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

Cheng Feng [1,3] Venkata Reddy Palleti [2] Aditya Mathur [2]
Deeph Chana [3]

[1]Siemens Corporate Technology

[2]Singapore University of Technology and Design

[3]Imperial College London

February 26, 2019

Background

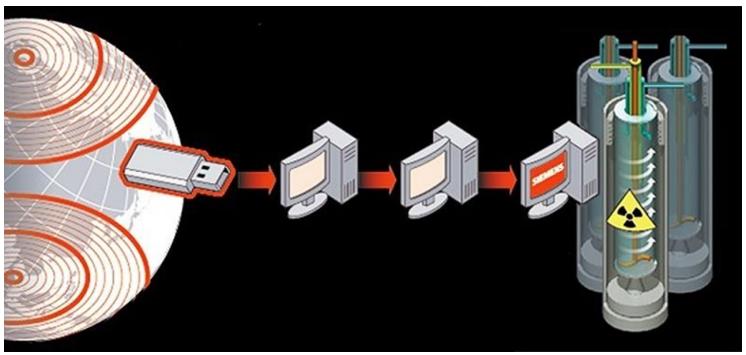
Industrial control systems

Industrial control systems (ICS) are combinations of software and hardware that monitor and manage industrial processes.

They can be found in many national infrastructures, e.g., gas pipelines, power plants, water treatment facilities, etc.

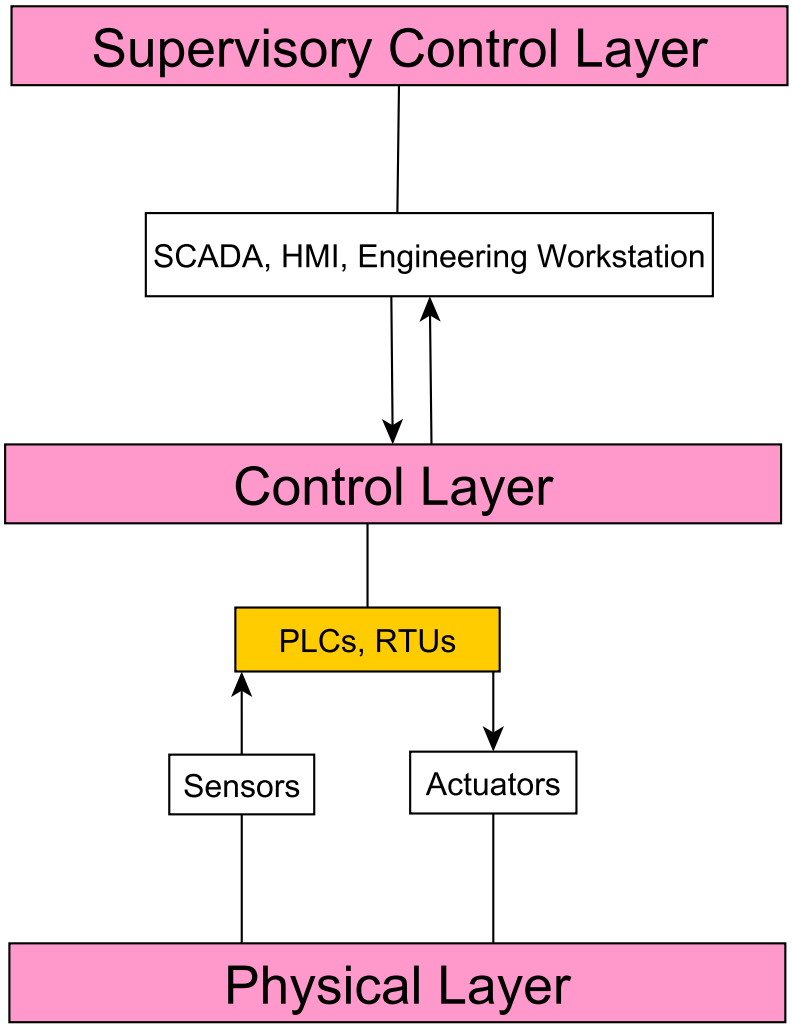
ICS have evolved from isolated networks to being **heavily connected with wider networks and services**.

- enhance operation efficiency and reduce maintenance costs
- lead to new cyber security vulnerabilities.



Background

ICS architecture



- Supervisory Control Layer: higher level supervisory monitoring and control function, e.g., Human machine interface, **Anomaly Detection systems**
- Control Layer: communicate and issue control commands to field devices
- Physical Layer: the physical process is directly monitored and controlled

Anomaly Detection Mechanisms for ICS

- Device-based: use the idea of device fingerprinting to detect intrusive devices
- Program-based: discover anomalous behaviour by checking the control or data flow in the control programs on programmable controllers.
- Network-based: reveal anomalies by investigating the network traffic flow such as the header, payload, timing and sequence of messages in the ICS network
- **Process-based**: look directly at the physical process variables such as sensor readings and actuator states, and their mathematical relationships to identify anomalies

Process-based Anomaly Detection

A common method:

- build a predictive model, e.g., AR, LDS, RNN models:

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

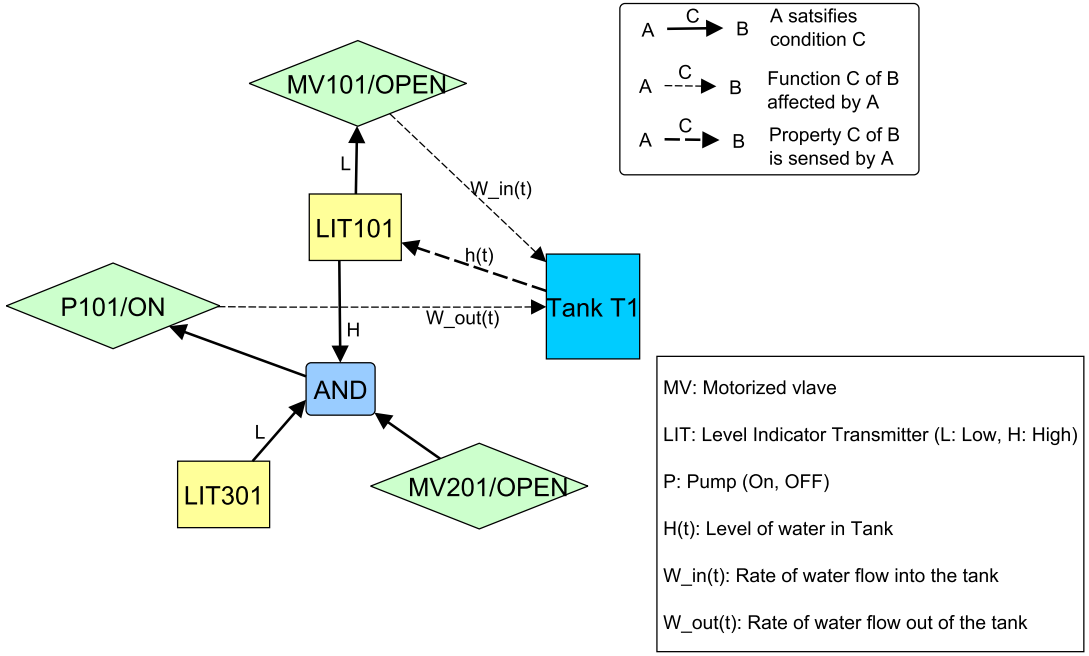
  - $\mathbf{x}^{\{t-p:t-1\}}$ the sensor measurements from time $t-p$ to $t-1$
  - $\mathbf{u}^{\{t-p:t-1\}}$ the actuator states from time $t-p$ to $t-1$
  - $\hat{\mathbf{x}}^{(t)}$ the predicted sensor measurements at time t
- An alarm will be raised when the residual error $\|\hat{\mathbf{x}}^{(t)} - \mathbf{x}^{(t)}\| > \tau$
- **Hard to define a decision boundary** to separate normal and abnormal sensor measurements; vulnerable to stealthy attacks.

Our work aims to propose a novel process-based detection model based on invariant rules.

Invariant Rules

Invariant rules:

- physical conditions that must be satisfied for any given state of an ICS
- generally defined by system engineers during the design phase of the system → this **manual process** is inefficient and suboptimal



$$\text{LIT101-H} \Rightarrow \text{MV101=OPEN}$$

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

Can we derive invariant rules from a purely **data-driven** perspective?

Problem Statement

$\mathcal{D}^{\{1:T\}} = \{\mathbf{d}^1, \mathbf{d}^2, \ldots, \mathbf{d}^T\}$: a time-series data log in which each signal $\mathbf{d}^t = \{\mathbf{x}^t, \mathbf{u}^t\}$ consists of two vectors capturing sensor measurements and actuator states, respectively.

$\mathcal{I} = \{i_1, i_2, \ldots, i_k\}$: 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 $I^t \subseteq \mathcal{I}$.

Formally, we define an invariant rule as follows:

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

An example:

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

Learning Steps

Given an arbitrary ICS data log $\mathcal{D}^{\{1:T\}}$, decompose the learning process into two steps:

- **Predicate Generation**:
  - 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 $\mathcal{I}$, transform the data log $\mathcal{D}^{\{1:T\}}$ into a database of itemsets $I^{\{1:T\}} = \{I^1, I^2, \ldots, I^T\}$.
  - mine **meaningful invariant rules** from the database $I^{\{1:T\}}$ which can be used for anomaly detection in the ICS.

Predicate Generation

Discrete variables (usually representing acuator states): enumerate the possible states, e.g., for a pump actuator we generate predicates $Pump = ON$ and $Pump = OFF$

Continuous variables (usually representing sensor measurements): two strategies utilize the dynamic of control systems:

- **Distribution-driven strategy**:
  - assume there are hidden control states to govern the update of sensor readings at each time step
  - Derive predicates based on which hidden control state a sensor update occurs
- **Event-driven strategy**:
  - assume the updates of actuator states are triggered by reaching critical values of sensor measurements.
  - Derive predicates because on the (pre,post) conditions of the event triggers

Distribution-driven Strategy

Assumption: there are K hidden control states to govern the update of a sensor:

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

μ_k: the expected update under hidden control state k.
ε_k: Gaussian noise under this state.

Infer hidden control states:

- Fit a GMM for the update of each sensor to capture the distributions of the underlying hidden control states.
- Decide value of K by criterion such as minimizing BIC score.

Generate predicates: based on K control states, generate predicates: $\{\Delta x^t \sim \mathcal{N}_1, \ldots, \Delta x^t \sim \mathcal{N}_K\}$.

Event-driven Strategy

Events: discrete changes on actuator states, e.g., pump is switched from ON to OFF.

Find event triggers:

- Define T_e as the set of time points at which the event e occurs.
- fit a lasso regression model for the values of sensor measurements at the time steps in T_e:

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

- A trigger for event e is found if:

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

Generate predicates: for each event trigger, we generate predicates:

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

Define Meaningful Invariant Rules

Conditions must be satisfied:

Minimum Support Condition

An invariant rule $X \Rightarrow Y$ is meaningful, then:

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

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

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 $\sigma(X \cup Y) = \sigma(U \cup W)$.

Mine Meaningful Invariant Rules

Mining meaningful invariant rules can be treated as a problem of **association rule mining**.

Steps:

- Mine all closed frequent itemsets with the multiple minimum support conditions via algorithms like CFP-growth, CFP-growth++, etc.

- Given any closed frequent itemset, partition the itemset Y into two non-empty subsets, X and Y-X, a rule $X \Rightarrow Y - X$ is generated if $\sigma(Y)/\sigma(X) = 1$.

Experiment Setup

- Based on datasets collected from a water distribution testbed (WADI) and a secured water treatment plant (SWAT).
- Data splited into training and testing set, both were collected every one second from the testbeds non-stoppable for several days.
- Various type of insider attacks injected in the testing set.
- Performance compared with invariant rules defined by system engineers and a residual-error based detection model.

Experiment Result

WADI

For any given attack type i, we consider this attack type is detectable by a model if $TPR_i > k \times FPR$, then we calculate:

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

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	$\tau_e = 1 \times 10^{-3}$

Experiment Result

SWAT

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	$\tau_e = 1 \times 10^{-3}$

Further Insights

why data-driven invariants outperform design-based invariants

Reason 1: more robust to noise on sensor measurements, can reduce FPR.

Example: the following design-based invariant rule:

$$1\_LT\_001 < 60 \Rightarrow 1\_MV\_004 = OFF$$

causes **55** false positives in the experiment on the WADI testbed.

The corresponding data-driven invariant rule:

$$1\_LT\_001 < 59.0399179104 \Rightarrow 1\_MV\_004 = OFF$$

which causes **zero** false positives instead.

Further Insights

why data-driven invariants outperform design-based invariants

Reason 2: the data-driven approach can generate a significantly larger invariant rule set, thus it has more chance to detect anomalies.

WADI	No. of Rules
Design-based	22
$\tau_e = 1 \times 10^{-4}$	3259
$\tau_e = 1 \times 10^{-3}$	45847

SWAT	No. of Rules
Design-based	38
$\tau_e = 1 \times 10^{-4}$	5805
$\tau_e = 1 \times 10^{-3}$	17737

Further Insights

why data-driven invariants outperform design-based invariants

Reason 3: 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.

Example: 65% of the data-driven invariant rules generated in SWaT case study span non-neighboring stages. However, the design-based invariant rules only capture the relationship between the sensors and actuators within the same or neighboring stages.

Further Insights

pros and cons compared with residual error-based method

Pros:

- more robust to process noise, enjoys relatively high TPR with a very low FPR
- self-explanatory, information can be used for further diagnosis
- more robust to stealthy attacks

Cons:

- larger delay
- nonlinear relationships are not explicitly considered

Conclusion

- Invariant rules can be automatically learned from ICS data log using a combination of several machine learning and data mining techniques.
- Using data-driven invariant rules to do process-based anomaly detection can achieve higher performance than using design-based rules
- Can achieve higher TPR compared with the residual error-based method with similar FPR.
- Possible extensions: adding nonlinear predicates, sequential invariant rules, etc.

End

Thank you!

Questions?