Smart Environment for Adaptive Learning of Cybersecurity Skills

Jan Vykopal, Pavel Seda, Valdemar Švábenský, and Pavel Čeleda

***Abstract*—Hands-on computing education requires a realistic learning environment that enables students to gain and deepen their skills. Available learning environments, including virtual and physical laboratories, provide students with real-world computer systems but rarely adapt the learning environment to individual students of various proficiency and background. We design a unique and novel smart environment for the adaptive training of cybersecurity skills. The environment collects a variety of student data to assign a suitable learning path through the training. To enable such adaptiveness, we propose, develop, and deploy a new tutor model and a training format. We evaluate the learning environment using two different adaptive trainings attended by 114 students of various proficiency. The results show that students were assigned tasks with a more appropriate difficulty, which enabled them to successfully complete the training. Students reported that they enjoyed the training, felt the training difficulty was appropriately designed, and would attend more training sessions like these. Instructors can use the environment for teaching any topic involving real-world computer networks and systems because it is not tailored to particular training. We freely release the software along with exemplary training so that other instructors can adopt the innovations in their teaching practice.**

***Index Terms*—Adaptive and intelligent educational systems, intelligent tutoring systems (ITSs), learning environments, security, virtual laboratories.**

I. Introduction

MASTERING cybersecurity requires extensive knowledge and skills, ranging from a wide area of theoretical concepts to practical skills with operating systems, command-line tools, and system vulnerabilities [1]. At the same time, more and more students with different backgrounds are entering the field of cybersecurity [2]. As a result, it is difficult for instructors to conduct hands-on cybersecurity training that would match the proficiency of all the students.

Existing cybersecurity training offerings are based on static scenarios with limited or no adaptiveness to an individual student [3]. Although the instructor can intervene to help students interactively, this is feasible only in relatively small classes, and not every student actively asks for help. The interactive help is especially complicated during online training (e.g., forced by restrictions caused by the COVID-19 pandemic [4]).

We see the opportunity to address the instructors' problem and improve the students' learning experience using a *smart learning environment* (SLE). This environment considers students' proficiency and adapts the learning content using data about student actions and performance in ongoing training. As a consequence, low-performing students are not overwhelmed by too difficult tasks, and high performers are not bored by too simple assignments. In the end, each student benefits from the adaptive training compared to the static assignments. Instructors benefit from efficient management, as well as monitoring of the learning environment and actions of individual students. An SLE, thus, saves the precious time of instructors, which they can spend on assisting individual students who struggle.

We reviewed the literature on SLE and related technologies such as remote laboratories, intelligent tutoring systems (ITSs), and adaptive learning systems. There are many works and systems for various learning domains, such as engineering, technology, science, foreign languages, and mathematics [5]. However, we have not found any smart network laboratory that would assign hands-on cybersecurity tasks to students based on their proficiency and performance in ongoing training featuring computer and network systems. Therefore, we have been iteratively developing and evaluating a learning environment with this capability. Since cybersecurity is a complex domain encompassing diverse technical knowledge and skills, creating an SLE for it represents a substantial research challenge.

The aims of this article are to 1) introduce the design of a *smart network laboratory* for training that involves computer networks, operating systems, and vulnerable applications and 2) evaluate the laboratory in an authentic teaching of cybersecurity skills. Our smart laboratory uses an unique tutor model and a training format, which are not present in state-of-the-art network laboratory environments. We evaluated our laboratory in field studies with 114 students of various proficiency participating in either on-site or remote training sessions. The objectives of the evaluation are to investigate 1) how efficiently were individual learners distributed to tasks of various difficulty and 2) stakeholders' experience of using our laboratory. The results show that students persisted in the adaptive

Manuscript received 15 December 2021; revised 31 August 2022; accepted 17 October 2022. Date of publication 21 October 2022; date of current version 14 June 2023. This work was supported by the ERDF project CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence under Grant CZ.02.1.01/0.0/0.0/16_019/0000822. An earlier version of this paper was presented in part at the 2021 IEEE Frontiers in Education Conference. DOI: 10.1109/FIE49875.2021.9637252. *(Corresponding author: Jan Vykopal.)*

This work involved human subjects or animals in its research. The author(s) confirm(s) that all human/animal subject research procedures and protocols are exempt from review board approval.

The authors are with the Masaryk University, 601 77 Brno, Czech Republic (e-mail: vykopal@ics.muni.cz; seda@fi.muni.cz; valdemar@mail.muni.cz; celeda@ics.muni.cz).

Digital Object Identifier 10.1109/TLT.2022.3216345

training and successfully completed more tasks compared to nonadaptive training. The students also reported they enjoyed the adaptive training, felt the training difficulty was appropriate, and would attend more adaptive training sessions.

The rest of this article is organized as follows. Section II summarizes related work and introduces SLEs, their core functions, and existing systems providing these functions for teaching cybersecurity hands-on. Section III introduces our smart laboratory for learning cybersecurity skills, used methods, and technological components. Section IV details the instructor's and student's view of the SLE. Section V describes a case study of using the developed SLE in authentic teaching in on-site and remote settings, and Section VI reports and discusses the results. Finally, Section VII concludes this article.

II. Background and Related Work

Our work is related to remote laboratories, ITSs and adaptive learning, and especially to SLEs.

Remote laboratories have been researched, developed, and used for teaching of various science and engineering disciplines for more than two decades [6], [7], [8]. Some laboratories collect data about students' interaction with the laboratory to provide learning analytics for teachers and learners [9], [10], [11], [12], for instance, an identification of common students' mistakes and remedial actions [13], [14]. Other laboratories provide automated student assessment or personalized assignments for each student [15], [16]. However, there is no published laboratory that would provide adaptive learning features described in this article.

The research of ITS and adaptive learning environment is well established [17], [18]. There are examples of successful tutoring systems for various fields of computer science, such as SQL-Tutor [19] or ProTuS [20], or systems created by various authoring tools [21], even by nonprogrammers [22]. However, to the best of our knowledge, there are no ITS for hands-on cybersecurity training in a networked laboratory environment.

A. Smart Learning Environments

A recent and thorough literature review by Tabuenca et al. [5] has shown that the term *SLE* is used inconsistently in the literature. The authors consolidated the terminology and synthesized core functions and characteristics of SLEs. In the rest of this article, we use the terms presented in the review. Its authors concluded that "the smartness in SLEs is the quality of a system to provide assistance for students or teachers considering their barriers for learning."

Next, the review identified four key components of SLEs:

1) *stakeholders*—students and teachers;
2) *space*—physical or virtual environment where learning occurs;
3) *system* providing smartness to the SLE by its core functions *sense*, *analyze*, and *react*;
4) *tools and technology* that facilitate students learning.

The *system* collects data from the learning context (the *sense* function), processes the collected data (the *analyze* function), and suggests actions to ease learning constraints (the *react* function). These functions are performed using *tools and technologies* such as data processing or visualization.

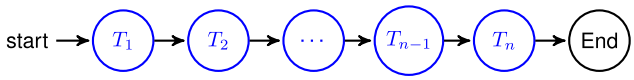

Fig. 1. Linear structure of training consisting of several tasks (T).

Tabuenca et al. [5] also identified affordances of SLEs reported in 68 empirical studies published from 2000 to 2019. Here, we list the four most frequent affordances.

1) *Adaptation, customization, and personalization* (*adaptable* onwards) refers to adjusting the learning environment considering the stakeholders' context, for instance, providing adapted and personalized environment for each student.
2) *Tracking and monitoring* (*traceable* onwards) refers to recording data from the stakeholders' context throughout learning activities using sensors installed in the environment.
3) *Feedback and recommendations* (*recommendation* onwards)—information provided by the SLE based on stakeholders' actions during learning activities, for instance, providing feedback just after answering the question.
4) *Patterns, activity, and behavior identification* (*pattern recognition* onwards)—the analysis of the collected data and identification of patterns related to stakeholders' behavior and their context, for example, identification of students' engagement when playing an educational game.

B. Environments for Learning Cybersecurity Skills

Cybersecurity skills are taught using interactive learning environments featuring emulated networks, information technology systems, or applications [23], [24]. These learning environments range from relatively simple Capture the Flag (CTF)[1] platforms [26], to sophisticated cyber ranges [27]. They enable individual students to learn by solving a set of tasks (T), which are often ordered linearly as depicted in Fig. 1.

The completion of each task is assessed by the environment, which checks whether the student submitted the correct answer, generated the expected network traffic, or changed the system state in the required way. Some platforms allow instructors to define static hints, which are provided to students on-demand when needed. Examples of these platforms are Hack The Box [28], TryHackMe [29], Project Ares [30], THREAT-ARREST [31], and KYPO Cyber Range Platform [32].

The role of the instructors who use these platforms shifts from being an active intermediary between learning content and students to a facilitator of learning who employs the platform and its features. Once a training starts, the instructors monitor students' progress using the insights automatically

[1] CTF is a popular form of gamified cybersecurity training in an informal setting. A successful solution of a CTF task yields a textual string called *flag*, which the learner submits in the learning environment to prove reaching the solution [25].

provided by the platform, such as those presented in [33]. The insights are generated using the methods of learning analytics [34] and educational data mining [35], which leverage data from educational contexts to understand and improve teaching and learning [36], [37]. If the instructors see students who need help, they can intervene appropriately.

C. How Smart Are Existing Environments for Learning Cybersecurity Skills?

Although Tabuenca et al. [5] did not discover any SLE built specifically for learning cybersecurity or related fields such as networking or operating systems, there are a few works that include some of the SLE core functions.

Cyber ranges [27] and CTF platforms [26] are learning technologies for cybersecurity that often employ data collection (the *sense* function). Maennel [38] reviewed digital datasets collected in cybersecurity training, which include timing information, commands, action counts, and input logs. However, as Weiss et al. [39] pointed out, the subsequent analysis of these data (the *analyze* function) is often limited to binary scoring of learners.

A rare exception is a work by Deng et al. [40] who evaluated a personalized laboratory environment that analyzes student activities. Examples of these activities include "mouse click, mouse hover, command line activity, and time spent inside a virtual machine" for cybersecurity training. Data about these activities are used as features to train a classifier to determine students' learning style. Subsequently, the system personalizes the style and presentation of the study materials for individual students. The SLE proposed by us differs in its goal: we aim to provide learners with adaptively chosen tasks of suitable difficulty.

To conclude, almost no environment for learning cybersecurity skills is advanced enough to offer actionable steps for supporting learning (the *react* function).

III. Smart Laboratory for Learning Cybersecurity Skills

The proposed smart laboratory (further *KYPO SLE*) is based on KYPO CRP [32], a platform we have been developing and using for hands-on cybersecurity training. Fig. 2 shows KYPO SLE mapped to the overall composition of an SLE presented in [5, Fig. 3]. Here, we detail the key SLE components in the context of learning cybersecurity skills:

1) *stakeholders*—instructors and students. Instructors prepare and supervise training activities in the virtual learning environment for students who perform these activities.
2) *spaces*—a virtual environment that a student can use from anywhere with a stable Internet connection, most commonly from home, school, or workplace.
3) *System*: KYPO CRP enhanced by these SLE core functions:

a) *Sense:* It Collects actions that students performed in the virtual environment, for instance, commands typed in the emulated environment (training sandboxes) or answers submitted to the training portal (see Section III-B);

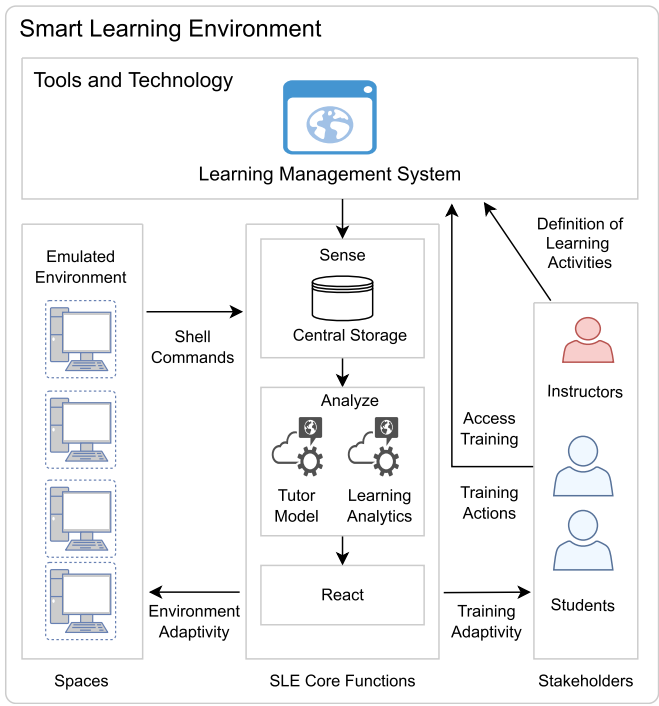

Fig. 2. Architecture overview of components of KYPO SLE.

b) *Analyze:* It Processes the collected data and provides them as input to a novel *tutor model* described in Section III-C1, which determines the most suitable learning path for each student. It also processes the data for creating the visualization of students' progress and performance for both students and instructors;
c) *React:* It presents the most suitable task for each student based on the output of the tutor model and evaluates the task completion (see Section III-D).
Using the terminology of adaptive learning systems, our SLE provides *task-loop adaptivity* [18].

4) *Tools and technology:* The virtual environment students interact with is hosted in a cloud or locally at personal computers (PCs) (such as a PC in a school laboratory or students' own laptops). In addition, the SLE is designed so that students need only a web browser to participate in training.

A. Generic Format of an Adaptive Training

To enable the *adaptable* affordance of KYPO SLE, we proposed a generic structure for adaptive cybersecurity training. In general, the training can contain an arbitrary number of phases and tasks. Each phase represents a learning activity. Each task in the phase exercises the same skills but varies in difficulty. Fig. 3 shows an example of such structure with five phases: three with two tasks and two with three tasks of various difficulty.

The training consists of several components: the introduction (Intro), the pretraining assessment (A), training phases (P_x) including variant tasks (T_y), decision components ($\mathrm{P_D}$), and post-training questionnaire (Q).

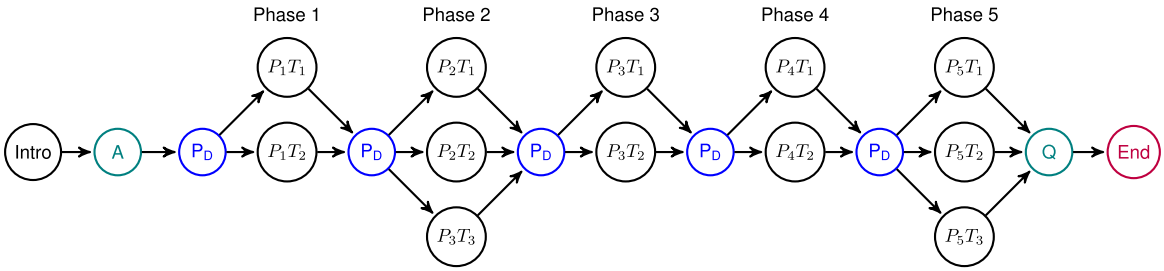

Fig. 3. Graph structure of adaptive cybersecurity training with pretraining assessment (A), decision component (P_D) applying the proposed model, and a post-training questionnaire (Q). This exemplary training contains five phases (P_x) with different number of tasks (T_y).

First, the introduction (Intro) familiarizes the student with the training and communicates necessary information before the training starts.

The pretraining assessment (A) is the first component of collecting data about students' knowledge and skills. The questions asked in the pretraining assessment are grouped into *question groups* based on their relation to specific training phases. Each question can be assigned into several question groups since they can be relevant to more phases. For each training phase, we set the *minimal ratio* of knowledge to determine whether the student's knowledge or self-reported skills are sufficient or not. For example, the minimal ratio can be set to 100%, which would mean that the students need to know answers to all the questions or self-report a defined level of skills for a particular phase. In particular, pretraining assessment should mostly include knowledge quizzes, as students' self-assessment can be inaccurate [41], [42].

The training phases contain tasks (T_y) that vary in difficulty but all aim at practicing the same topic. The decision component assigns exactly one task from the given phase. This assignment is based on the student performance in previous phases and on the results of the pretraining assessment. Students interact with their dedicated emulated environment, typically by entering shell commands, to find an answer: proof they completed the task. The student performance is measured by time, used commands, submitted answers, and a solution displayed in the phase. These performance indicators were selected based on the capabilities of the KYPO CRP platform and aligned with the review of metrics in cybersecurity exercises [38]. The tasks are denoted as T_1, T_2,..., T_n, where T_1 represents the most difficult task in the phase and T_n is the easiest task in the same phase. We refer to T_1 as the *base task* and T_2,..., T_n as *variant tasks*. Furthermore, the decision component (P_D) processes the students' performance and knowledge to assign a suitable task from the training phase.

Finally, the post-training questionnaire (Q) is an optional part of training, which enables instructors to collect immediate feedback from the students. Depending on the training objectives, the post-training questionnaire can be the same as or different from the pretraining questionnaire.

B. Sense—Collect Data

KYPO SLE collects answers from the pretraining assessment, training actions, and shell commands from the learning environment. All these data are further required by the tutor model, which selects the most suitable task for each student (see Section III-C).

1) Pretraining Assessment and Training Actions: The learning management system (LMS) is a key component of the SLE. It presents students with the pretraining assessment and tasks that have to be completed in the emulated environment. The LMS collects answers from a questionnaire at the beginning of the training (the state A in Fig. 3) and audits training actions that students make while they work on tasks (P_xT_y) in the training phases.

The training actions include answers submitted by the student in all phases, the action of revealing the task solution, and the action of correct/wrong answer to complete the task. All these data are timestamped and saved to the central storage. For instance, when a student submits an incorrect answer (e.g., `.invoices2021`), the system audits current timestamp in Epoch time (e.g., `1621524941312`), the type of the training action (`action.training.WrongAnswerSubmitted`), user pseudoidentifier (e.g., `5`), and the training run identifier (e.g., `3`). The data are stored as JSON records.

2) Shell Commands: When students interact with the emulated environment, they enter commands in shells such as BASH or Metasploit Console. These commands are captured at hosts in the environment in real-time and forwarded using the Syslog Protocol [43] to the central storage using Elastic Stack [44]. The commands are stored in JSON and timestamped with microsecond precision.

For example, a command `ssh alice@server` executed by a student in the Linux terminal at a machine in the emulated environment is timestamped and audited using Syslog as a string (see Fig. 4). Then, it is transformed into JSON and forwarded to the central storage as an entry for further processing [32]. This way, the submitted commands can be correlated with the pretraining assessment and training actions of the same student.

All hosts in the emulated environment use clock synchronization via the network time protocol [45]. This setting is a key requirement for time correlating the captured commands with training actions and other data. The architecture for collecting shell commands is detailed in [46].

C. Analyze—Select the Most Suitable Task

When designing the "Analyze" function of KYPO SLE, we had to deal with constraints specific to cybersecurity hands-on training. These include: 1) heterogeneity of training definitions,

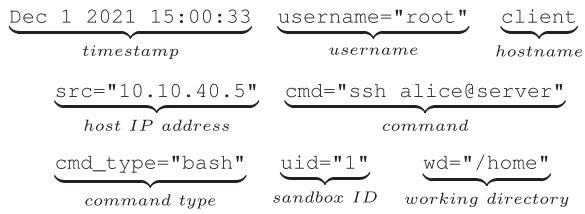

Fig. 4. Log entry for a command executed on one machine in an emulated environment [32].

which can have different phases and relations between them; 2) a limited volume of data to find statistical patterns; 3) complexity of the performed tasks; and 4) the inability to collect more in-depth data about students before the training. We designed a novel tutor model that processes the collected student data and computes the number of the most suitable task in a particular phase for each student [47].

1) Tutor Model: Let us denote the variables $\boldsymbol{p}$, $\boldsymbol{k}$, $\boldsymbol{a}$, $\boldsymbol{t}$, and $\boldsymbol{s}$, which are the binary vectors on the correctness or incorrectness of prerequisites for a particular training phase. Vector $\boldsymbol{p}$ is defined as follows: $\boldsymbol{p} = (\,p_1 \quad p_2 \quad \dots \quad p_m\,)$, where m is the number of training phases. The other vectors use the analogous notation.

1) $\boldsymbol{p}$ represents the (in)correctness of answers from the pretraining assessment.
2) $\boldsymbol{k}$ indicates if the student used the expected key commands in the command line within the given task.
3) $\boldsymbol{a}$ denotes whether the student submitted the expected answers to the task.
4) $\boldsymbol{t}$ contains the information if the task was completed in a predefined time.
5) $\boldsymbol{s}$ contains the information whether the student asked to reveal the solution for the task.

The model is defined by (1)–(3). By (1), we get the *decision matrix* $\boldsymbol{W}$ with weights for the individual phases' metrics. It is specific for each training phase. The weights represent the relationships between phases and their metrics. The value of the weight determines the importance of the metric to the phase. For instance, consider training with six phases where the third phase deepens the topic exercised in the first phase. In this case, we set the weights in the third matrix so that the selected weights for the metrics from the first phase are nonzero. The other performance metrics with weights set to zero are ignored.

The weights have to be manually set by the instructor since each training is unique. The number of decision matrices is equal to the number of training phases. The symbols $\pi, \kappa, \alpha, \theta$, and σ denote the columns in the decision matrices and the $i = 1, \dots, m$ are the rows in the decision matrices.

By (2), we get the *student's performance* based on the defined metrics and their weights for completed phases. The value of the performance is in the interval of [0,1]. In (2), s is multiplied by a, k, and t to distinguish between students who satisfy a, k, and t metrics without using a solution and solved the task on their own.

By (3), we get *the number of the most suitable task* y in phase x for a particular student (1 is T_1, 2 is T_2, and so on):

$$\boldsymbol{W}^{(x)} = \left(w_{ij}^{(x)}\right), \quad i = 1, \dots, m, j = \pi, \kappa, \alpha, \theta, \sigma \tag{1}$$

$$f(x) = \frac{\sum_{i=1}^{x}\left[p_i w_{i\pi}^{(x)} + s_i\left(k_i w_{i\kappa}^{(x)} + a_i w_{i\alpha}^{(x)} + t_i w_{i\theta}^{(x)} + w_{i\sigma}^{(x)}\right)\right]}{\sum_{i=1}^{x}\left(w_{i\pi}^{(x)} + w_{i\kappa}^{(x)} + w_{i\alpha}^{(x)} + w_{i\theta}^{(x)} + w_{i\sigma}^{(x)}\right)} \tag{2}$$

$$T_y = \begin{cases} n_x, & \text{if } f(x) \text{ is equal to } 0 \\ \text{trunc}(n_x[1 - f(x)]) + 1, & \text{otherwise} \end{cases} \tag{3}$$

where

$x =$ the phase a student is entering,
$y =$ the order of the task in a phase,
$T_y =$ the most suitable task of the phase x for the student,
$n_x =$ the number of variant tasks in the phase x,

$$p_i = \begin{cases} 1, \text{if question group } i \text{ from A is correctly answered} \\ 0, \text{otherwise,} \end{cases}$$

$k_i =$ commands corresponding to the phase i were used,
$e_i =$ expected time to complete of the phase i,
$o_i =$ student's completion time in the phase i,

$$t_i = \begin{cases} 1, \text{if } o_i < e_i \text{ in phase } i \\ 0, \text{otherwise,} \end{cases}$$

$$s_i = \begin{cases} 1, \text{if the solution of the phase } i \text{ is not displayed} \\ 0, \text{otherwise,} \end{cases}$$

$a_i =$ answers corresponding to the phase i were submitted.

2) Model Assumptions: The proposed model requires several assumptions that must be met by any SLE that would use it for hands-on cybersecurity training [47].

1) The learning environment has to collect the required data: the pretraining assessment answers p, commands typed by the students k, the submitted answers a, phase completion time t, and the action of displaying the solution s.
2) The model expects that some tasks are related; otherwise, it will heavily rely only on the pretraining assessment that may not be sufficient to capture students' proficiency.
3) The pretraining assessment question groups have to be mapped to the training phases to distinguish the level of knowledge and self-reported skills for a particular phase.
4) The model assumes that the tasks in the phases are sorted so that T_1 is the most difficult task, T_2,..., T_{n-1} are gradually easier tasks than T_1, and T_n is the easiest task.

To ease the unified design and run of the training, we add the following constraints that simplify the model assumptions.

1) The students' performance in a phase is evaluated in the same way in all the tasks.
2) The observed metrics are binary. Other metrics of students' performance, such as similarity of the submitted answers to the correct ones, are either unavailable or ignored.

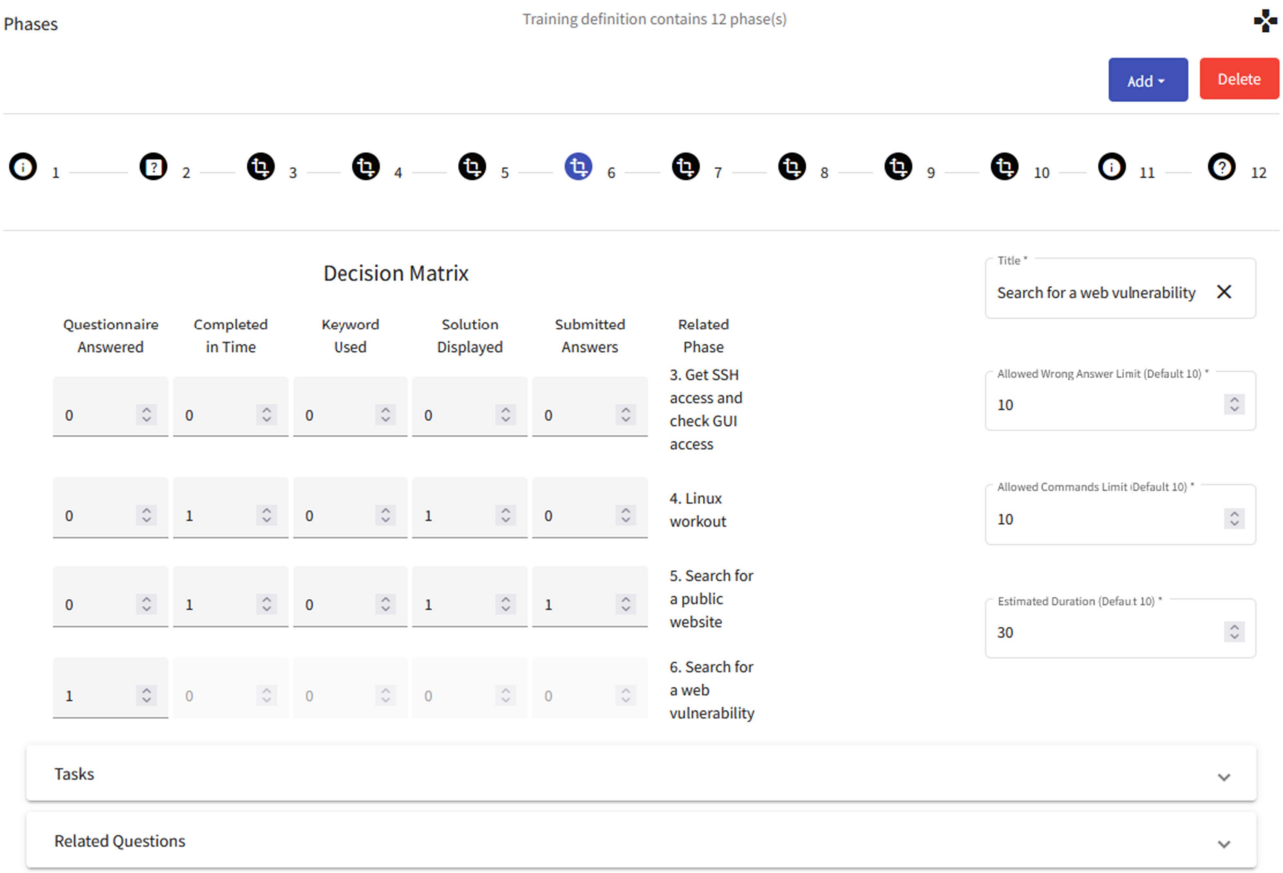

Fig. 5. Instructor's user interface for setting weights for Phase 6 of the Knowledge Base training via the Decision matrix. The instructor can also set task content and related questions from the pretraining assessment (available under Tasks and Related Questions accordions, respectively). This interface allows setting all weights for all phases in the adaptive training as depicted in Fig. 12. *Note:* This screenshot from KYPO SLE contains also nontraining phases (Intro, A, Q), so the numbering of phases does not align with the phase numbering in other figures (e.g., Sankey diagrams).

The model was developed with the aim to reinforce the cybersecurity training with respect to the commonly used performance metrics [38]. Nevertheless, it can be applied in any domain collecting such data.

D. *React—Serve the Selected Task*

When the student transitions between phases, the P_D component (see Fig. 3) is applied. This component uses the model described in Section III-C1 to assign the most suitable task in the next phase. When the task is assigned to the student, the task content is shown to the student. Each student can receive different task content.

IV. Stakeholders' Usage of the Smart Laboratory

In this section, we describe interactions of instructors and students with KYPO SLE before, during, and after the training.

A. *Instructor's View*

1) Before the Training: At first, the instructor(s) have to prepare the training: task assignments, the correct answers, and emulated environment. The learning activities have to be split into several phases, as described in Section III-A. For each phase, the instructor designs several tasks of varying difficulty to serve students of various proficiency. Furthermore, for each phase, the instructor sets model weights to define logical relations between phases and their metrics.

Fig. 5 shows user interface of KYPO SLE for setting the weights of preceding phases for the sixth phase. In this example, the instructor set the weight for *Questionnaire Answered* assigned to the sixth phase, and *Completed in Time* and *Solution Displayed* metrics for the fifth and the fourth phase, and for *Submitted Answers* metric in the fifth phase. The weights set to nonzero values determine which metrics will be used by the SLE for computing the most suitable task in the sixth phase. To ease the design of the model weights, we provide a tool assisting the instructors with the adaptive training design [48].

Finally, the instructor deploys the created training for a particular training session for a predefined number of students. The SLE automatically creates the emulated environment for the defined number of students and generates a unique access token, which the instructor distributes to the students.

2) During the Training: After the students enter the training session, the instructor monitors their progress using visual analytics provided by Sankey diagram and a progress chart.

The Sankey diagram (see Fig. 7) enables the instructor to monitor the overall progress of all the students in the training. The instructor might provide additional help to students who enter the easier tasks and still struggle. The progress chart

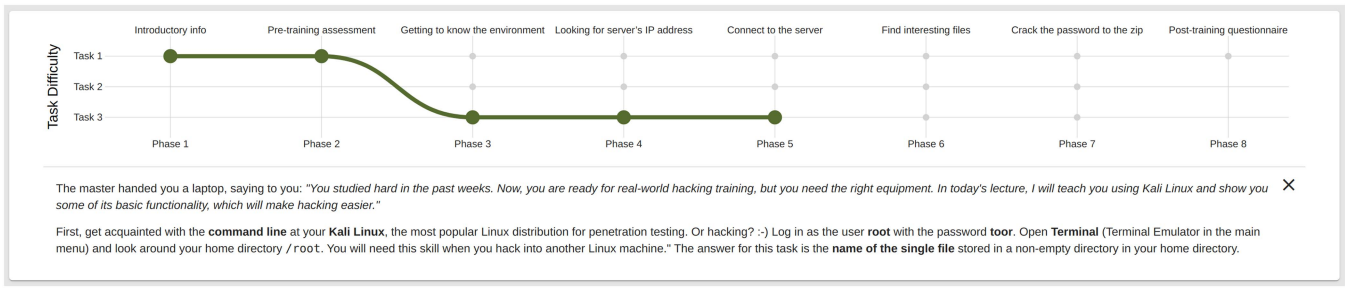

Fig. 6. Visualization showing the student's training path and the tasks' description.

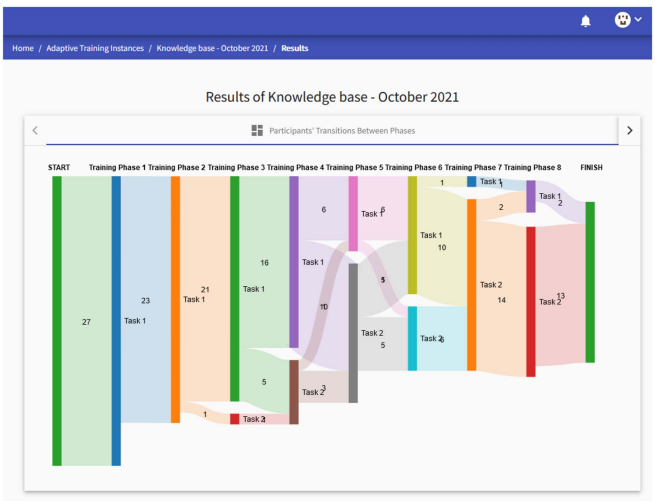

Fig. 7. Visualization showing the real-time progress of students during the adaptive training in KYPO CRP. It shows the number of students in particular phases and tasks.

(see Fig. 6) provides a detailed view of the progress and pathway of a selected student.

3) After the Training: When the training is over, the visualization of student progress is shown to the instructor and students. While instructors see the pathways of all students in one view (as in Fig. 7), each student sees only their own pathway (as in Fig. 6). The instructor can easily identify the critical training phases and give feedback to students for future learning or improve the training.

B. Student's View

1) Before the Training: Before the training, the students receive a URL to the web portal of KYPO SLE, requirements for the student's system used for accessing the SLE and access token to enter a particular training. Then, the students log into the system using their credentials and enter the access token to start the training. In that moment, one instance of an existing emulated environment is assigned to the student.

2) During the Training: First, the students read an introduction to the training and continue with the pretraining assessment of their theoretical knowledge and self-reported levels of skills. After the students complete this assessment, they enter the training phases to exercise cybersecurity skills. The training phases involve practical tasks performed in the student's own instance of the emulated environment. The students are not explicitly informed that training is adapted to their current performance and proficiency.

3) After the Training: When a student finishes the training, their progress is visualized to them to provide feedback and insights for future learning. Fig. 6 shows an example of such visualization. The student can see their path through the training. If the path moves in the lower parts (variant tasks, such as P3T3), this indicates missing knowledge or skills required by a particular task since the student did not satisfy prerequisites for more difficult tasks (such as P3T2 or P3T1). In addition, the student can see the assignment of any task by selecting bullets in the grid representing all the tasks in the training.

V. Case Study Setup

This section describes the case study of using KYPO SLE in teaching practice. The study evaluates the smart features of the learning environment in different contexts.

A. Study Objectives

The objective of the study is to investigate 1) how efficiently were individual learners distributed to tasks of various difficulty and 2) stakeholders' experience of using KYPO SLE. In the case of students, we are interested whether the laboratory eases their learning. In particular, we study whether low-performing students are provided with easier tasks, which enables them to complete the training in expected time. In the case of instructors, we analyze how much time and effort is saved by KYPO SLE compared to a manual assignment of training tasks to each student by instructors. Our study is conducted in two different contexts: a training session with and without the instructor's supervision.

B. Study Design

We followed the approach of action research [49], which is closely related to design-based research [50]. Both the methods are extensively used in applied and educational research. Their methodology involves developing a prototype that addresses a practical problem, testing it in an authentic context, performing a small-scale evaluation, and iterating the development further based on the lessons learned from the evaluation [25].

At first, we enhanced our existing KYPO Cyber Range Platform with data collection features described in Section III-B and implemented a prototype of the tutor model presented in Section III-C1. Along with that, we created the first adaptive

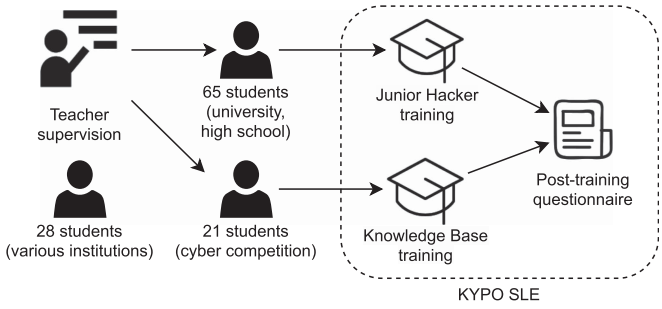

Fig. 8. Study design: 114 students completed one of two adaptive trainings deployed in KYPO SLE and answered a post-training questionnaire. Most training sessions (86 students) were facilitated by the instructor, but some (28 students) were not.

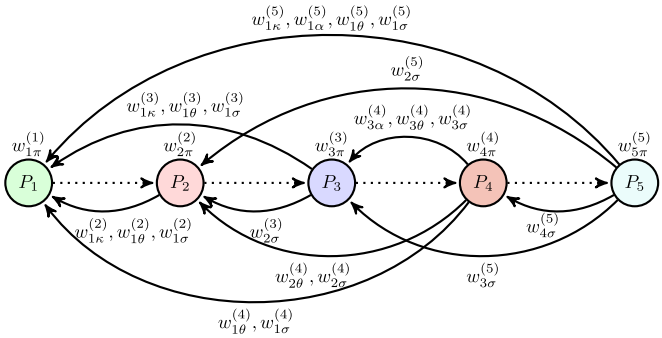

Fig. 10. Relationships between all the phases of Junior Hacker training. P_x is a phase x and $w_{ij}^{(x)}$ is weight for phase x and metric ij [47].

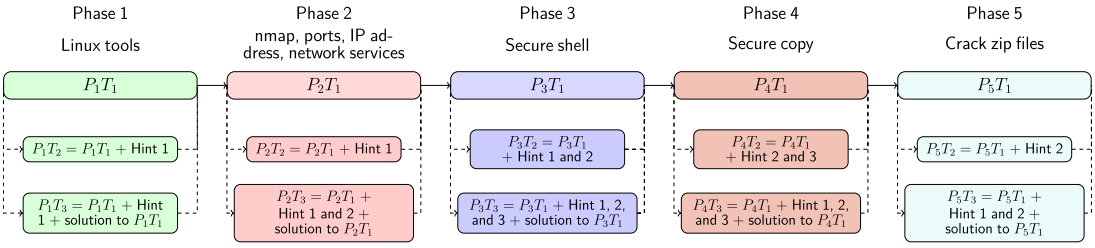

Fig. 9. Phases of the Junior Hacker adaptive training. Assignments of variant tasks enhance base tasks by hints or the solution [47].

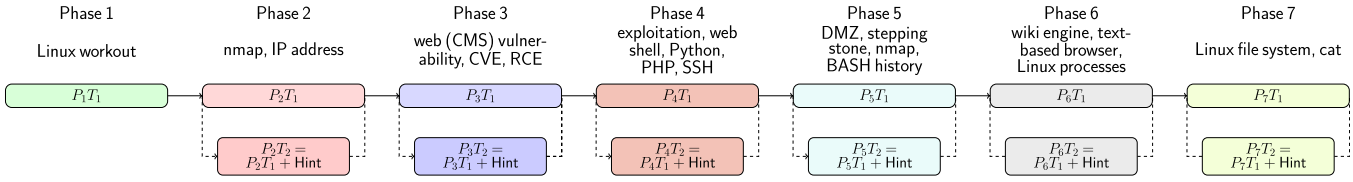

Fig. 11. Phases of the Knowledge Base adaptive training that follows the proposed generic format. The assignment of Task 2 enhances Task 1 by a hint.

training following the proposed generic format described in Section III-A. Then, we held the first training session with 24 participants and published the initial results [47].

Based on the lessons learned, we integrated the prototype of the tutor model with a user interface described in Section III-D and created a full-fledged SLE, which is publicly available [51]. We then designed another adaptive training and held additional training sessions to show the versatility of the training format and KYPO SLE. In total, we held ten training sessions with 114 participants in two different trainings.

Both trainings were designed to last 2 h to fit our classes. They were first tested by experienced instructors and then used in this study. The second training was intentionally designed with more phases but less tasks to highlight capabilities and limitations of the proposed training format and tutor model.

Fig. 8 visualizes the study framework. The role of instructors during the supervised sessions was only to provide technical assistance related to using the SLE. Specifically, the instructors did not provide any hints on training tasks .

C. Adaptive Trainings

1) Junior Hacker Training: This training consists of the pretraining assessment with eight questions and five phases covering topics depicted in Fig. 9. Each training phase features one base task and two variant tasks, including one presenting the step-by-step solution. The task with the solution is assigned to students who would not match any phase prerequisites. In the first training phase, basic Linux tools are practiced in three tasks (P_1T_1, P_1T_2, and P_1T_3). Task P_1T_2 contains the same assignment as P_1T_1 and provides Hint 1. The third task P_1T_3 contains the assignment from P_1T_1 with Hint 1 and the solution to that task. The subsequent training phases apply the same pattern that differs only in the content of the tasks, hints, and solution provided. The relationships between the training phases expressed as weights of each phase in the proposed tutor model are shown in Fig. 10.

2) Knowledge Base Training: This training consists of the pretraining assessment with eight questions and seven phases covering topics depicted in Fig. 11. Each phase contains one base task and one variant task, which enhances the assignment of the base task with a specific recommended tool or steps needed for finishing the phase. In contrast to the Junior Hacker

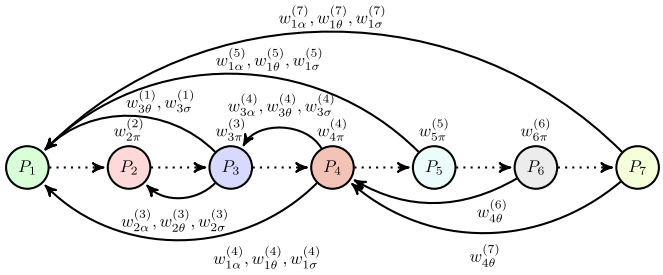

Fig. 12. Relationships between all the phases of Knowledge Base training. P_x is a phase x and $w^{(x)}_{ij}$ is weight for phase x and metric ij.

training, this training contains fewer inter-related phases, as shown in Fig. 12. However, the student performance in the first phase on Linux essentials (P_1) is considered when determining the suitable task in all other phases but the second phase (P_2). This was a design decision motivated by 1) the fact that the basic skills to use the Linux system were a strong prerequisite in this training, and 2) the intent to demonstrate the versatility of the proposed training format and tutor model.

D. Participants

In total, 114 individuals of diverse demographic characteristics (age, education, experience, and background) participated in our study. The participants' age ranged from 18 to 37. They consisted of high school students, university students, and university graduates, all focusing on computing and related technical disciplines. Since the participants' expertise in cybersecurity varied, they represented a suitable sample for demonstrating the capabilities of our adaptive SLE.

Eighty-six participants attended the training under the supervision of one, two, or three instructors, either on-site or remote via video conference. Sixty-five participants were undergraduate students and graduates of Masaryk University (MU) and the Brno University of Technology (BUT), both located in Brno, Czech Republic. In addition, this group included four high school students completing an internship at MU. Twenty-one participants were senior high school students and bachelor students of other universities, the finalists of the Czech national cybersecurity competition.

In addition, 28 participants attended the training remotely without any guidance (unsupervised training). They came from various institutions including industry companies (such as IBM and Kyndryl) or the two universities (MU and the BUT).

Table I summarizes the information about the trainings. All the participants attended voluntarily because of their interest in security.

E. Data Collection

The participants were assigned the Junior Hacker or Knowledge Base training described in Section V-C. They were informed that the estimated time for completing the training is up to 2 h. The supervised training sessions were held on-site in a computer laboratory or remotely via video conference in a time period between December 2020 and September 2021.

TABLE I
INFORMATION ABOUT THE FIELD STUDIES AND THE PARTICIPANTS.

Training date	Training modality	Participants' institution	Survey responses / num. participants
Dec 2, 2020	remote	MU	9 / 9
Dec 4, 2020	remote	MU	7 / 7
Dec 11, 2020	remote	MU	4 / 4
Jan 14, 2021	remote	MU	4 / 4
May 25, 2021	remote	MU	19 / 19
May 26, 2021	remote	BUT	8 / 10
May 28, 2021	remote	BUT	8 / 8
Jul 22, 2021	hybrid	Various	17 / 21
Sep 9, 2021	on-site	High school	4 / 4
Oct–Nov 2021	unsupervised	Various	15 / 28
			Total: 95 / 114

MU = Masaryk University, Czech Republic.
BUT = Brno University of Technology, Czech Republic.

TABLE II
WORDING OF THE POST-TRAINING QUESTIONNAIRE [47]

No.	Question
Q1	Did you feel the tasks were designed so that you can complete the training in a timely manner?
Q2	Did you feel you got stuck at some point during the training?
Q3	How much did you enjoy the training?
Q4	Did you feel the training should be more difficult for you?
Q5	Did you feel you would like the training to be longer with additional tasks to solve?
Q6	Would you like to play more cybersecurity training sessions like this one?

The primary role of the instructor(s) was only to assist students with access to the virtual laboratory or to troubleshoot any technical issues that might occur during the training. In contrast, the unsupervised session took place without any instructor's presence and support. Students could choose any time in October and November 2021 when they wanted to take the training and interacted only with our laboratory.

We collected all data available in KYPO SLE, i.e., students' answers to questions from the pretraining assessment, training actions, and shell commands. Both trainings contain a post-training Likert-scale questionnaire about their training experience (see Table II). Students who did not finish the training (i.e., did not reach the post-training questionnaire) were asked to fill in an additional questionnaire about issues they encountered during the training.

The study was waived from review by the university institutional review board as the collected data are anonymous and reported aggregately. In addition, all the participants provided informed consent to use the collected data for research purposes.

VI. RESULTS AND DISCUSSION

We now report and discuss the results of the study. We distinguish training sessions with instructor supervision (on-site and remote) and without any supervision (fully remote). Next, we discuss the effort required to run adaptive training with

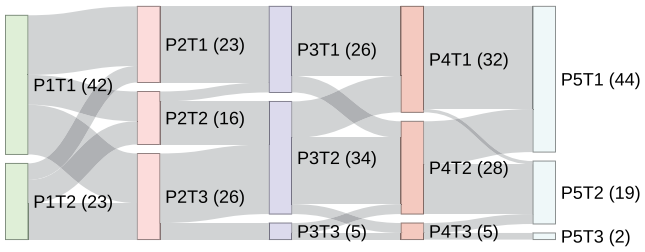

Fig. 13. Transitions of 65 students between particular tasks in Junior Hacker training. P_xT_y denotes task T_y in the phase P_x. The number of students solving the task is in brackets.

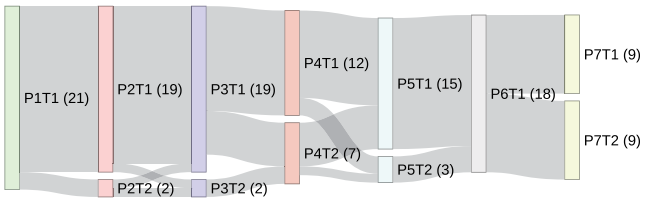

Fig. 14. Transitions of 21 students between particular tasks in Knowledge Base training. P_xT_y denotes task T_y in the phase P_x. The number of students solving the task is in brackets. The two students quit the training in phase P_3.

and without the SLE. Finally, we report limitations of the study and lessons learned.

A. Adaptive Training With Instructor's Supervision

1) Junior Hacker Training: This training was finished by all 65 participants. Fig. 13 shows the transitions of all the participants between tasks (P_xT_y) in all the training phases of this training. The diversity of transitions shows that the SLE enabled all the participants to finish the training, yet by completing less difficult tasks.

Furthermore, the transitions from more difficult to easier tasks between phases indicate that the participants had different issues with different tasks. In the first phase, 23 students assessed their knowledge of Linux basic commands as "None" or "Low." These answers determined the P_1T_2 task for them. In the second phase, $w^{(2)}_{1\kappa}$, $w^{(2)}_{1\theta}$, $w^{(2)}_{1\sigma}$, and $w^{(2)}_{2\pi}$ metrics were evaluated. Twenty-three students were assigned to the hardest (base) task P_2T_1 since they correctly answered the question related to Phase 2 and successfully finished Phase 1. Sixteen students were assigned to P_2T_2 mostly due to their inability to complete Phase 1 in the expected time; others entered too many commands or did not correctly answer the question assigned to Phase 2. The last group of 26 students was assigned to the P_2T_3 task mainly since they claimed to have "None" or "Low" skills in searching for opened network ports. In total, 51 students incorrectly answered the question $w^{(2)}_{2\pi}$ assigned to P_2, 25 students exceeded the shell commands limit $w^{(2)}_{1\kappa}$ in P_1, 29 students exceeded the expected time $w^{(2)}_{1\theta}$ in P_1, and six students displayed the solution $w^{(2)}_{1\sigma}$ in P_1. In the remaining phases, the students were assigned T_1 if they performed well or the other tasks (T_2 or T_3) due to various issues in related phases or pretraining assessment.

2) Knowledge Base Training: This training was finished by 18 out of 21 (86%) participants. Fig. 14 shows the transitions of 21 participants between tasks (P_xT_y) in the phases of Knowledge Base training. This training session was attended by the senior high school students and undergraduates who were finalists of the Czech national cybersecurity competition.

Although we expected better performance of this group, Fig. 14 shows that students also solved easier variants of the tasks in all the phases except Phase 1. This phase named "Linux workout" contains only one task, so all the students were assigned to it. In the second phase, two students failed to answer that the `nmap` tool is used for scanning network ports. In the third phase, two students were provided with the P_3T_2 task. One student revealed solutions in the first two phases, exceeded the estimated time in P_1, and failed to answer the questions relevant to the third phase. The other student exceeded the time in the first two phases and failed to answer the question assigned to the third phase. Furthermore, two students exited the training. In the third phase, seven students fell into the P_4T_2 task. Out of the seven students, two revealed the solution from Phases 1 and 3. The other five students had different issues: one submitted too many wrong answers and revealed the solutions, and the others failed to complete the previous phases in an expected time, submitted too many wrong answers, and revealed the solutions. In the fourth, fifth, and sixth phases, the students faced various issues such as exceeding the time to complete, submitting wrong answers, revealing solutions, or providing incorrect answers from pretraining assessment. Owing to these deficiencies, the students were assigned easier tasks in the respective phases.

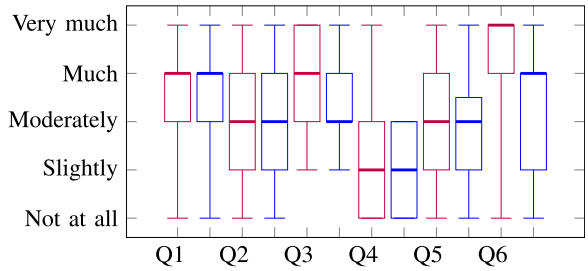

Fig. 15. Post-training questionnaire answers to Q1–Q6 in the survey from 80 students (red—Junior Hacker, blue—Knowledge Base).

Fig. 15 presents answers to questions from the post-training questionnaire listed in Table II for both Junior Hacker and Knowledge Base training. The participants reported that tasks of both trainings were appropriately designed so that they have successfully completed the training in time (Q1). The majority of participants of both trainings (70% in Junior Hacker, 68% in Knowledge Base) did not get stuck *Much* nor *Very much* during the training (Q2). The participants of both trainings enjoyed the learning experience (Q3). Junior Hacker training was rated higher than Knowledge Base. The majority of participants (51% in Junior Hacker, 63% in Knowledge Base) felt the trainings should be only *Slightly* or *Not at all* more difficult (Q4), which indicates that the provided tasks are not overwhelming yet keep the participants appropriately motivated. Only one participant of Junior Hacker training thought that the training should be *Very much* more difficult. Next, the participants engaged in both trainings and would like to continue if possible (Q5). Finally, the participants of both trainings would like to join another similar training (Q6).

This was unequivocal for those who participated in Junior Hacker training. Opinions of participants of Knowledge Base training were mixed, though still mainly positive.

To conclude, we see KYPO SLE caters to the students with various proficiency. Otherwise, these students would likely have not completed the training if using other state-of-the-art cybersecurity training platforms.

B. Adaptive Training Without Instructor's Supervision

Since the training sessions with the supervision were a success, we investigated the limits of the proposed approach. We prepared one training session with Knowledge Base training open for two months for anyone interested. We expected that the adaptive training would reduce the participants' failure rate and enable them to complete the training as we have seen in our supervised training sessions. However, out of 28 participants joining this session, only 15 successfully finished it. We, therefore, asked these participants who did not have the opportunity to fill in the post-training questionnaire to give us feedback on the training. We specifically asked if the participants encountered any issues during the training. Four students provided us with the following answers:

1) "*I could not for some reason access a file specified by [the] task—it looked it was not there for some reason, but maybe I did something wrong.*"
2) "*I did not know how to finish the task even with the provided solution.*"
3) "*I only started the training to see what is it about. I wanted to play it later, but due to COVID, I didn't manage to do so. I'll try it later.*"
4) "*Something interrupted me while participating in this training. Otherwise I would [have] finished [the] whole training.*"

The first two answers may indicate an issue in the design of this particular training, which discouraged the student from continuing. Students tend to stop the training and never come back in such cases. The third and fourth answers show that these students were forced to stop the training due to unforeseen circumstances that might be more distracting during the unsupervised training. To conclude, this particular training does not seem suitable for running in the unsupervised mode.

C. Effort Required to Run Adaptive Trainings

Table III shows descriptive statistics of training actions and shell commands entered by students who finished the supervised training (86 students, 2 trainings). Each participant performed 36 actions and typed 131 commands on average during one training session lasting about 2 h. In addition, they also filled in the pretraining assessment comprising eight questions. The total amount of data is so vast that it is infeasible to process manually, thus necessitating automation.

To support this argument, we now estimate how much time an instructor familiar with a state-of-the-art environment collecting these data would need to analyze the data manually. Our estimates come from the manual analysis performed in our initial study [47]. Without the SLE, the instructor would evaluate the pretraining assessment answers and map them to the relevant training phase. This evaluation may take tens of seconds for each student. Before each training phase, the instructor would need to analyze captured shell commands (searching for keywords, counting the commands) and training actions of each participant (counting the number of wrong answers, searching whether a solution was taken). This analysis may take tens of seconds, perhaps a minute or more in training events with tens of participants or more. This time estimation is based on the experience of four instructors that organized the first four training sessions in Table I when the SLE was not fully integrated into the KYPO CRP. Finally, the instructor would need to combine all these results to compute the suitable task for each participant using the tutor model. While the instructor is extremely busy and overwhelmed at that time, the student is only waiting to be assigned the next task. Using this "manual" approach, the instructor can handle only a few students. However, for medium to large classes, the manual approach does not scale. This example clearly supports the necessity of an SLE for running adaptive hands-on cybersecurity training sessions. What is more, automated task assignments by the SLE enable instructors to focus on providing additional help to struggling students.

TABLE III
DESCRIPTIVE STATISTICS OF TRAINING ACTIONS AND COMMANDS ENTERED IN KYPO SLE BY 86 STUDENTS (65 FROM JUNIOR HACKER AND 21 FROM KNOWLEDGE BASE TRAINING)

Training	Min	Max	Mean	Median	Total
Training actions					
Junior Hacker	7	45	28	25	1415
Knowledge Base	23	88	43	41	897
Both	7	88	36	33	2312
Commands					
Junior Hacker	12	155	83	74	4557
Knowledge Base	54	556	180	150	3775
Both	12	556	131	112	8332

The means are rounded to the nearest whole number.

D. Limitations

In this evaluation, the Knowledge Base training has only two tasks in each phase. Providing more tasks may increase the probability that the participant will get a more suitable task and increase their overall student experience.

We challenged our approach and studied whether the SLE can fully substitute a human instructor. The results of Knowledge Base training in an unsupervised mode showed that this is still not feasible. However, we might obtain better results with the Junior Hacker training, which we consider easier than Knowledge Base.

Another aspect that may negatively affect the unsupervised training session is that the SLE cannot easily recognize whether the student is thinking about the task (while not

producing any training action or typing the command) or interrupted the training for a while. The latter may mislead the tutor model using the "completed in time" metric.

E. Lessons Learned

For easier adoption of the developed SLE, we highlight the main lessons learned and provide general recommendations. All lessons are based on our experience from adaptive trainings in an authentic setting. Each lesson is illustrated with a concrete example.

1) Adjust the Weights in the Model Carefully: Inappropriate settings of weights in the decision matrices of the tutor model may lead to suboptimal transitions through the training tasks. The instructor(s) should verify the training with simulated students who perform differently to test that the model weights are set correctly. To reduce the complexity of such simulation, the instructor can use assisting tools described in [48].

Next, the instructor may stress critical prerequisites for a particular phase by setting a greater value of an important weight. For instance, all weights but one were set to one in the Knowledge Base training. The weight of timely completion in Phase 4 was set to two for Phase 7 to express its importance.

2) Training Content Must Be Thoroughly Designed and Tested: The SLE significantly helps the instructor to prepare and run the adaptive hands-on cybersecurity training. However, when the training content is not designed properly, (e.g., long and difficult Phase 6 in the Knowledge Base training), the students might get stuck in the task due to the misunderstanding of the task or the insufficient number of easier tasks. To design trainings more effectively, instructors may benefit from the documented guidelines [52].

3) Beginning of the Training Affects Its Progress: The training sessions are mostly held in a limited time frame (such as class). The pretraining assessment questionnaire should be brief and follow best practices for educational assessment [53], [54]. It should also be complemented by one or two phases with a single task that evaluates the skills of the students. For instance, Phase 1 in the Knowledge Base training served this purpose. The combination of quizzes, skill self-assessment, and skill evaluation provides a solid foundation for the tutor model.

4) Design as Many Tasks for Each Phase as Possible: To cater to students of various proficiency, the training should provide several variant tasks in each phase. If there are only two tasks in a phase as in the Knowledge Base training, some students may still struggle and need the instructor's assistance. However, a higher number of tasks increases the instructor's effort in preparing the training.

5) Design at Least Some Relationships Between the Training Phases: KYPO SLE relies on the collected data and the model settings. If the instructor sets the model weights so that there are no relationships between any phases, tasks will be assigned only based on the pretraining assessment questionnaire. This might not truly reflect the students' proficiency before entering particular tasks.

VII. Conclusion

The proposed smart learning environment KYPO SLE is, to the best of our knowledge, one of the first SLEs for hands-on cybersecurity training. The main objective of KYPO SLE is to provide an optimal individual learning path in hands-on training to improve the students' experience. To achieve that, we designed a new tutor model and a new training format that supports a graph structure to enable different learning paths for each student. The tutor model processes questionnaire answers and training actions from the LMS and shell commands from the emulated environment. Based on these data, it determines the most suitable task for each individual in the training.

We implemented the training format, data collection, and the tutor model and evaluated the developed SLE with 114 participants from a wide variety of institutions (high schools, universities, and companies). The evaluation showed that the proposed tutor model and the adaptive training format are generic enough to be used for various training sessions with different topics. Furthermore, the developed SLE can increase the students' ability to successfully complete the hands-on training and, thus, increase their positive experience. Without the SLE, instructors would not be able to process the complex and voluminous learning data required for determining the most suitable task. Finally, to ease the adoption of the proposed SLE, we released it as an open-source project [51] together with a detailed documentation [55] and an exemplary definition of an adaptive training [56].

A. Affordances of KYPO SLE

Our smart laboratory qualifies as an SLE because it fulfills the six characteristic features identified by Tabuenca et al. [5]. Specifically, it is or has:

- *adaptable*—it adjusts the learning environment so that it is adaptive and personalized for each student;
- *tracking and monitoring*—the instructor can monitor progress of each student during the training and revisit the results of each individual student after the training;
- *feedback and recommendations*—tasks assigned to students are determined based on the student's assessment and current performance;
- *pattern recognition*—the instructor can define patterns that are searched for in students' data during the training. These patterns are essential for selecting the most suitable task for each student;
- *efficient*—the laboratory enables assigning tasks of appropriate difficulty with respect to students' proficiency and current performance;
- *effective*—the laboratory enables more students to complete the training compared to the nonadaptive training where all students are provided with the same tasks regardless of students' proficiency and performance.

B. Open Challenges

We identified two distinct directions for possible future work.

1) Machine Learning for Setting the Tutor Model: The parameters of the tutor model are now set by instructors based on their expertise, the content of the tasks, and their relations between phases. Exploring how to employ machine learning algorithms should optimize metrics selection and weight settings. The application of machine learning algorithms will be challenging due to the typically small number of participants in each training session, their diverse proficiency, and the complexity of performed tasks.

2) Conditional Phases: The current format of the adaptive training assumes each student will pass through each training phase. Enhancing the format by allowing to skip some phases if certain conditions are met during the training can open new opportunities.

ACKNOWLEDGMENT

The authors would like to thank all researchers and developers of KYPO Cyber Range Platform who transferred research ideas into real open-source software.

REFERENCES

[1] D. Mouheb, S. Abbas, and M. Merabti, *Cybersecurity Curriculum Design: A Survey*. Berlin, Germany:Springer, 2019, pp. 93–107.
[2] M. Bashir, C. Wee, N. Memon, and B. Guo, "Profiling cybersecurity competition participants: Self-efficacy, decision-making and interests predict effectiveness of competitions as a recruitment tool," *Comput. Secur.*, vol. 65, pp. 153–165, 2017.
[3] C. Braghin, S. Cimato, E. Damiani, F. Frati, L. Mauri, and E. Riccobene, "A model driven approach for cyber security scenarios deployment," in *Comput. Secur.*. Cham, Switzerland: Springer, 2020, pp. 107–122.
[4] R. S. Putri, A. Purwanto, R. Pramono, M. Asbari, L. M. Wijayanti, and C. C. Hyun, "Impact of the COVID-19 pandemic on online home learning: An explorative study of primary schools in Indonesia," *Int. J. Adv. Sci. Technol.*, vol. 29, no. 5, pp. 4809–4818, 2020.
[5] B. Tabuenca et al., "Affordances and core functions of smart learning environments: A systematic literature review," *IEEE Trans. Learn. Technol.*, vol. 14, no. 2, pp. 129–145, Apr. 2021.
[6] J. Ma and J. V. Nickerson, "Hands-on, simulated, and remote laboratories: A comparative literature review," *ACM Comput. Surv.*, vol. 38, no. 3, 2006, Art. no. 7-es.
[7] T. Alkhaldi, I. Pranata, and R. I. Athauda, "A review of contemporary virtual and remote laboratory implementations: Observations and findings," *J. Comput. Educ.*, vol. 3, no. 3, pp. 329–351, 2016.
[8] I. Grout, "Remote laboratories as a means to widen participation in STEM education," *Educ. Sci.*, vol. 7, no. 4, 2017, Art. no. 85.
[9] H.-D. Wuttke, M. Hamann, and K. Henke, "Learning analytics in online remote labs," in *Proc. IEEE 3rd Exp. Int. Conf.*, 2015, pp. 255–260.
[10] C. N. Tulha, M. A. G. Carvalho, and L. N. de Castro, "LEDA: A learning analytics based framework to analyze remote labs interaction," in *Proc. 9th ACM Conf. Learn. Scale*, 2022, pp. 379–383.
[11] P. Orduña, A. Almeida, D. López-de Ipiña, and J. Garcia-Zubia, "Learning analytics on federated remote laboratories: Tips and techniques," in *Proc. IEEE Glob. Eng. Educ. Conf.*, 2014, pp. 299–305.
[12] J. García-Zubíaet al., "Dashboard for the VISIR remote lab," in *Proc. IEEE 5th Exp. Int. Conf.*, 2019, pp. 42–46.
[13] H. Considine, A. Nafalski, and Z. Nedic, "Understanding common student mistakes in the remote laboratory NetLab," in *Proc. IEEE Int. Conf. Teaching, Assessment, Learn. Eng.*, 2018, pp. 266–271.
[14] H. Considine, A. Nafalski, and M. Milosz, "An automated support system in a remote laboratory in the context of online learning," in *Educating Engineers Future Ind. Revolutions*, M. E. Auer and T. Rüütmann, Eds. Cham, Switzerland:Springer, 2021, pp. 657–665.
[15] A. A. Benattia, A. Benachenhou, and M. Moussa, "Development of an automatic assessment in remote experimentation over remote laboratory," in *Smart Ind. and Smart Educ.*, M. E. Auer and R. Langmann, Eds. Cham, Switzerland:Springer, 2019, pp. 136–143.
[16] A. L. Gonçalves, L. M. Carlos, J. B. da Silva, and G. R. Alves, "Personalized student assessment based on learning analytics and recommender systems," in *Proc. IEEE 3rd Int. Conf. Portuguese Soc. Eng. Educ.*, 2018, pp. 1–7.
[17] E. Mousavinasab, N. Zarifsanaiey, S. R. N. Kalhori, M. Rakhshan, L. Keikha, and M. G. Saeedi, "Intelligent tutoring systems: A systematic review of characteristics, applications, and evaluation methods," *Interact. Learn. Environ.*, vol. 29, no. 1, pp. 142–163, 2021.
[18] V. Aleven, E. A. McLaughlin, A. Glenn, and K. Koedinger, "Instruction based on adaptive learning technologies," in *Handbook of Res. on Learn. and Instruct.*, R. E. Mayer and P. A. Alexander, Eds. New York, NY, USA: Routledge, 2016, pp. 522–560.
[19] A. Mitrovic and S. Ohlsson, "Implementing CBM: SQL-Tutor after fifteen years," *Int. J. Artif. Intell. Educ.*, vol. 26, no. 1, pp. 150–159, Mar. 2016.
[20] B. Vesin, K. Mangaroska, and M. Giannakos, "Learning in smart environments: User-centered design and analytics of an adaptive learning system," *Smart Learn. Environ.*, vol. 5, no. 1, pp. 1–21, 2018.
[21] D. Dermeval, R. Paiva, I. I. Bittencourt, J. Vassileva, and D. Borges, "Authoring tools for designing intelligent tutoring systems: A systematic review of the literature," *Int. J. Artif. Intell. Educ.*, vol. 28, no. 3, pp. 336–384, 2018.
[22] V. Aleven et al., "Example-tracing tutors: Intelligent tutor development for non-programmers," *Int. J. Artif. Intell. Educ.*, vol. 26, no. 1, pp. 224–269, 2016.
[23] N. Chouliaras, G. Kittes, I. Kantzavelou, L. Maglaras, G. Pantziou, and M. A. Ferrag, "Cyber ranges and TestBeds for education, training, and research," *Appl. Sci.*, vol. 11, no. 4, 2021, Art. no. 1809.
[24] M. Swann, J. Rose, G. Bendiab, S. Shiaeles, and F. Li, "Open source and commercial capture the flag cyber security learning platforms—A case study," in *Proc. IEEE Int. Conf. Cyber Secur. Resilience*, 2021, pp. 198–205.
[25] V. Švábenský, "Automated feedback for cybersecurity training," Ph.D. dissertation, Faculty Informat., Masaryk University, Brno, Czech Republic, 2022.
[26] S. Kucek and M. Leitner, "An empirical survey of functions and configurations of open-source capture the flag (CTF) environments," *J. Netw. Comput. Appl.*, vol. 151, 2020, Art. no. 102470.
[27] M. M. Yamin, B. Katt, and V. Gkioulos, "Cyber ranges and security testbeds: Scenarios, functions, tools and architecture," *Comput. Secur.*, vol. 88, 2020, Art. no. 101636.
[28] Hack the Box, 2022. [Online]. Available: https://www.hackthebox.com/
[29] TryHackMe, 2022. [Online]. Available: https://www.tryhackme.com/
[30] *Project Ares*, Circadence, Boulder, CO, USA, 2022. [Online]. Available: https://projectares.academy
[31] G. Hatzivasilis et al., "Modern aspects of cyber-security training and continuous adaptation of programmes to trainees," *Appl. Sci.*, vol. 10, no. 16, 2020, Art. no. 5702.
[32] J. Vykopal, P. Čeleda, P. Seda, V. Švábenský, and D. Tovarňák, "Scalable learning environments for teaching cybersecurity hands-on," in *Proc. IEEE Front. Educ. Conf.*, 2021, pp. 1–9.
[33] R. Ošlejšek, V. Rusňák, K. Burská, V. Švábenský, J. Vykopal, and J. Čegan, "Conceptual model of visual analytics for hands-on cybersecurity training," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 8, pp. 3425–3437, Aug. 2021.
[34] C. Lang, G. Siemens, A. Wise, and D. Gašević, *Handbook of Learning Analytics*, 1st ed. Beaumont, AB, Canada: Society for Learning Analytics Research, 2017.
[35] C. Romero, S. Ventura, M. Pechenizkiy, and R. S. Baker, *Handbook of Educational Data Mining*. Boca Raton, FL, USA: CRC Press, 2010.
[36] C. Romero and S. Ventura, "Educational data mining and learning analytics: An updated survey," *Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov.*, vol. 10, no. 3, 2020, Art. no. e1355.
[37] C. Hundhausen, D. Olivares, and A. Carter, "IDE-Based learning analytics for computing education: A process model, critical review, and research agenda," *ACM Trans. Comput. Educ.*, vol. 17, no. 3, 2017, Art. no. 11.
[38] K. Maennel, "Learning analytics perspective: Evidencing learning from digital datasets in cybersecurity exercises," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops*, 2020, pp. 27–36.
[39] R. Weiss, M. E. Locasto, and J. Mache, "A reflective approach to assessing student performance in cybersecurity exercises," in *Proc. 47th ACM Tech. Symp. Comput. Sci. Educ.*, 2016, pp. 597–602.

[40] Y. Deng, D. Lu, C.-J. Chung, D. Huang, and Z. Zeng, "Personalized learning in a virtual hands-on lab platform for computer science education," in *Proc. IEEE Front. Educ. Conf.*, 2018, pp. 1–8.
[41] V. Švábenský and J. Vykopal, "Challenges arising from prerequisite testing in cybersecurity games," in *Proc. 49th ACM Tech. Symp. Comput. Sci. Educ.*, 2018, pp. 56–61.
[42] J. Mirkovic and P. A. Peterson, "Class capture-the-flag exercises," in *Proc. USENIX Summit Gaming, Games, Gamification Secur. Educ.*, 2014.
[43] G. Rainer and A. GmbH, "The Syslog protocol," Internet Requests for Comments, RFC Editor, RFC 5424, Mar. 2009. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5424.txt
[44] *The Elastic Stack*, Elastic NV, Mountain View, CA, USA, 2021. [Online]. Available: https://www.elastic.co/elastic-stack/
[45] D. Mills, J. Martin, J. Burbank, and W. Kasch, "Network time protocol version 4: Protocol and algorithms specification," *Internet Requests for Comments*, RFC Editor, RFC 5905, Jun. 2010. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5905.txt
[46] V. Švábenský, J. Vykopal, D. Tovarňák, and P. Čeleda, "Toolset for collecting shell commands and its application in hands-on cybersecurity training," in *Proc. IEEE Front. Educ. Conf.*, 2021, pp. 1–9.
[47] P. Seda, J. Vykopal, V. Švábenský, and P. Čeleda, "Reinforcing cybersecurity hands-on training with adaptive learning," in *Proc. IEEE Front. Educ. Conf.*, 2021, pp. 1–9.
[48] P. Seda, J. Vykopal, P. Čeleda, and I. Ignác, "Designing adaptive cybersecurity hands-on training," in *Proc. IEEE Front. Educ. Conf.*, 2022, pp. 1–9.
[49] D. E. Avison, F. Lau, M. D. Myers, and P. A. Nielsen, "Action research," *Commun. ACM*, vol. 42, no. 1, pp. 94–97, Jan. 1999.
[50] T. Anderson and J. Shattuck, "Design-based research: A decade of progress in education research?," *Educ. Res.*, vol. 41, no. 1, pp. 16–25, 2012.
[51] *KYPO Cyber Range Platform*. Masaryk Univ., Brno, Czech Republic, 2022. [Online]. Available: https://gitlab.ics.muni.cz/muni-kypo-crp
[52] M. Gáliková, V. Švábenský, and J. Vykopal, "Toward guidelines for designing cybersecurity serious games," in *Proc. 52nd ACM Tech. Symp. Comput. Sci. Educ.*, 2021, pp. 1275–1275.
[53] A. W. Astin and A. L. Antonio, *Assessment for Excellence: The Philosophy and Practice of Assessment and Evaluation in Higher Education*. Lanham, MD, USA: Rowman & Littlefield Publishers, 2012.
[54] G. Petty, *Teaching Today: A Practical Guide*. Cheltenham, U.K.: Nelson Thornes, 2009.
[55] *KYPO Cyber Range Platform: Documentation*, Masaryk Univ., Brno, Czech Republic, 2022. [Online]. Available: https://docs.crp.kypo.muni.cz/
[56] M. Gáliková, V. Švábenský, and J. Vykopal, "Junior Hacker Adaptive Training," 2022. [Online]. Available: https://gitlab.ics.muni.cz/muni-kypo-trainings/games/junior-hacker-adaptive

Jan Vykopal received the Ph.D. degree in computer systems and technologies from Masaryk University, Brno, Czech Republic, in 2013.

He is an Assistant Professor with Masaryk University, Brno, Czech Republic. He teaches cybersecurity and researches how to teach it better. He has been with KYPO Cyber Range Platform since its early beginnings in 2013. He has been designing and organizing various cybersecurity games and exercises, including the Czech national defense exercise, since 2015. He also organizes summer schools for finalists of the Czech national cybersecurity competition.

Pavel Seda received the M.Sc. degree in communications and informatics from the Brno University of Technology, Brno, Czech Republic, in 2017, the M.Sc. degree in applied informatics from Masaryk University, Brno, Czech Republic, in 2018, and the Ph.D. degree in electrical engineering from Brno University in 2022.

From 2014 to 2018, he was a Java Developer with IBM, Czech Republic. His research interests include cybersecurity, technologies, optimization, and cybersecurity education.

Valdemar Švábenský received the Ph.D. degree in data-driven support of hands-on cybersecurity training from Masaryk University, Brno, Czech Republic, in 2022.

His research interests include methods for automatic analysis of training data to generate tailored feedback for students and instructors.

Dr. Švábenský received the Best Paper Award at the ACM SIGCSE Technical Symposium on Computer Science Education 2020 and 2022 conferences. He also received two university-wide awards for the contribution to teaching computer science.

Pavel Čeleda received the Ph.D. degree in informatics from the University of Defence, Brno, Czech Republic, in 2007.

He is currently an Associate Professor with Masaryk University, Brno. He is a Principal Investigator of the KYPO Cyber Range project and co-Principal Investigator of the C4e Center of Excellence. His main research interests include traffic analysis, situational awareness, and cybersecurity testbeds for research and education. These research topics are the subject of many projects, collaborations, and supervised Ph.D. dissertations.