Teaching ICS Security in Blended Classroom Environment

Petr Matoušek
Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic
matousp@fit.vutbr.cz

Ondřej Ryšavý
Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic
rysavy@fit.vutbr.cz

***Abstract*—Demonstration of real industrial equipment, manufacturing processes, and control communication is essential for students of technical universities and colleges to improve their technical competencies and knowledge. To build an industrial control systems (ICS) lab with real hardware and control software usually requires a significant amount of finances. However, the lab equipment is usually accessible only to a limited number of students. A possible alternative is a blended classroom environment that combines real inexpensive devices connected with a configurable simulation environment. In such an environment, groups of students can create their own experiments and observe a near real-life behavior of an ICS system.**

In this paper we demonstrate how a blended ICS classroom can be built of the Factory I/O 3D software simulator and real UniPi PLC devices equipped with digital and analog inputs. Using such an environment, students may design a set of non-trivial manufacturing scenarios, e.g., a production line, and make experiments with ICS components. The paper presents the topology and equipment of the blended ICS classroom. We also introduce two lab scenarios focused on the security of ICS processes and analysis of Modbus communication in the ICS environment.

***Index Terms*—simulation, industrial system, blended classroom, ICS security, ICS testbed**

I. Introduction

Industrial Control Systems (ICS) have become more important today due to the massive automation and digitization of manufacturing processes and the increase of intelligent control in power, gas and water treatment facilities. An ICS system controls industrial processes, robotic manufacturing, home and office automation systems, or intelligent transport. It interconnects physical processes with intelligent control devices like Programmable Logic Controllers (PLCs), Remote Terminal Units (RTUs), or Intelligent Electronic Devices (IEDs) that are usually connected to a Human Machine Interface (HMI) and operated by a Supervisory Control and Data Acquisition (SCADA) server.

The operation of an ICS/SCADA system significantly differs from traditional IT services build upon TCP/IP both in design and requirements. Common ICS protocols are proprietary (Siemens S7, Modicon Modbus, OPC) or standardized by ISO/IEC (Goose, MMS, or DLMS). They are mostly implemented over the data link layer (layer 2) or on the application layer (layer 7) of the ISO stack in contrast to IT services. ICS systems are usually a part of the critical infrastructure. Thus, they must maintain a high level of system availability and operational resilience [1] which is not required by IT services. Security ICS systems was traditionally implemented by dedicated lines and systems that were physically separated from the outer world. Today, with the convergence of IT and OT technologies, ICS/SCADA systems are vulnerable to common attacks known from Internet world. Due to the limited computational power of ICS/SCADA devices and high requirements on the system performance with low delays, it is not easy to implement security measures to ICS/SCADA systems as it happened in IT services.

The importance of ICS/SCADA systems requires that students of technical schools oriented on IT, electrical engineering, automation, or robotics should be acquainted with the principles, features and behavior of these systems. When teaching classical networking courses, student can easily explore behavior of network devices and services using the lab equipment that includes switches, routers, or WiFi access points. Common network applications and services can be installed on common laptops and desktops, or in the virtualized environment using Virtual Box or VMWare software where students may carry out experiments.

Implementation of an ICS/SCADA system in the classroom environment is not so straightforward and has two major limitations. First, building an ICS system as an off-the-shelf solution is expensive and not easily extendable [2]. On the other hand, to design and assemble a functional ICS system using component parts (a so called "Do it yourself", DIY approach) is a time consuming process that requires particular skills and expert knowledge. The second limitation relates to the physical processes that are controlled by an ICS system and that are not easy to emulate.

Holm et al. [3] mention three basic approaches how to create an ICS testbed. The first one includes *real hardware and software*. Such testbed provides a very high degree of fidelity but its reconfiguration and maintenance is difficult. Of course, it costs highly. The second option employs *simulation*. Simulation models can be easily reconfigured, maintained and

978-1-7281-9327-4/21/$31.00 ©2021 IEEE

are scalable. However, such models have limited fidelity in comparison to a real world scenarios. The third approach is based on virtualization. The virtual container emulates a real environment where we can run multiple instances of system processes and use an actual software instead of simulation. Virtualization can be combined with real physical inputs as showed in our solution.

A. Structure of the text

The paper is structured as follows. After Introduction we give an overview of the related work where we mention various approaches of how to build an ICS testbed. Section III explains the background of SCADA system and Modbus protocol. Section IV describes our virtual testbed with all components that include a Factory I/O simulation software, PLCs with UniPI technology and an HMI interface programmed in Python. Section V gives an example of two lab scenarios that can be run on our testbed. The last section concludes our work.

B. Contribution

This work has two main parts: first, we present how to build a near real-world ICS testbed using virtualized environment and real PLCs. The entire testbed can be purchased with costs under 2600 Euros (prices in 2020). The second contribution includes two lab scenarios: (i) control and monitoring of the factory production line, (ii) detection of cyber security attacks against Modbus communication. Detailed description of the environment with configuration files and user manual will be available at the project web site.

II. Related Work

In this section we give a short overview of existing ICS testbeds and experiences obtained by their authors. We mention both testbeds with real hardware and software as well as testbeds with virtualized components.

In 2015, Holm et al. [3] published a survey of 30 existing ICS testbeds from over the world. The authors observed their architectures and evaluated standard requirements as fidelity, repeatability, accuracy, and self execution of tests. However, a detailed description of each of the testbeds is missing.

Green et al. present in [4] their experience with building an ICS testbed with a range of devices (PLCs, HIMs, RTUs) and software. They describe the testbed design and architecture together with experience obtained. There findings are expressed in ten points. These points have been later updated in [2]. The updated paper also describes a water treatment plant testbed built with physical components. The authors also describe a training and prototyping testbed created using FactoryIO software and real Siemens and Allan Bradley PLCs.

A modular design of an ICS testbed for security research is introduced in [5]. The authors define five layers: management, user, infrastructure bridge, experimental and remote access. For each layer they present a list of possible implementation tools that include hardware devices as well as commercial or open source software. Their study gives a good overview of available technologies for building a virtualized ICS testbed.

Behavior of a virtualized ICS testbed may differ from real devices. This was demonstrated by Alves et al. [6] who examined the fidelity of a virtual SCADA testbed composed of an OpenPLC software running in the virtual machine under Linux to a physical testbed where an OpenPLC was running on the hardware UniPI board. The authors noticed that the virtual OpenPLC processes the logic faster than the physical one due to a more powerful hardware. On the other hand, the response of the virtualized PLC was slower than physical because of the task scheduler in the operating system.

A combination of a virtualized system and real physical devices in building an ICS testbed was presented in NISTIR 8089 report [1] where the authors showed three scenarios: (i) a chemical plant with a continuous process and slow dynamics implementing the Tennessee Eastman problem [7], (ii) a cooperative robotic assembly line for smart manufacturing with a discrete state process and fast dynamics, and (iii) an intelligent transport system. Their report focuses on testing the system performance and resilience during the cyber attacks.

One of the most important tasks related to ICS testbeds is how to emulate physical processes that are controlled by ICS systems. Gillen et al. [8] describe a full-scale ICS testbed that emulates a cooling system of Oak Ridge National Laboratory's supercomputer. The testbed is equipped with PLCs, HMI, sensors and actuators. In order to emulate an actual physical system, they obtained a 30-days historian data from the production system. They processed and converted historian data into the emulators that were running on Raspberry PIs. Their experiments with the testbed showed similar behavior.

To build and maintain a full-scale physical system is very expensive and unfeasible for teaching purposes. The solution is to create an ICS testbed with a small-scale physical model where students learn principles of an ICS system and provide their own experiments. Here, we mention successful implementations of such testbeds. Ahmed et al. [9] created in the University of New Orleans a testbed with three physical processes: a gas pipeline, a power distribution system, and a wastewater treatment system. Their paper describes how the testbed is used for teaching PLC programming, forensics, etc.

An ICS testbed with small-scale physical system for power energy was presented by S. Mocanu [10]. The testbed is used in Grenoble for both teaching and research. Researchers in Singapore University created two full-scale ICS testbeds for research and training on ICS security: the smart grid testbed EPIC [11], [12] and the water treatment testbed SWaT [13].

A virtualized ICS testbed for training and teaching was also implemented by Čeleda et al. [14]. Their testbed contains PLCs built on UniPi platform, I/O modules, HMI, linear motor and communication gateway. Their platform is primarily used for teaching and training cyber security of ICS systems. They also designed Cyber Security of ICS system course.

This paper presents a SCADA testbed built of several PLCs, I/O Factory simulation software, and a HMI interface programmed using Python. It is a scalable and open solution that provides students an opportunity to create new physical processes and configure control operations of the system.

III. Preliminaries

A. SCADA system

Supervisory control and data acquisition (SCADA) is a (distributed) control system architecture for industrial processes, including manufacturing, process control, power generation, fabrication, and refining. The SCADA principles can be used for building both large and small systems. An example of small SCADA system architecture is shown in Fig. 1.

- *Supervisory control and data acquisition (SCADA) system* includes five components [6]: a physical system (process), wire bridge, human machine interface (HMI), and programmable logic controllers (PLCs). A physical process is connected via the wire bridge with a PLC that monitors and controls a state of a physical process using sensors and actuators. The PLC is connected to an HMI interface that enables an operator to see the state of the system on a visual display and control its behavior.
- *Human Machine Interface (HMI)* is a software application that displays information about the current process state. It also enable an operator to control processes using simple control elements (buttons,touch screen). HMI receives data about running processes from PLCs using ICS control protocols, e.g., Modbus.
- *Programmable Logic Controller (PLC)* is an industrial digital controller that is connected to the physical process unit. It constantly monitors the state of its input lines (sensors) and makes decision on what to do with its output lines (actuators) based on a user program. It sends status data via industrial network to the HMI or SCADA server.

When creating an ICS/SCADA testbed, we need to implement all these components as real physical devices and connectors, virtualized applications, or a simulation software. Typically, PLCs and HMI are physical devices while a physical system is simulated or emulated by a software.

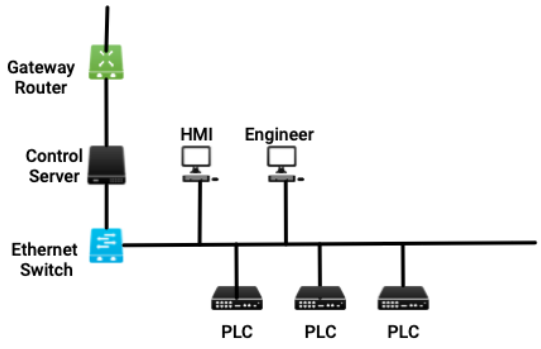

Fig. 1. A local ICS network topology

B. Modbus protocol

The Modbus protocol is one of the most widespread communication protocol in ICS networks. Originally, it was developed for serial (RS232) links. Currently, it is transmitted over TCP. The Modbu/TCP communicates using a master/slave architecture. The master node pulls the information from a slave device. The Modbus network contains a single master node and up to 247 slave nodes. Modbus defines a simple set of operations to read and write coils, registers and file records. A Modbus/TCP communication system uses TCP protocol for transferring Modbus application messages. The standard mandates to use TCP port 502 for the Modbus/TCP server.

The Modbus protocol data unit (PDU) consists of:

- *Function code* - defines the function and therefore also the type of the payload data. When a message is sent from a Client to a Server device the function code field tells the server what kind of action to perform.
- *Data* - the data which meaning should be interpreted according to the function. The data field of messages sent from a client to server devices contains additional information that the server uses to take the action defined by the function code. In the reverse order, the data field can carry the response data or can be empty.

The Modbus communication consists of the exchange of query and response messages called a transaction. Once the request of the query message has been processed by a server, a response message is sent back by the server. Depending on the result of the processing the response may indicate the error using an exception function code.

The Modbus protocol provides a set of functions to read from and write to data sources. Following basic types of data sources are defined: Coil, Discrete Input, Input Register, Holding Register, File, Queue. The data sources are addressable using a 16 bit wide pointer. An example of Read Input Registers function is provided in Fig. 2. The function requests to read values of 40 registers starting at address 48. The result is expected to provide 80 bytes of data content as each register has 16 bits.

The Modbus protocol does not provide any form of security to protect the communication against eavesdropping, modification, or insertion.

```
▼ Modbus/TCP
    Transaction Identifier: 10626
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 255
▼ Modbus
    .000 0100 = Function Code: Read Input Registers (4)
    Reference Number: 48
    Word Count: 40
```

Fig. 2. An Example of Read Input Register Function Message

IV. Virtual ICS testbed

In order to enable students to gain experience with industrial systems, we have assembled a virtual ICS environment. To provide perception close to reality the virtual ICS testbed consists of a simulated physical environment with advanced visualization, hardware PLC and HMI components. The virtual testbed can simulate possibly large and expensive real physical environments. Not only the costs are significantly reduced but

the simulated environment is also safe for experiments. The students can freely modify the physical environment simply by loading a new scenario from the predefined collections or build their own from the predefined simulation components. The virtual ICS can represent various small to mid-size industrial control environments. The main components of a single virtual ICS is as follows (Fig. 4):

- Environment simulation software is represented by Factory I/O software[1], which not only simulates the physics of industrial processes, e.g., production lines, but also provides appealing 3D visualization (see Fig. 3). Factory I/O provides a collection of components based on the most common industrial equipment such as sensors, operators, stations, warning devices, walkways, etc. Moreover, the software package offers the possibility to extend the simulation with new components using the provided API.
- The environment is controlled by PLCs. Contrary to the simulated environment, the real hardware PLCs are utilized. To reduce the costs and increase the flexibility we employ PLC units based UniPi Neuron platform[2] and Advantech USB Modules[3]. UniPi is an extension board for Raspberry Pi minicomputer that implements PLC designed for smart homes, commercial and industrial applications. Advantech USB-4750, resp. 4704 are USB data acquisition modules with digital, resp. analog I/O channels that add measurement and control capability. PLCs' digital input and outputs are connected to the PC running factory simulation software using the digital I/O USB module.
- The HMI can run on the standalone computer or on the same machine as the simulated industrial environment. Any HMI software that communicates with PLCs using one of the industrial protocols, e.g., Modbus or DNP3 can be used. In our case, we implemented the HMI using the Python QT library[4].
- We use our ICS testbed primarily for security education and research. Therefore, the testbed exposes networking infrastructure enabling for demonstrating various security scenarios. HMI and ICSs are interconnected by the LAN switch integrated with a router. We used a configurable Mikrotik router that supports port mirroring for monitoring purposes and VPNs for providing remote access.

The detailed list of used components, their parameters, and prices are in Table I. The costs of the entire virtual ICS environment with a single PLC is lower than 2300 euros.

V. Scenarios

The virtual ICS testbed can be used for a number of experiments. In this section, we present two possible scenarios. The first case demonstrates how the testbed can serve students for understanding of specifics of control systems monitoring and management and basic principles of the ICS. The second case focuses on monitoring ICS network traffic and its security analysis.

[1]https://factoryio.com/

[2]See https://www.unipi.technology/products/neuron-3.

[3]See https://www.advantech.com/products/1-2mlkno/usb-4750/mod_43dfaaf0-a44c-4437-a8c8-0f7460c30b26

[4]See https://pypi.org/project/PyQt5/.

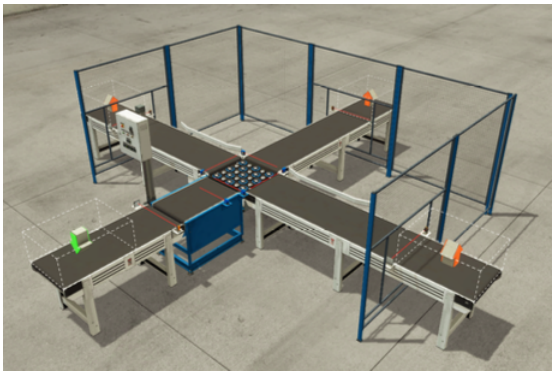

Fig. 3. A screenshot of Factory.IO visualization

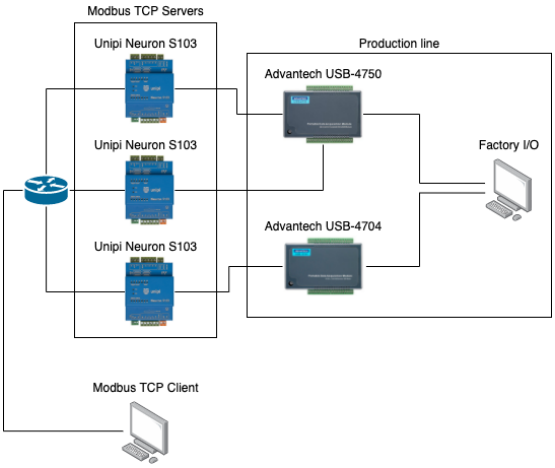

Fig. 4. The topology of Virtual ICS testbed

A. Simulation of Industrial Processes

Using the virtual ICS environment it is possible to create and simulate various industrial systems. We are only limited by the capability of used simulation software that provides a set of predefined components to create the industrial setup. The simulation software consists of a rich library of industrial parts, including robot stations, sensors, conveyors, loaders, operators, elevators, and many others. The software comes with predefined setups of different complexity or it is possible to create a new virtual factory using the available parts. For educating students in the ICS system design we can use the testbed for the following training scenarios:

- *Design of factory processing lines:* students learn how to build a processing line by composing basic components. The students need to be aware of physical conditions and limitations when deploying the hardware parts. On the other hand, they are shed from too many details needed to solve in real deployments.
- *Control of the factory processes:* students are asked to write a software to control the factory processes. They

Item	Parameters	Price (EUR)	Pieces	Total (EUR)
Factory.io	The 3D factory simulation software.	695	1	695
Mini PC	An Intel NUC Kit mini PC to run the simulation software and the HMI.	400	1	400
Router	Mikrotik Routerboard RB750Gr3.	60	1	60
I/O USB Module	Advantech USB-4750-BE I/O module.	240	2	480
PLC	UniPi Neuron PLC.	320	3	960
Total		2255		3135

TABLE I
THE COMPONENT LIST OF THE ICS TESTBED

are taught principles of PLC programming, ICS control and monitoring. Because of the virtual environment, they can immediately see the effect of the programs written. Since we do not use any major brand of PLC devices, the students are not fixed to a single vendor's environment but are rather exposed to the general principles of PLC programming.

- *Troubleshooting problems and component failures:* as for all computer-based systems, debugging and troubleshooting is an essential skill. Students can test different configurations of the system without being worried about possible damage caused by system malfunctions. Thanks to the advanced visualization the effect of any change in the control software can be immediately observed which provides the necessary feedback improving the learning experience.

The expected learning outcomes of this scenario include understanding basic principles of ICS, cyber-physical components and software control of industrial system processes.

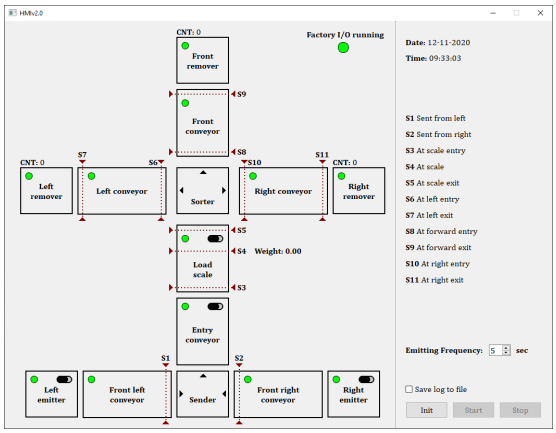

Fig. 5. The simple HMI of the system

An example exercise uses a scenario of a sorting line which sorts emitted objects by weight to various directions (see Fig. 3 for the visualization of virtual factory). It contains two object emitters, several belt conveyors, several sensors, three object removers, two pop-up wheel sorters and one weight scale. Each component can be controlled by an I/O port of the connected PLC. Software part implements a control station program that communicates via Modbus to PLCs of the production line. The control program is written in Python using pyModbusTCP library[5]. Also the HMI (see Fig. 5) is implemented for comfortable control of production line. This HMI is created using PyQT toolkit. Although the HMI supports only basic functionality, it is sufficient to demonstrate the principles of controlling industrial systems. The HMI creates an event log that is useful for providing post-mortem forensic analysis of the system behavior.

B. ICS Traffic Security

Industrial Control Systems often form critical infrastructures such as water, gas or electricity distribution, smart grids, or various types of manufacturing. The virtual testbed can be used for education and experiments with ICS security enabling various paths. For instance, cyber security vulnerabilities can be tested on functional control systems. Students can find and deploy exploits to understand the implications of the vulnerability. For example, the following attacks are easy to deploy. They convey an illustrative demonstration of the implications:

- *Command injection attack.* The attack is executed at the HMI. As a part of the attack, the command for the `Entry` conveyor to stop is sent. It is possible to see that when the conveyor is stopped while the emitted object was on it. The consequence is that it did not get to the weight scale. Thus, the entry sensor on the weight scale does not register the object, and the whole process line is non-functional.
- *Command modification attack.* To demonstrate this attack, the value of the object's weight is modified, which leads to an incorrect sorting function. This attack assumes that the attacker can intercept the communication and alter Modbus messages. Because Modbus does not protect the integrity of the messages the attack is easy to deploy.

Another possible use of the testbed is to monitor an ICS network traffic and create long-term datasets for different simulated systems. Datasets can be a source for application of various anomaly detection methods including statistical profiling, machine learning approaches, etc.

VI. CONCLUSION

Industrial control systems are often considered to be a part of critical infrastructure. Historically, these systems were deployed in the electrical and control systems engineering areas. As modern systems are often complex networked installations, software design and network communication are

[5]See https://pypi.org/project/pyModbusTCP/

more involved. Computer engineers often do not have enough understanding of ICS principles and technology. This paper presented a testbed that aims to provide a suitable environment for computer engineering students to understand the ICS area. The testbed is built around the virtual factory simulated by the Factory.IO software package. The HMI and PLC devices are physical components to provide access to basic control components and their communication. Using the testbed, we proposed two basic scenarios that cover (i) control and monitoring of the factory production line, (ii) analysis of cyber security attacks against ICS communication. The scenarios can be used as hands-on lab exercises, or students can create their own experiments and observe a near real-life behavior of an ICS system.

ACKNOWLEDGMENT

This work is supported by the Brno University of Technology project "Application of AI methods to cyber security and control systems", no. FIT-S-20-6293 (2020-2022) and the National Centre of Competence project "Traffic Analysis and Security Operations for ICS/SCADA (TRACTOR), no. TN01000077 (2019-2020) supported by the Technology Agency of the Czech Republic. The authors also appreciate contribution of Ján Pristaš and Mária Masárová who participated in practical building of the virtual testbed.

REFERENCES

[1] R. Candell, K. Stouffer, and T. Zimmerman, "An Industrial Control System Cybersecurity Performance Testbed," National Institute of Standards and Technology, Tech. Rep. NISTIR-8089, 2015.

[2] J. Gardiner, B. Craggs, B. Green, and A. Rashid, "Oops I Did It Again: Further Adventures in the Land of ICS Security Testbeds," in *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy*, ser. CPS-SPC'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 75–86. [Online]. Available: https://doi.org/10.1145/3338499.3357355

[3] H. Holm, M. Karresand, A. Vidström, and E. Westring, "A survey of industrial control system testbeds," in *Secure IT Systems*, S. Buchegger and M. Dam, Eds. Cham: Springer International Publishing, 2015, pp. 11–26.

[4] B. Green, A. Lee, R. Antrobus, U. Roedig, D. Hutchison, and A. Rashid, "Pains, Gains and PLCs: Ten Lessons from Building an Industrial Control Systems Testbed for Security Research," in *10th USENIX Workshop on Cyber Security Experimentation and Test (CSET 17)*. Vancouver, BC: USENIX Association, Aug. 2017. [Online]. Available: https://www.usenix.org/conference/cset17/workshop-program/presentation/green

[5] B. Green, R. Derbyshire, W. Knowles, J. Boorman, P. Ciholas, D. Prince, and D. Hutchison, "ICS testbed tetris: Practical building blocks towards a cyber security resource," in *13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20)*. USENIX Association, Aug. 2020. [Online]. Available: https://www.usenix.org/conference/cset20/presentation/green

[6] T. Alves, R. Das, and T. Morris, "Virtualization of industrial control system testbeds for cybersecurity," in *Proceedings of the 2nd Annual Industrial Control System Security Workshop*, ser. ICSS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 10–14. [Online]. Available: https://doi.org/10.1145/3018981.3018988

[7] J. Downs and E. Vogel, "A plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245 – 255, 1993, industrial challenge problems in process control. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0098135493800181

[8] R. E. Gillen, L. A. Anderson, C. Craig, J. Johnson, A. Columbia, R. Anderson, A. Craig, and S. L. Scott, "Design and Implementation of Full-Scale Industrial Control System Test Bed for Assessing Cyber-Security Defenses," in *2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 2020, pp. 341–346.

[9] I. Ahmed, V. Roussev, W. Johnson, S. Senthivel, and S. Sudhakaran, "A SCADA System Testbed for Cybersecurity and Forensic Research and Pedagogy," in *Proceedings of the 2nd Annual Industrial Control System Security Workshop*, ser. ICSS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1–9. [Online]. Available: https://doi.org/10.1145/3018981.3018984

[10] S. Mocanu, "ECO-Lab : an open lab for flexible distant education in digital technologies for energy," in *27th EAEEIE Annual Conference*, Grenoble, France, Jun. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01537803

[11] S. Adepu, N. K. Kandasamy, and A. Mathur, "EPIC: An Electric Power Testbed for Research and Training in Cyber Physical Systems Security," in *Computer Security - ESORICS 2018 International Workshops, CyberICPS 2018 and SECPRE 2018, Barcelona, Spain, September 6-7, 2018, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 11387. Springer, 2018, pp. 37–52. [Online]. Available: https://doi.org/10.1007/978-3-030-12786-2_3

[12] A. Siddiqi, N. O. Tippenhauer, D. Mashima, and B. Chen, "On practical threat scenario testing in an electric power ics testbed," in *Proceedings of the Cyber-Physical System Security Workshop (CPSS), co-located with ASIACCS*, June 2018.

[13] A. Mathur and N. O. Tippenhauer, "Swat: A water treatment testbed for research and training on ics security," in *Proceedings of Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater)*, April 2016.

[14] P. Čeleda, J. Vykopal, V. Švábenský, and K. Slavíček, "KYPO4INDUSTRY: A Testbed for Teaching Cybersecurity of Industrial Control Systems," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1026–1032.