Implementation of Programmable CPS Testbed for Anomaly Detection

Hyeok-Ki Shin Woomyo Lee Jeong-Han Yun HyoungChun Kim

The Affiliated Institute of ETRI
Republic of Korea
{hkshin721, wmlee, dolgam, khche}@nsr.re.kr

Abstract

A large number of studies have provided datasets in the field of CPS security research, but the rate of actually using these datasets is low. It is difficult to objectively compare and analyze research results based on different testbeds or datasets. Our goal is to create public datasets for CPS security researchers working for anomaly detection. It is challenging for individuals to collect long-term datasets repeatedly for a large number of scenarios. This can lead to mistakes and inaccurate information. The process of collection must be comfortable and automated. For this purpose, we constructed a testbed in which three physical control systems (GE turbine, Emerson boiler, and FESTO water treatment system) can be combined with each other through the dSPACE Hardware-in-the-loop (HIL) simulator. We have built an environment that can automatically control each sensor and control point remotely. Using this environment, it is possible to collect datasets while repeatedly driving a large number of benign/malicious scenarios for a long period while minimizing human effort. We will develop and release CPS datasets using the testbed in the future.

1 Introduction

Cyber-physical systems (CPSs) are used in a variety of national core infrastructures such as waterworks, railways, transportation, and power plants. Abnormal or malicious behaviors in these CPSs can pose a serious threat to society.

Anomaly detection studies for CPS security have been carried out mainly in the field of network traffic [6, 8, 11]. A recent category of techniques focuses on changes in the physical states of control systems. With respect to detect CPS-specific attacks, numerous studies have been conducted to detect security incidents while monitoring the CPS operation status [1, 4, 5]. It is necessary to obtain various datasets for such investigations. However, this is extremely difficult.

The characteristics of normal operating conditions can be best identified by collecting information from an actual control system. However, it is difficult to accurately evaluate the performance of anomaly detection because there are limitations in the experiments used to reproduce the abnormal state in an actual CPS. Numerous researchers have built testbeds to conduct various experiments. In addition, a large number of studies have provided datasets in the field of security research, but the rate of actually using these datasets is low [12]. It is difficult to objectively compare and analyze research results based on different testbeds or datasets. Numerous public datasets are required to begin research in one field.

There have been a few recent works on dataset generation for CPS research [7, 9, 10]. There are certain drawbacks of using the currently available CPS datasets presented in literature [3].

- **Criteria of abnormal states.** Most data labeling is performed manually. Hence, there may be a problem with the accuracy of labeling information, particularly time. Additionally, sensor information may not be stabilized immediately after attacks but may gradually return to the normal state. There is lack of information on analyzing the system behaviors in normal and abnormal conditions.
- **Same attack in different environments.** Anomaly-based detection can detect the same attack depending on the operating situation. For accurate performance analysis, the same attack scenario must be collected several times in different operating situations.
- **Monitoring manipulation attack.** An attacker can manipulate monitoring information using human-machine interface (HMI)/Historian/SCADA for attack concealment. While there are attacks that send malicious control commands, such as Stuxnet, and transmit data by pretending to be a normal situation on an HMI, there are no available scenarios among open datasets.
- **Human effort.** It is challenging for individuals to collect long-term datasets repeatedly for a large number of scenarios. This can lead to errors and inaccurate information. The process of collection must be comfortable and automated.

We are developing a dataset that can be used for anomaly detection based on CPS operation information. For this purpose, we constructed a testbed in which three physical control systems (GE, Emerson, and FESTO) can be combined with each other through the dSPACE (HIL) simulator. To overcome the abovementioned problems and generate datasets for various attack scenarios, we considered the following points when constructing the testbed.

- We have built an environment that can automatically control each sensor and control point remotely. This makes it possible to collect datasets while repeatedly driving a large number of benign/malicious scenarios for a long period as well as minimizing human effort.
- Our testbed provides a method that manipulates monitoring information to prevent an operator or security equipment from recognizing an attack situation.
- We propose two methods to facilitate the changing of the testbed according to different scenarios without changing the control logic of each control device, and we implement the methods in our testbed.
- We build a PID simulator to predict the influence on the surrounding physical system according to the operation range of the main control point and to derive an appropriate attack point for reproducing attack scenarios.

The rest of this paper is organized as follows. In Section 2, we propose an attack model of CPSs and methods to reproduce the attack model in the testbed. Section 3 introduces our power plant testbed that consists of three single processes (GE turbine, Emerson boiler, and FESTO water treatment system) and an HIL simulator. Section 4 describes the process of reproducing the benign and malicious scenarios in the power plant testbed using the attack model. Finally, we discuss conclusions and future work in Section 5.

2 Attack Generation Based on Process Control Loop

We focus on recreating the effects of an attack, not its specific vectors. It is difficult for an attacker to control the entire system. Therefore, some devices and sensors may be manipulated and attacked.

We assume that an attacker has the following abilities:

1) An attacker can access field devices (sensors and actuators) and forge sensor values and control commands.
2) An attacker can access control devices (DCS and PLC) and forge sensor values, control commands, and monitoring signals through a control logic unit.
3) An attacker can manipulate monitoring signals to hide attacks against SCADA systems and internal security appliances.

2.1 Process Control Loop (PCL) Model

The process control loop (PCL) model is used to ensure the validity of an attack by defining an attack model and evaluating the inconsistency of the PCL. The PCL model describes the behavior of an individual component of a PCL, except for a controlled process, as a physical domain, as shown in Fig. 1. An ICS generally contains multiple PCLs to help a process output run in a stable manner by utilizing sensors, controllers, and actuators. A PCL uses one of six fundamental control strategies to design a controller, i.e., feedback control, feedforward control, cascade control, split-range control, ratio control, and override select control. All these techniques can be represented by four types of variables, i.e., the setpoint (SP), process variable (PV), control output (CO), and control parameters (CPs). For the desired SP value of the process output, a controller calculates the CO value from the SP and PV values measured by sensors at the process output and applies a correction based on a control algorithm with CPs. The CO value is transmitted to an actuator to adjust the process output. An HMI helps an operator monitor the PV value and control the SP and CP values.

An inconsistent PCL can be identified by a control algorithm model. To obtain an accurate model of a control algorithm, it is necessary to derive a mathematical relationship from PLC or DCS control logic to consider internal parameters such as the range limit, rate limit, and sampling interval. In general, well-known controller models, such as PID controllers, can obtain high-accuracy numerical models using data-based grey-box model estimation or neural networks. We discuss a PID controller as an example in Chapter 3.

2.2 Manipulating PCL Components

The manipulation of PCL components is accomplished through signal injection during each component's signaling process or parameter modification in the middle of data processing.

First, the SP value can be tampered by the parameters of a

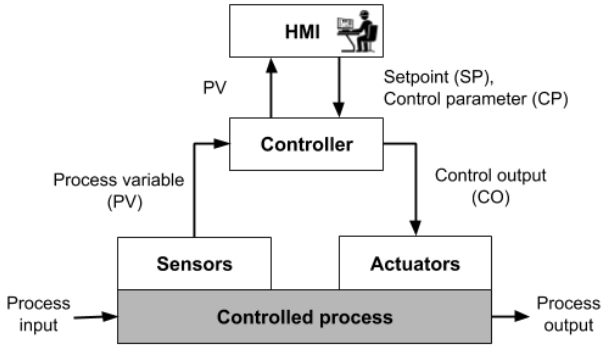

Figure 1: Process control loop model

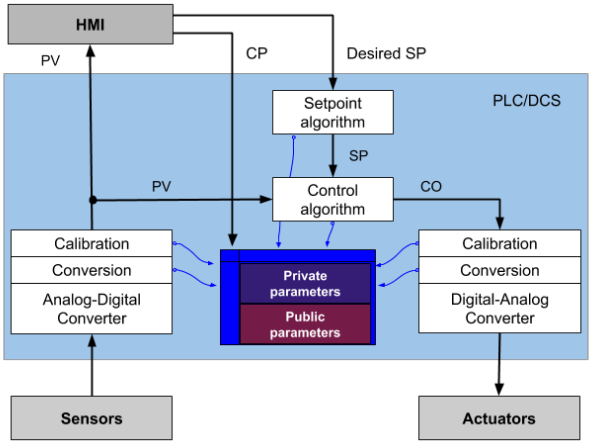

Figure 2: Control logic parameters of PCL components

setpoint algorithm shown in Fig. 2. In general, when an operator sets the desired SP value, a controller gradually increases or decreases the SP output through limiting the rate of the SP value for system safety. For example, if the desired SP value of turbine speed is higher than the current value, the SP is gradually increased according to a predetermined slope. This leads to a decrease in physical fatigue.

Next, the PV and CO values can be manipulated in signal transmission, data conversion, or a sensor calibration process. However, the manipulation of the PV and CO during signal transmission rarely occurs in reality because hardware must be installed between a controller and a sensor. The PV, which is an analog input value, is converted into a process value and a calibration step, and the CO is processed in the reverse order of the PV. Hence, the PV and CO values can be tampered by the scaling and biasing parameters for data conversion. For example, a replay attack can be performed by setting the scaling factor to zero and injecting prerecorded normal data to the biasing value. In addition, we can use the parameters for sensor calibration, which is required to linearize the output to the sensor input. In numerous cases, it is limited to the valid range of the sensor after correcting the output for each input section using a piecewise linear function.

Finally, the CP value is a parameter of the control algorithm; for example, the control response of the system can be manipulated by changing the proportional gain, integral gain, or differential gain value of the PID controller. Common control mechanisms, such as PID control, include input and output scaling and bias parameters for general use and can be used as critical attack operation points.

Meanwhile, to implement attack scenarios of various types and difficulties by manipulating the PCL components, it is necessary to change the manipulation order and values of each parameter differently. Let us consider an example of an attack scenario where false reports are sent to an HMI and a controller's output is manipulated arbitrarily. The attack target PV value can be replayed with the prerecorded PV value. In addition, it can be simultaneously changed to a value larger than the current value of the scaling factor of the controller or the process output can be manipulated by injecting an attacker-defined pattern that maximizes system fatigue.

The attacks on each PCL component are performed in the order of pre-configuration, false data injection, and post-configuration. First, the pre-configuration is required to configure system settings before false data injection. This enables a manipulated value to be immediately applied to a target component and then false data injection begins. The false data injection step is the phase during which data manipulation actually occurs. After all data injection operations are completed, all modifications are restored to the normal condition in the post-configuration phase and no other operation occurs until the control state of the PCL becomes normal. This is for the clear and easy analysis of the impact on each operation.

In a false data injection phase, we can change the target value of PCL components to a specific value or slightly increase or decrease the value according to a user-defined pattern by referring to the current value. This makes it possible to implement attack scenarios of various difficulty levels by maintaining controller consistency at a certain level or higher.

False data injection in real systems can be achieved by forcing another value to a critical data point or embedding malicious control logic. Some DCS controllers have externally accessible public parameters for the optimization of control logic, and data injection is possible only by changing the public parameters without changing the control logic or adding hardware. In many cases, however, the control logic parameters of most PLCs and DCSs are private parameters that cannot be changed over a network. Hence, it is necessary to modify private parameters to refer to the memory value that can be accessed externally. False signal injection is only possible through the installation of additional hardware [2].

2.3 Attack Effects of PCL Components

The goal of this study is not to reproduce attacks but to simulate the result of the attacks. Table 1 shows the possible attacks that can occur through PCL manipulation.

We can simulate multiple attack scenarios by simultaneously or sequentially manipulating multiple PCL components. Several PCL components can be manipulated for each process control loop used in each system. In general, the SP, CO, and PC are used to modulate actuator control commands, and the PV is mainly used to hide an attack by modulating monitoring information. The accessible PC varies depending on the configuration of a manufacturer and control logic.

2.4 Validation of the PCL Attack Effects

The PID controller simulation is used to validate the effectiveness of attacks on PCL components. All attacks performed

Target	Attack Effects	PCL
Sensor	- Physical access to equipment change and configuration manipulation. - Sensor circuit breakdown through electromagnetic wave injection. - False signal injection via sensor communication interface.	PV
Actuator	- Change configuration manipulation through physical access. - False signal injection via communication interface. - Change settings via the management interface (portable maintenance device).	CO
PLC, DCS	- Manipulate set-points, actuator control commands, control parameters, etc. through control logic changes.	CO, PV, CP
HMI	- Set-point change by acquiring HMI access authority. - False monitoring information injection via HMI communication interface.	SP, PV

Table 1: Mapping PCL manipulation to attack effects

for the purpose of data collection ultimately change the actuator output and the process output. Therefore, it is possible to identify the increase in the effectiveness of the attack by evaluating the inconsistency of the PCL. That is, we can obtain the residues between the CO value changed by forgery and modification of the SP, PV, and CP values and the CO value estimated from the simulation model of the PCL. The PCLs in our testbed are implemented by a PID control algorithm. We used the actual collected data to evaluate the effectiveness of the attack by comparing the estimated results through the simulation of the PID controller in Appendix A. The values are used as the labeling information of attack data.

3 A Programmable CPS Testbed with Real-Time HIL Simulator

To create a richer dataset by utilizing multiple small-size control systems, we constructed a testbed in which three independent real industrial control systems ware interconnected via a real-time HIL simulator, which has a 500 MW steam power plant model balancing with 100 MW pumped-storage hydro plant model. Moreover, to minimize the effort involved during dataset collection, the testbed could be run automatically according to normal and attack scenarios and a simulator was developed for each PID process to provide information that could help data analysis.

3.1 Overall Architecture

In numerous cases, small testbeds for research purposes have a limited number of changeable PCL components and a simple control process. This limits the reproducibility of various scenarios. Emerson's boiler control system, GE's turbine control system, and FESTO's water treatment control system are used, which are constructed in small sizes by utilizing components that are actually used in industrial environments.

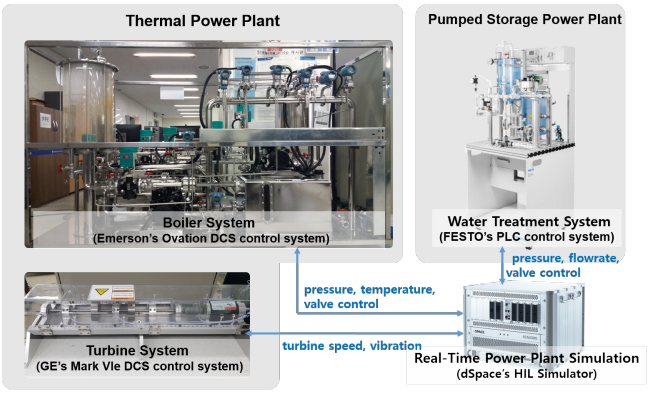

Figure 3: Programmable CPS testbed

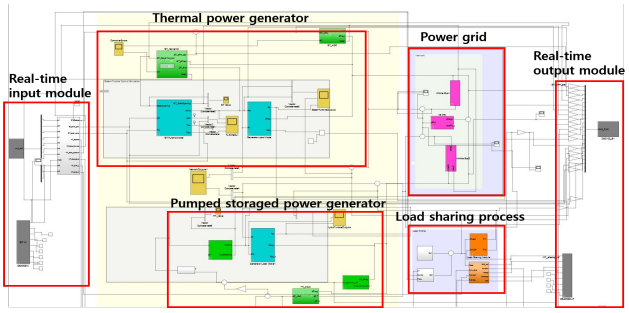

Figure 4: Simulink model of HIL simulator

We built a single complex plant simulation system with various PCL components by linking the abovementioned three independent control systems with the HIL simulator. The Emerson boiler control system and the GE turbine control system simulated the thermal power plant in conjunction with the HIL simulator, and the FESTO water treatment control system simulated the pumping power plant in conjunction with the HIL simulator. The thermal power plant and pumped-storage power plant modeled in the HIL simulator were located in the same area. They were connected to an infinite bus and had a combined power generation environment model in which one input load was distributed and developed at each power plant.

- HIL power plant simulator: An HIL-based simulator was developed to combine three control systems (boiler, turbine, and water treatment) to form a combined power generation system. To control power generation according to input load, it is necessary to determine the valve opening/closing rate of the thermal power plant and pumped-storage power plant to determine the valve opening/closing rate of each control system (boiler and water treatment systems).

- GE turbine system : This is a control system for the turbine speed control and vibration monitoring of the thermal power plant. The RPM of the actual turbine

control system is controlled according to the frequency of the HIL simulator.

- Emerson boiler system: This a system for controlling boiler pressure, temperature, and water level in the thermal power plant. It controls the opening and closing rate of the main valve of the actual boiler control system according to the steam valve opening rate of the thermal power plant in the HIL simulator. The modified pressure and temperature values of the main piping are transmitted to the HIL simulator in real time to determine the amount of generated power.

- FESTO water treatment system: This is a system for controlling the water level of the pumped-storage power plant. It controls the opening and closing rate of the water valve of the actual water treatment system according to the opening and closing rate of the pumping station in the HIL simulator. Accordingly, the hydraulic pressure, flow rate, and water level of the upper water tank (dam) are transmitted to the HIL simulator in real time to determine the amount of generated power.

3.2 HMI Automator

As shown in Fig. 5, we have developed a tool to schedule HMI tasks for a long period without human intervention. Our system is a semi-automated system with an HIL simulator. Therefore, humans have to set the process variables, such as the SP for the planned process output.

In this case, there are several practical limitations to reproduce various normal and abnormal situations over a long period, and it is difficult to maintain consistency each time when an attack is performed in the same situation. In addition, data labeling by operators can degrade the accuracy and reliability of data.

The implemented tool sets the SP value at a scheduled day and time for each PCL. Scheduled tasks can be executed

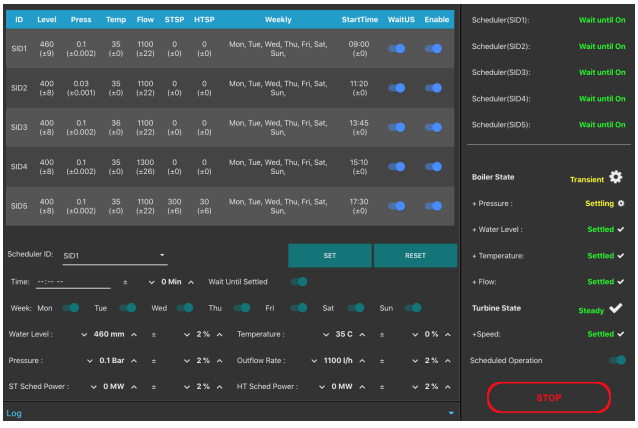

Figure 5: HMI Automator

only in the steady state of the PCL. In addition, the task start time and SP value can be randomly changed within a predetermined range so that various operation scenarios can be generated under the same setting conditions. As the time of occurrence of the control event generated by the scheduler, the SP value, and stabilization can be utilized as metadata, the reliability of PCL data as learning data is improved and the post analysis of PCL data is facilitated.

3.3 Attack Tool with PCL Manipulation

We have developed an attack tool that remotely controls the PCL component physically implemented in the PLC/DCS system according to an attack scenario and records the details of the attack to label data. Through attack task scheduling, the values of the PLC/DCS attack point on the OPC are sequentially manipulated and the attack active command is transmitted to the controller-in-the-middle [2] for signal manipulation. After the attack task for one PCL component is terminated, the PCL waits until it reaches the steady state, thereby reducing the interference between PCL attacks and facilitating the analysis of the attack effect. In a few cases, multiple tasks can be executed simultaneously to allow for multiple attacks regardless of whether they are steady state or not.

System	Process	PCL components	
Water treatment	Level control	SP	Low and High level set-points
		PV	level sensor
		CO	Pump, Discharge valve
		PC	Analog I/O calibration
Turbine	RPM control	SP	RPM set-point
		PV	RPM sensor
		CO	Turbine
		PC	Analog I/O calibration
Boiler	Pressure control	SP	Main tank pressure set-point
		PV	Pressure sensor
		CO	Pressure control valve
		PC	PID gain, Analog I/O calibration
	Return tank level FF control	SP	Return tank level set-point
		PV	Level sensor, Discharge flow sensor
		CO	Level control valve
		PC	PID gain, Analog I/O calibration
	Discharge flow control	SP	Return tank flow set-point
		PV	Discharge flow sensor
		CO	Flow control valve
		PC	PID gain, Analog I/O calibration

Table 2: A part of the PCL components in our testbed

In our testbed, we have identified 26 PCL components that can be manipulated in the 5 important PCLs that affect the entire system (Table 2). The testbed uses a PID control algorithm and has a complex configuration including feedback, feedforward, and cascade. In particular, boiler systems have many PCL components compared to the turbine and the water treatment control system, including all of these complex configurations.

4 Attack Scenarios on CPS Testbed

We can express various benign scenarios and attack scenarios by manipulating PCL components in the programmable CPS testbed. For better understanding, we present a benign scenario through random changes in the SP and suggest attack scenarios through CO, SP, PV changes in the water treatment system and the boiler system.

4.1 Benign Scenarios

A normal operating scenario assumes that an operator routinely operates the control facility via the HMI. The operator monitors the current sensor PV value displayed on the HMI and changes the SPs of various control devices to operate the control facility. When the control process moves out of the normal range, physical values, such as pressure and water level, are changed abnormally and the hybrid control system enters an abnormal operating state.

Through experiments, we confirmed the normal range of the SP where the entire process was stable while changing the value for each SP (Table 3). We used the HMI operation task scheduler to periodically set the SPs to random values within the normal range to represent the benign scenario.

System	Process control loop	Set-points	Min/Max	Normal range
Water treatment	Level control	Low level set-point	0/100	0∼20
		High level set-point	0/100	70∼90
Turbine	RPM control	RPM set-point	0/4000	300 3000
Boiler	Pressure control	Main tank pressure set-point	0/10	0 ∼2
	Return tank level FF control	Return tank level set-point	0/1600	350 ∼450
	Discharge flow control	Return tank flow set-point	0/2500	800∼1200

Table 3: Normal range of the set-points

4.2 Attack Scenarios

An abnormal operating condition implies that a few control facilities are out of the normal range and operate in an unpredictable state owing to an attack or device malfunction. We can express various attack scenarios by combining attack time, PCL components, and attack methods on the power plant testbed. To help understand the attack model, we present attack scenarios in the water treatment system and the boiler system.

4.2.1 Scenario 1: Pump CO Forgery Attack

In the benign scenario, when the power load on a combined power plant is low, power can be supplied only through thermal power generation. Therefore, the pumped-storage hydroelectric power station stops power generation and drives the pump to raise the water level of the dam.

Attack scenario 1 changes the S1-level-CO1 at the PCL components to make the pump malfunction and simultaneously changes the S1-level-PV1 to modulate the level monitoring signal transmitted to the HMI (Table 4). The PCL component, S1-level-CO1, refers to the pump control command (CO1) in the level control process (level) of the water treatment system (S1). The pump control command is the voltage across the pump motor, 0 to 10 V. The PCL component, S1-level-PV1, refers to a level sensor measurement (PV1) in the level control process (level) of the water treatment system (S1).

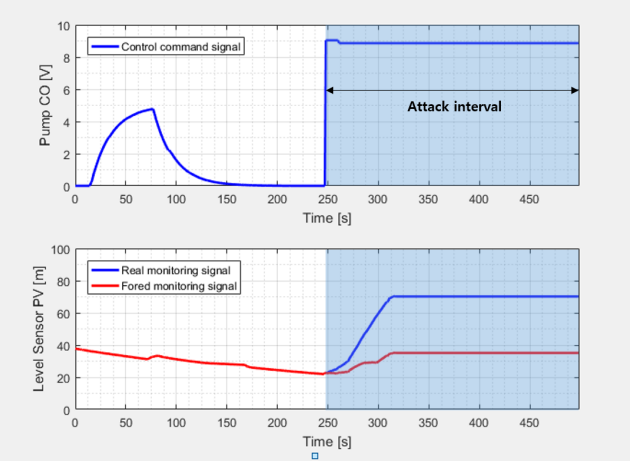

Figure 6: Pump CO Forgery attack (Scenario 1)

It is possible to modulate S1-level-CO1 by changing ***WaterpumpGain*** and ***WaterpumpBias*** among the control logic symbols of the PLC that control the water treatment system. ***WaterpumpGain*** and ***WaterpumpBias*** are the gain and bias of the pump control command CO, and they are set as 1 and 0 in the steady state, respectively, so that CO is not modulated. When the attack starts, ***WaterpumpGain*** changes from 1 to 0 and ***WaterpumpBias*** changes from 0 to 10 to fix the pump control voltage CO at 10 V. Therefore, the pump motor continues to operate at the maximum power and the water level of the dam continuously rises and reaches the dangerous level. In the first graph shown in Fig. 6, the voltage applied to the pump shows that 10 V is continually output when the attack starts.

If the operator is monitoring the dam level through the HMI screen, the attacker can modulate S1-level-PV1 to prevent the operator from recognizing the occurrence of the attack. To modulate level sensor measurement, S1-level-PV1 can be modulated by changing ***LevelSensorGain*** and ***LevelSensorBias*** among the PLC's control logic symbols. ***LevelSensorGain*** and ***LevelSensorBias*** are the gain and bias of the level sensor value and they are set as 1 and 0, respectively, in the steady state, so that the PV is not modulated.

When the attack starts, the PV value is linearly reduced to hide the rise in water level. The PV is decreased gradually by linearly changing ***LevelSensorGain*** from 1 to 0.5. The second graph in Fig. 6 shows the actual water level of the dam

System	Process	PCL components	Symbol of control logic	Attack metheod	Normal	Attack
water treatment(S1)	Level	S1-Level-CO1 (pump control command)	WaterpumpGain	A1) Constant	1	0
			WaterpumpBias	A1) Constant	0	10
		S1-Level-PV1 (level sensor measurement)	LevelSensorGain	A2) Linear	1	(1:-0.01:0.5)
			LevelSensorBias	-	0	0

Table 4: Configuration of the attack scenario 1

System	Process	PCL components	Symbol of control logic	Attack metheod	Normal	Attack
Boiler(S3)	Pressure	S3-Pressure-SP1 (press set-point)	PressSPGain	A1) Constant	1	0
			PressSPBias	-	0	0
		S3-Pressure-PV1 (pressure sensor measurement)	PressSensorGain	A1) Constant	1	0
			PressSensorBias	R) Replay	0	replay signal

Table 5: Configuration of the attack scenario 2

and the false water level information sent to the HMI.

4.2.2 Scenario 2: Pressure SP Forgery Attack

In the benign scenario, the pressure SP of the boiler system is set to a value between 0 and 2 to control the opening and closing rate of the piping valve. An attacker can perform an attack by setting the pressure SP high in the press control process to break the pipe at high pressure, or to set the SP low to close the piping valve.

The attack scenario 2 change the S3-pressure-SP1 at the PCL components to make the valve malfunction and simultaneously change the S3-Pressure-PV1 to modulate the level monitoring signal transmitted to the HMI (Table 5). The PCL component, S3-pressure-SP1, refers to the Press set-point(SP1) in the press control process (Press) of the boiler system (S3). The press set-point is the normal range is 0 to 2. The PCL component, S3-Pressure-PV1, refers to a pressure sensor measurement (PV1) in the press control process (Press) of the boiler system (S3).

It is possible to conduct S3-pressure-SP1 by changing ***PressSPGain*** and ***PressSP*** in the control tags of the PLC. ***PressSPGain*** and ***PressSPBias*** are the gain and bias of the pressure SP, respectively. When the attack starts, ***PressSP-Gain*** changes from 1 to 0 and ***PressSPBias*** maintains a value of 0. Therefore, the piping valve is closed and the pressure applied to the piping is rapidly lowered. The first graph shown in the Fig. 7, the Press SP becomes 0 when the attack starts.

If the operator is monitoring the pressure change of the pipe through the HMI screen, the attacker can modulate S3-pressure-PV1 to prevent the operator from recognizing the occurrence of the attack. To modulate pressure sensor measurement, S3-pressure-PV1 can be modulated by changing ***PressSensorGain*** and ***PressSensorBias*** among the PLC control logic symbols. ***PressSensorGain*** and ***PressSensorBias*** are the gain and bias of the level sensor value and they are set as 1 and 0 respectively, in the steady state so that PV is not modulated. When the attack starts, the PV value for a certain period is replied to forge the pressure monitoring value. To do this, the ***PressSensorGain*** was changed from 1

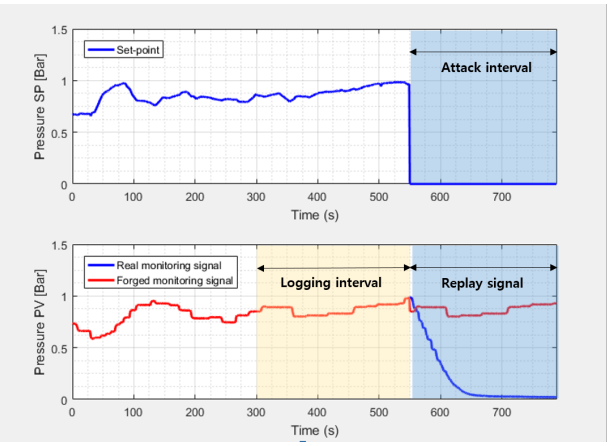

Figure 7: Pressure SP forgery attack (Scenario 2)

to 0 to remove the actual PV value and add a reply signal to the ***PressSensorBias*** to modulate the monitoring signal. The second graph shown in Fig. 7 shows the actual press of the pipe and the false press information transmitted to the HMI. It can be confirmed that the previous pressure sensor signal is replayed when the attack starts.

5 Conclusion and Future Work

Although there are a growing number of studies for CPS security, there is still a lack of open data sets that can be used for research. Our testbed is designed to generate accurate data sets for various scenarios while minimizing human effort.

Our goal is to create datasets for CPS security researchers working for anomaly detection. We will develop and release CPS datasets using this testbed in the future.

References

[1] Chuadhry Mujeeb Ahmed, Jianying Zhou, and Aditya P. Mathur. Noise matters: Using sensor and process noise

fingerprint to detect stealthy cyber attacks and authenticate sensors in cps. In *Proceedings of the 34th Annual Computer Security Applications Conference*, ACSAC '18, pages 566–581, 2018.

[2] Seungoh Choi, Woomyo Lee, Hyeok-Ki Shin, Jeong-Han Yun, and Sin-Kyu Kim. POSTER: CPS security testbed development using controller-in-the-middle. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (AsiaCCS)*, pages 829–831, 2018.

[3] Seungoh Choi, Jeong-Han Yun, and Sin-Kyu Kim. A comparison of ics datasets for security research based on attack paths. In *The 13th International Conference on Critical Information Infrastructures Security (CRITIS)*, 2018.

[4] Wenbo Ding and Hongxin Hu. On the safety of iot device physical interaction control. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 832–846, 2018.

[5] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. A systematic framework to generate invariants for anomaly detection in industrial control systems. In *Network and Distributed Systems Security (NDSS)*, 2019.

[6] David Formby, Preethi Srinivasan, Andrew Leonard, Jonathan Rogers, and Raheem Beyah. Who's in control of your control system? device fingerprinting for cyber-physical systems. In *Network and Distributed System Security Symposium (NDSS)*, 2016.

[7] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A dataset to support research in the design of secure water treatment systems. In *The 11th International Conference on Critical Information Infrastructures Security (CRITIS)*, 2016.

[8] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H. Hartel. Through the eye of the plc: Semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, ACSAC '14, 2014.

[9] Antoine Lemay and José M. Fernandez. Providing scada network data sets for intrusion detection research. In *Proceedings of the 9th USENIX Conference on Cyber Security Experimentation and Test*, CSET'16, 2016.

[10] Nicholas R Rodofile, Thomas Schmidt, Sebastian T Sherry, Christopher Djamaludin, Kenneth Radke, and Ernest Foo. Process control cyber-attacks and labelled datasets on s7comm critical infrastructure. In *Australasian Conference on Information Security and Privacy*, pages 452–459, 2017.

[11] Jeong-Han Yun, Yoonho Hwang, Woomyo Lee, Hee-Kap Ahn, and Sin-Kyu Kim. Statistical similarity of critical infrastructure network traffic based on nearest neighbor distances. In *Research in Attacks, Intrusions, and Defenses (RAID)*, pages 577–599, 2018.

[12] Muwei Zheng, Hannah Robbins, Zimo Chai, Prakash Thapa, and Tyler Moore. Cybersecurity research datasets: Taxonomy and empirical analysis. In *Proceedings of the 11th USENIX Conference on Cyber Security Experimentation and Test*, CSET'18, 2018.

Appendix A. Validating the Attack Effects on PID Controller

The CO value of the PID controller can be estimated through a recursive neural network using the SP and PV values. The PID controller calculates the proportional, derivative, and integral terms for an error input, $E[k] = SP[k] - PV[k]$, and outputs the weighted sum of the three terms. As the PID controller in a PLC/DCS is a discrete-time system with a sampling interval T, the differential equation of the PID controller can be represented as

$$\begin{aligned} CO[k] =& CO[k-1] + \left(K_P + K_I T + \frac{K_D}{T}\right) E[k] \\ & - \left(K_P + \frac{2K_D}{T}\right) E[k-1] + \frac{K_D}{T} E[k-2] \end{aligned} \tag{1}$$

where the coefficients (K_P, K_I, K_D) are the gains of the PID controller. In many cases, the PID gains and sampling interval of Eq. 1 are not known directly, and furthermore the PID control loop includes additional blocks such as saturation and rate limiter. The saturation block limits the input value to the upper and lower saturation values, and the rate limiter limits the rate of change of input signal.

To obtain the models of PID feedback, feedforward, and cascade control loops, we used the Nonlienar AutoRegressive Exogenous (NARX) feedback neural network Toolbox on MATLAB, which is mainly used for time series data modeling. This is because the PID control mechanism in Eq. 1 has a time-delayed recursive model of the controller output with error inputs as shown in Fig. 8.

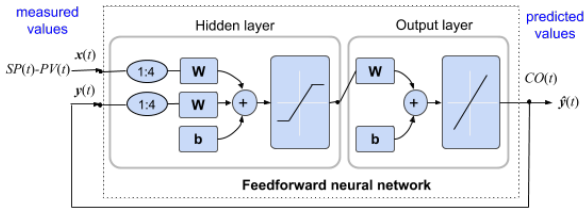

Figure 8: Recurrent neural network (NARX network closed-loop architecture) for modeling a PID controller

The NARX neural network regresses past independent n-sample input values and past m-sample output values to predict the next output value as shown in Fig. 8. We set n and m to four at least greater than two From Eq. 1 and use 10 neurons in the hidden layer. The inputs $x(t)$ of the neural network for modeling a feedforward control must be two-dimensional with a process error and a feedforward PV value. A cascade control loop has a structure in which two PID blocks are connected successively, so that both the input and the output of the learning network are two-dimensional.

Fig. 9 shows the control output of the feedforward PID controller of the liquid level using the training and testing data collected for two days.

The closed-loop network outputs a relatively large error in the steady-state during the first 22 h. However, it could validate the attack effects for the PLC components when false data injection with three types of data patterns occurs in one of SP, PV, and CO, as shown in Fig. 9.

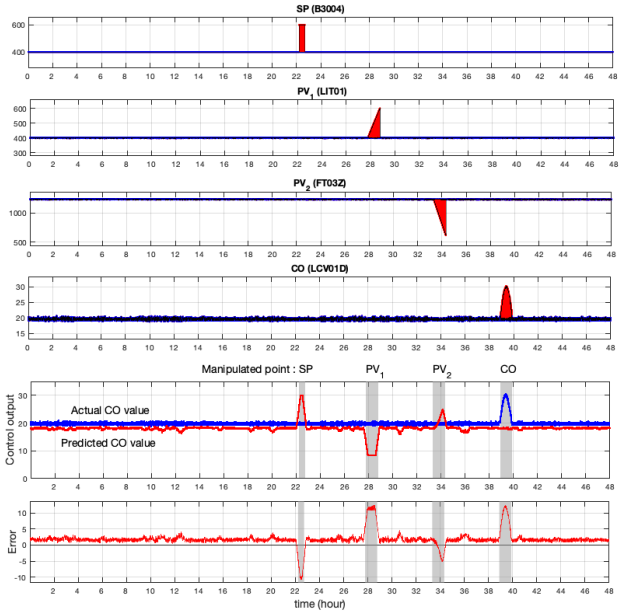

Figure 9: Residue of the estimated CO using PID simulation for the actual data collected for two days

Appendix B.
PCL Dataset Description

A data sample is collected every second for a set of 9 PCLs, consisting of 9 SPs, 19 COs, and 40 PVs. Under normal operating conditions, they are stored on the time-series database about 66 MB per day, and Fig. 10 shows a portion of the data collected over a week.

Meanwhile, all attacks performed for the purpose of data collection were logged with the labels in Table 6.

Figure 10: A portion of the data collected for a week

Label	Description	Format
Start Time	time when an attack scenario starts	POSIX time
End Time	time when an attack scenario ends	POSIX time
Target PCL	target PCL name	String
Target Variables	PCL variables (i.e. SP, PVs, CO)	String array
Attack Intent	Description of the intended affect of the attack	String
Attack Methods	data injection method	String array
Attack Values	injected values under the attack	Float array
PCL Errors	error between the SP and PV for all PCLs	Float array
PCL Inconsistency	residue of PID control output for all PCLs	Float array

Table 6: Attack labels