# KYPO4INDUSTRY: A Testbed for Teaching Cybersecurity of Industrial Control Systems

Pavel Čeleda
Masaryk University
Czech Republic
celeda@ics.muni.cz

Jan Vykopal
Masaryk University
Czech Republic
vykopal@ics.muni.cz

Valdemar Švábenský
Masaryk University
Czech Republic
svabensky@ics.muni.cz

Karel Slavíček
Masaryk University
Czech Republic
slavicek@ics.muni.cz

## ABSTRACT

There are different requirements on cybersecurity of industrial control systems and information technology systems. This fact exacerbates the global issue of hiring cybersecurity employees with relevant skills. In this paper, we present KYPO4INDUSTRY training facility and a course syllabus for beginner and intermediate computer science students to learn cybersecurity in a simulated industrial environment. The training facility is built using open-source hardware and software and provides reconfigurable modules of industrial control systems. The course uses a flipped classroom format with hands-on projects: the students create educational games that replicate real cyber attacks. Throughout the semester, they learn to understand the risks and gain capabilities to respond to cyber attacks that target industrial control systems. Our described experience from the design of the testbed and its usage can help any educator interested in teaching cybersecurity of cyber-physical systems.

## CCS CONCEPTS

• **Hardware**; • **Computer systems organization → Embedded and cyber-physical systems**; • **Security and privacy**; • **Social and professional topics → Computing education**;

## KEYWORDS

training facility, modular testbed, cyber-physical systems, industrial control systems, ICS, SCADA, cybersecurity education, syllabus

**ACM Reference Format:**
Pavel Čeleda, Jan Vykopal, Valdemar Švábenský, and Karel Slavíček. 2020. KYPO4INDUSTRY: A Testbed for Teaching Cybersecurity of Industrial Control Systems. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366908

*SIGCSE '20, March 11–14, 2020, Portland, OR, USA*


## 1 INTRODUCTION

Industrial control systems (ICS) provide vital services, such as electricity, water treatment, and transportation. Although these systems were formerly isolated, they became connected with information technology (IT) systems and even to the Internet. Figure 1 shows the ISA-95 enterprise reference architecture that describes the connection between the functions of ICS and IT systems [29]. This connection of processes in the cyberspace and the physical world has reduced costs and enabled new services. However, the ICS assets became vulnerable to new threats and ever-evolving cyber threat landscape [41].

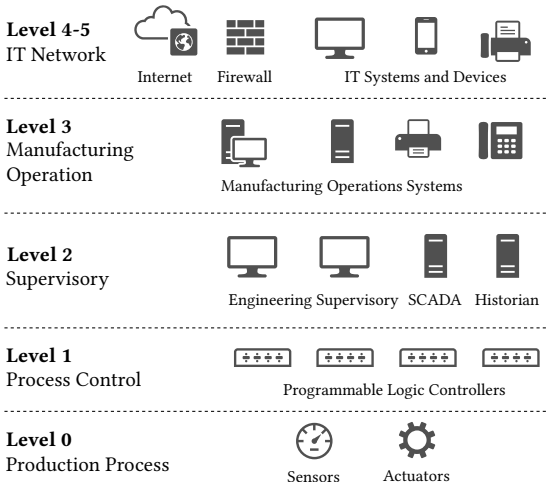


**Figure 1: The ISA-95 architecture: A hierarchical model of enterprise-control system integration [29]**

ICSs are made to maintain the integrity and availability of production processes and to sustain conditions of industrial environments. Their hardware and software components are often custom-built and tightly integrated. However, IT systems use off-the-shelf hardware and software and have different operational characteristics and security objectives [25].

Traditional cybersecurity courses are falling short in training ICS security [7], since they focus on exploiting and defending IT assets. To teach ICS security, a training facility (testbed) is needed to

model a real-world ICS system [17] and to provide hands-on experience. However, building and operating a realistic cyber-physical testbed using standard industrial equipment is expensive. It incorporates equipment such as programmable logic controllers (PLC), input/output modules, sensors, actuators, and other devices.

This paper addresses how to teach ICS cybersecurity to computer science students. Currently, the majority of students has intermediate knowledge level of IT cybersecurity but is unfamiliar with ICS principles. Our work brings two main contributions. First, we share our experience with the design and acquisition of KYPO4INDUSTRY testbed. Second, we describe a course syllabus to deliver cybersecurity training in a simulated industrial environment. The course uses a flipped classroom format [6] with hands-on projects replicating real cyber attacks. The students learn to understand the risks and gain capabilities to respond to cyber attacks that target ICS.

This paper is organized into five sections. Section 2 provides an overview of hands-on activities for teaching cybersecurity in IT and ICS. Section 3 describes the ICS training facility, lists the main components, and provides implementation details. Section 4 provides a detailed description of the design, content, and assessment methods of the ICS cybersecurity course. Finally, Section 5 concludes the paper and outlines future work.

## 2 RELATED WORK

Cybersecurity knowledge and skills are usually taught through classroom lectures complemented with labs, exercises, and home assignments. Such a combination of theory and practice is essential in training cybersecurity experts, since the number of cyber attacks and the ingenuity of attackers is ever-growing. This section presents the current best practice for teaching cybersecurity in IT and ICS.

### 2.1 Teaching Cybersecurity in IT

The three most popular types of IT cybersecurity training are hands-on assignments, capture the flag (CTF) games, and cyber defense exercises (CDX). Hands-on assignments include working with cybersecurity tools, usually in a virtual environment. An example collection of such assignments is SecKnitKit [31], a set of virtual machines (VMs) and corresponding learning materials. Using ready-made VMs offers a realistic and isolated environment with minimal setup, which is well-suited for cybersecurity training. Alternatively, online learning platforms, such as Root Me [23], provide a set of cybersecurity challenges that the learners solve locally or online.

CTF is a format of cybersecurity games and competitions in which the learners solve various cybersecurity tasks. Completing each task yields a textual string called flag, which is worth a certain amount of points. There are two main variations of the CTF format: Jeopardy and Attack-Defense.

In Jeopardy CTF, such as PicoCTF [8], learners choose the tasks to solve from a static collection of challenges presented in a web interface. The challenges are divided into categories such as cryptography, reverse engineering, or forensics. Learners solve the tasks locally at their machines or interact with a remote server. Jeopardy CTFs can thus accommodate hundreds of players at the same time.

In Attack-Defense CTF, such as iCTF [36], teams of learners each maintain an identical instance of a vulnerable computer network. Each team must protect its network while exploiting vulnerabilities in the networks of other teams. Successful attacks yield flags, which, along with maintaining the availability of the network services, contribute to the teams' score.

While anyone can participate in hands-on training or CTF games, CDX is a complex cybersecurity exercise for professionals, often from military or government agencies or dedicated cybersecurity teams [12, 39]. Learners are divided into blue teams responsible for maintaining and defending a complex network infrastructure against attacks of an external red team. The blue teams must preserve the availability of the network services for end-users and respond to prompts from law enforcement groups and journalists. Beyond IT systems, some exercises feature simulated critical infrastructure (e.g., electricity grid or transportation).

### 2.2 Teaching Cybersecurity in ICS

Teaching ICS relies on components that are likely to be encountered in operational environments. Testbeds are built to replicate the behavior of ICS and incorporate a control center, communication architecture, field devices, and physical processes [32]. Holm et al. surveyed the current ICS testbeds and reported on their objectives and implementation [17]. Most testbeds focus on cybersecurity – vulnerability analysis, tests of defense mechanisms, and education. Testbed fidelity is essential for training activities and the level of provided courses. High-fidelity testbeds are rare, and most testbeds use simulations, scaled-down models, and individual components [7]. ICS courses cover beginner and intermediate levels of training.

Virtualized, purely software-based testbeds are built upon virtual PLCs and devices modeled in software [3]. They can be highly flexible and imitate any real environment with an arbitrary number of various devices. Their main drawback is the lack of look and feel of the operational environment. Users who are accustomed to using the real equipment might perceive purely software-based testbeds as a computer game and not as training for real situations. An example of such testbed is a system for assessment of cyber threats against networked critical infrastructures [30].

Hardware-based testbeds are used, for example, in training operating personnel of chemical and nuclear plants. Apart from these, there are other specialized ones, such as PowerCyber [16], which is designed to closely resemble power grid communication utilizing actual field devices and Supervisory Control and Data Acquisition (SCADA) software. This testbed allows to explore cyber attacks and defenses while evaluating their impact on power flow. Ahmed et al. [1] presented SCADA testbed that demonstrates three industrial processes (a gas pipeline, a power transmission and distribution system, and a wastewater treatment plant) in a small scale. To do so, it employs real-world industrial equipment, such as PLCs, sensors, or aerators. These are deployed at each physical process system for local control and monitoring, and the PLCs are connected to a computer running human-machine interface (HMI) software for monitoring the status of the physical processes. The testbed is used in a university course on ICS security. Students can observe the industrial processes, learn ladder logic programming in various programming environments, and observe network traffic of multiple communication protocols.

In 2016, Antonioli et al. [5] prepared SWaT Security Showdown, the first CTF event targeted at ICS security. The game employed

Secure Water Treatment (SWaT), a software-based testbed available at Singapore University of Technology and Design [22]. Selected twelve international teams from academia and industry were invited. The game was divided into two phases: online Jeopardy and on-site Attack-Defense CTF. The first part served as a training session and included novel categories related to the ICS realm. The on-site CTF lasted two days. The teams visited the testbed on the first day. The next day, they had three hours to attack the SWaT testbed. The authors devised a dedicated scoring system for the assessment of attacks launched by the teams. The scoring evaluated the impact of the attacks on the physical and monitoring processes of the testbed, and the ability to conduct attacks that are not discovered by ICS detection systems deployed in the testbed.

Chothia and de Ruiter [9] developed a course at the University of Birmingham on penetration testing techniques of off-the-shelf consumer Internet of Things (IoT) devices. Students were tasked to analyze device functionality, write up a report, and give a presentation of their findings.

## 3 KYPO4INDUSTRY: ICS TRAINING FACILITY

In this section, we describe the hardware and software components of the ICS testbed. The ICS training takes place in a specialized physical facility, which has been frequently used for university courses [37], international CDXs [39], and extracurricular events. The room contains six large tables, each with three seats, three desktop PCs, and ICS hardware devices. As Figure 2 shows, the devices within the testbed infrastructure are interconnected and so can communicate with each other. The tables are portable to allow the instructor to rearrange the room for various activities, including team assignments, student presentations, and group discussions.

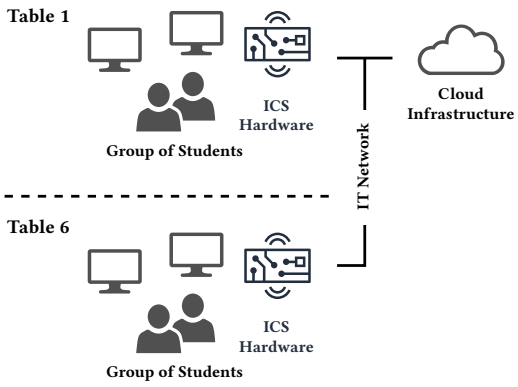


**Figure 2: Training facility setup**

### 3.1 Hardware Components

Based on the discussions with our partners and our experience, we defined these requirements on the hardware components of the KYPO4INDUSTRY testbed:

- *Open-hardware* – full access to hardware and software to avoid vendor-lock and other proprietary limitations, unlimited software manipulation, and community support.
- *Performance* – the PLC processor and memory (RAM, FLASH) must be sufficient to host operating-system with virtualization support (containers) and TCP/IP networking.
- *Communication interfaces* – wired and wireless communication buses for connecting peripherals and devices in the testbed. Industry standards like Ethernet, Wi-Fi, Bluetooth, USB, RS-485, and 1-Wire must cover both IT and ICS environments.
- *Inputs* – digital inputs to read binary sensors and devices such as buttons, switches, and motion sensors. Analog inputs to measure voltage from temperature, pressure, and light sensors.
- *Outputs* – digital outputs to switch binary actuators (LEDs, relays, motors), seven-segment displays, and graphical display (touchscreen) for human-machine interface.
- *Physical dimension* – hardware setup which will provide a cyber-physical experience (allow manipulation and observation of physical processes), multiple devices mounted in the same control panel, tabletop and mobile setup.
- *Safety* – durable equipment and a tamper-resistant installation, all cabling and connectors should be concealed to prevent (un)intended tampering during hands-on training, and electrical safety – avoid grid power parts.

Figure 3 shows the proposed hardware architecture. The hardware components of the control panel include PLCs, I/O modules, touchscreen, linear motor, and communication gateway.

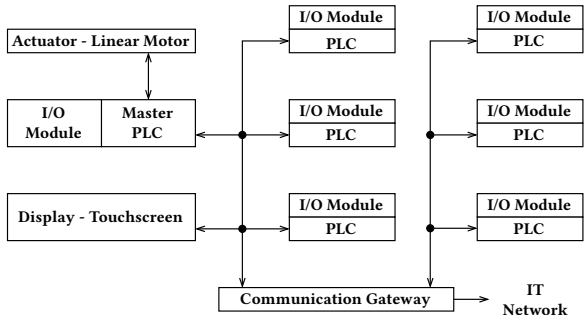


**Figure 3: Control panel block diagram**

PLC devices are a fundamental component of the control panel. When choosing a suitable PLC platform, it was essential for us that it leverages well-known hardware and has an industrial appearance. We chose the UniPi platform, which uses the popular Raspberry Pi single-board computer [15] and industrial casing. The UniPi Neuron M103 [34] model is used as the master PLC, and slave PLCs use UniPi Neuron S103 [35]. Both versions use Raspberry Pi 3 Model B with four-core 1.2 GHz CPU and 1 GB RAM. The Neuron PLC is DIN rail mountable, requires 24 V DC power supply, and has the following interfaces:

- 10/100 Mbit Ethernet, Wi-Fi, Bluetooth,
- four USB 2.0 ports, Micro SD port,
- RS-485, 1-Wire interface,
- digital input and output pins,
- one analog input and one analog output port.

The control panel uses two I/O module types. The first one connects the master PLC to three large-area LEDs, two buttons, one

key switch, and two motion detectors. The master PLC controls the linear motor through the RS-485 interface. The second type connects slave PLC with three large-area LEDs, two buttons, high power led (heating), 1-wire digital thermometer, and light sensor (analog input). Slave PLC uses RS-485 to control two-digit seven-segment display and 1.54" e-paper module.

10" LCD touchscreen is used to display technology processes. Dedicated Raspberry Pi module controls LCD via the HDMI interface and the touch panel via USB. A mechanical demonstrator (actuator) uses a linear motor. It includes DRV8825 stepper motor driver, ATmega 328 MCU, two end-stops switches, and three infrared position sensors. A network switch (MikroTik CRS125-24G) connects all PLC devices. Switch manages the flow of data between PLCs (100 Mbit Ethernet network) and incorporates routing functionality to connect the control panel to the IT network.

We built ten control panels to place at the top of the table and six as a movable trolley (see Figure 4). The tabletop setup is space-efficient, and the portable trolley provides mobility. The control panel is easy to handle; it requires only a power cord to connect to the mains electricity supply and Ethernet cable to connect to the IT network. The power consumption of one control panel is less than the power consumption of a desktop computer ($\leq 200$ W).

**Figure 4: Physical hardware setup of the ICS testbed**

## 3.2 Software Components

The physical equipment provides fidelity of the operational environment, but software is needed to replicate the functions and behavior of various ICS systems. Figure 5 shows the proposed software architecture based on the simplified ISA-95 model.

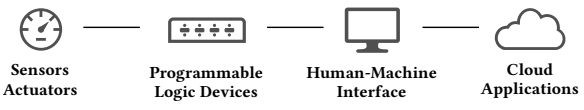


**Figure 5: Interfaces between software components**

Based on our experience from developing and delivering hands-on cybersecurity courses, we defined the following requirements on software components of KYPO4INDUSTRY testbed:

- *Open-source model* – access to source code and full software control, no licensing fees and licensing obstacles, community support, and collaboration.
- *Operating system* – a fully-fledged operating system with Raspberry PI support, operating-system-level virtualization, and high-speed networking.
- *Orchestration* – the ability to manage all testbed devices - configuration management and application deployment, automated preparation of testbed environment.
- *Communication protocols* – support for numerous legacy and emerging communication protocols used in ICS and IT environment.

The software stack of ICS testbed includes Linux OS (Debian optimized for PLC devices), Docker ecosystem [18], and on-premise OpenStack [14] cloud environment. We combine cloud deployment (virtual machines in OpenStack) with physical devices (PLCs, sensors, and actuators) to create ICS systems with varying levels of fidelity.

Automated orchestration of the testbed environment is of utmost importance. The central testbed controller runs as a virtual appliance. It provides management and monitoring of ICS testbed and contains Docker repository for PLC devices. The PLC devices are pre-installed with Debian OS and enabled Docker support. Using Docker containers simplifies software deployment and configuration of testbed components.

The openness of the used software allows us to implement virtually any new software component. We focus on two use-cases: widely deployed systems and new emerging technologies. Communication protocols and application interfaces are essential to creating a complete ICS system. There are dozens of industrial protocols, and many new protocols are being proposed every year. Widely deployed protocols are Modbus and DNP3 [19]. They have been used for decades for communications between ICS devices. The new emerging protocols represent MQTT [24] and REST [28].

# 4 ICS CYBERSECURITY COURSE DESIGN

This section presents our proposed ICS cybersecurity course that employs the ICS testbed. While the previous section described the hardware and software components of the testbed, it did not deal with content. One of our motivations for this course, apart from student learning, is that the students will create training content for the testbed. When writing this section, we followed the guidelines for planning new courses [40] and Joint Task Force on Cybersecurity Education (JTF) Cybersecurity Curricula 2017 [26].

## 4.1 Course Goals and Covered Topics

The overall goal of the course is to provide undergraduate students with an awareness of threats within the ICS domain via hands-on experience. As in the *authentic learning* framework [21], the focus is on solving real-world problems and learning by doing. The students' final product of the course is a training game for exercising both attacks at and defense of a selected industrial process. Our students previously created such games in the IT domain [37].

The primary JTF curriculum Knowledge Area (KA) the course covers is System Security, with Knowledge Units (KU) of Common System Architectures, System Thinking, and System Control. The

secondary KAs are Component Security (KU Component Testing), Connection Security (KU Network Defense), Data Security (KU Secure Communication Protocols), and Organizational Security (KU Systems Administration). We also marginally include the KA Social Security (KUs Cybercrime and Cyber Law). Finally, the course focuses not only on technical skills but also enables students to exercise communication, presentation skills, and time management.

## 4.2 Course Format

The course is aimed at computer science university students, namely undergraduates with a basic background in computer networks and security. The recommended prerequisite is completing our Cyber Attack Simulation course in the IT domain [37]. The initial run of the course is prepared for 6 students; however, the training facility described in Section 3 can accommodate up to 20 students who can work in pairs using the 16 control panels (see Figure 4). The course spans the whole semester (13 weeks). It is taught in a flipped classroom format [6] with 2-hour long weekly lab sessions, various homework assignments, and a hands-on semester project.

The necessary infrastructure includes, apart from the ICS testbed, also a CTF game infrastructure for running students' games (such as CTFd [10] or KYPO cyber range platform [38]), and Gitlab repositories for students' projects. We appreciate the effort of the open-source community, such as learning resources, documentation, and countless projects [2, 13, 33], which will help students to understand the used software.

## 4.3 Course Syllabus

Table 1 provides an overview of the course syllabus, student deliverables, and assessment methods. The course is divided into three parts: basics of ICS, development of an ICS training game, and its presentation and submission.

*ICS Principles.* Since we expect the students to have little knowledge of ICS, the first class session will motivate the topic by presenting examples of past cyber attacks such as Stuxnet [20]. The goal is to demonstrate the real-world impact of ICS incidents. We will follow by explaining the related terms, such as critical information infrastructures, and the corresponding legal regulations (such as a national Act on cybersecurity). For their homework, the students will individually choose a real, publicly-known attack on ICS and present it to others next class (in 15 minutes, including Q&A). After the presentations, the homework assigned in week 2 will be reading this paper and the papers we reference in the related work.

In week 3, the students familiarize themselves with the ICS testbed. They will complete several hands-on labs to learn the basic operational features of HW and SW components of the testbed. At the end of the class, they will discuss in groups how to demonstrate the known attacks using ICS testbed. As an individual homework assignment, they will search for existing ICS security threat landscape reports/lists, like the OWASP Internet of Things Project [27].

The following week, each student will present their results. The group will discuss the severity of each threat, and which of them can or cannot be demonstrated on the KYPO4INDUSTRY testbed to understand the capabilities and limitations of the testbed. The individual homework for the next week will be to prepare a 1-page written survey of CTF games in the ICS domain.

In the week 5 class, the students will engage in a pair activity of merging their reports to create a shared list of existing CTF games for the whole class. The motivation is to have a knowledge base of inspiration for students' games. The activity will follow with a short discussion centered around the question, "What features should an engaging game have?" The instructor will then briefly lecture on the principles of gamification [4] and provide an illustrative example to help students in their later assignment. The homework for the next week will be to think about a topic of student's game, which processes and threats the student will focus on, and how the student can use the ICS testbed for it. The instructor will highlight the specifics of ICS processes, and point out that they are threatened by different types of attacks than conventional IT systems. This homework starts the semester project phase.

*Game Development.* Week 6 starts with an activity in which pairs of students "peer-review" each others' discovered threats using the Security Threat Modelling Cards [11]. Students who finish will proceed to one-on-one consultations with the instructor to discuss the topic and the process of the game (output of the previous homework). Afterward, the students start working on the game narrative (storyline) and design the game flow, including the separation of tasks into levels. For their homework, the students will finish this design and send the draft to the instructor to receive formative feedback. The instructor will review the drafts and send comments before the next class.

In week 7, students will individually continue to develop their game, particularly the PLC-related part (Layer 1 of the ISA-95 architecture, see Figure 1). The instructor will then briefly lecture on the importance of the proper setting of learning outcomes and prerequisites, including examples from existing games. The students will use these instructions in their homework and add the learning outcomes and prerequisites to the description of their game.

Week 8 is dedicated to finishing the development of the PLC-related part and development and configuration of the Supervisory part (Layer 2 of the ISA-95 architecture). Students have to deliver an alpha version of their game for the dry run before the next class.

Week 9 starts with the dry run of students' games in pairs. Each student plays the game of another student for 45 minutes and takes notes about the learning experience. Then they switch roles. Afterward, the students are instructed on how to file a good bug report and report their feedback on the game in Gitlab. The instructor will review the submitted bug reports before the next class. The optional homework is to improve the games based on the dry run.

Week 10 starts with a short presentation of demonstrative examples of filed bug reports chosen by the instructor. For the rest of the class, students improve the games based on the feedback from the dry run.

In week 11, the students document their game and automate its deployment in the ICS testbed. They must submit the final version of their game three days before the next class, the course finale.

*Game Presentation and Submission.* In week 12, the students take part in organizing a Hacking Day – a public event during which other students of the university can play the created games. This event has two goals: motivating the students to work on their projects and popularizing ICS cybersecurity. Our experience from hosting such an event in the IT domain is described in [37].

**Table 1: The schedule and the structure of the ICS cybersecurity course, along with the student deliverables and their contribution to the total course grade (with 10% being for active participation in class), and important tasks for the instructor**

| Week | Class content | Student homework task (% of the grade) | Instructor tasks |
|---|---|---|---|
| 1 | Motivation, real attacks, legal issues | Prepare a presentation about an ICS attack (5%) | — |
| 2 | Student presentations of chosen attacks | Read this paper and some of the references | Grade the presentations |
| 3 | Hands-on labs on ICS testbed familiarization | Write an ICS security threat landscape report (5%) | — |
| 4 | Threat discussion, demo on ICS testbed | Write a short survey of CTF games in ICS (5%) | Grade the reports |
| 5 | Merge surveys, introduce game concepts | Select threats for your game | Grade the surveys |
| 6 | Threat modeling, storyline, consultation | Write a game draft | Check the game drafts |
| 7 | Preparing ICS part, educational objectives | Add learning outcomes and prerequisites | Check the game drafts |
| 8 | Preparing ICS and IT part | Prepare an alpha version of the game | Deploy the games |
| 9 | Dry run of the games with peers | Improve the game, submit bug reports (5%) | Review bug reports |
| 10 | Bug presentations, game improvement | Improve the game | — |
| 11 | Documentation, automation, deployment | Submit the game for presentation (50%) | Deploy the games |
| 12 | Public run of the games | Write a reflection from the public run (5%) | Oversee the event |
| 13 | Final reflections | Fix any issues that emerged in the public run (15%) | Grade the games |

Finally, week 13 is dedicated to students' reflections and the Hacking Day wrap-up in a focus group discussion. If any issues emerged in their game during the Hacking Day, they must fix them.

## 5 CONCLUSIONS

We shared the design details of KYPO4INDUSTRY, a testbed for teaching ICS cybersecurity in a hands-on way. Moreover, we proposed a novel university course that employs the testbed. The students will practically learn about threats associated with the ICS domain, develop an educational cyber game, and exercise their soft skills during multiple public presentations. The acquired skills will be essential for the computer science undergraduates who will be responsible for cybersecurity operations of an entire organization in their future career. We suppose that more organizations will employ cyber-physical systems, and so understanding of ICS-specific features will constitute an advantage for the prospective graduates.

### 5.1 Experience and Lessons Learned

Although using simple microprocessor systems (e.g., development boards) in teaching is popular, these systems do not replicate complex ICSs. Cyber-physical systems are unique and change with the physical process they control. The proposed testbed provides ten tabletop control panels and six mobile installations. In total, students can work with 148 PLCs, which use the popular Raspberry Pi single-board computers. The individual components (PLCs, sensors, actuators) are available off the shelf; however, the challenge is to build a hardware setup that will replicate the ICS in a laboratory environment. Addressing this challenge involves multiple engineering professions and requires external collaboration.

### 5.2 Future Work

The presented testbed is modular; therefore, it can be gradually upgraded as new advances in the field will emerge in the future. We rely on open-source components that are supported by large communities of users and developers. Still, there is room for future work on the content of training scenarios and novel instruction methods in the ICS domain. Another interesting research idea is to develop methods for creating cyber games and compare whether they work the same in the IT and ICS domain.

## ACKNOWLEDGMENTS

This research was supported by the ERDF project *CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence* (No. CZ.02.1.01/0.0/0.0/16_019/0000822).

## REFERENCES

[1] Irfan Ahmed, Vassil Roussev, William Johnson, Saranyan Senthivel, and Sneha Sudhakaran. 2016. A SCADA System Testbed for Cybersecurity and Forensic Research and Pedagogy. In *Proceedings of the 2Nd Annual Industrial Control System Security Workshop (ICSS '16)*. ACM, New York, NY, USA, 1–9. https://doi.org/10.1145/3018981.3018984

[2] Thiago Alves. 2019. *OpenPLC project.* Retrieved November 30, 2019 from https://www.openplcproject.com

[3] Thiago Alves, Rishabh Das, Aaron Werth, and Thomas Morris. 2018. Virtualization of SCADA testbeds for cybersecurity research: A modular approach. *Computers & Security* 77 (2018), 531–546. https://doi.org/10.1016/j.cose.2018.05.002

[4] Leonard A. Annetta. 2010. The "I's" Have It: A Framework for Serious Educational Game Design. *Review of General Psychology* 14, 2 (2010), 105–112. https://doi.org/10.1037/a0018985

[5] Daniele Antonioli, Hamid Reza Ghaeini, Sridhar Adepu, Martin Ochoa, and Nils Ole Tippenhauer. 2017. Gamifying ICS Security Training and Research: Design, Implementation, and Results of S3. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy (CPS '17)*. ACM, New York, NY, USA, 93–102. https://doi.org/10.1145/3140241.3140253

[6] Jacob Lowell Bishop, Matthew A Verleger, et al. 2013. The flipped classroom: A survey of the research. In *ASEE national conference proceedings, Atlanta, GA*. 1–18.

[7] Jonathan Butts and Michael Glover. 2015. How Industrial Control System Security Training is Falling Short. In *Critical Infrastructure Protection IX*, Mason Rice and Sujeet Shenoi (Eds.). Springer International Publishing, Cham, 135–149. https://doi.org/10.1007/978-3-319-26567-4_9

[8] Peter Chapman, Jonathan Burket, and David Brumley. 2014. PicoCTF: A Game-Based Computer Security Competition for High School Students. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*. USENIX Association, San Diego, CA, 10. https://www.usenix.org/conference/3gse14/summit-program/presentation/chapman

[9] Tom Chothia and Joeri de Ruiter. 2016. Learning From Others' Mistakes: Penetration Testing IoT Devices in the Classroom. In *2016 USENIX Workshop on Advances in Security Education (ASE 16)*. USENIX Association, Austin, TX, 8. https://www.usenix.org/conference/ase16/workshop-program/presentation/chothia

[10] Kevin Chung. 2017. Live Lesson: Lowering the Barriers to Capture The Flag Administration and Participation. In *2017 USENIX Workshop on Advances in Security Education (ASE 17)*. USENIX Association, Vancouver, BC, 6. https://www.usenix.org/conference/ase17/workshop-program/presentation/chung

[11] Tamara Denning, Batya Friedman, and Tadayoshi Kohno. 2013. *The Security Cards*. Retrieved November 30, 2019 from https://securitycards.cs.washington.edu

[12] Chris Eagle. 2013. Computer security competitions: Expanding educational outcomes. *IEEE Security & Privacy* 11, 4 (2013), 69–71. https://doi.org/10.1109/MSP.2013.83

[13] OpenJS Foundation. 2019. *Node-RED: Low-code programming for event-driven applications*. Retrieved November 30, 2019 from https://nodered.org

[14] OpenStack Foundation. 2019. *OpenStack: Open-source software platform for cloud computing*. Retrieved November 30, 2019 from https://www.openstack.org

[15] Raspberry Pi Foundation. 2019. *Raspberry Pi*. Retrieved November 30, 2019 from https://www.raspberrypi.org

[16] Adam Hahn, Ben Kregel, Manimaran Govindarasu, Justin Fitzpatrick, Rafi Adnan, Siddharth Sridhar, and Michael Higdon. 2010. Development of the PowerCyber SCADA Security Testbed. In *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research (CSIIRW '10)*. ACM, New York, NY, USA, Article 21, 4 pages. https://doi.org/10.1145/1852666.1852690

[17] Hannes Holm, Martin Karresand, Arne Vidström, and Erik Westring. 2015. A Survey of Industrial Control System Testbeds. In *Secure IT Systems*, Sonja Buchegger and Mads Dam (Eds.). Springer International Publishing, Cham, 11–26. https://doi.org/10.1007/978-3-319-26502-5_2

[18] Docker Inc. 2019. *Docker: Enterprise Container Platform*. Retrieved November 30, 2019 from https://www.docker.com

[19] Eric D. Knapp and Joel Thomas Langill. 2014. *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems* (2nd ed.). Syngress Publishing. https://doi.org/10.1016/C2013-0-06836-3

[20] Ralph Langner. 2011. Stuxnet: Dissecting a Cyberwarfare Weapon. *IEEE Security Privacy* 9, 3 (May 2011), 49–51. https://doi.org/10.1109/MSP.2011.67

[21] Marilyn M Lombardi. 2007. Authentic learning for the 21st century: An overview. *Educause learning initiative* 1, 2007 (2007), 1–12.

[22] A. P. Mathur and N. O. Tippenhauer. 2016. SWaT: a water treatment testbed for research and training on ICS security. In *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*. 31–36. https://doi.org/10.1109/CySWater.2016.7469060

[23] Root Me. 2019. *Hacking and Information Security learning platform*. Retrieved November 30, 2019 from https://www.root-me.org

[24] MQTT. 2019. *Message Queuing Telemetry Transport*. Retrieved November 30, 2019 from https://mqtt.org

[25] Lee Neitzel and Bob Huba. 2014. Top ten differences between ICS and IT cybersecurity. *InTech* 61, 3 (2014), 12–18.

[26] Joint Task Force on Cybersecurity Education. 2017. *Cybersecurity Curricular Guideline*. Retrieved November 30, 2019 from https://cybered.acm.org

[27] Open Web Application Security Project. 2019. *OWASP Internet of Things (IoT) Project*. Retrieved November 30, 2019 from https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project

[28] Leonard Richardson and Sam Ruby. 2007. *Restful Web Services* (first ed.). O'Reilly.

[29] Bianca Scholten. 2007. *The Road to Integration: A Guide to Applying the ISA-95 Standard in Manufacturing*. International Society of Automation. 234 pages.

[30] Christos Siaterlis and Béla Genge. 2014. Cyber-physical Testbeds. *Commun. ACM* 57, 6 (June 2014), 64–73. https://doi.org/10.1145/2602575

[31] Ambareen Siraj, Blair Taylor, Siddarth Kaza, and Sheikh Ghafoor. 2015. Integrating Security in the Computer Science Curriculum. *ACM Inroads* 6, 2 (May 2015), 77–81. https://doi.org/10.1145/2766457

[32] Keith Stouffer, Suzanne Lightman, Victoria Pillitteri, Marshall Abrams, and Adam Hahn. 2015. NIST 800-82: Guide to Industrial Control Systems (ICS) Security. *National Institute of Standards and Technology* (2015), 247. https://doi.org/10.6028/NIST.SP.800-82r2

[33] The Information Trust Institute (ITI) at the University of Illinois. 2019. *ICS Security Tools, Tips, and Trade*. Retrieved November 30, 2019 from https://github.com/ITI/ICS-Security-Tools

[34] UniPi.technology. 2019. *UniPi Neuron M103*. Retrieved November 30, 2019 from https://www.unipi.technology/unipi-neuron-m103-p95

[35] UniPi.technology. 2019. *UniPi Neuron S103*. Retrieved November 30, 2019 from https://www.unipi.technology/unipi-neuron-s103-p93

[36] Giovanni Vigna, Kevin Borgolte, Jacopo Corbetta, Adam Doupé, Yanick Fratantonio, Luca Invernizzi, Dhilung Kirat, and Yan Shoshitaishvili. 2014. Ten Years of iCTF: The Good, The Bad, and The Ugly. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*. USENIX Association, San Diego, CA, 7. https://www.usenix.org/conference/3gse14/summit-program/presentation/vigna

[37] Valdemar Švábenský, Jan Vykopal, Milan Cermak, and Martin Laštovička. 2018. Enhancing Cybersecurity Skills by Creating Serious Games. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2018)*. ACM, New York, NY, USA, 194–199. https://doi.org/10.1145/3197091.3197123

[38] Jan Vykopal, Radek Oslejsek, Pavel Celeda, Martin Vizvary, and Daniel Tovarnak. 2017. KYPO Cyber Range: Design and Use Cases. In *Proceedings of the 12th International Conference on Software Technologies - Volume 1: ICSOFT*. INSTICC, SciTePress, 310–321. https://doi.org/10.5220/0006428203100321

[39] Jan Vykopal, Martin Vizvary, Radek Oslejsek, Pavel Celeda, and Daniel Tovarnak. 2017. Lessons Learned From Complex Hands-on Defence Exercises in a Cyber Range. In *2017 IEEE Frontiers in Education Conference (FIE)*. 1–8. https://doi.org/10.1109/FIE.2017.8190713

[40] Henry M. Walker. 2016. CURRICULAR SYNCOPATIONS: Planning and Organizing a Course for the First Time. *ACM Inroads* 7, 4 (Nov. 2016), 12–17. https://doi.org/10.1145/2987377

[41] Bonnie Zhu, Anthony Joseph, and Shankar Sastry. 2011. A Taxonomy of Cyber Attacks on SCADA Systems. In *Proceedings of the 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing (ITHINGSCPSCOM '11)*. IEEE Computer Society, Washington, DC, USA, 380–388. https://doi.org/10.1109/iThings/CPSCom.2011.34