Designing Adaptive Cybersecurity Hands-on Training

Pavel Seda
Masaryk University
Brno, Czech Republic
seda@fi.muni.cz

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

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

Igor Ignác
Masaryk University
Brno, Czech Republic
ignaci@mail.muni.cz

***Abstract*—This Research To Practice Full Paper presents an instructor guide and a tool to improve the creation of cybersecurity hands-on training with adaptive learning support. Adaptive learning uses students' performance and skills to assign suitable tasks to improve their learning experience. While it is well-established in many domains, it is rarely used in operating systems, networking, and cybersecurity. In this paper, we improve and present how to ease the creation and optimization process of adaptive hands-on training by instructors. To the best of our knowledge, this paper is one of the first works investigating the process of creating cybersecurity training with adaptive learning. The training uses metrics such as pre-training assessment and performance during the previous tasks in training to assign suitable tasks for each student. With the help of the developed tool, we demonstrate how metrics settings influence the students' transitions between training tasks. The instructors can easily visualize students' transitions throughout the training. This approach helps the instructors adapt the metrics to predict students' transitions between tasks for each training session. The results from performed simulations show that our tool might increase the efficiency of the adaptive training and students' experience even more. Using the experience from the simulations and past training sessions, we propose the design process for the whole creation of adaptive training. This design process is general enough to be adopted by other domains such as operating systems and networking that may use adaptive learning techniques for their hands-on assignments. We have released the tool and all the software components under an open-source license, so other instructors can freely use and adopt them.**

***Index Terms*—adaptive learning, cybersecurity, evaluation, tool, tutor authoring, tutor model**

I. Introduction

Cybersecurity learning necessitates a wide scope of tools and concepts such as operating systems, command-line tools, programming languages, and system vulnerabilities [1]. This wide scope of required knowledge and skills makes it difficult to design training that fits the capabilities of all the students in small to large classes. Further, since the cybersecurity area is quite popular these days, many students from different backgrounds (even non-technical) are entering cybersecurity programs and classes [2].

Instructors can help the students during hands-on training, however, it is impossible in large classes to actively assist every student. The difficulty with assisting the students is also increased by the complexity of the training. The hands-on cybersecurity training among the wide scope of tools and concepts includes real-world scenarios. These can involve complex network topologies and require deep knowledge of specific tools that made the instructors' assistance much more difficult compared to the introductory programming courses [3].

In the computer science domain, adaptive learning techniques are well-established for adapting tasks for students. In the domain of cybersecurity, there is a lack of solutions that supports this. Cybersecurity learning environments offer static task assignments with limited adaptiveness to the environment [4]. However, there are some attempts that apply adaptive learning techniques to the cybersecurity learning environments [5]. The preparation of such training is quite difficult for the instructor since there is no software support to set up the training metrics. Our main goal is to improve the instructors' preparation for the learning environment using adaptive learning for hands-on training.

In this paper, we present the design of adaptive training and optimization process assisting the instructor in the design phase of hands-on cybersecurity training with adaptive learning. We developed a tool that uses students' knowledge and performance to show the simulated path through the tasks in adaptive training. Using the tool, we demonstrate the importance of the design part of the training and its impact on the students' path through the training. The design process and the tool are verified using a case study using a training session with 19 graduate students. The developed tool is open-sourced [6] and fully integrated into the production release of the KYPO CRP platform [7].

This paper is organized into six sections. Section II describes adaptive learning techniques in computer science and cybersecurity. Section III introduces the design process and recommendations for instructors preparing adaptive hands-on training. Section IV describes the tool and how the instructor can benefit from such a tool. Section V describes the data on which the simulations were performed, including the tutor model used in the tool and the teaching context. Section VI concludes the paper with its practical impact and recommendations for instructors.

II. Related Work

Adaptive learning techniques use computer algorithms, nowadays especially artificial intelligence [8], to adjust the

pedagogical content for the learners and their current state of knowledge. These techniques were introduced in the 1970s [9], and the research area still receives considerable interest. Personalized learning achievable by adaptive techniques was identified by the US National Academy of Engineering as one of the Grand Challenges for Engineering [10].

Since adaptive learning is well-established in computer science, we first review relevant papers in this general domain. Then, we focus on cybersecurity education, particularly hands-on training in an interactive lab environment with a focus on general guidelines and supporting tools for instructors.

A. Adaptive Learning in Computer Science

Adaptive learning techniques have been more or less established in educational fields in recent years. The main reason to establish these techniques is to reflect learners' current knowledge with suitable tasks throughout their learning path.

Over the years, a few strategies have been used in the education sector. In 2012 Colorado Technical University used adaptive learning based on assessments and a faculty-driven approach in one of their computer science courses [11]. Through these courses, each student had their own learning path based on their knowledge and performance. This approach resulted in a higher pass rate in the courses. Moreover, as the authors state, based on student surveys, students' engagement was more prominent in comparison with the non-adaptive structure of the courses.

Khosravi et al. [12] present lessons gained from using the Ripple system, which proposes appropriate learning activities for relational database students. The authors discovered that gamification, such as prizes and leaderboards, is a crucial aspect of the learning system for motivating students.

Aptitude tests assess student knowledge and thus create the student model for Adaptive Learning Systems (ALS). Several aptitude tests exist to determine the skill task of programming students. For example, [13] proposes and evaluates one such test. Although in the domain of programming, this topic has been explored since the 1960s [14], there is no standardized test for cybersecurity.

The previous findings are suitable building blocks for adaptive training in cybersecurity. However, additional aspects need to be considered. These include the option to acquire previous knowledge about students, the variety of data obtainable about the student during the training, the limited time frame for a training session, and the option to modify training phases during the training session.

B. Adaptive Learning in Cybersecurity

Adaptive learning itself is settled as one of the learning techniques in the pedagogical sphere. On the other hand, adaptive learning in cybersecurity is a relatively new research field.

Haag et al. [15] propose a virtual lab with educational enhancements, including a prototype of ALS. It enables hands-on experience to learn BASH commands essential for cybersecurity. The system also provides personalized feedback based on the given knowledge base for a given course exercise. However, it does not provide a fully adaptive training design in terms of changing the tasks or training environment based on the students' performance.

In our previous work [5], we presented an adaptive training model for creating adaptive cybersecurity training. We designed a new training format that changes the original linear structure of the training to a graph-based structure. The training is divided into phases where each phase has one or more tasks on the same topic. The tasks in a phase are ordered from the hardest to the easiest. The model then assigns a suitable task for each student in each phase based on the dedicated performance metrics. These include pre-training assessment, task completion time, solution displayed, submitted answers, and entered commands metrics. Students' evaluation of the newly created type of training provided affirmative conclusions. The results had shown a greater increase in students' ability to finish the training successfully. Apart from that, the overall positive experience from the training was also noticeable. The proposed model is now part of KYPO CRP platform [7]. The platform provides extensive options for collecting data even from adaptive training events [16]. However, the platform lacks support for instructors preparing adaptive training. There are neither guidelines for designing the training, nor a tool for simulation of students' transitions set by the tutor model.

Therefore, considering the non-trivial process of creating adaptive cybersecurity hands-on training and its rare usage in this field, we believe this to be the reason for the absence of research in this area.

III. Adaptive Training Design

We present a process of an effective design of adaptive training, which maximizes the learning experience of students from the training. To design an adaptive training, we proceed in these steps: a) learning environment preparation, b) setting learning objectives, c) base tasks design, d) variant tasks design, e) data gathering and evaluation, and f) testing.

A. Learning Environment Preparation

This step includes the review of the capabilities of the learning environment where the training will be held. Cybersecurity hands-on training usually consists of emulated virtual machines and networks that have to be accessible. In the environment, the training instructor needs to focus on data gathering, available computing resources, and constraints that the environment might bring (e.g., virtual private network requirements). Data gathering is an important aspect to evaluate the training results. Also, the hardware and software resources have to be allocated before the training. Cybersecurity hands-on training requires a significant amount of resources since it needs to replicate the network and hosts from real use cases for multiple students. Each student has their own instance of networked environment.

B. Setting Learning Objectives

The learning objectives represent the first step in the design of training phases. When the instructor identifies the learning objectives they can split the training for particular consecutive tasks. To review available tactics and techniques in cybersecurity, we can leverage existing frameworks, such as MITRE ATT&CK [17]. In this step, it is suitable to know the target audience in advance. Otherwise, the instructor can create training that did not match the needs of students.

C. Base Tasks Design

This step includes the design of base tasks that the students are solving in the hands-on training. The important part of this step is to design the tasks so they are consecutive. Otherwise, the model for adaptive learning can not rely on students' performance metrics from previous tasks. The evaluation in such a case would rely only on theoretical knowledge from pre-training assessment. Furthermore, the tasks should be easy to understand, grammatically and technically correct, and should contain a description of the answer format [18].

D. Variant Tasks Design

This step includes the design of variant tasks of the base tasks. These tasks are assigned to lower- or higher-performing students to better meet their proficiency. First, instructor should decide how many variant tasks for each task you would design. The more is theoretically better, however, also more difficult (time for preparation, negligible difference between tasks). Second, the content of the variant tasks should be as much similar to the base task. Different students should have the feeling that they are undertaking the same training.

E. Data Gathering and Evaluation

The learning environment has to collect and evaluate the data to be able to perform the task adaptivity. Furthermore, it is recommended to store the training data in a suitable format and visualize them using visual analytics [19]. Next, it can be useful for further development of adaptive learning techniques or for other research directions.

F. Testing

This part is a continuous process where the instructor has to verify that the training works as expected. The testing includes checking the learning environment and reviewing the learning objectives and the designed tasks. It is suitable to test the whole solution with another person before the training to avoid unexpected issues.

IV. Tool Assisting Instructor in the Design Phase

The non-trivial process of creating and optimization of adaptive training is the main problem the instructor faces. The adaptive training is based on the model described in [5]. The model uses several weights to make the training truly adaptive to trainees' performance. Therefore, setting the model properly to have the correct task assignment is a crucial part of the design process of training. Tuning parameters of the model requires a significant time investment. During the process of creating entirely new training, the instructor has to work with at that time non-existing trainees' performance. Therefore, the instructor has to figure out the transition path of a trainee to validate the model settings and compute suitable tasks by hand.

In this section, we first introduce two possible use cases of the tool. Next, the requirements for this tool will be considered separately, as they differ in complexity and resource demands based on the use case. Finally, we present the overall design of the tool. Moreover, a detailed description of the tool is encouraged with schemes describing its components.

A. Tool Use-Cases

The main goal of the tool [6] is to ease the non-trivial process of creating adaptive training for instructors. In order to achieve this, the tool helps with the creation process of training definition. More precisely, it is available in the user interface where the model for the given training is specified. This way, the instructor can create simulation runs of trainees with their model and adjust it in real-time. Also, it is possible to use the tool with data exported from past training sessions. This approach allows the instructor to work with a more extensive set of data which are also actual performances of trainees in given training.

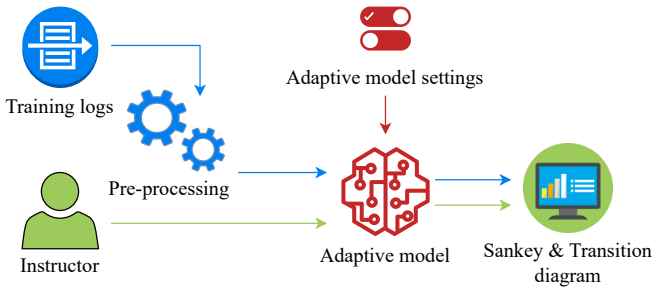

Fig. 1. Architecture of the pre-training and post-training module.

1) Pre-training Module: Figure 1 demonstrates implementation of the pre-training module highlighted in green background. Additionally, the common components of both modules are marked with red color. The pre-training module consists of two primary components illustrated in Figure 3. Namely, the performance matrix and transition graph. First, the performance matrix serves the instructor to simulate trainees' performance. Additional details about the performance matrix are presented in the paper presenting the model itself [5]. Second, the transition graph component shows suitable tasks assigned to a trainee (see Figure 5).

The instructor is required to provide the following metrics: correctness of answers in the pre-training assessment, training completion time, entered commands in the console, information on whether the student displayed a solution, and submitted answers. These metrics serve as a simulation of trainees' performance in training.

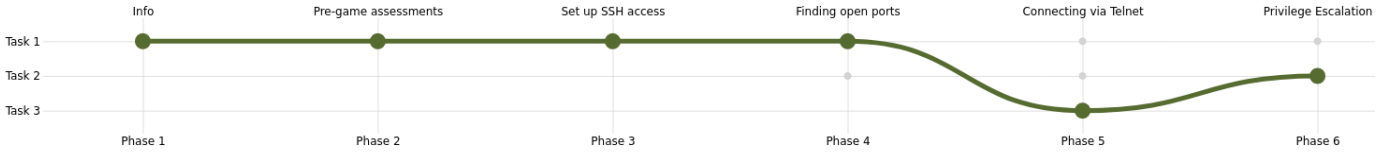

Fig. 2. Performance graph of a single trainee in the pre-training module.

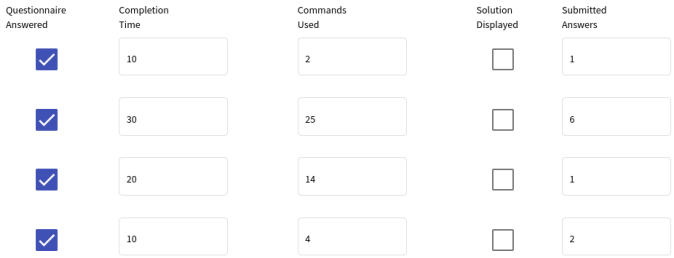

Fig. 3. Performance settings of a single trainee in the pre-training module.

The second component, based on the input from the previous component, computes the trainees' pathway through the training using the model presented in [5].

2) Post-training Module: Post-training module consists of two main components, similar to the pre-training module. These components are highlighted with blue background in Figure 1. Data from previous training instances serve as an input for this module. The first component represents the configurable model [5] of the training instance. The second component illustrates a Sankey diagram representing pathways for every trainee previously present in the training.

Figure 3 presents model settings component that shows current phases of training definition associated with training instance. This component can be of use to change the previously configured adaptive model used in the training instance. The instructor can modify metrics for every phase of adaptive training in this section. Apart from the pre-training module, only model settings can be adjusted, not the performance itself. Changes to the model and performance are displayed in the line graph shown in Figure 2.

Figure 5 displays the Sankey diagram generated from the performance of trainees and metrics of the adaptive model. Every bar of this graph represents one task of the phase of the adaptive training. Following the graph from left to right, the distribution of trainees among tasks can be seen. Where tasks with a lower number, e.g., Task 1 in Training Phase 2 represent the most challenging task in Phase 2. On the other hand, tasks with a higher number represent easier tasks, e.g., Task 3 in Phase 3.

B. Requirements for the Learning Environment

We consider two types of scenarios according to the use cases of the tool individually. Both pre-training and post-training modules of the tool require additional features within the instance they are being deployed, respectively.

The pre-training module is set in a design process of the adaptive training for the mentioned platform. The proposed design and best practices for composing such adaptive training are discussed in Section III. The expected user of the tool proposed in this work is an instructor with experience in the cybersecurity field. Moreover, the instructor should be capable of creating content for the training.

On the contrary, the post-training module has some more advanced requirements as it does simulations on post-training data. Naturally, the module inherits all the requirements to the instructor and platform mentioned earlier. Furthermore, it adds more requirements to the platform. Apart from assisting in the development of revised versions of the training definition, it also helps with the analysis itself. To simulate the pathways of trainees that were present in training, it requires exported logs from analyzed training. Command entries, training events, and answers to the questionnaire must be properly logged [20] to obtain accurate simulations. Moreover, the module makes use of an existing implementation of the adaptive model [5] and performs simulations with the help of existing software services provided by the platform instance within which they are integrated.

C. Tool Design

The post and pre-training modules are built upon the common adaptive model. Furthermore, the post-training module requires additional components that help it pre-process exported training data and generate the Sankey diagram.

First, the common adaptive model component is used by both modules to compute suitable tasks for trainees based on their performance in training. In the pre-training module, the instructor simulates trainee performance. Trainees' performance is the only user input needed for this component apart from model settings on which the instructor decided to run the simulation. On the contrary, the post-training module requires single user input in the form of an adaptive model setting. The trainees' performance is pre-computed and is not modifiable as the module works with the data from a past training session.

More precisely, the post-training module uses an additional component visible in Figure 1. The component pre-processes the data from previous training. As mentioned earlier, trainees' performance is computed from logs that contain training events and commands. Apart from that, the answers in the pre-training assessment are taken into notice. Combining this information, an immutable input for the post-training module is pre-computed and used for further simulations with variable user-defined adaptive model settings.

Last, the tool uses graphical components to demonstrate assigned suitable tasks. The post-training module uses the Sankey diagram for presenting transitions of all trainees. In

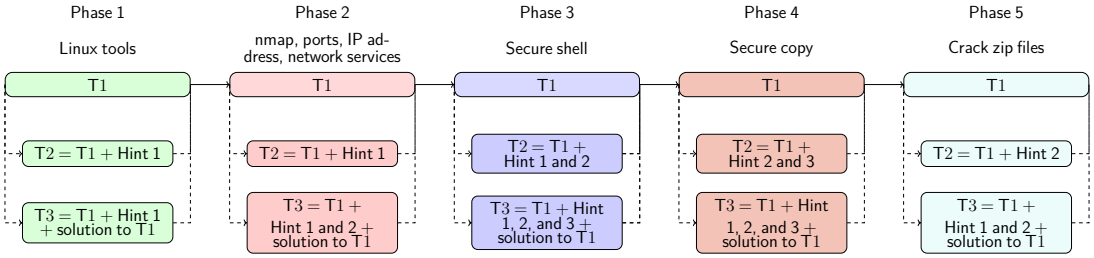

Fig. 4. Phases of the Junior Hacker adaptive training. Assignments of variant tasks enhance base tasks by hints or the solution [5].

contrast, the pre-training module uses a simplified version of the Sankey diagram to present only a single trainee.

V. Case Study

We investigate the potential of adaptive learning in cybersecurity hands-on training using a case study. The study uses a generic adaptive training format [5] on training with data from 19 graduate students. The goal of the study is to validate the proposed tool and highlight the potential of well-designed adaptive learning. Furthermore, the study presents the proposed design process for instructors on how to effectively design adaptive hands-on training.

A. Case Study Setup

1) Teaching Context and Students: We report the adaptive training process and tool usage using a training session with 19 students. The students were graduate students of Masaryk University. The session was designed for two hours and all the students provided informed consent for using their anonymized data for research purposes.

The students were not informed that the training is adaptive. The students entered the training and read the introductory info of the training including the settings of the environment. Further, they filled the pre-training assessment and continued with the training tasks including the usage of a virtualized environment. At the end of the training, the students filled in their feedback. Using the data from the training we used the post-training module to simulate the students' progression through the training. The simulation included a different setup tutor model highlighting its impact on the progression of students through the training.

2) Adaptive Training Design Usage: For the design of the training named "Junior Hacker Training" [21], we follow the steps mentioned in the Section III.

The preparation of the learning environment in our case represents the allocation of essential resources in the OpenStack cloud. This includes hundreds of virtual CPUs and tens of GBs of RAM. Further, we tested essential parts of the adaptive learning environment in the KYPO CRP platform. These include auditing of events and commands, visualizations, and all the remaining parts of the software related to the training.

The learning objectives of this training are centered around the application of essential tools for students of cybersecurity or related fields. These include using Linux tools and commands, such as `ssh`, `nmap`, `scp` and others.

For the design of base tasks, we first considered the time frame for the training. It was held in the seminar lab session, so we designed it to be two hours long and has five phases as depicted in Figure 4. The first phase contains Linux tools to practice the usage of the command line. The second phase targets the usage of the `nmap` command to search for an opened ports. The third phase targets the usage of `ssh` as an essential tool for remote operations on the servers. The fourth phase practices `scp` command for copying files from/to the server. Finally, the fifth phase exercised cracking encrypted ZIP archive.

For the variant task design, we extend these phases (base tasks) so each phase contains one base task and two variant tasks. Further, each phase features a task presenting the step-by-step solution. This was a last-resort task for students who would not match any phase prerequisites. In the first training phase, basic Linux tools are practiced in three variant tasks (T1, T2, and T3). Task T2 contains the same assignment as T1 and provides Hint 1. The third task T3 contains the assignment from T1 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 solutions provided. The tasks were assigned to each student by the proposed model presented in Section V-B. The recommendations for the instructors creating tasks in adaptive training are presented in Section VI-B.

In the data gathering and evaluation step, we checked all the data that are collected in the KYPO CRP platform so none of the input data is missing. Further, we checked that the Sankey diagram and other visualizations are generated correctly.

Testing was performed continuously in each of the steps of the training preparation. The developed tool significantly helps to verify that the training is correctly set. Instructors should not omit this step, otherwise, the data might be invalid and the collected results will be misleading. In the worst scenario, the training cannot be held since some of the essential functionalities are not working.

B. Tutor Model

The case study was performed on the model assigning suitable tasks for each student of the training presented in [5]. The model consists of three equations. The first defines the weights for the designated metrics. The metrics include pre-training assessment, training completion time, entered com-

mands in the console, submitted answers, and information on whether the student displayed a solution. The second equation computes the performance of each student based on the "score" the student achieved divided by the maximal possible score for selected metrics. The results from the second equation are in the interval of $[0, 1]$. The third equation takes the result from the second equation and the number of variant tasks in a phase. The third equation divides the interval $[0, 1]$ into the same number of intervals as the number of variant tasks. Based on that, the equation returns suitable tasks for each student. The first interval is Task 1 (T1), the second interval is Task 2 (T2), and so on.

Although the model was piloted with performance metrics [22] used for cybersecurity training, it can be applied in any domain collecting the same or similar data.

1) Initial Model Settings: To use the model, the weights for the listed metrics for each training phase must be set. These weights indicate the relationships between training phases. For simplicity, we set these weights to zero or one in our case study. One indicates the relationship and zero indicates that there is no relationship between the phases.

Considering the phases depicted in Figure 4, the model is set as follows. Each training phase relies on a pre-training assessment associated with a particular phase. The first phase relies only on pre-training assessment since no other performance indicators are available at that time. The second phase relies on commands, task completion time, and solution displayed metrics from phase one. The third phase relies on the solution displayed metric from phase two and from phase one relies on commands, solution displayed and task completion time metrics. The fourth phase relies on solution displayed, commands, submitted answers from phase three, and on solution displayed and task completion time metrics from phases two and one. The fifth phase relies on task completion time from phases four, three, and two, and on commands, task completion time, solution displayed, and submitted answers from phase one.

2) Modified Model Settings: The modified model settings are the same as the initial model settings except we do not consider the pre-training assessment metric. So none of the phases uses pre-training assessment metrics in the evaluation to assign a suitable task.

C. Results and Discussion

We report the results of the study and summarize the impact of the developed tool and model settings on trainees' experience.

1) Results of Simulations: Figure 5 shows the transitions of 19 students between training phases and their tasks in the past training. Based on the transitions visible on the graph we can easily see the variety in students' skills. Even in the first training phase (P1), there is a group of students who were provided with the easiest task based on the pre-training assessment questionnaire. Weights of phase P1 were set to take the results of the pre-training assessment in process of selecting the suitable task. The selection of weights in the following phases further determined the suitable tasks for each student. Whether the student received the easiest task (step-by-step solution) can be based on exceeding the estimated time in the previous phase or displaying its solution.

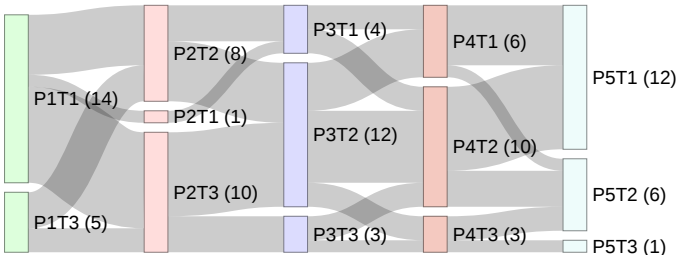

Fig. 5. Transitions of 19 students between particular tasks in training for *initial model settings*. P*X*T*Y* denotes task T*Y* in the phase P*X*. The number of students solving the task is in brackets.

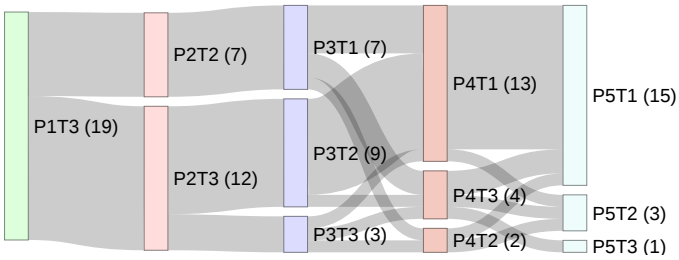

Fig. 6. Transitions of 19 students between particular tasks in training for modified model settings *without pre-training assessment metric*. P*X*T*Y* denotes task T*Y* in the phase P*X*. The number of students solving the task is in brackets.

Furthermore, in Figure 5 we can identify a particular group of five students that transitioned from the easiest task of phase P1 to the second hardest task in P2. We can observe that these students were given the easiest task even when in the following phase they belonged to a group of students with the second hardest task. These phenomena can be observed in another group of students as well. In phase P1, there is a group of eight students who after the pre-training assessment obtained the hardest task. Later, following phase P2, this group of students received the easiest task in the phase. These observations show that the variability of task assignments is significantly impacted by the model settings and the variety of students' performance.

To conclude, we decided to remove the impact of the filling of the pre-training questionnaire from model settings. This alternation is made within every phase of simulated training. More precisely, we set the weight of whether the trainee answered the questionnaire or not to zero.

Modified model weights show a significant increase in non-trivial task occurrence in Figure 6. This tendency is more visible in phase P2. The main reason for this is that the earlier phases depend more on the pre-training questionnaire. In contrast, further phases use the number of commands, submitted answers, etc., as the weight for assigning the suitable task. These differences highlight the importance of model settings and their impact on the task assignment for students.

2) Discussion: The modifiability of the model settings is relatively spacious, and therefore it is a non-trivial task to choose the settings appropriately. The instructor is capable to set the weights in the model to any non-negative real number. Therefore, the number of possible combinations is infinite. In our simulations, we restricted the value of the possible weights to binary. By this constraint, the complexity of possible combinations of settings is exponential.

This complexity of weights settings affects the difficulty of training preparation for the instructor. The instructor has to "manually" compute the expected task assignment for differently performing students. Moreover, the instructor is not able to validate the model settings during that process. The instructor had to test the model settings by pre-computing the results "manually" or running the training and acting as differently performing students. Such a process is very laborious and costly since it requires significant resources (e.g., instructors' time and resources of the learning environment). The developed tool can reduce the time and resources required for testing.

Furthermore, adaptive learning techniques seem to be promising since the students of different proficiency did not get the same tasks. The students were assigned different tasks based on their performance during the training. All the students successfully finished the training.

VI. Conclusion

Task adaptivity in cybersecurity hands-on training seems to be a promising approach to increase the students learning experience. In the domain of cybersecurity, this approach is especially essential since more and more students with different backgrounds are entering this field. In this paper, we enhanced our previous work [5] with a design process to ease the creation of adaptive training for instructors. Further, we developed a tool [6] that simplifies the testing and optimization phase for the instructors to verify that the adaptive training is set up properly. To ease the adoption of the developed tool, we integrated it into the KYPO CRP platform [7].

A. Practical Impact

The adaptive training design helps the instructors to follow a guide to successfully prepare for a hands-on training. The example training in the case study supports the usage of adaptive training design. It provides a more easily understandable way of how it can be used.

The proposed tool simulating the students' transitions between tasks significantly helps in the design phase. This tool supports the optimization of the existing and new training in terms of better metrics selection and weighting. It makes the instructors' view of the path transitions through the training for differently performing students easier. Furthermore, it saves tens of minutes or even hours to verify the training settings. It avoids the need to allocate necessary hardware and software resources to run the whole training whenever we modify the training settings.

B. Recommendations for Instructors

1) Design smaller tasks rather than a few bigger ones: The tasks of training that can be made adaptive should separate learning outcomes into individual tasks. This structure simplifies the design of pre-training assessments, variant tasks, and hints in each task.

2) For each base task, provide variant tasks that even a low-performing student can complete: The proficiency of the students and their performance in the training session varies a lot. If the instructor does not provide variant tasks for low-performing students, they can get stuck as if there were no variant tasks.

3) For each base task, provide a default task containing a worked-out solution: As a last-resort option, we suggest adding a default variant task with step-by-step instructions on how to complete it. This default task enables even low-performing students to continue.

4) Use the same approach for unplugged training: The used adaptive training format can be easily applied to tabletop exercises, which do not involve any lab environment. Pre-training assessment and measurements of variables capturing student progress (such as time to complete) can be done without a training platform.

Acknowledgment

This research was supported by 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] D. Mouheb, S. Abbas, and M. Merabti, *Cybersecurity Curriculum Design: A Survey*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019, pp. 93–107. [Online]. Available: https://doi.org/10.1007/978-3-662-59351-6_9

[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," *Computers & Security*, vol. 65, pp. 153–165, 2017. [Online]. Available: https://doi.org/10.1016/j.cose.2016.10.007

[3] R. Beuran, D. Tang, C. Pham, K.-i. Chinen, Y. Tan, and Y. Shinoda, "Integrated framework for hands-on cybersecurity training: Cytrone," *Computers & Security*, vol. 78, pp. 43–59, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2018.06.001

[4] C. Braghin, S. Cimato, E. Damiani, F. Frati, L. Mauri, and E. Riccobene, "A Model Driven Approach for Cyber Security Scenarios Deployment," in *Computer Security*, A. P. Fournaris, M. Athanatos, K. Lampropoulos, S. Ioannidis, G. Hatzivasilis, E. Damiani, H. Abie, S. Ranise, L. Verderame, A. Siena, and J. Garcia-Alfaro, Eds. Cham: Springer International Publishing, 2020, pp. 107–122. [Online]. Available: https://doi.org/10.1007/978-3-030-42051-2_8

[5] P. Seda, J. Vykopal, V. Švábenský, and P. Čeleda, "Reinforcing Cybersecurity Hands-on Training With Adaptive Learning," in *2021 IEEE Frontiers in Education Conference (FIE)*. New York, USA: IEEE, 2021, p. 1–9. [Online]. Available: https://doi.org/10.1109/FIE49875.2021.9637252

[6] I. Ignác, P. Seda, J. Vykopal, and P. Čeleda, "KYPO Adaptive Model Simulator," 2022. [Online]. Available: https://gitlab.ics.muni.cz/muni-kypo-crp/frontend-angular/components/kypo-adaptive-model-simulator

[7] J. Vykopal, P. Čeleda, P. Seda, V. Švábenský, and D. Tovarňák, "Scalable Learning Environments for Teaching Cybersecurity Hands-on," in *2021 IEEE Frontiers in Education Conference (FIE)*. New York, USA: IEEE, 2021, pp. 1–9. [Online]. Available: https://dx.doi.org/10.1109/FIE49875.2021.9637180

[8] K. Colchester, H. Hagras, D. Alghazzawi, and G. Aldabbagh, "A Survey of Artificial Intelligence Techniques Employed for Adaptive Educational Systems within E-Learning Platforms," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 7, no. 1, pp. 47–64, 2016. [Online]. Available: https://doi.org/10.1515/jaiscr-2017-0004

[9] J. R. Carbonell, "AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction," *IEEE Transactions on Man-Machine Systems*, vol. 11, no. 4, pp. 190–202, 1970. [Online]. Available: https://doi.org/10.1109/TMMS.1970.299942

[10] National Academy of Engineering. (2008) NAE Grand Challenges For Engineering. National Academy of Engineering. [Online]. Available: http://www.engineeringchallenges.org/9127.aspx

[11] R. Cai, "Adaptive Learning Practice for Online Learning and Assessment," in *Proceedings of the 2018 International Conference on Distance Education and Learning*, ser. ICDEL '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 103–108. [Online]. Available: https://doi.org/10.1145/3231848.3231868

[12] H. Khosravi, S. Sadiq, and D. Gasevic, "Development and Adoption of an Adaptive Learning System: Reflections and Lessons Learned," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 58–64. [Online]. Available: https://doi.org/10.1145/3328778.3366900

[13] M. Tukiainen and E. Mönkkönen, "Programming aptitude testing as a prediction of learning to program," in *Proceedings of the 14th Annual Workshop of the Psychology of Programming Interest Group*. London, UK: Psychology of Programming Interest Group, 2002, pp. 45–57.

[14] M. A. Howell, J. W. Vincent, and R. A. Gay, "Testing Aptitude for Computer Programming," *Psychological reports*, vol. 20, no. 3_suppl, pp. 1251–1256, 1967. [Online]. Available: https://doi.org/10.2466/pr0.1967.20.3c.1251

[15] J. Haag, H. Vranken, and M. van Eekelen, *A Virtual Classroom for Cybersecurity Education*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019, pp. 173–208. [Online]. Available: https://doi.org/10.1007/978-3-662-59351-6_13

[16] V. Švábenský, J. Vykopal, P. Seda, and P. Čeleda, "Dataset of shell commands used by participants of hands-on cybersecurity training," *Data in Brief*, vol. 38, p. 107398, 2021. [Online]. Available: https://doi.org/10.1016/j.dib.2021.107398

[17] MITRE, "MITRE ATT&CK," 2022. [Online]. Available: https://attack.mitre.org/

[18] M. Gáliková, "Methods for Designing Educational Cybersecurity Games," Master's thesis, Masaryk University, 2021. [Online]. Available: https://is.muni.cz/th/uovmy/Galikova-thesis.pdf

[19] R. Ošlejšek, V. Rusňák, K. Burská, V. Švábenský, J. Vykopal, and J. Čegan, "Conceptual Model of Visual Analytics for Hands-on Cybersecurity Training," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 8, pp. 3425–3437, 2021. [Online]. Available: https://doi.org/10.1109/TVCG.2020.2977336

[20] V. Švábenský, J. Vykopal, D. Tovarňák, and P. Čeleda, "Toolset for Collecting Shell Commands and Its Application in Hands-on Cybersecurity Training," in *2021 IEEE Frontiers in Education Conference (FIE)*. New York, USA: IEEE, 2021, pp. 1–9. [Online]. Available: http://dx.doi.org/10.1109/FIE49875.2021.9637052

[21] M. Gáliková, V. Švábenský, and J. Vykopal, "Junior Hacker Adaptive Training," 2021. [Online]. Available: https://gitlab.ics.muni.cz/muni-kypo-trainings/games/junior-hacker-adaptive

[22] K. Maennel, "Learning Analytics Perspective: Evidencing Learning from Digital Datasets in Cybersecurity Exercises," in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, 2020, pp. 27–36. [Online]. Available: https://doi.org/10.1109/EuroSPW51379.2020.00013