does not exist in the training data.

Algorithm 2 Data identification with RTI+:

Require: A (multi-)set of timed strings S_+
Ensure: A small PDRTA $\mathcal{A}$ for S_+
  Construct a timed prefix $\mathcal{A}$ tree from S_+, let $Q' = \emptyset$
  for all all transitions $\delta = \langle q, q', a, [m, m'] \rangle$ from $\mathcal{A}$, **do**
    Evaluate all possible merges of q' with states from Q'
    Evaluate all possible splits of δ
    if the lowest split p-value< 0.05 **then**
      perform this split
    else if the highest merge p-value> 0.05 **then**
      perform this merge
    else
      add q to Q'
    end if
  end for

6 EXPERIMENTS

This section presents the evaluation of TABOR and discussion based on the learning experience.

6.1 Evaluation

The traditional way of evaluating anomaly detection is essentially a *point-based* approach. It considers multivariate time series data at each time point as an isolated instance. However, most of practical attacks in real life happen in a continuous period of time, such as the attack scenarios in the SWaT dataset that last continuously from minutes to hours. For the method we need to determine how many attack scenarios can be detected and the coverage in each

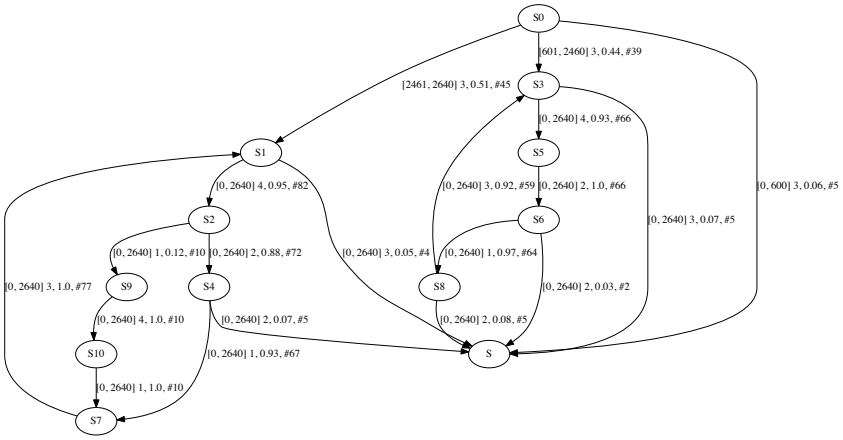


Figure 10: Timed automaton learned from LIT101. 1, 2, 3, 4 are symbols for QD, SC, SU, QU. S1 is the sink state, which is introduced due to fact that some sequences in the training data have very low frequencies of occurrence. The sink states are left without split or merge with other states due to lack of evidence for statistical testing.

Algorithm 3 K2

Require: A set of n nodes, and ordering on the nodes, an upper bound u on the number of parents a node may have, and a dataset D containing m cases.
Ensure: Parent of each node
  for all $i \coloneqq 1$ to n do, **do**
    $\pi_i \coloneqq \emptyset$
    $P_{old} \coloneqq f(i, \pi_i)$;
    OKToProceed $\coloneqq$ true;
    while OKToProceed and $|\pi_i| < u$ **do**
      let z be the node in Pred(x_i)$-\pi_i$ that maximized $f(i, \pi_i \cup \{z\})$
      $P_{new} \coloneqq f(i, \pi_i \cup \{z\})$
      if $P_{new} > P_{old}$ **then**
        $P_{old} \coloneqq P_{new}$
        $\pi_i \coloneqq \pi_i \cup \{z\}$
      else
        OKToProceed $\coloneqq$ false
      end if
    end while
    write parents node in each node
  end for

detected scenario. The traditional scoring methods, such as precision and recall, do not suffice because they only look at data points instead of windows [26]. In this paper, a novel way of evaluating anomaly detection in CPS system is proposed by borrowing the concept of *time series discord* from the data mining community. The mining task of time series discord is actually finding abnormal subsequences in time series data [24, 35], which is similar to the *scenario-based* or *window-based* detection goal in this paper. As illustrated in Figure 12, a false positive is a detected subsequence without an overlap between any ground-truth scenario. For the case of true positive, the coverage percentage (proportion of overlap length and total ground-truth scenarios length) evaluates the quality of detection coverage, while the penalty score (with time as unit) evaluates the length of detection outside the overlap. A good detection result should have a high coverage percentage (close to 1) and a small number of penalty scores.

All types of alarms are listed as follows:

(1) Timed automaton
  (a) Event error: an invalid event at given state;
  (b) Timing error: an event duration outside valid timing guard;
  (c) State error: reaching a sink state, where the computation halts.
(2) Bayesian network: a zero probability entry in the CPD
(3) Out of alphabet (OOA): the sensor value exceeds the threshold (i.e., $min - c\sigma$ and $min - c\sigma$), and the actuator value did not occur in the training data.

In Section 4 it is mentioned that the sensors in SWaT are grouped based on their different properties. For the LIT and DPIT sensors,

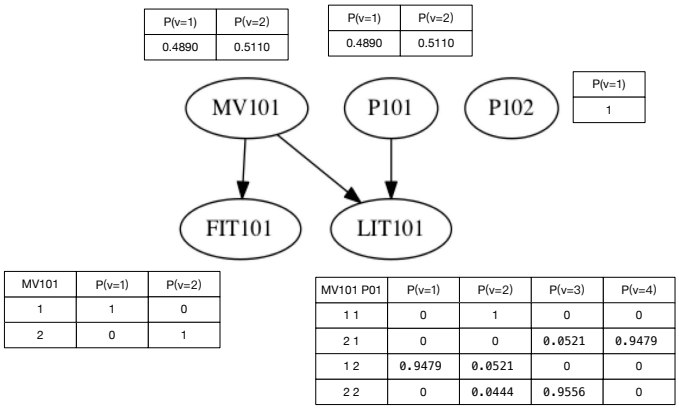


Figure 11: Bayesian network learned from P1. The unit in the table from the first column and the first row indicates that both MV101 and P101 are closed, the probability that water level quickly decreases (QD) is 0. Note that the actuators' status: open and closed are denoted as 2 and 1, respectively.

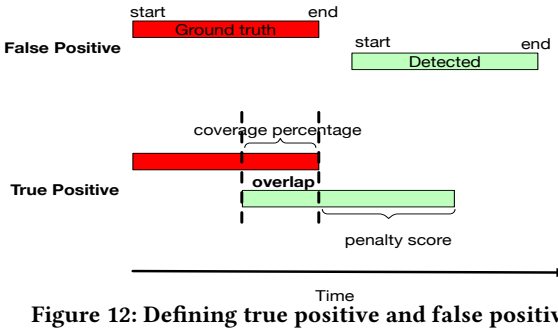


Figure 12: Defining true positive and false positive.

the signal shows regular patterns. The TA and the BN models are learned as shown in models 1, 5, 6, and 9. One key question is how to fuse the results from the TA and the BN model. Considering the high cost of false positive in a large water plant, a conservative *and* strategy is used to fuse the results, i.e. only adopting alarms raised from both the TA and the BN model. However, the OOA errors are adopted into the final result because they tend to show obviously abnormal patterns with a high priority. Table 2 shows the results of using the fused results and the single result from each model. It is imaginable that using an *or* strategy will get more true positives and much more false positives. Table 3 presents detailed results from each model. Figure 13 shows an example of the result fused from different types of alarms.

For the chemical sensors AIT and pressure sensor PIT, only the deviation of the differential is checked, i.e., an out of alphabet type. Because the resulting timed strings do not show stationary behaviors, a single symbol checking is thus deployed for anomaly detection. Examples from AIT202 and PIT501 are shown in Figure 14 and Figure 15.

To make a comprehensive comparison with existing literature, the point based recall evaluation in each attack scenario is also conducted in this paper, as shown in Table 4. Our proposed model successfully detects 24 out of 36 scenarios, while the DNN and the

Table 2: Comparison only using TA or BN

Model number	Method	PF	TP	HP (%)	PS (s)
1	TABOR	0	9	7.85	1889
1	TA	7	26	87.48	189516
1	BN	0	5	1.95	629
1	OOA	0	5	5.90	1260
5	TABOR	0	3	63.36	70
5	TA	16	13	73.95	47645
5	BN	0	4	64.03	70
5	OOA	0	0	0	0
6	TABOR	0	5	65.12	145
6	TA	3	24	81.32	188996
6	BN	0	3	63.32	33
6	OOA	0	2	1.80	112
9	TABOR	0	4	61.67	233
9	TA	0	3	60.23	856
9	BN	0	2	59.40	73
9	OOA	0	4	61.67	233

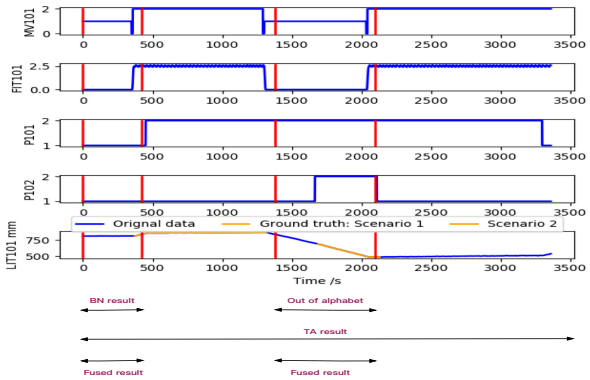


Figure 13: An example of fused results. The timed string of this example is (3,420)(2,960)(1,720)(3,1260), which is not fired by the TA as a whole sequence. The probability of the aligned event $P(LIT = 3)|(MV = 1, P101 = 1) = 0$. The actuator P102 is never seen to be open in the training data.

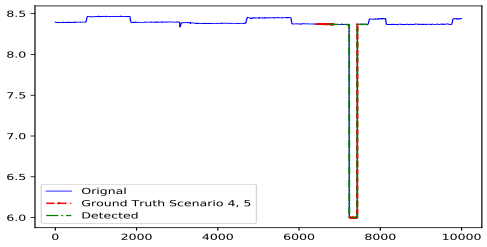


Figure 14: An example of the detection result from the chemical measurement sensor AIT202.

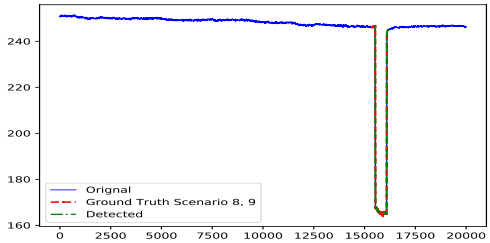


Figure 15: An example of detection results from the press measurement sensor PIT501.

Table 3: Results of each model

Model Number	FP	TP	Detected Scenarios	HP (%)	PS (s)
1	0	9	1, 2, 16, 21, 25, 28, 29, 30, 31	7.85	1889
2	0	0	0	0	0
3	0	2	4, 5	0.60	637
4	0	0	0	0	0
5	0	3	7, 22, 23	63.50	117
6	0	5	7, 13, 22, 23, 30	65.12	145
7	0	0	0	0	0
8	0	7	8, 9, 17, 23, 33, 34, 35	11.99	2363
9	0	4	9, 17, 23, 35	61.67	233
10	0	6	8, 9, 17, 23, 34, 35	3.22	1867
11	0	6	8, 9, 17, 23, 33, 35	5.50	1360
12	0	1	23	0.38	513
13	0	9	8, 9, 14, 15, 17, 23, 32, 34, 35	4.17	1354
14	0	6	8, 9, 17, 23, 32, 35	3.11	965
15	0	7	8, 9, 17, 23, 32, 34, 35	2.36	1127
16	0	7	8, 9, 17, 23, 32, 34, 35	4.50	877

SVM detects 12 and 19, respectively. The further overall comparison is shown in Table 5. The proposed model has slightly more false positives but a much better recall, thus the overall performance in F-measure is superior over the DNN and the SVM models with 2-3% relative improvement. Moreover, the runtime comparison in Table 6 shows that the computation is highly efficient in the proposed model. The key advantage is that the original multivariate signal is partitioned into groups and segmented to dramatically reduce the dimension and the size of training data. The learning of RTI+ and K2 are both polynomial time. The computation complexity of testing each event is just $O(1)$ in both the TA and BN model.

6.2 Discussion

Precise segmentation on the basis of sensor data is difficult due to the noise. Also, the classification of segments with very close differential values, e.g., SU and SC in the signals from LIT101, pose challenges to a robust detection. Another more important question is the ending time of an attack scenario. The researchers who designed the attacks in SWaT claim that the time interval between two consecutive attacks is large enough for the stabilization of the SWaT system. However, just for the LIT sensors, one cycle of water fluctuation takes more than one hour (Figure 6), while the shortest time interval of attack in the SWaT system is less than 10 minutes. An example false positive result in Model 6 detected by TA is shown in Figure 16. An obvious abnormal pattern is seen in the sensor signal. However, no attack actually took place. The irregular signal is caused by the stabilization following the attack scenarios 8 and 9. A better, and more fair, way of evaluation is an open question in the anomaly detection of CPS, e.g., the ending time of an attack should be on the basis of no more impact on the system rather than the end point of an attack behavior. The main reason for a false negative is the conservative result fusion strategy from the TA and the BN model. How to combine the results of different models and make a tradeoff between sensitivity and robustness are challenging problems. Dealing with concurrent attacks on a same node of TABOR is also a challenging problem. Our system is only able to separate them if different types of alarms are raised.

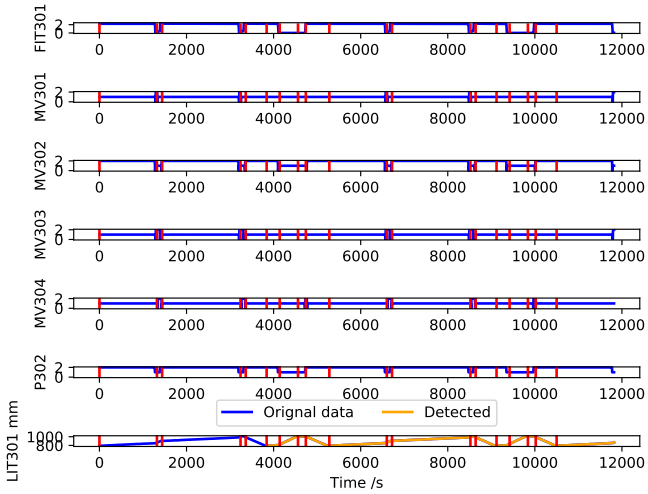


Figure 16: An example of detection results from PIT501. The false alarm is caused by the stabilization procedure after an attack.

7 CONCLUSION AND FUTURE WORK

In this paper, a novel graphical framework is proposed to learn the local behavior of a complex water treatment plant. The model profiles the normal behavior of the SWaT system, which is further used for anomaly detection. This technique can be considered as a combination of machine learning and specification-based detection. On one hand, it provides an inexpensive and automated learning approach for specification mining from an industrial control system without the need of expert knowledge. On the other hand, the resulting specification-like model is highly interpretable and useful for the validation and the localization of abnormal sensors or actuators in the system.

We have already started working on a state-based version of TABOR, i.e., modeling sequential control behavior of actuators in

Table 4: Points evaluation in each scenario. The second column of scenario number is consistent with the original attacks description. However, some of them do not have any actual impact in SWaT. Basically only 36 scenarios are counted in this paper and the literature.

NO.	NO. Scenario	Description of attack	DNN	SVM	TABOR
1	1	Open MV-101	0	0	0.049
2	2	Turn on P-102	0	0	0.930
3	3	Increase LIT-101 by 1mm every second	0	0	0
4	4	Open MV-504	0	0.035	0.328
5	6	Set value of AIT-202 as 6	0.717	0.720	0.995
6	7	Water level LIT-301 increased above HH	0	0.888	0
7	8	Set value of DPIT as >40kpa	0.927	0.919	0.612
8	10	Set value of FIT-401 as <0.7	1	0.433	0.994
9	11	Set value of FIT-401 as 0	0.978	1	0.998
10	13	Close MV-304	0	0	0
11	14	Do not let MV-303 open	0	0	0
12	16	Decrease water level LIT-301 by 1mm each second	0	0	0
13	17	Do not let MV-303 open	0	0	0.597
14	19	Set value of AIT-504 to 16 uS/cm	0.123	0.13	0.004
15	20	Set value of AIT-504 to 255 uS/cm	0.845	0.848	0.997
16	21	Keep MV-101 on continuously; Value of LIT-101 set as 700mm	0	0.0167	0.083
17	22	Stop UV-401; Value of AIT502 set as 150; Force P-501 to remain on	0.998	1	0.998
18	23	Value of DPIT-301 set to >0.4 bar; Keep MV-302 open; Keep P-602 closed	0.867	0.875	0
19	24	Turn off P-203 and P-205	0	0	0
20	25	Set value of LIT-401 as 1000; P402 is kept on	0	0.009	0
21	26	P-101 is turned on continuously; Set value of LIT-301 as 801mm	0	0	0.999
22	27	Keep P-302 on continuously; Value of LIT401 set as 600mm till 1:26:01	0	0	0.196
23	28	Close P-302	0.936	0.936	1.000
24	29	Turn on P-201; Turn on P-203; Turn on P-205	0	0	0
25	30	Turn P-101 on continuously; Turn MV-101 on continuously; Set value of LIT-101 as 700mm; P-102 started itself because LIT301 level became low	0	0.003	0.999
26	31	Set LIT-401 to less than L	0	0	0
27	32	Set LIT-301 to above HH	0	0.905	0
28	33	Set LIT-101 to above H	0	0	0.890
29	34	Turn P-101 off	0	0	0.990
30	35	Turn P-101 off; Keep P-102 off	0	0	0.258
31	36	Set LIT-101 to less than LL	0	0.119	0.889
32	37	Close P-501; Set value of FIT-502 to 1.29 at 11:18:36	1	1	0.998
33	38	Set value of AIT402 as 260; Set value of AIT502 to 260	0.923	0.927	0.996
34	39	Set value of FIT-401 as 0.5; Set value of AIT-502 as 140 mV	0.940	0	0.369
35	40	Set value of FIT-401 as 0	0.933	0.927	0.997
36	41	Decrease LIT-301 value by 0.5mm per second	0	0.357	0

Table 5: Points based evaluation

Method	Precision	Recall	F measure
DNN	**0.98295**	0.67847	0.80281
SVM	0.92500	0.69901	0.79628
TABOR	0.86171	**0.78803**	**0.82322**

Table 6: Runtime comparison

Model Number	Training	Testing
DNN	2 weeks	8 hours
SVM	30 min	10 min
TABOR	**214 s**	**33** s

SWaT instead of relying on the segmentation and alignment of sensor signal. This extension also aims at discovering very complex concurrent events of CPS in state space without independent event assumption in Bayesian network. In the near future, a construction-and-correct idea can be implemented in the learning procedure, e.g., any false positive and false negative examples are considered as counter examples to improve the learning result. The behavior modeling of network traffic and network attack detection are also important in the future work, because currently in SWaT dataset, the attacker is assumed sitting inside the network. Last but not least, the learned model can actually be used as a simulation controller, which can be used for attack-response platforms in further research.

REFERENCES

[1] S. Adepu and A. Mathur. 2016. Generalized Attacker and Attack Models for Cyber Physical Systems. In *2016 IEEE 40th Annual Computer Software and Applications*

Conference (COMPSAC). 283–292.

[2] Sridhar Adepu and Aditya Mathur. 2016. An investigation into the response of a water treatment system to cyber attacks. In *High Assurance Systems Engineering (HASE), 2016 IEEE 17th International Symposium on*. IEEE Computer Society, 141–148.

[3] Sridhar Adepu and Aditya Mathur. 2016. Using process invariants to detect cyber attacks on a water treatment system. In *IFIP International Information Security and Privacy Conference*. Springer, 91–104.

[4] S. Adepu, G. Mishra, and A. Mathur. 2017. Access Control in Water Distribution Networks: A Case Study. In *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*. 184–191.

[5] Chuadhry Mujeeb Ahmed, Carlos Murguia, and Justin Ruths. 2017. Model-based Attack Detection Scheme for Smart Water Distribution Networks. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 101–113.

[6] Patricia Bouyer, Fabrice Chevalier, and Deepak D'Souza. 2005. Fault diagnosis using timed automata. In *International Conference on Foundations of Software Science and Computation Structures*. Springer, 219–233.

[7] A.A. Cardenas, S. Amin, and S. Sastry. 2008. Secure Control: Towards Survivable Cyber-Physical Systems. In *Distributed Computing Systems Workshops, 2008. ICDCS '08. 28th International Conference on*. 495 –500.

[8] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry. 2011. Attacks against process control systems: Risk assessment, detection, and response. In *ACM Symp. Inf. Comput. Commun. Security*.

[9] Edmund M Clarke and Paolo Zuliani. 2011. Statistical Model Checking for Cyber-Physical Systems.. In *ATVA*, Vol. 11. Springer, 1–12.

[10] Pamel Cobb. 2015. German Steel Mill Meltdown: Rising Stakes in the Internet of Things. (2015). https://securityintelligence.com/german-steel-mill-meltdown-rising-stakes-in-the-internet-of-things/

[11] Gregory F Cooper and Edward Herskovits. 1992. A Bayesian method for the induction of probabilistic networks from data. *Machine learning* 9, 4 (1992), 309–347.

[12] Sriharsha Etigowni, Dave Jing Tian, Grant Hernandez, Saman Zonouz, and Kevin Butler. 2016. CPAC: securing critical infrastructure with cyber-physical access control. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 139–152.

[13] Wei Gao and Thomas H Morris. 2014. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *The Journal of Digital Forensics, Security and Law: JDFSL* 9, 1 (2014), 37.

[14] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. 2017. Anomaly Detection in Cyber Physical Systems Using Recurrent Neural Networks. In *High Assurance Systems Engineering (HASE), 2017 IEEE 18th International Symposium on*. IEEE, 140–145.

[15] D. Hadžiosmanović, R. Sommer, E. Zambon, and Pieter H. Hartel. 2014. Through the Eye of the PLC: Semantic Security Monitoring for Industrial Processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*. ACM, New York, NY, USA, 126–135.

[16] ics-cert [n. d.]. https://ics-cert.us-cert.gov/. ([n. d.]).

[17] icsCERTAdvisory [n. d.]. ICS-CERT Advisories https://ics-cert.us-cert.gov/advisories. ([n. d.]).

[18] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M Poskitt, and Jun Sun. 2017. Anomaly Detection for a Water Treatment System Using Unsupervised Machine Learning. *arXiv preprint arXiv:1709.05342* (2017).

[19] Goh J., Adepu S., Junejo K. N, and Mathur A. 2016. A Dataset to Support Research in the Design of Secure Water Treatment Systems. In *The 11th International Conference on Critical Information Infrastructures Security (CRITIS)*. Springer, New York, USA, 1–13.

[20] Austin Jones, Zhaodan Kong, and Calin Belta. 2014. Anomaly detection in cyber-physical systems: A formal methods approach. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, IEEE Computer Society, 848–853.

[21] Khurum Nazir Junejo and Jonathan Goh. 2016. Behaviour-Based Attack Detection and Classification in Cyber Physical Systems Using Machine Learning. In *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*. ACM, 34–43.

[22] Eunsuk Kang, Sridhar Adepu, Daniel Jackson, and Aditya P Mathur. 2016. Model-based security analysis of a water treatment system. In *Proceedings of the 2nd International Workshop on Software Engineering for Smart Cyber-Physical Systems*. ACM, 22–28.

[23] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. 2001. An online algorithm for segmenting time series. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*. IEEE, 289–296.

[24] Eamonn Keogh, Jessica Lin, Sang-Hee Lee, and Helga Van Herle. 2007. Finding the most unusual time series subsequence: algorithms and applications. *Knowledge and Information Systems* 11, 1 (2007), 1–27.

[25] Yakup Koç, Martijn Warnier, Piet Van Mieghem, Robert E Kooij, and Frances MT Brazier. 2014. The impact of the topology on cascading failures in a power grid model. *Physica A: Statistical Mechanics and its Applications* 402 (2014), 169–179.

[26] Alexander Lavin and Subutai Ahmad. 2015. Evaluating Real-Time Anomaly Detection Algorithms–The Numenta Anomaly Benchmark. In *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. IEEE, 38–44.

[27] Edward A. Lee. 2008. *Cyber Physical Systems: Design Challenges, http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html*. Technical Report UCB/EECS-2008-8. EECS Department, University of California, Berkeley. http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html

[28] Robert Lipovsky. 2016. New wave of cyberattacks against Ukrainian power industry. (January 2016). http://www.welivesecurity.com/2016/01/11.

[29] Y. Liu, P. Ning, and M. Reiter. 2009. False data injection attacks against state estimation in electric power grids. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*. 21–32.

[30] Yao Liu, Peng Ning, and Michael K Reiter. 2011. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)* 14, 1 (2011), 13.

[31] Alexander Maier, Asmir Vodencarevic, Oliver Niggemann, Roman Just, and Michael Jaeger. 2011. Anomaly detection in production plants using timed automata. In *8th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*. 363–369.

[32] John Mulder, Moses Schwartz, Michael Berg, Jonathan Roger Van Houten, Jorge Mario, Michael Aaron King Urrea, Abraham Anthony Clements, and Joshua Jacob. 2013. *WeaselBoard: Zero-Day Exploit Detection for Programmable Logic Controllers*. Technical Report. tech. report SAND2013-8274, Sandia National Laboratories.

[33] Oliver Niggemann, Benno Stein, Asmir Vodencarevic, Alexander Maier, and Hans Kleine Büning. 2012. Learning Behavior Models for Hybrid Timed Systems.. In *AAAI*, Vol. 2. 1083–1090.

[34] Paul Oman and Matthew Phillips. 2007. Intrusion detection and event monitoring in SCADA networks. *Critical Infrastructure Protection* (2007), 161–173.

[35] Haemwaan Sivaraks and Chotirat Ann Ratanamahatana. 2015. Robust and accurate anomaly detection in ECG artifacts using time series motif discovery. *Computational and mathematical methods in medicine* 2015 (2015).

[36] John A. Stankovic. 2016. Research Directions for Cyber Physical Systems in Wireless and Mobile Healthcare. *ACM Trans. Cyber-Phys. Syst.* (Nov. 2016), 1:1–1:12.

[37] Keith Stouffer, Joe Falco, and Karen Scarfone. 2011. *Guide to industrial control systems (ICS) security*. Technical Report 11. NIST special publication. 800–82 pages. http://csrc.nist.gov/publications/nistpubs/800-82/SP800-82-final.pdf

[38] Stavros Tripakis. 2002. Fault diagnosis for timed automata. In *FTRTFT*, Vol. 2469. Springer, 205–224.

[39] David I. Urbina, Jairo A. Giraldo, Alvaro A. Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, and Henrik Sandberg. 2016. Limiting the Impact of Stealthy Attacks on Industrial Control Systems. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. 1092–1105.

[40] Sicco Verwer, Mathijs de Weerdt, and Cees Witteveen. 2010. A likelihood-ratio test for identifying probabilistic deterministic real-time automata from positive data. In *International Colloquium on Grammatical Inference*. Springer Berlin Heidelberg, 203–216.

[41] Sicco Ewout Verwer. 2010. *Efficient identification of timed automata: Theory and practice*. Ph.D. Dissertation. TU Delft, Delft University of Technology.

[42] Sicco E Verwer, Mathijs M De Weerdt, and Cees Witteveen. 2006. Identifying an automaton model for timed data. In *Benelearn 2006: Proceedings of the 15th Annual Machine Learning Conference of Belgium and the Netherlands, Ghent, Belgium, 11-12 May 2006*.

[43] Asmir Vodenčarević, Hans Kleine Büning, Oliver Niggemann, and Alexander Maier. 2011. Using behavior models for anomaly detection in hybrid systems. In *Information, Communication and Automation Technologies (ICAT), 2011 XXIII International Symposium on*. IEEE, 1–8.

[44] Sharon Weinberger. 2011. Computer security: Is this the start of cyberwarfare? *Nature* 174 (June 2011).

[45] K. Wilhoit and S. Hara. 2015. The real world evaluation of cyber-attacks against ICS system. In *Society of Instrument and Control Engineers of Japan (SICE), 2015 54th Annual Conference of the*. 977–979.

[46] Samuel S Wilks. 1938. The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The Annals of Mathematical Statistics* 9, 1 (1938), 60–62.

[47] Xi Zheng and Christine Julien. 2015. Verification and validation in cyber physical systems: research challenges and a way forward. In *Proceedings of the First International Workshop on Software Engineering for Smart Cyber-Physical Systems*. IEEE Press, 15–18.