# Diagnosing PLC Using Control Invariants

Hua Yu, *Zhejiang University*
Zeyu Yang, *Zhejiang University*
Liang He, *University of Colorado Denver*
Peng Cheng, *Zhejiang University*
Jiming Chen, *Zhejiang University*

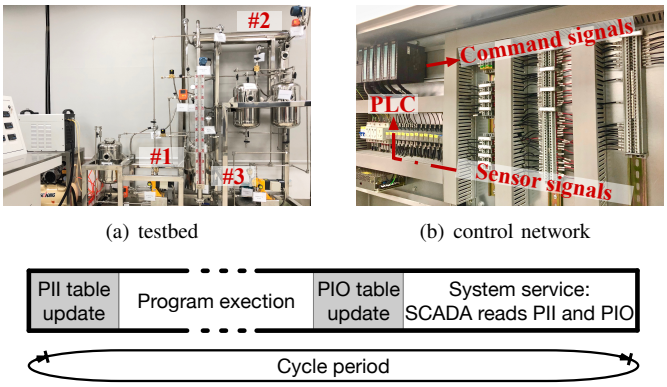
***Abstract*—Programmable Logic Controllers (PLCs) are the ground of control systems, which are however, vulnerable to a variety of cyber attacks, especially for networked real-time control systems. To mitigate this issue, we design `PLC-Sleuth`, a novel non-invasive intrusion detection/localization system for PLCs, grounding on a set of control invariants i.e., the correlations between sensor readings and the concomitantly triggered PLC commands that exist pervasively in all control systems. Specifically, taking the system's Supervisory Control and Data Acquisition log as input, `PLC-Sleuth` abstracts/identifies the system's control invariants as a control graph using data-driven structure learning, and then monitors the weights of graph edges to detect anomalies thereof, which is in turn, a sign of intrusion.**

## I. Introduction

**Background.** Commodity *Programmable Logic Controllers* (PLCs) are proved vulnerable to cyber attacks. Researchers have identified >36.7K PLCs that can be accessed by scanning the ports of common communication protocols, such as Modbus and Siemens S7. Symantec has also confirmed the feasibility of hijacking a number of mission-critical PLCs, such as acquiring credentials of the target PLC to administer destructive payloads. Keliris and Maniatakos have designed an autonomous process to compromise PLCs, facilitating the executable program injection or firmware modification. After compromising the PLC, adversaries can mount a variety of attacks, including but not limited to:

- *Command Injection Attacks.* The malware TRITON was launched remotely in Saudi Arabia in 2017 [1] to disturb the operations of safety actuators in a petrochemical facility. These actuators were originally designed to take remedial actions in case of emergency, while TRITON instead sent them adversarial commands to shut down the facilities.
- *Cooperative Stealthy Attacks.* To hide attacks from being detected, adversaries can even mount advanced stealthy attacks to PLCs, by not only tampering control commands, but also cooperatively forging the Supervisory Control and Data Acquisition (SCADA) logs with historical (and normal) data. An example of this stealthy attack is the worm Stuxnet which damaged hundreds of centrifuges [2]. A firmware vulnerability of Allen Bradley PLCs exposed in 2017 [3] allows modifying the PLCs' control commands and forging sensor readings.

**Attack Model.** Combining the attack behavior and PLC running scheme (see Fig. 1(c)), we consider the following attack model targeting at the PLC of a given control system:




(a) testbed (b) control network

(c) Sequence of CPU executions in PLC [4]

Fig. 1. The Secure Ethanol Distillation System (SEDS) prototype in our lab.

a) The attacker can mount the command injection attack during the program execution period by downloading malicious code and sending faked protocol packets to PLC.
b) Atop the injection attack, the attacker can deliver cooperative stealthy attacks by further replaying normal sensor readings to the PLC's process-image input (PII) table. Note that SCADA reads sensor readings from PII table, which happens after the execution of malicious code. This way, the attacker can deceive the system monitor to conclude a normal operation even if a successful attack has been launched [2].
c) The attacker cannot modify the record of actual commands issued during system operation, i.e., any forged commands issued by the attacker will be logged as they are. This is because PIO table logs all commands and remains unchanged after the execution of the program [5]. By reading from the PIO table, SCADA obtains control commands that are actually issued.

**Protecting PLCs Using Control Invariants.** To mitigate the above issues, we design a non-invasive data-driven intrusion detection system (IDS) for PLCs, which we call `PLC-Sleuth`. The foundation of `PLC-Sleuth` is a set of *control invariants* that exist pervasively in all control systems: the control commands issued by PLCs correlate strongly with the concomitant sensor readings. `PLC-Sleuth` automatically identifies these control invariants using system's SCADA logs, and abstracts them as a control graph, in which the nodes represent system variables, and the weights of edges quantify the strength of correlations between system variables. `PLC-Sleuth` then captures the normal behavior of the weights of graph edges using data-driven approaches,

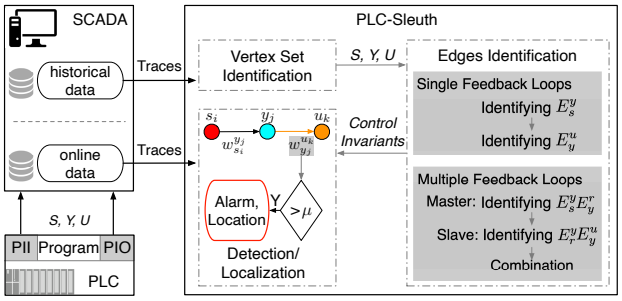



Fig. 2. Architecture of PLC-Sleuth.

and detects, at runtime, the anomalies — edges with abnormal weights indicate the control channels between corresponding system variables have been compromised.

In this demo, we use our prototype of an ethanol distillation control system, as shown in Fig. 1, called SEDS (Secure Ethanol Distillation System), to demonstrate how the PLC-Sleuth detects and localizes PLC intrusions at real time, which extends our recently accepted paper [6].

## II. Design of PLC-Sleuth

Fig. 2 presents the logic flow of PLC-Sleuth: abstracting the control invariants of the real-time system-of-interest by identifying the variable's connections, and monitoring the time series of invariants to detect and localize PLC attacks.

### A. Abstracting Control Invariants

For a given control system, PLC-Sleuth identifies and abstracts its control invariants — defined by the system variables and the interactions thereof — using a *control graph*. Specifically, the control graph is defined as a weighted and directed acyclic graph $\mathcal{G}(\mathtt{V}, \mathtt{E}, \mathtt{W})$, where: (i) $\mathtt{V}$ is the set of nodes representing system variables (i.e., setpoints $S$, sensor readings $Y$, and commands $U$), (ii) $\mathtt{E}$ is the set of directed edges connecting nodes in $\mathtt{V}$, including a control error edge set $E_s^y$ connecting nodes in $S$ to nodes in $Y$, and a control command edge set $E_y^u$ connecting nodes in $Y$ to nodes in $U$, and (iii) $\mathtt{W}$ is the set of edge weights, representing the strength of the correlations described by $\mathtt{E}$. Specifically,

- **Error Weight Set $W_s^y$.** The weight of control error edge $w_{s_i}^{y_j} \in W_s^y$ captures the difference between system state $y_j$ and the corresponding setpoint $s_i$.
- **Command Weight Set $W_y^u$.** A novel correlation — characterizing transition dependency of $u$ and $y$ using the posterior probability distribution of their time series — is proposed to define command weight $w_{y_j}^{u_k} \in W_y^u$. Specifically, we define the *transition mutual information (TMI)* as,

$$TMI(x_i, u_k) = \sum_{\xi \in \mathcal{X}^\tau} \sum_{\eta \in \mathcal{U}^\gamma} p(\xi, \eta) \log \left( \frac{p(\xi, \eta)}{p(\xi) p(\eta)} \right), \quad (1)$$

where $x_i$ is the control error between $s_i$ and $y_j$, $\tau$ and $\gamma$ are the length of sequences used for characterizing $y$ and $u$ transitions. This way, PLC-Sleuth calculates the command weight $w_{y_j}^{u_k}$ as $TMI(x_i, u_k)/H(u_k)$, where $H(u_k)$ is the entropy of the recent sequence of commands $u_k$.

PLC-Sleuth abstracts the control graph using data-driven structure learning.

### B. Detection/Localization of PLC Intrusions

PLC-Sleuth then detects/localizes, at runtime, PLC intrusions using an online norm model constructed using $\mathcal{G}$.

**Intrusion Detection.** Tampering the control command will violate the control invariants between commands and the corresponding measurements, i.e., the weights in $\mathcal{G}(\mathtt{V}, \mathtt{E}, \mathtt{W})$. PLC-Sleuth uses a memory-based method, such as the nonparametric CUSUM, to alarm operators if an anomaly is detected in the weights of edges in $E_y^u$.

**Intrusion Localization.** It is trivial to localize the forged command if only one abnormal edge is detected, i.e., the command responsible for that edge is compromised. When multiple anomalies are detected, PLC-Sleuth localizes the forged command as the one triggering the anomaly alarm first. This greedy strategy is intuitive because cascaded anomalies require a longer time to cause instability to control systems, when compared to the directly forged command.

## III. Demonstration

We will demonstrate PLC-Sleuth through two showcases deployed on SEDS:

- **The online construction of control graph** using SEDS's SCADA logs, showing the on-the-fly installation of PLC-Sleuth on a real-time control system, and the increasing accuracy as system runs.
- **The detection/localization of PLC intrusions** using the constructed control graph, under two attack scenarios on SEDS: (i) Attack-1: tampering control commands and control algorithm parameters, by downloading malicious control program and sending faked protocol packets to PLC, and (ii) Attack-2: replaying normal sensor measurements, atop of Attack-1, to deceive SCADA.

The first showcase focuses on the real-time capability and applicability to the critical industrial control system, and the second demonstrates the detection/localization effectiveness and timeliness for the two PLC intrusion scenarios mentioned above.

## References


[1] Symantec Security Response. After Triton, Will the Industrial Threat Landscape Ever be the Same? https://www.symantec.com/blogs/feature-stories/after-triton-will-industrial-threat-landscape-ever-be-same, 2018.

[2] Nicolas Falliere, Liam O Murchu, and Eric Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security Response.*, 5(6):29, 2011.

[3] Luis Garcia, Ferdinand Brasser, Mehmet Hazar Cintuglu, Ahmad-Reza Sadeghi, Osama A Mohammed, and Saman A Zonouz. Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit. In *Network and Distributed Systems Security (NDSS) Symposium*, 2017.

[4] Gary A Dunning. *Introduction to programmable logic controllers*. Cengage Learning, 2005.

[5] Hans Berger. *Automating with STEP7 in STL and SCL: programmable controllers Simatic S7-300/400*. Publicis Publishing, 2012.

[6] Zeyu Yang, Liang He, Peng Cheng, Jiming Chen, David K.Y. Yau, and Linkang Du. PLC-Sleuth: Detecting and Localizing PLC Intrusions Using Control Invariants. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020.