Automated Feedback for Participants of Hands-on Cybersecurity Training

Valdemar Švábenský, Jan Vykopal, Pavel Čeleda and Ján Dovjak

Faculty of Informatics, Masaryk University, Botanická 68a, Brno, 60200, Czech Republic.

*Corresponding author(s). E-mail(s): valdemar@mail.muni.cz; Contributing authors: vykopal@fi.muni.cz; celeda@fi.muni.cz; dovjak@mail.muni.cz;

Abstract

Computer-supported learning technologies are essential for conducting hands-on cybersecurity training. These technologies create environments that emulate a realistic IT infrastructure for the training. Within the environment, training participants use various software tools to perform offensive or defensive actions. Usage of these tools generates data that can be employed to support learning. This paper investigates innovative methods for leveraging the trainee data to provide automated feedback about the performed actions. We proposed and implemented feedback software with four modules that are based on analyzing command-line data captured during the training. The modules feature progress graphs, conformance analysis, activity timeline, and error analysis. Then, we performed field studies with 58 trainees who completed cybersecurity training, used the feedback modules, and rated them in a survey. Quantitative evaluation of responses from 45 trainees showed that the feedback is valuable and supports the training process, even though some features are not fine-tuned yet. The graph visualizations were perceived as the most understandable and useful. Qualitative evaluation of trainees' comments revealed specific aspects of feedback that can be improved. We publish the software as an open-source component of the KYPO Cyber Range Platform. Moreover, the principles of the automated feedback generalize to different learning contexts, such as operating systems, networking, databases, and other areas of computing. Our results contribute to applied research, the development of learning technologies, and the current teaching practice.

Keywords: cybersecurity education, interactive learning environments, hands-on training, virtual labs, shell commands

1 Introduction

Practical training is crucial to improving one's skills and competencies. However, the hands-on practice may not be enough on its own. Formative feedback – information intended to modify a learner's thinking or behavior to improve learning (Shute, 2008) – is crucial for deepening the learning gains. Formative feedback explains to the learner how they progressed, what they did well, and what they can improve. Nevertheless, this feedback is typically served manually by instructors, which is a time-consuming task. As a result, the feedback is delayed and does not scale for large groups of learners.

This shortcoming can be remedied by leveraging data generated by learners (Švábenský, Vykopal, Čeleda, & Kraus, 2022). In cybersecurity training, students often use various command-line tools since many tools do not have a graphical user interface. As a result, they generate command history data, which include submitted commands with their arguments and metadata, such as timestamps. We aim to analyze these data to provide automated, personalized feedback after the training. The feedback focuses on conformance to the expected solution, solution patterns, and error discovery.

Apart from supporting individual learners, instructors also benefit from the data analytics associated with the feedback. For example, aggregated insights about common mistakes or solution approaches help instructors understand the students and adapt the instruction to the students' needs. These insights can also be presented during the training to improve the instructors' classroom situational awareness.

1.1 Application Areas of Learning

In cybersecurity, hands-on training frequently involves practicing cyber attacks and defense in an *interactive learning environment* (ILE). By ILE, we mean a computer system, such as a testbed or a cyber range (Yamin, Katt, & Gkioulos, 2020), that allows practicing cybersecurity skills in a realistic IT infrastructure.

An ILE allows learners to practice tasks that involve attacking and defending emulated computer systems without impacting real systems. For example, these tasks may include:

- penetration testing of a vulnerable system,
- cracking passwords,
- configuring network defense mechanisms,
- analyzing network communication,
- administrating operating systems.

The target audience for such training is security professionals and computer science students at a university level.

While we demonstrate the application of our research in the security domain, the feedback principles apply also to other areas. For example, when teaching databases, SQL tools provide command history (DelftStack, 2023) that can be processed analogously to commands from cybersecurity training. In networking classes, configuring firewall rules can be represented as a sequence of commands (Sehl & Vaniea, 2018). Even though graphical integrated development environments exist, command-line tools are still essential for teaching programming (Malmi, Utting, & Ko, 2019).

In general, the feedback methods proposed in this paper are applicable to any exercise in which the student activity can be represented by high-level "commands". Examples include tasks with a clearly defined set of possible actions (Hao, Shu, & von Davier, 2015) or games for teaching computational thinking (Rowe et al., 2021).

1.2 Contributions and Structure of This Paper

First, this paper provides a thorough review of current literature (Section 2). We survey related studies and explain the key terms to familiarize readers with state of the art.

Second, we propose and implement open-source learning software for generating post-training feedback (Section 3). The software automatically processes command-line data from hands-on training to provide insights for learners and instructors. It is based on principles synthesized from the literature review.

Third, to demonstrate the value of such software, we deploy and evaluate it in five field studies. We examine the feedback's understandability and usefulness in authentic teaching contexts with 58 students from universities in two countries.

Section 4 describes our methods for collecting and analyzing research data. Section 5 presents and discusses the findings and proposes future research challenges. Finally, Section 6 concludes and summarizes the practical impact of this work.

2 Background and Related Work

This section reviews related publications about educational feedback on learning and technologies for delivering it.

2.1 Definition of Feedback and Motivation for Providing It

Educational feedback is "information provided by an agent (e.g., teacher, peer, book, parent, self, experience) regarding aspects of one's performance or understanding" (Hattie & Timperley, 2007).

Researchers and practitioners agree that quality feedback to students has a significant positive impact on the learning process (Petty, 2009, p. 480). It is "a major predictor of both good teaching and good learning" (Shephard, 2019, p. 283). Challenging exercises with extensive performance feedback "support students' engagement in independent-learning activities" (Thomas, 1993).

However, the frequency of feedback in typical classrooms is low (Hattie & Timperley, 2007), so this element of learning offers room for improvement.

Our work aims to provide learners with formative feedback that is automatically generated based on the learners' data. Compared to summative feedback, which focuses on the achieved outcomes (such as the grade), formative feedback is "information communicated to the learner that is intended to modify his or her thinking or behavior to improve learning" (Shute, 2008).

2.2 Features, Content, and Delivery of Feedback

Based on the theory formulated by Hattie and Timperley (Hattie & Timperley, 2007), feedback should answer three questions: *(1) Where am I going?* (goal definition), *(2) How am I going?* (progress assessment), and *(3) Where to next?* (determining future steps).

Each question can be communicated at four levels (Hattie & Timperley, 2007):

- *task*: information about how well a task is being accomplished (e.g., corrective feedback on the result),
- *process*: information about the processes underlying the task completion (e.g., explaining how to perform a task),
- *self-regulation*: information about how one's actions are guided (e.g., setting learning goals), and
- *self*: personal affective comments (e.g., praise), which is often less effective in terms of learning gains.

This paper focuses on feedback at the *task* and *process* levels, though the feedback may influence the *self-regulation* level as well. For example, communicating to a student how they progressed may lead them to set new learning goals, such as "I need to review how networking in Linux works".

How feedback is delivered matters (Mouratidis, Lens, & Vansteenkiste, 2010). If it is meaningful, specific, and perceived as legitimate, it supports persistence and well-being. Fraser et al. (Fraser, Ngoon, Weingarten, Dontcheva, & Klemmer, 2017) state that feedback should be positive, specific, and hint toward the solution of the learning problem.

Kölling and McKay (Kölling & McKay, 2016) proposed 13 criteria based on which to judge learning systems for novice programmers. One of those criteria is that the system should provide automated feedback. Two features of such feedback were highlighted as especially important: it should be available when the learner needs it, and it should be helpful in order to support learning.

Bodily and Verbert (Bodily & Verbert, 2017) reviewed 93 articles about learning analytics dashboards and reporting systems for students. They identified that such systems must help students in two key aspects: understanding what happened and recommending what to do next. The post-training feedback in this paper focuses on "understanding what happened".

In the field of learning dashboards, automatic analysis of learning data to provide feedback to students is a highly relevant research topic. It was identified as one of the "important open issues and future lines of work" based

on a literature review of 55 papers (Schwendimann et al., 2017). In the field of cybersecurity, this topic is also an open issue, since research on automated feedback in cybersecurity training is not yet widely established.

2.3 Manual and Automated Feedback

Although grading students and providing feedback to them manually is the traditional approach, computing instructors consider it "the highest priority pain point" in their classes (Codio, 2019). This activity is time-consuming, especially in large classes or at institutions that lack resources for teaching assistants.

As a result, manual feedback is often delayed: students have to wait for it for several days or even weeks (Vykopal, Švábenský, & Chang, 2020). This diminishes the feedback's educational value. In addition, manual feedback can be inaccurate if student data are extensive and complex (Švábenský, Weiss, et al., 2022). The data may contain many variables, solution approaches, and behavioral patterns. Lastly, providing feedback manually may be complicated during online learning (such as when forced by the COVID-19 pandemic restrictions (Pokhrel & Chhetri, 2021)).

An obvious advantage of automated feedback is its timeliness. It saves the instructor's time and reaches the student much faster, increasing its educational impact. In addition, automated feedback is usually more accurate (free from human error), more fair (free from human bias), scalable, easily replicable, and more anonymous for the student. Even if full automation is not viable or suitable, some form of automated feedback can be an excellent complement to manual feedback.

Although research into the effects of automated feedback in computing education is in its early phases, automated feedback has disadvantages as well. Its perception by stakeholders likely suffers from "algorithm aversion", meaning that humans irrationally prefer to receive information from other humans, even if the algorithmic, automated solution is more accurate (Dietvorst, Simmons, & Massey, 2015). Another drawback is that unless it is clear how the feedback is generated, it can be perceived as less transparent, as is the issue with almost every data-driven reporting system (Lepri, Oliver, Letouzé, Pentland, & Vinck, 2018).

2.4 Automated Feedback Systems in Computing Education

Computing educators' need for faster assessment has propelled the rise of feedback systems for programming classes. These systems run instructor-defined tests on student code and provide tailored feedback messages. For example, Web-CAT (Edwards & Pérez Quiñones, 2008) includes a library that allows the instructor to associate a hint with each test case. When the submitted code fails a test, the corresponding hint is displayed as feedback (Haldeman et al., 2018).

However, some students abuse tools for autograding of code (Baniassad, Zamprogno, Hall, & Holmes, 2021). For example, they submit trial-and-error corrections to their code until the autograding tool reports no errors. Moreover, some automated tools may inadvertently encourage this practice (Buffardi & Edwards, 2015). One of the possible solutions is providing high-level conceptual feedback instead of detailed test results (Cordova, Carver, Gershmel, & Walia, 2021).

Apart from hints (Elkherj & Freund, 2014; T.W. Price et al., 2019) and the results of code testing (Bruzual, Montoya Freire, & Di Francesco, 2020), feedback can consist of suggested code edits (T. Price, Zhi, & Barnes, 2017; T.W. Price, Dong, & Lipovac, 2017), step-by-step examples (Wang et al., 2020), encouragement messages (Marwan, Gao, Fisk, Price, & Barnes, 2020), pre-prepared instructor prompts (Pardo, Jovanovic, Dawson, Gašević, & Mirriahi, 2019), rule-based prompts (Chen, Ciborowska, & Damevski, 2019; Harden, Gusukuma, Bart, & Kafura, 2021), peer-to-peer assessment (Fraser et al., 2017; Kulkarni, Bernstein, & Klemmer, 2015), and generative grading (Malik et al., 2021).

Keuning et al. (Keuning, Jeuring, & Heeren, 2018) compared 101 tools that provide automated feedback on programming assignments. Most tools provide a similar type of feedback, focusing on pointing out mistakes (97 tools), followed by suggesting the next steps (45 tools). The tools predominantly use techniques such as automated testing and static analysis of code. Some of these techniques could be adapted to cybersecurity exercises in the command line. However, to the best of our knowledge, there has been no published attempt to do so. Furthermore, techniques that work in the programming domain are not transferable to other types of tasks, such as network traffic analysis.

Although programming is the most researched domain of computing education, automated feedback was explored in other domains as well. Marchiori (Marchiori, 2022) used a tool for assessing command-line tasks in system-level courses. It enables students to check their solutions locally and instructors to grade them. Bezáková et al. (Bezáková, Hemaspaandra, Lieberman, Miller, & Narváez, 2020) proposed a feedback system for teaching automata, grammars, and regular expressions. The system checks exact string submissions and compares them to the canonical solution. This approach is also the most commonly used method identified in surveying 63 publications about feedback in online learning systems (Cavalcanti et al., 2021). However, it does not apply to many security problems since they rarely have a single correct answer that is easy to check.

The majority of automated solutions generate summative feedback (Kovanović, Joksimović, Gašević, Hatala, & Siemens, 2017). While they provide a quick overview and help instructors with grading, they do not support students' learning. Summative feedback does not provide insight into aspects such as learning strategies (Macfadyen & Dawson, 2010). Instead, formative feedback that identifies "weaknesses and suggestions for overcoming them" (Kovanović et al., 2017) is valuable for increasing the educational impact.

Finally, there are two types of educational data mining models: *descriptive* and *predictive* (Peña-Ayala, 2014). Descriptive models explain the structure and relations within the mined data. Predictive models "estimate unknown or future values of dependent variables based on the" related independent variables. To provide formative feedback, we build descriptive models from the collected and analyzed data. However, generating feedback on open-ended assignments poses a major research challenge, because these assignments have multiple correct approaches to the solution (Chow, Yacef, Koprinska, & Curran, 2017).

2.5 Automated Feedback Systems in Cybersecurity Education

When it comes to providing feedback to students, cybersecurity instructors face similar challenges as instructors in other computing domains. Yet, few automated systems address these challenges. The current feedback systems support mostly real-time situational awareness during the training. However, the feedback to learners after the training is limited, and teachers lack insight into the educational impact on learners (Ošlejšek et al., 2020).

SERA (Agudo, Rios, & Nieto, 2019) is a framework for automated feedback on cybersecurity assignments. It allows defining conditions for checking the student-generated data. For example, suppose a student has to generate an encryption key. A condition C can verify if the key is at least 2048 bits long. Based on whether the check passes or fails, the student receives a feedback message associated with the condition C. This aspect of SERA resembles the systems for displaying instructor-defined prompts or hints. Our system provides post-training feedback to support reflection after learning.

The typical feedback for cyber defense exercises and cybersecurity competitions is summative, in the form of score dashboards (Chung, 2017; Doupé et al., 2011; Ošlejšek et al., 2020). One of the exceptions is Frankenstack (Kont, Pihelgas, Maennel, Blumbergs, & Lepik, 2017; Pihelgas & Kont, 2021), an automated feedback framework for supporting attacking teams in defense exercises. It monitors the exercise infrastructure to detect activities such as indicators of compromise in the network. Then, it displays real-time visual feedback about ongoing attacks on the defending teams. Contrary to our system, which shows post-training feedback to enhance learning, Frankenstack provides situational awareness to the exercise participants during the training.

A slight limitation of research that has been done in this field is that different papers use different metrics to evaluate their feedback. As a result, it is not possible to quantitatively compare these feedback interventions.

To conclude the review of state of the art, feedback must enable learners to understand what they did well and in which areas they can improve. However, the related work does not yet address some of the core challenges in providing post-exercise feedback. These include improving the understanding of learning through meaningful uses of data (Ošlejšek et al., 2020), as well as supporting reflection after the training. We aim to address these challenges using the system described in the next section.

3 Learning Technologies for Hands-on Training and Automated Feedback

This section details the infrastructure for conducting cybersecurity training and the training format. Then, it explains the data collection and the proposed feedback system.

3.1 Interactive Learning Environment

The training can be hosted in any ILE that enables practicing hands-on tasks in a realistic setting. A key requirement is that the ILE must also collect data from the training to enable feedback provision. Figure 1 shows the conceptual lifecycle of training and automated feedback.

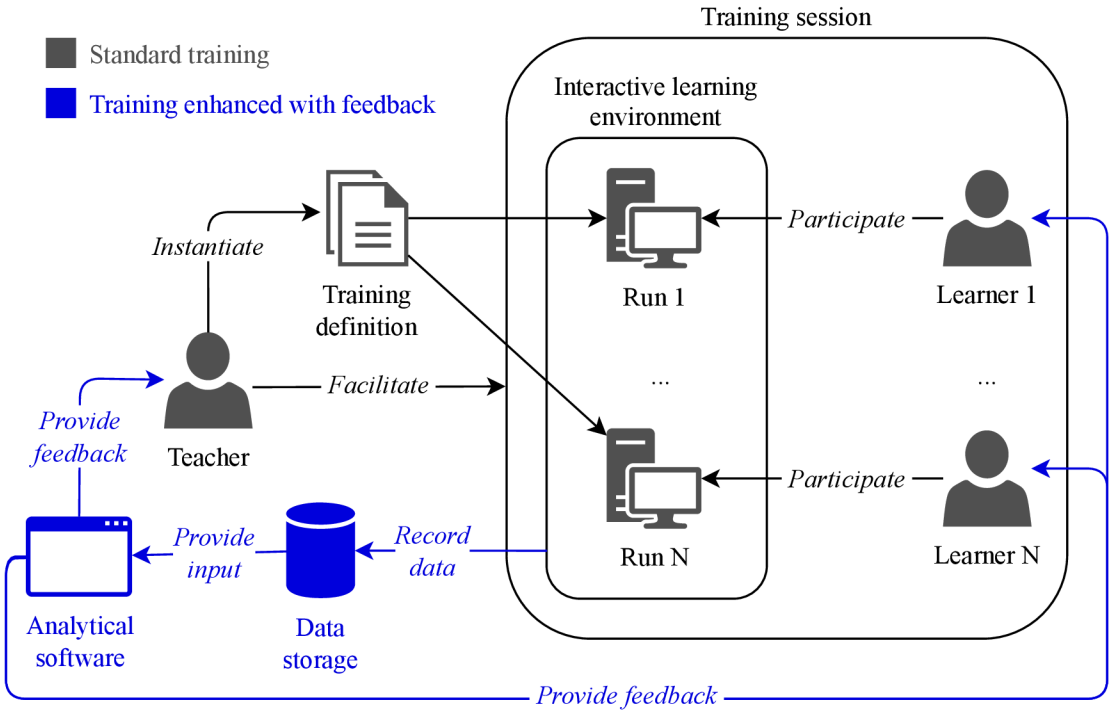

Fig. 1 The generic lifecycle of hands-on training in an ILE enhanced with automated feedback.

For this paper, we selected to host hands-on cybersecurity training in KYPO Cyber Range Platform (only KYPO CRP from now on), an advanced cloud-based ILE for emulating complex IT systems. KYPO CRP uses virtual machines (VMs) with full-fledged operating systems and software to emulate real-world infrastructures, networks, and applications. KYPO CRP is open-source software (Masaryk University, 2023a), and its design and architecture are documented in a previous publication (Vykopal, Čeleda, Seda, Švábenský, & Tovarňák, 2021).

3.2 Cybersecurity Training Format and Content

In the scope of our research, the training's learning objectives cover technical skills, especially data, network, and system security as defined by the CSEC2017 curricular guidelines (Joint Task Force on Cybersecurity Education, 2017). The training tasks involve gradually compromising an emulated IT system, following the stages of Mandiant's Attack Lifecycle (Mandiant, 2013, p. 27). Teaching such skills supports the development of "adversarial mindset" (OConnor, 2022). This is essential for (future) cybersecurity experts since it enables them to understand cyber attacks and set up effective defenses (Švábenský, Vykopal, Cermak, & Laštovička, 2018). Like other state-of-the-art teaching frameworks (Chothia, Holdcroft, Radu, & Thomas, 2017; Hall et al., 2022), the training features an engaging storyline to motivate the participants.

The training in our research features mainly cybersecurity command-line tools. The learner controls a VM with Kali (Offensive Security, 2023): a Linux operating system distribution for penetration testing. Kali provides the necessary command-line tools, which are often taught in offensive security courses (Ghafarian, 2019). Using Kali, the learner completes a sequence of assignments that involve attacking intentionally vulnerable hosts in the emulated network.

For this paper, we used an open-source training named *Locust 3302* (Masaryk University, 2021; Švábenský, Weiss, et al., 2022), in which the trainees have to scan a suspicious server using `nmap` (Lyon, n.d.), identify a vulnerable service, and exploit it using Metasploit (Offensive Security (OffSec Services Limited), 2023) to gain access. Then, they have to copy a private SSH key, crack its passphrase with John the Ripper (`john`) (Openwall, n.d.), and use it to access another host. Students work on the training tasks individually, as Figure 2 shows. However, they are free to use optional on-demand hints included in the training or ask the instructor for assistance, mainly regarding the potential technical issues with the ILE.

Fig. 2 Students in a computer lab are working on the cybersecurity training *Locust 3302*.

3.3 Data Collection

In order to provide feedback on the learners' actions, we used an open-source toolset for collecting shell commands from the training machines (Švábenský, Vykopal, Tovarňák, & Čeleda, 2021). While the learners solve the training assignments in a command-line interface, the toolset automatically acquires their submitted commands and the associated metadata, such as timestamps, hostnames, and IP addresses of the machines where the commands are executed. Then, the data are formatted as JSON records (see Figure 3) and stored in dedicated storage. These data are automatically provided as input for the feedback system (see Section 3.5 for architectural details).


```
{
    "timestamp"  : "2022-04-13T10:11:34+01:00",
    "username"   : "root",
    "hostname"   : "attacker",
    "ip"         : "10.2.154.7",
    "sandbox_id" : "3",
    "wd"         : "/home",
    "cmd"        : "nmap 172.1.18.7",
    "cmd_type"   : "bash-command"
}
```


Fig. 3 Example of a single log record from a command history of one trainee.

3.4 Requirements and Design of the Feedback System

We proposed a system that extends the ILE by generating post-training feedback (i.e., provided after the student finishes the training). This feedback is based on the student command-line data. The process of creating the system consisted of four stages similar to those we followed in previous work (Ošlejšek, Rusňák, Burská, Švábenský, & Vykopal, 2019):

1. Understanding the domain constraints, available data, and the resulting insights.
2. Defining requirements on the system in accordance with state of the art and the educational goals of the training.
3. Prototyping, iterative design, and development.
4. Performing field studies with computer science students.

Initially, we hosted a focus group discussion with five instructors from Masaryk University, Czech Republic, who had a minimum of one year and a maximum of 11 years of experience teaching cybersecurity at a university level. We asked them to brainstorm requirements that users would have on the feedback. Then, we combined the focus group outputs with our teaching experience and the literature review (Section 2) to define key properties of the feedback. To address the needs of educational stakeholders, feedback provided by the system should be:

- *Automated*: the feedback must be generated at scale, without manual interventions for any number of learners. As a result, learners can work with the system independently.
- *Timely*: the feedback must occur as soon as possible, during or just after the training session. This allows learners to reflect on their progress while they still remember the details of their activities and decisions (Ošlejšek et al., 2020). However, this does not mean the feedback must always be immediate. For example, providing instant error corrections during an exercise can distract the learner (Hattie & Timperley, 2007).
- *Detailed on demand*: the feedback must provide an overview as well as detailed per-trainee information, which should be displayed on request so that it is not obtrusive. The presentation must be based on straightforward principles that are easy to understand, such as common visualization elements. As a result, learners can reduce extraneous cognitive load and examine their learning activity when they are ready and interested.
- *Personalized*: the feedback must be specific for each learner based on their actions in the training. As a result, learners can engage with feedback that is accurate and relevant to them, which increases their learning gains compared to a passive acceptance (Nicol & Macfarlane-Dick, 2006).

3.5 Technological Architecture and Implementation

Figure 4 illustrates the feedback system architecture. First, the trainee interacts with machines deployed in the ILE to complete the training tasks. While doing so, the command-line inputs and metadata are automatically recorded (Švábenský, Vykopal, Tovarňák, & Čeleda, 2021). Logs from all sandboxes (isolated environments with the training machines) of all trainees are stored in Elasticsearch (Elasticsearch, 2023) database.

Subsequently, the *feedback service* requests data from this database, processes them, and passes the generated feedback to the *training service*, which associates the feedback with the corresponding trainee. Finally, the trainee can display the interactive visualizations in the web portal.

The feedback generation for a trainee starts after the trainee finishes the last task. This is independent of other trainees in the training session. Since the computation is fast (less than three seconds from the trainee's perspective), there is no need to perform pre-computation during the training, even though it would be possible since the data are collected gradually throughout the training.

The whole system has minimal data storage requirements. Our previous dataset (Švábenský, Vykopal, Seda, & Čeleda, 2021) of 21459 log records from 275 trainees takes only 4.2 MB of space in total. This means that on average, each trainee produces only about 16 KB of log data per training session. The largest log file in our dataset has 66 KB. For these reasons, the feedback provision scales well, even for large groups of students. As the final remark, the feedback system backend is implemented in Java and the frontend in Angular.

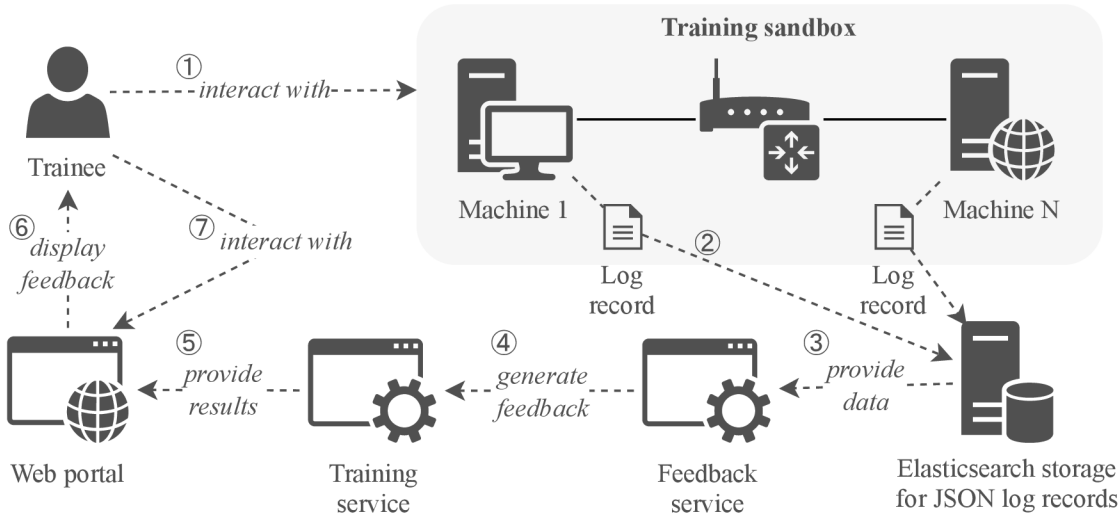

Fig. 4 An overview of the architecture of the automated feedback system. The architecture is generic, so it can be adapted to other ILEs and learning contexts outside cybersecurity.

3.6 Four Modules of the Feedback System

We prototyped and tested four methods implemented as distinct software modules for providing individual feedback (Demčák, 2021), which were then integrated into the ILE. The first two modules help the students answer the questions *Where am I going?* and *How am I going?*, while the remaining two modules focus on the latter question from a different perspective.

3.6.1 Reference Graphs

This method, which was first introduced in (Švábenský, Weiss, et al., 2022), provides an overview of the training structure. The reference graph is a static representation of a particular training, and it is the same for all trainees. It shows the step-by-step sequence of actions that a trainee should perform to successfully complete the training.

For a given training, the teacher first defines the *reference solution* in JSON format. After the reference solution is defined, the ILE uses it to automatically generate the *reference graph* that visualizes the expected solution.

Reference graph definition is performed manually based on the teacher's expert knowledge. Nevertheless, this step is not time-consuming if the teacher is familiar with the training content. If novel solutions are discovered later, the reference solution can always be updated to cover alternative approaches to the training tasks.

Figure 5 shows a small section of the reference graph for a particular training. (For the purposes of this paper, the graph formatting is more compact.) The graph may include three types of nodes:

- *Mandatory*: states that must be reached to solve the task.
- *Optional*: states that are not necessary to reach but are not erroneous, such as displaying the help for a tool.

- *Branching*: states used to model alternative solutions, i.e., different paths that can lead to a successful solution. Taking one of the possible paths is enough to consider the solution successful.

The directed edges represent the commands that must be executed to progress from one state to the next. Each edge is annotated with the required command and arguments.

For example, if a trainee then executes the following sequence of commands, that would be considered a correct solution with the minimal possible number of steps:

```
msfconsole
use exploit/linux/backdoor
set LHOST 10.2.154.7
set RHOST 172.1.18.7
set RPORT 4200
exploit
```

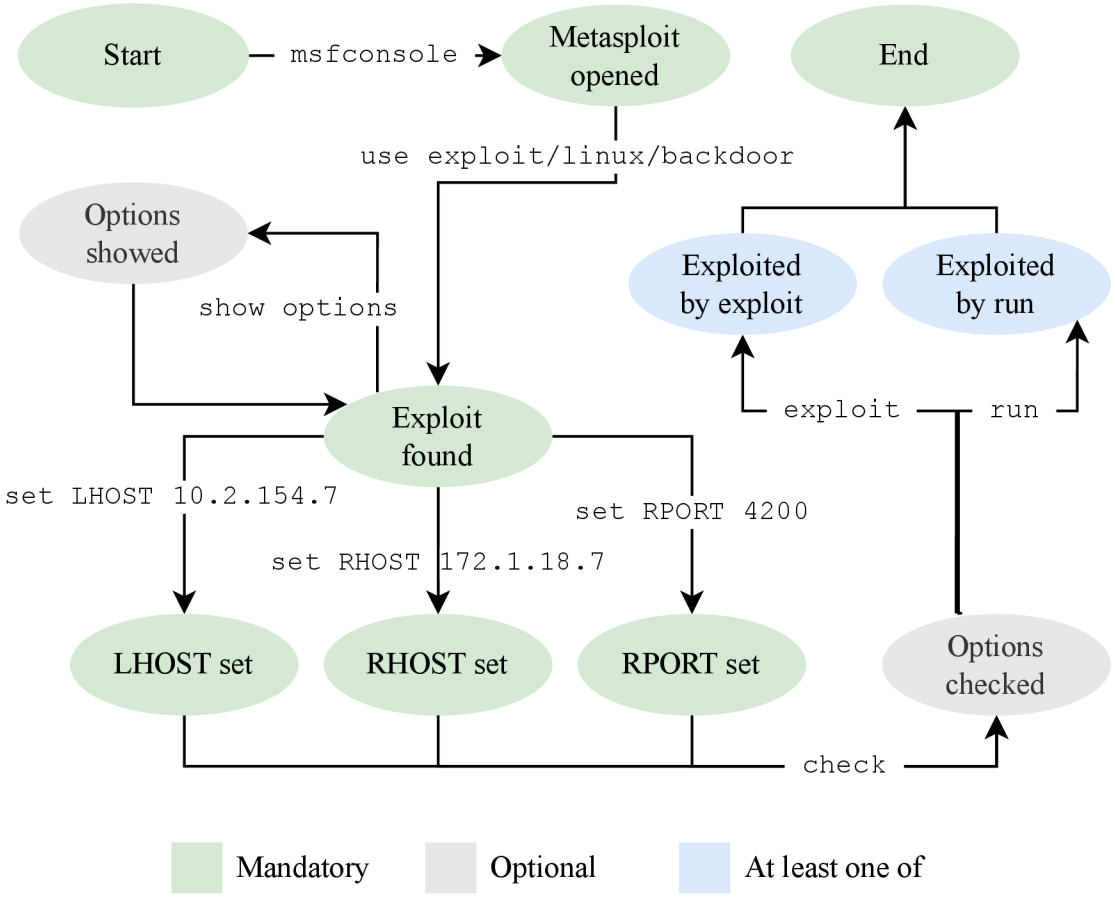

Fig. 5 This reference graph displays a training phase focused on using the Metasploit tool. The graph demonstrates modeling actions that can be completed in any order, optional actions, and branching.

The reference graph can be shown to trainees as post-training feedback to visualize the correct solution. Moreover, it enables the automated generation of *trainee graphs*.

3.6.2 Trainee Graph

This graph displays individual trainee's progress and deviations from the reference graph (Švábenský, Weiss, et al., 2022). For example, the trainee whose graph is shown in Figure 6 executed the following sequence of commands:

```
nmap 172.1.18.7
use exploit/linux/backdoor
set LHOST 10.2.154.7
set RHOST 172.1.18.7
set RPORT 4200
check
exploit
```

As the name indicates, each trainee receives a personalized trainee graph. It is created from the learner's submitted commands, which are automatically mapped to the reference graph. Each command is processed sequentially. The algorithm attempts to match it with a regular expression representing the correct solution defined by the instructor. The learner can see only their own graph, while the teacher can display the graph of any learner.

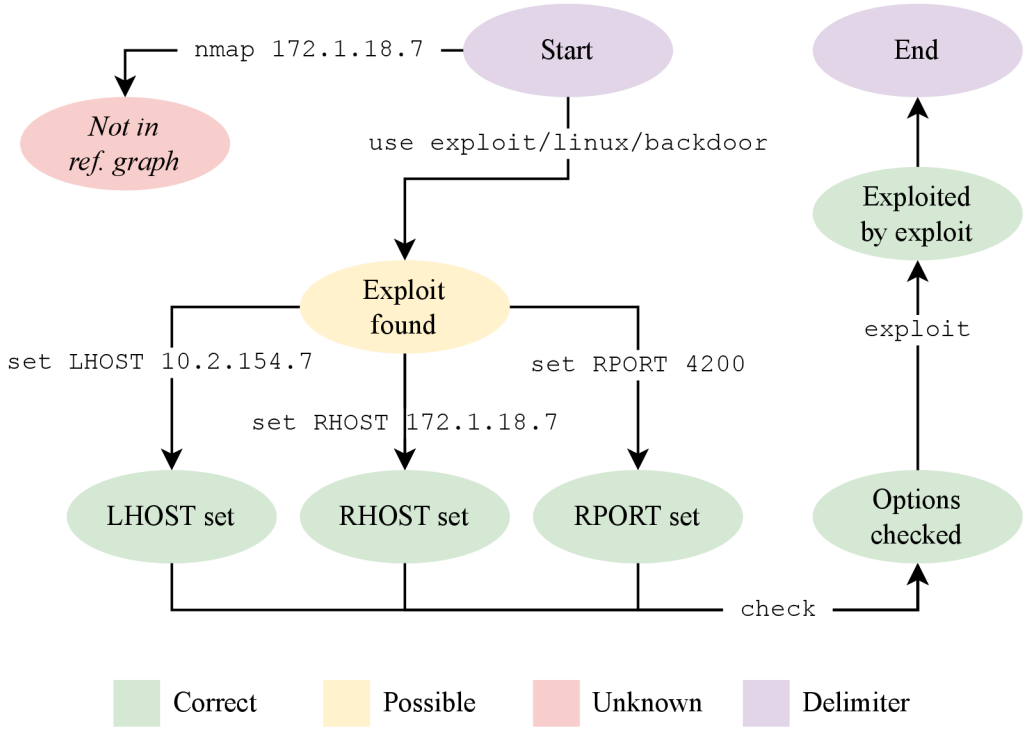

Fig. 6 A simplified trainee graph belonging to a learner in a training.

The graph may include four types of nodes:

- *Correct*: actions successfully mapped to the reference graph (conforming to the expected solution).
- *Possible*: actions with possibly missing prerequisites.
- *Unknown*: actions that were likely erroneous or unnecessary.
- *Delimiter* nodes to indicate the start/end of each "level" (training phase).

Trainee graphs serve many purposes:

- They show which tasks the learners achieved, which were problematic, and what approaches were used.
- They allow comparing a learner's expected and observed activities and solution paths.
- They enable distinguishing between the effective and ineffective pathways.
- They may reveal unexpected and potentially novel ways to solve a task, if they show a pathway through the exercise that the instructor did not expect.

Both reference and trainee graphs feature a unified format, and the semantics of nodes and edges are the same. To ease the interpretation, the color-coded legend is always visible, and subgraphs corresponding to individual levels are separated. Since the graphs can be extensive in longer training sessions, it is possible to interactively zoom in and view their parts.

3.6.3 Command Timeline

Another module of the feedback system is an interactive timeline of submitted commands, as shown in Figure 7. Using the timeline, the learner can see a linear overview of their commands. At the same time, the teacher can display such a timeline for any learner in the training session. The module is interactive, showing on-demand details of each command by clicking on it. These details include the command type, its arguments, and the IP address of the machine on which it was executed.

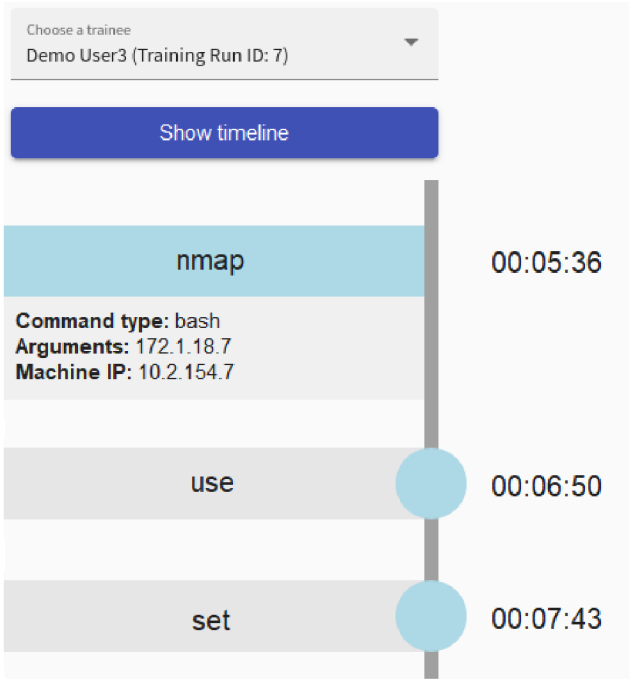

Fig. 7 Command timeline of a "Demo User 3", (adapted from (Masaryk University, 2023b)), showing the first three commands of the same trainee as in Figure 6.

The timeline is a simple yet effective view of the learner's activity ordered chronologically. Although a trainee graph also captures chronological order by associating the nodes left-to-right and top-to-bottom, the order of commands may not be immediately apparent. The command timeline complements the graph perspective, allowing the learner to focus on a specific stage of the training. For example, the learner can identify a sequence of commands corresponding to a problematic task. They can also identify and review difficult training phases based on long gaps between successive commands.

3.6.4 Command Analysis

The final module provides the error analysis of commands belonging to a selected learner. The analysis results (see Figure 8) include all incorrect commands, which are sorted descending based on the tool frequency and can be filtered by error type.

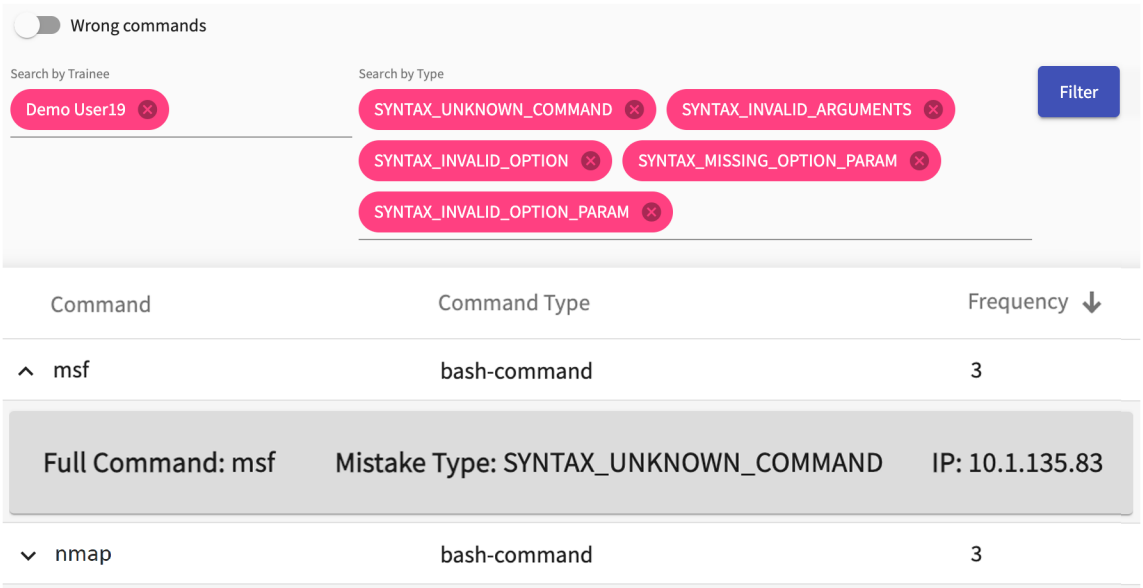

Fig. 8 Snippet from a command analysis of a learner (adapted from (Masaryk University, 2023b)).

Currently, seven error types are supported. The module automatically distinguishes five types of syntactic errors: *unknown command*, *incorrect argument*, *incorrect option*, *missing mandatory parameter*, and *incorrect parameter*. In addition, two semantic errors are recognized: *unknown IP address* and *incorrect IP address*. The error analysis is based on comparing the submitted commands with pre-defined JSON files, which represent the commands' syntax and semantics in the training. The syntactical analysis is executed first. If the command is syntactically correct, it is then analyzed semantically.

Consistently with the other feedback modules, the learner can see only their own activity, while the teacher can display the command analysis of any learner in the class. Clicking on an entry in the table shows more details about the command: its arguments, IP address of execution, and the error type if

that command is incorrect. This analysis is useful to reveal the most frequently or rarely used commands, typical errors, and deep insights such as whether certain commands are often associated with certain errors. The purpose is to provide a data-driven understanding of student mistakes, which can then be identified and remedied.

4 Methods for Feedback Evaluation

In order to evaluate the feedback provided by the proposed software, we seek to answer the following research questions:

- RQ1: How *understandable* is the feedback for students?
- RQ2: How *useful* is the feedback for students?
- RQ3: How can the feedback be *improved*?

We conducted five field studies in authentic teaching contexts. Students completed the *Locust 3302* cybersecurity training (see Section 3.2) in KYPO CRP (see Section 3.1), interacted with all four modules of their auto-generated feedback, and answered a questionnaire evaluating the feedback.

We followed the recommendations from an extensive literature review (Keuning et al., 2018), stating that for achieving the highest accuracy, educational tools should be evaluated on their own, not as a part of a course evaluation or other research. Otherwise, it is "difficult to isolate and measure the effect of the tool itself". In addition, it is essential to assess the effectiveness of different types of feedback and their combinations (Keuning et al., 2018, Section 10.4), which we focus on in this paper.

4.1 Questionnaire Design

Several previous studies evaluated feedback effectiveness using a questionnaire (Bruzual et al., 2020; Marwan et al., 2020). Our questionnaire asked a set of questions with a uniform structure, which gradually targeted different aspects of the feedback system. The questions had the following form, where X had the values from the set {*reference graph*, *trainee graph*, *command timeline*, *command analysis*}:

- X is *understandable*. (rating on a Likert scale)
- X is *useful*. (rating on a Likert scale)
- What do you like/dislike about X? (open question)

The Likert scale offered the following options: 1 – strongly disagree, 2 – disagree, 3 – neutral, 4 – agree, 5 – strongly agree. We also included open-ended questions so that participants could explain their rating choices or provide comments.

In addition, each feedback module was targeted by one or two extra questions specific to that module's function, which were answered on the same Likert scale. For example: *I was able to easily understand the sample solution from the reference graph.* These questions were added after the first field study.

4.2 Participants of the Field Studies

Table 1 lists all five field studies we conducted. The sample of 58 distinct participants consisted of undergraduate and graduate students of computer science. All of them attended the training session voluntarily, and the decision to participate or not had no influence on their course grades. 36 students of Masaryk University, Czech Republic (MU) responded to an open invitation because of their interest in cybersecurity. 22 students of Hague University of Applied Sciences, Netherlands (HUAS) attended the training session as an optional part of their university class on Linux and security.

Table 1 Information about the field studies and the participants. MU = Masaryk University, Czech Republic. HUAS = Hague University of Applied Sciences, Netherlands.

Training date	Training modality	Participants' institution	Survey responses / num. participants
Apr 22, 2021	remote	MU	9 / 10
Feb 18, 2022	in-person	MU	13 / 13
Feb 24, 2022	hybrid	MU	13 / 13
Apr 25, 2022	in-person	HUAS	10 / 10
Apr 28, 2022	in-person	HUAS	0* / 12
			Total: 45 / 58

**The last session did not yield any data because of a technical error.*

The training interface, tasks, and questionnaire were worded in English. Communication with the instructor(s) took place in the students' respective native language (Czech, Slovak, or Dutch). We did not systematically collect any demographic information about the students in order to save time and ease the burden on the study participants, but all students were Czech, Slovak, or Dutch nationals, and all of them were adults at least 18 years of age or older.

The students were not incentivized in any way. The vast majority did not attend any cybersecurity training in KYPO CRP before, and none of the students completed the training *Locust 3302* before. The fastest student finished the training in 30 minutes, the slowest in three hours. The average completion time was less than two hours. An average student submitted 71 commands throughout the training.

In the case of MU students, self-selection bias (Heckman, 2010) may be present. However, we mitigated it by including the students of HUAS in the study. This multi-national, multi-institutional framework addresses the limitations of many research papers in computing education (Guzdial & du Boulay, 2019), which often focus on a single-institutional study. Moreover, we employed different training modalities (in-person, virtual, and hybrid) to cover various use cases. Overall, the study sample represents a broad range of computing students with different backgrounds.

4.3 Field Study Progress

All five field studies had a uniform structure. Each student completed the training individually at their own pace. After a student finished the training, they interacted with the feedback for as long as they wished. Students spent approximately five minutes on this step. Since this interaction was brief, there was no need to cut down on the training time.

Subsequently, the student started filling in the questionnaire. The feedback remained available in another browser tab, so the students could (and many of them did) return to the feedback while completing the questionnaire.

The students could ask the instructor(s) any questions. The whole process was independent of other students, so once a student completed all the steps, they were free to leave.

4.4 Privacy and Ethical Measures

Before collecting any data, we discussed the research with the institutional review board of MU and the instructors from HUAS. We obtained a waiver from the ethical committee of MU since we intentionally do not collect any personally identifiable information that could reveal the learners' identity. The data are anonymous and cannot be linked to specific individuals.

The learners agreed to complete the anonymous questionnaire for research via informed consent before the training. We minimized the extent of data collection to gather only the data necessary for the research. Most importantly, we ensured the learners would not be harmed by the research. They all had the right to stop participating at any time without any restrictions (although no one exercised this right).

4.5 Methods for the Data Analysis

Out of the 58 participants, we received questionnaire responses only from 45. Twelve participants could not answer due to an unexpected technical issue in the last training session, which has been fixed now. One participant did not answer because of time constraints.

To analyze the Likert scale ratings (RQ1 and RQ2), we computed descriptive and inferential statistics in R (R Core Team, 2023). Even though Likert scale data are ordinal by definition, we treated them as interval data (Carifio & Perla, 2008; Norman, 2010; Wu & Leung, 2017), which is applicable. This was also demonstrated in related research on automated feedback (Arends, Keuning, Heeren, & Jeuring, 2017; Bruzual et al., 2020). To analyze the open comments (RQ3), we performed qualitative coding (Saldaña, 2021) of students' answers.

5 Results and Discussion

We now answer the three research questions, discuss the results, and propose ideas for future research.

5.1 Quantitative Analysis (RQ1 and RQ2)

The first two research questions examine the understandability and usefulness of the feedback. Figure 9 displays bar charts grouped in four quadrants. Each quadrant corresponds to one of the four feedback formats (reference graph, trainee graph, command timeline, and command analysis). The individual charts present descriptive statistics of the questionnaire answers. Then, the following sections of this text discuss the feedback further.

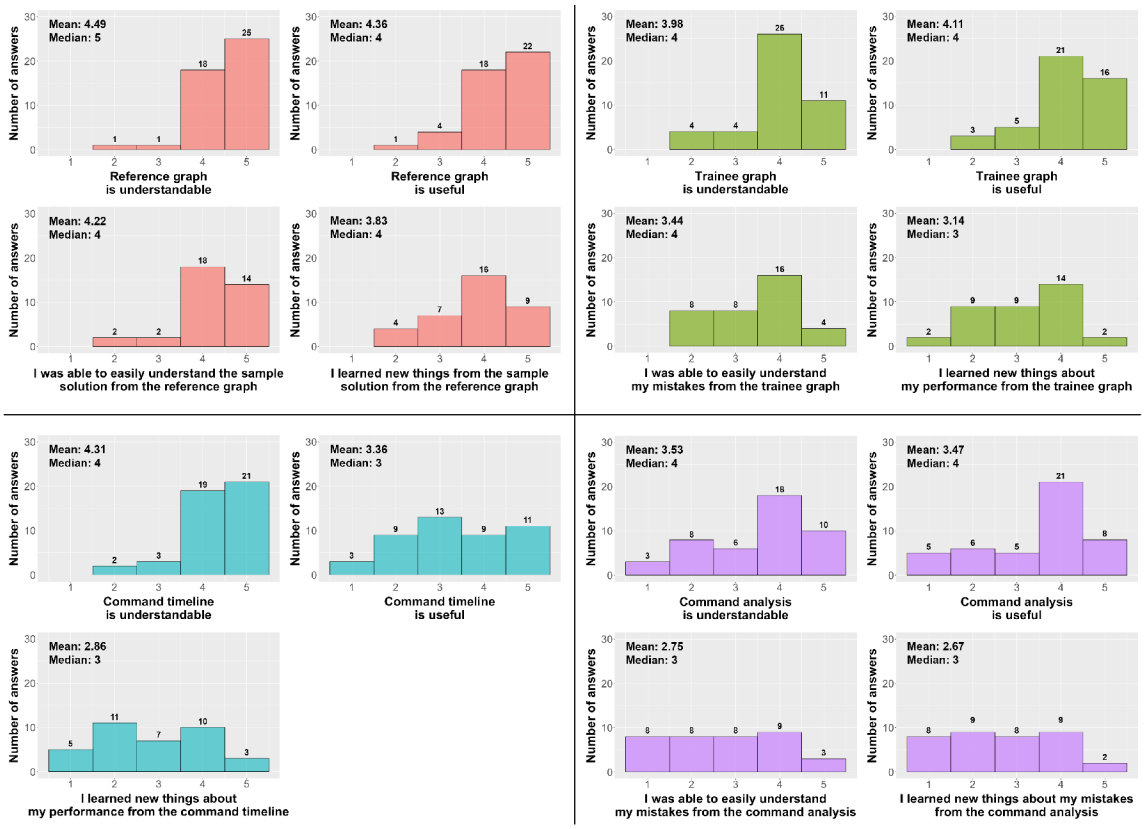

Fig. 9 Questions and the number of responses regarding the feedback modules. The number of responses is $n = 45$ for the questions in the 1st and 3rd row, and $n = 36$ for the remaining questions, which were added after the first field study. The horizontal axis represents the Likert scale defined in Section 4.1.

5.1.1 Understandability and Usefulness of the Individual Feedback Formats

Overall, the trainees expressed the largest agreement regarding the understandability and usefulness of the *reference graph*. The *trainee graph* scored reasonably well, also. As we expected, the *command timeline* was understandable. However, the trainees did not consider it much useful. The *command analysis* received the most varied and conflicting responses, which are explained in Section 5.2.

5.1.2 Correlation of Understandability and Usefulness

We statistically tested the hypothesis that *for each feedback format, there is a correlation between understandability and usefulness*. We assumed that these characteristics would correlate positively. Should that not be the case, it may indicate a problem with the feedback design.

We used the sample of $n = 45$ data points corresponding to the study participants. The significance level was set to $\alpha = 0.05$, which is the typical cutoff (Miller & Ulrich, 2019). The Pearson correlations for the four feedback formats were moderate: .56, .48, .32, and .58, respectively. In all cases, the correlations were statistically significant, confirming our hypothesis. The p-values were well below the α level, ranging from $3 \cdot 10^{-2}$ to $3 \cdot 10^{-5}$.

5.1.3 Mutual Difference of Understandability

Based on Figure 9, the average values of understandability of the four feedback formats were 4.49, 3.98, 4.31, and 3.53 out of 5. We examined whether these differences are statistically significant. Therefore, we tested the hypothesis that *all feedback formats have mutually different understandability*.

To compare the understandability across the four feedback formats, we used Welch's one-way analysis of variance (ANOVA) test. Again, $n = 45$ and $\alpha = 0.05$. In this case, the independent variable (factor) was one of the four feedback formats, while the dependent variable was the numerical measurement of its understandability.

ANOVA works well even for data that are not normally distributed (Schmider, Ziegler, Danay, Beyer, & Buhner, 2010) (which happens in practical use cases since educational phenomena are often distributed non-normally). Although there are non-parametric tests for non-normally distributed data, parametric tests are generally stronger, so we decided to use ANOVA. We checked two assumptions:

- The individual measured values are independent – this is true since each student completed the questionnaire only once and independently of other students.
- The variance of measured values across all groups is the same (homoscedasticity) – this assumption is false, since the Levene test returned $p = 0.03$. Therefore, instead of the standard ANOVA, we performed Welch's ANOVA, which accounts for the unequal variance.

Welch's ANOVA test yielded $p = 4 \cdot 10^{-5}$, supporting the hypothesis that the understandability of individual feedback types differs significantly.

5.1.4 Mutual Difference of Usefulness

We tested an analogous hypothesis to the previous case, stating that *all feedback formats have mutually different usefulness*. We followed the same procedure. Since the Levene test returned $p = 0.02$, we used Welch's ANOVA again. The test supported the hypothesis, yielding $p = 6 \cdot 10^{-6}$.

5.2 Qualitative Analysis (RQ3)

The third research question examines how the feedback can be improved. Two researchers performed open coding of the questionnaire responses. We summarize the findings for each of the four feedback modules below.

5.2.1 Reference Graph

The trainees appreciated that the reference graph was simple and illustrated the intended correct solution(s) well. Since the reference solution can differ from their own, they were able to learn alternative approaches to the task. Overall, they suggested the least amount of improvements, which were minor:

- Displaying the full command in a verbatim form next to an edge like in Figure 5, instead of a more complex annotation listing the command type, tool, and arguments, as is currently implemented in KYPO CRP.
- Adding the ability to view the graph per individual training phases ("levels"), as well as a full-screen view.

5.2.2 Trainee Graph

Apart from the previous points, which apply to the trainee graph as well, the trainees raised the following concerns:

- Some actions that contributed to the correct solution (or at least not impeded it) were classified as unknown. Fixing this requires updating the reference graph, which is currently performed manually.
- If several different argument combinations for a command are attempted, the edge description becomes too long. A solution would be to show it in rows or on-demand.
- Trainee graph could be displayed side-by-side with the reference graph or even trainee graphs of other students to enable better comparison.
- It would be beneficial to discuss the trainee graph with the teacher so that the student can better understand how to fix their mistakes.

5.2.3 Command Timeline

As the quantitative analysis indicated, the timeline was not much useful for the trainees. They noted that the time information can be valuable for identifying the exact command succession, time gaps, and frequency of command usage. We realized this information might be more useful for instructors and researchers than for trainees. In addition, they suggested several improvements:

- Closer interconnection with error analysis to distinguish incorrect commands.
- Hiding or grouping information about repeated and similar commands, mistakes, and especially typos.
- Displaying the timeline in sections split according to the training phases ("levels").
- Displaying the commands in a single column as opposed to two, and including a tooltip that each entry is clickable to reveal further information.

5.2.4 Command Analysis

This module yielded the most negative responses.

- Trainees were displeased with false positives and negatives in the error analysis. Like in the reference graph, fixing this would require updating the definition files that enable error recognition.
- Suggestions on how to fix the errors could be added to support student learning.
- Error categories could be described in more detail since the short descriptions were not always understandable.
- The graphical interface was not user-friendly. By default, no error category was displayed, and individual categories had to be selected one by one. Some categories included no errors but still could be selected; instead, they could have been hidden or disabled.

5.3 Lessons Learned

Students found the feedback valuable overall, even though the presented proof of concept can still be significantly improved. A slight disadvantage is that the reference and trainee graph modules require the instructor to define the expected solution, and the error analysis requires the syntactic and semantic definition files. However, these extra steps must be performed only once for particular training and can be reused in future training sessions. An alternative solution would be to display only those types of feedback that the instructor deems necessary. Currently, the feedback software displays all four types of feedback by default.

Automated feedback systems in general, including the one presented in this paper, have several other limitations. First, it is difficult to automatically assess all factors relevant to the exercise, such as the teaching context, personal learning goals, or strengths/weaknesses of specific students. As a result, the generated automated feedback may not be fully relevant to every individual student. Second, it is difficult to automatically evaluate soft skills, such as critical thinking and time management, which are also important for success in cybersecurity exercises.

5.4 Discussion of the Feedback System

Even though there is room for improvement, the feedback system as a whole addresses the nine key aspects (typeset in *italics*) that need to be considered when designing a learning analytics reporting system (Bodily & Verbert, 2017). The *system goal* is to provide post-training feedback to learners and teachers participating in hands-on cybersecurity training. The feedback is automated, timely, detailed on demand, and personalized. This *addresses the needs* of learners to understand the learning content, as well as of teachers to understand how their students learn.

To generate the feedback, we *collect data* consisting of training actions (commands) from the training environment, which are then analyzed to identify

educational insights. The *feedback is presented* using a combination of visual and textual techniques within the training platform. The *reason for using this representation* is that it effectively displays the training data, combining well-known elements (e.g., a timeline) with novel approaches (e.g., a trainee graph).

Students and instructors *use the system* immediately after the training ends. This way, they are not distracted during the training yet still remember the details of their activities and decisions. In the evaluation, the learners reported that they *perceived the feedback* from the system as understandable and mostly useful. In addition, they *consider the system* to be generally usable but warranting some improvements (see Section 5.2). The *effect on student behavior or achievement* is to be determined in the upcoming evaluations.

5.5 Open Research Challenges

In terms of feedback generation, an interesting research challenge is automatically updating the reference graph based on new solutions from students. Analogously, automating the updates of definition files for error analysis would decrease the occurrence of false positive and false negative discoveries. Lastly, measuring factors relevant to the teaching and learning context and incorporating them in the feedback would increase its pedagogical value and relevance to individual students.

In terms of feedback evaluation, future work can experimentally compare the learning of two groups of students: those who receive the feedback and those who do not. A randomized trial setup can measure whether the feedback impacted future learning. Both groups would receive the same knowledge test relevant to the learning outcomes of the training, and the performance of the two groups on the test would be compared. In addition, the data analytics in the feedback modules can be evaluated with instructors to determine whether the software effectively supports classroom situational awareness.

Another challenge is extending the system to provide feedback during the training, not only after. Then, it would be useful to compare these two modalities to determine what type of feedback is the most effective at which stage. This mirrors the question *How and when is it best to give students feedback to improve learning?* It was identified as the fifth most pressing issue in a survey of computing educators about which problems they want computing education researchers to investigate (Denny, Becker, Craig, Wilson, & Banaszkiewicz, 2019). More challenges regarding automated feedback (focusing on programming education) are discussed in (Malmi et al., 2019, Section 21.5.4).

6 Conclusion

Timely and personalized feedback to students is crucial to support their learning. It helps them understand the learning content, confirms what they did well, and helps correct their mistakes. Therefore, we designed, implemented, and evaluated four modules for providing feedback on complex cybersecurity exercises.

The quantitative analysis confirmed our intuition that the feedback modules are generally understandable and potentially useful. The qualitative interpretation of data revealed valuable suggestions for improvement and extension of the feedback. The key takeaway is that the proposed feedback methods and their principles apply to various ILEs and learning contexts. Other instructors can adopt or adapt our results to enhance the learning technologies they use.

6.1 Implications for Practice

From the research perspective, we conducted a multi-national, multi-institutional study that examined the feedback from the perspective of various students. The conditions and modalities of training varied to accommodate different use cases. The data were analyzed using mixed methods. This addresses the gap in almost 28% of current feedback systems that provided only anecdotal evidence of feedback effectiveness or even no evidence at all (Keuning et al., 2018). The graph modules were shown to have the largest impact, while the command timeline and analysis modules were not substantially effective.

Developers of learning technologies may implement a similar system and build upon our work, since all source code of the training platform and feedback modules is freely available under a permissive license. The insights described in this paper are not limited to cybersecurity education, since the demonstrated principles of feedback can be applied to other domains.

Regarding the practical implications on teaching, cybersecurity instructors may deploy KYPO CRP and use the modules in their classes. Receiving automated feedback is especially relevant in remote education when instructors have limited awareness of what students do on their computers (Švábenský, Vykopal, Tovarňák, & Čeleda, 2021). This proved invaluable in our online teaching practice during the COVID-19 pandemic in 2020–2021.

6.2 Publicly Available Supplementary Materials

In the spirit of open science, materials associated with this paper are available under open-source licenses. This represents a valuable contribution since the open sharing of supplementary materials is not a standard practice in cybersecurity education research (Švábenský, Vykopal, & Čeleda, 2020). The materials include:

- The interactive learning environment KYPO CRP (Vykopal et al., 2021), including the feedback software (Masaryk University, 2023c, 2023d).
- The training *Locust 3302* and others (Masaryk University, 2021).
- The modular logging toolset (Švábenský, Vykopal, Tovarňák, & Čeleda, 2021).
- The questionnaire, collected data, analysis scripts, and qualitative coding table developed for this paper (Švábenský, Vykopal, Čeleda, & Dovjak, 2023).

Statements and Declarations

Funding. Funded by the European Union under Grant Agreement No. 101087529. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Acknowledgments. Parts of this publication build upon the first author's Ph.D. thesis. We thank Ján Demčák and the development team of KYPO CRP for their excellent implementation work. Next, we thank the students and instructors who participated in the field studies. We also thank Lydia Kraus for her help with qualitative coding of the questionnaire responses. Finally, we thank Radek Pelánek for thoughtful comments on early versions of this paper.

Competing interests. The authors have no competing interests to declare.

Availability of data and materials. The accompanying data and materials are published in a permanent, free, open-source repository on Zenodo (Švábenský et al., 2023).

Authors' contributions. *Valdemar Švábenský*: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Visualization, Project Administration. *Jan Vykopal*: Investigation, Resources, Writing - Review and Editing, Supervision. *Pavel Čeleda*: Writing - Review and Editing, Supervision, Funding acquisition. *Ján Dovjak*: Methodology, Software, Formal analysis, Investigation, Resources, Writing - Review and Editing, Visualization

References

Agudo, I., Rios, R., Nieto, A. (2019). Personalized Computer Security Tasks with Automatic Evaluation and Feedback. *Proceedings of the 2019 SIGED International Conference on Information Systems Education and Research.* Association for Information Systems (AIS). Retrieved from https://www.nics.uma.es/pub/papers/1835.pdf

Arends, H., Keuning, H., Heeren, B., Jeuring, J. (2017). An Intelligent Tutor to Learn the Evaluation of Microcontroller I/O Programming Expressions. *Proceedings of the 17th Koli Calling International Conference on Computing Education Research* (p. 2–9). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3141880.3141884

Baniassad, E., Zamprogno, L., Hall, B., Holmes, R. (2021). STOP THE (AUTOGRADER) INSANITY: Regression Penalties to Deter Autograder Overreliance. *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (p. 1062–1068). New York, NY, USA:

Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3408877.3432430

Bezáková, I., Hemaspaandra, E., Lieberman, A., Miller, H., Narváez, D.E. (2020). Prototype of an Automated Feedback Tool for Intro CS Theory. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (p. 1311). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3328778.3372598

Bodily, R., & Verbert, K. (2017). Review of Research on Student-Facing Learning Analytics Dashboards and Educational Recommender Systems. *IEEE Transactions on Learning Technologies*, *10*(4), 405-418. Retrieved from https://doi.org/10.1109/TLT.2017.2740172

Bruzual, D., Montoya Freire, M.L., Di Francesco, M. (2020). Automated Assessment of Android Exercises with Cloud-Native Technologies. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (p. 40–46). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3341525.3387430

Buffardi, K., & Edwards, S.H. (2015). Reconsidering Automated Feedback: A Test-Driven Approach. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (p. 416–420). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2676723.2677313

Carifio, J., & Perla, R. (2008). *Resolving the 50-year debate around using and misusing Likert scales* (Vol. 42) (No. 12). Retrieved from https://doi.org/10.1111/j.1365-2923.2008.03172.x

Cavalcanti, A.P., Barbosa, A., Carvalho, R., Freitas, F., Tsai, Y.-S., Gašević, D., Mello, R.F. (2021). Automatic feedback in online learning environments: A systematic literature review. *Computers and Education: Artificial Intelligence*, *2*, 100027. Retrieved from https://doi.org/10.1016/j.caeai.2021.100027

Chen, H., Ciborowska, A., Damevski, K. (2019). Using Automated Prompts for Student Reflection on Computer Security Concepts. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (p. 506–512). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3304221.3319731

Chothia, T., Holdcroft, S., Radu, A.-I., Thomas, R.J. (2017). Jail, Hero or Drug Lord? Turning a Cyber Security Course Into an 11 Week Choose

Your Own Adventure Story. *2017 USENIX Workshop on Advances in Security Education (ASE 17)* (pp. 1–11). Berkeley, CA, USA: USENIX Association. Retrieved from https://www.usenix.org/conference/ase17/workshop-program/presentation/chothia

Chow, S., Yacef, K., Koprinska, I., Curran, J. (2017). Automated Data-Driven Hints for Computer Programming Students. *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization* (p. 5–10). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3099023.3099065

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

Codio (2019). *Report: 2019 Survey of CS Teaching.* Online, accessed October 3, 2023. Retrieved from https://www.codio.com/research/2019-computer-science-teaching-survey

Cordova, L., Carver, J., Gershmel, N., Walia, G. (2021). A Comparison of Inquiry-Based Conceptual Feedback vs. Traditional Detailed Feedback Mechanisms in Software Testing Education: An Empirical Investigation. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (p. 87–93). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3408877.3432417

DelftStack (2023). *Get Command History in MySQL.* Online, accessed October 3, 2023. Retrieved from https://www.delftstack.com/howto/mysql/mysql-command-history/

Demčák, J. (2021). *Web Application for Automated Feedback to Participants of Cybersecurity Training* (Master thesis, Masaryk University, Faculty of Informatics). Retrieved from https://is.muni.cz/th/a4pjx/?lang=en

Denny, P., Becker, B.A., Craig, M., Wilson, G., Banaszkiewicz, P. (2019). Research This! Questions That Computing Educators Most Want Computing Education Researchers to Answer. *Proceedings of the 2019 ACM Conference on International Computing Education Research* (p. 259–267). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3291279.3339402

Dietvorst, B.J., Simmons, J.P., Massey, C. (2015). Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, *144*(1), 114. Retrieved from

https://doi.org/10.1037/xge0000033

Doupé, A., Egele, M., Caillat, B., Stringhini, G., Yakin, G., Zand, A., ... Vigna, G. (2011). Hit 'Em Where It Hurts: A Live Security Exercise on Cyber Situational Awareness. *Proceedings of the 27th Annual Computer Security Applications Conference* (pp. 51–61). New York, NY, USA: ACM. Retrieved from https://doi.org/10.1145/2076732.2076740

Edwards, S.H., & Pérez Quiñones, M.A. (2008). Web-CAT: Automatically Grading Programming Assignments. *SIGCSE Bull.*, *40*(3), 328. Retrieved from https://doi.org/10.1145/1597849.1384371

Elasticsearch (2023). *What is the ELK Stack?* Online, accessed October 3, 2023. Retrieved from https://www.elastic.co/what-is/elk-stack

Elkherj, M., & Freund, Y. (2014). A System for Sending the Right Hint at the Right Time. *Proceedings of the First ACM Conference on Learning @ Scale* (p. 219–220). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2556325.2567864

Fraser, C.A., Ngoon, T.J., Weingarten, A.S., Dontcheva, M., Klemmer, S. (2017). CritiqueKit: A Mixed-Initiative, Real-Time Interface For Improving Feedback. *Adjunct Publication of the 30th Annual ACM Symposium on User Interface Software and Technology* (pp. 7–9). Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3131785.3131791

Ghafarian, A. (2019). Using Kali Linux Security Tools to Create Laboratory Projects for Cybersecurity Education. *Proceedings of the Future Technologies Conference (FTC) 2018* (pp. 358–367). Cham: Springer International Publishing. Retrieved from https://doi.org/10.1007/978-3-030-02683-7_25

Guzdial, M., & du Boulay, B. (2019). The History of Computing Education Research. S.A. Fincher & A.V. Robins (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 11–39). Cambridge, United Kingdom: Cambridge University Press. Retrieved from https://doi.org/10.1017/9781108654555

Haldeman, G., Tjang, A., Babeş-Vroman, M., Bartos, S., Shah, J., Yucht, D., Nguyen, T.D. (2018). Providing Meaningful Feedback for Autograding of Programming Assignments. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (p. 278–283). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3159450.3159502

Hall, J.G., Mohanty, A., Murarisetty, P., Nguyen, N.D., Bahamón, J.C., Ramaprasad, H., Sridhar, M. (2022). Criminal Investigations: An Interactive Experience to Improve Student Engagement and Achievement in Cybersecurity Courses. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1* (p. 696–702). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3478431.3499417

Hao, J., Shu, Z., von Davier, A. (2015, Jan.). Analyzing Process Data from Game/Scenario-Based Tasks: An Edit Distance Approach. *Journal of Educational Data Mining*, *7*(1), 33–50. Retrieved from https://jedm.educationaldatamining.org/index.php/JEDM/article/view/JEDM072

Harden, J., Gusukuma, L., Bart, A.C., Kafura, D. (2021). A Specification Language for Matching Mistake Patterns with Feedback. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (p. 555–561). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3408877.3432481

Hattie, J., & Timperley, H. (2007). The Power of Feedback. *Review of Educational Research*, *77*(1), 81-112. Retrieved from https://doi.org/10.3102/003465430298487

Heckman, J.J. (2010). Selection Bias and Self-Selection. In S.N. Durlauf & L.E. Blume (Eds.), *Microeconometrics* (pp. 242–266). London: Palgrave Macmillan UK. Retrieved from https://doi.org/10.1057/9780230280816_29

Joint Task Force on Cybersecurity Education (2017). *Cybersecurity Curricular Guideline.* Online, accessed October 3, 2023. Retrieved from https://cybered.acm.org

Keuning, H., Jeuring, J., Heeren, B. (2018, September). A Systematic Literature Review of Automated Feedback Generation for Programming Exercises. *ACM Trans. Comput. Educ.*, *19*(1), 3:1–3:43. Retrieved from https://doi.org/10.1145/3231711

Kölling, M., & McKay, F. (2016). Heuristic Evaluation for Novice Programming Systems. *ACM Trans. Comput. Educ.*, *16*(3). Retrieved from https://doi.org/10.1145/2872521

Kont, M., Pihelgas, M., Maennel, K., Blumbergs, B., Lepik, T. (2017). Frankenstack: Toward real-time Red Team feedback. *2017 IEEE Military*

Communications Conference, MILCOM 2017 (pp. 400–405). New York, NY, USA: IEEE. Retrieved from https://doi.org/10.1109/MILCOM.2017.8170852

Kovanović, V., Joksimović, S., Gašević, D., Hatala, M., Siemens, G. (2017). Content Analytics: The Definition, Scope, and an Overview of Published Research. C. Lang, G. Siemens, A. Wise, & D. Gašević (Eds.), *Handbook of Learning Analytics* (pp. 77–92). Society for Learning Analytics Research (SoLAR). Retrieved from https://doi.org/10.18608/hla17

Kulkarni, C.E., Bernstein, M.S., Klemmer, S.R. (2015). PeerStudio: Rapid Peer Feedback Emphasizes Revision and Improves Performance. *Proceedings of the Second (2015) ACM Conference on Learning @ Scale* (pp. 75–84). Retrieved from https://doi.org/10.1145/2724660.2724670

Lepri, B., Oliver, N., Letouzé, E., Pentland, A., Vinck, P. (2018). Fair, Transparent, and Accountable Algorithmic Decision-making Processes. *Philosophy & Technology*, *31*(4), 611–627. Retrieved from https://doi.org/10.1007/s13347-017-0279-x

Lyon, G. (n.d.). *Nmap Reference Guide.* Online, accessed October 3, 2023. Retrieved from https://nmap.org/book/man-briefoptions.html

Macfadyen, L.P., & Dawson, S. (2010). Mining LMS data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*, *54*(2), 588-599. Retrieved from https://doi.org/10.1016/j.compedu.2009.09.008

Malik, A., Wu, M., Vasavada, V., Song, J., Coots, M., Mitchell, J., ... Piech, C. (2021). Generative Grading: Near Human-level Accuracy for Automated Feedback on Richly Structured Problems. *Proceedings of the 14th International Conference on Educational Data Mining.* International Educational Data Mining Society. Retrieved from https://educationaldatamining.org/EDM2021/virtual/static/pdf/EDM21_paper_94.pdf

Malmi, L., Utting, I., Ko, A.J. (2019). Tools and Environments. S.A. Fincher & A.V. Robins (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 639–662). Cambridge, United Kingdom: Cambridge University Press. Retrieved from https://doi.org/10.1017/9781108654555.022

Mandiant (2013). *APT1: Exposing One of China's Cyber Espionage Units* (Tech. Rep.). FireEye, Inc. Retrieved from https://www.mandiant.com/resources/apt1-exposing-one-of-chinas-cyber-espionage-units

Marchiori, A. (2022). Labtool: A Command-Line Interface Lab Assistant and Assessment Tool. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1* (p. 1–7). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3478431.3499285

Marwan, S., Gao, G., Fisk, S., Price, T.W., Barnes, T. (2020). Adaptive Immediate Feedback Can Improve Novice Programming Engagement and Intention to Persist in Computer Science. *Proceedings of the 2020 ACM Conference on International Computing Education Research* (p. 194–203). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3372782.3406264

Masaryk University (2021). *The listing of all cybersecurity games and common instructions.* Online, accessed October 3, 2023. Retrieved from https://gitlab.ics.muni.cz/muni-kypo-trainings/games/all-games-index

Masaryk University (2023a). *KYPO Cyber Range Platform.* Online, accessed October 3, 2023. Retrieved from https://docs.crp.kypo.muni.cz

Masaryk University (2023b). *KYPO Cyber Range Platform – Visualizations for Linear Training.* Online, accessed October 3, 2023. Retrieved from https://docs.crp.kypo.muni.cz/user-guide-basic/training-agenda/visualizations/visualizations-for-linear

Masaryk University (2023c). *KYPO Training Feedback.* Online, accessed October 3, 2023. Retrieved from https://gitlab.ics.muni.cz/muni-kypo-crp/backend-java/kypo-training-feedback

Masaryk University (2023d). *Training command graph visualization.* Online, accessed October 3, 2023. Retrieved from https://gitlab.ics.muni.cz/muni-kypo-crp/frontend-angular/components/kypo-command-visualizations

Miller, J., & Ulrich, R. (2019, 01). The quest for an optimal alpha. *PLOS ONE*, *14*. Retrieved from https://doi.org/10.1371/journal.pone.0208631

Mouratidis, A., Lens, W., Vansteenkiste, M. (2010). How You Provide Corrective Feedback Makes a Difference: The Motivating Role of Communicating in an Autonomy-Supporting Way. *Journal of Sport and Exercise Psychology*, *32*(5), 619–637. Retrieved from https://doi.org/10.1123/jsep.32.5.619

Nicol, D.J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: a model and seven principles of good feedback practice. *Studies in Higher Education*, *31*(2), 199-218. Retrieved from https://

doi.org/10.1080/03075070600572090

Norman, G. (2010). Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education*, *15*(5), 625–632. Retrieved from https://doi.org/10.1007/s10459-010-9222-y

OConnor, T. (2022). HELO DarkSide: Breaking Free From Katas and Embracing the Adversarial Mindset in Cybersecurity Education. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1* (p. 710–716). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3478431.3499404

Offensive Security (2023). *Kali Linux.* Online, accessed October 3, 2023. Retrieved from https://www.kali.org

Offensive Security (OffSec Services Limited) (2023). *Metasploit Unleashed.* Online, accessed October 3, 2023. Retrieved from https://www.offensive-security.com/metasploit-unleashed

Openwall (n.d.). *John the Ripper password cracker.* Online, accessed October 3, 2023. Retrieved from https://www.openwall.com/john

Ošlejšek, R., Rusňák, V., Burská, K., Švábenský, V., Vykopal, J. (2019, 10). Visual Feedback for Players of Multi-Level Capture the Flag Games: Field Usability Study. *Proceedings of the 16th IEEE Symposium on Visualization for Cyber Security* (pp. 1–11). New York, NY, USA: IEEE. Retrieved from https://doi.org/10.1109/VizSec48167.2019.9161386

Ošlejšek, R., Rusňák, V., Burská, K., Švábenský, V., Vykopal, J., Čegan, J. (2020, February). Conceptual Model of Visual Analytics for Hands-on Cybersecurity Training. *IEEE Transactions on Visualization and Computer Graphics*, *27*(8). Retrieved from https://doi.org/10.1109/TVCG.2020.2977336

Pardo, A., Jovanovic, J., Dawson, S., Gašević, D., Mirriahi, N. (2019). Using learning analytics to scale the provision of personalised feedback. *British Journal of Educational Technology*, *50*(1), 128-138. Retrieved from https://doi.org/10.1111/bjet.12592

Petty, G. (2009). *Teaching Today: A Practical Guide.* Nelson Thornes. Retrieved from https://geoffpetty.com/geoffs-books/teaching-today

Peña-Ayala, A. (2014). Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications*, *41*(4, Part 1), 1432–1462. Retrieved from https://doi.org/10.1016/j.eswa.2013.08.042

Pihelgas, M., & Kont, M. (2021). Frankenstack: Real-time Cyberattack Detection and Feedback System for Technical Cyber Exercises. *2021 IEEE International Conference on Cyber Security and Resilience (CSR)* (p. 396-402). Retrieved from https://doi.org/10.1109/CSR51186.2021.9527923

Pokhrel, S., & Chhetri, R. (2021). A Literature Review on Impact of COVID-19 Pandemic on Teaching and Learning. *Higher Education for the Future*, *8*(1), 133-141. Retrieved from https://doi.org/10.1177/2347631120983481

Price, T., Zhi, R., Barnes, T. (2017). Evaluation of a Data-Driven Feedback Algorithm for Open-Ended Programming. *International Conference on Educational Data Mining.* Retrieved from https://educationaldatamining.org/EDM2017/proc_files/papers/paper_36.pdf

Price, T.W., Dong, Y., Lipovac, D. (2017). ISnap: Towards Intelligent Tutoring in Novice Programming Environments. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (p. 483–488). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3017680.3017762

Price, T.W., Dong, Y., Zhi, R., Paaßen, B., Lytle, N., Cateté, V., Barnes, T. (2019). A Comparison of the Quality of Data-Driven Programming Hint Generation Algorithms. *International Journal of Artificial Intelligence in Education*, *29*(3), 368–395. Retrieved from https://doi.org/10.1007/s40593-019-00177-z

R Core Team (2023). *The R Project for Statistical Computing.* Vienna, Austria. Retrieved from https://www.r-project.org/

Rowe, E., Almeda, M.V., Asbell-Clarke, J., Scruggs, R., Baker, R., Bardar, E., Gasca, S. (2021). Assessing implicit computational thinking in Zoombinis puzzle gameplay. *Computers in Human Behavior*, *120*, 106707. Retrieved from https://doi.org/10.1016/j.chb.2021.106707

Saldaña, J. (2021). *The Coding Manual for Qualitative Researchers.* Sage Publications. Retrieved from https://books.google.com/books?id=

RwcVEAAAQBAJ

Schmider, E., Ziegler, M., Danay, E., Beyer, L., Buhner, M. (2010, 01). Is it really robust? Reinvestigating the robustness of ANOVA against the normal distribution assumption. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, *6*, 147-151. Retrieved from https://doi.org/10.1027/1614-2241/a000016

Schwendimann, B.A., Rodríguez-Triana, M.J., Vozniuk, A., Prieto, L.P., Boroujeni, M.S., Holzer, A., ... Dillenbourg, P. (2017). Perceiving Learning at a Glance: A Systematic Literature Review of Learning Dashboard Research. *IEEE Transactions on Learning Technologies*, *10*(1), 30-41. Retrieved from https://doi.org/10.1109/TLT.2016.2599522

Sehl, S., & Vaniea, K. (2018). Permission Impossible: Teaching Firewall Configuration in a Game Environment. *European Workshop on Usable Security (EuroUSEC).* Retrieved from https://doi.org/10.14722/eurousec.2018.23006

Shephard, K. (2019). Higher Education Pedagogy. S.A. Fincher & A.V. Robins (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 276–291). Cambridge, United Kingdom: Cambridge University Press. Retrieved from https://doi.org/10.1017/9781108654555.011

Shute, V.J. (2008). Focus on Formative Feedback. *Review of Educational Research*, *78*(1), 153-189. Retrieved from https://doi.org/10.3102/0034654307313795

Thomas, J.W. (1993). Promoting Independent Learning in the Middle Grades: The Role of Instructional Support Practices. *The Elementary School Journal*, *93*(5), 575-591. Retrieved from https://doi.org/10.1086/461741

Švábenský, V., Vykopal, J., Čeleda, P., Dovjak, J. (2023). *Supplementary Materials: Automated Feedback for Participants of Hands-on Cybersecurity Training.* Zenodo. Retrieved from https://doi.org/10.5281/zenodo.8405210

Vykopal, J., Čeleda, P., Seda, P., Švábenský, V., Tovarňák, D. (2021, 10). Scalable Learning Environments for Teaching Cybersecurity Hands-on. *Proceedings of the 51st IEEE Frontiers in Education Conference* (pp. 1–9). New York, NY, USA: IEEE. Retrieved from https://doi.org/10.1109/FIE49875.2021.9637180

Vykopal, J., Švábenský, V., Chang, E.-C. (2020). Benefits and Pitfalls of Using Capture the Flag Games in University Courses. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 752–758). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3328778.3366893

Wang, W., Rao, Y., Zhi, R., Marwan, S., Gao, G., Price, T.W. (2020). Step Tutor: Supporting Students through Step-by-Step Example-Based Feedback. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (p. 391–397). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3341525.3387411

Wu, H., & Leung, S.-O. (2017). Can Likert Scales be Treated as Interval Scales?—A Simulation Study. *Journal of Social Service Research*, *43*(4), 527-532. Retrieved from https://doi.org/10.1080/01488376.2017.1329775

Yamin, M.M., Katt, B., Gkioulos, V. (2020). Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. *Computers & Security*, *88*, 101636. Retrieved from https://doi.org/10.1016/j.cose.2019.101636

Švábenský, V., Vykopal, J., Cermak, M., Laštovička, M. (2018, 07). Enhancing Cybersecurity Skills by Creating Serious Games. *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (pp. 194–199). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3197091.3197123

Švábenský, V., Vykopal, J., Seda, P., Čeleda, P. (2021, September). Dataset of Shell Commands Used by Participants of Hands-on Cybersecurity Training. *Elsevier Data in Brief*, *38*. Retrieved from https://doi.org/10.1016/j.dib.2021.107398

Švábenský, V., Vykopal, J., Tovarňák, D., Čeleda, P. (2021, 10). Toolset for Collecting Shell Commands and Its Application in Hands-on Cybersecurity Training. *Proceedings of the 51st IEEE Frontiers in Education Conference* (pp. 1–9). New York, NY, USA: IEEE. Retrieved from https://doi.org/10.1109/FIE49875.2021.9637052

Švábenský, V., Vykopal, J., Čeleda, P. (2020, 03). What Are Cybersecurity Education Papers About? A Systematic Literature Review of SIGCSE and ITiCSE Conferences. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 2–8). New York, NY, USA: Association for Computing Machinery. Retrieved from

https://doi.org/10.1145/3328778.3366816

Švábenský, V., Vykopal, J., Čeleda, P., Kraus, L. (2022). Applications of Educational Data Mining and Learning Analytics on Data From Cybersecurity Training. *Springer Education and Information Technologies*. Retrieved from https://doi.org/10.1007/s10639-022-11093-6

Švábenský, V., Weiss, R., Cook, J., Vykopal, J., Čeleda, P., Mache, J., ... Chattopadhyay, A. (2022). Evaluating Two Approaches to Assessing Student Progress in Cybersecurity Exercises. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education*. New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3478431.3499414