HackEd: A Pedagogical Analysis of Online Vulnerability Discovery Exercises

Eric Zhang, Daniel Votipka, and Michelle L. Mazurek
University of Maryland
ezhang98@umd.edu, dvotipka@cs.umd.edu, mmazurek@cs.umd.edu

Abstract

Hacking exercises are commonly used for security education, but evidence of their efficacy as an educational intervention is limited. In this poster, we develop a set of pedagogical effectiveness dimensions, derived from the learning sciences and educational literature, but specific to hacking exercises. We review 30 popular online hacking exercises, evaluating whether and how they implement each pedagogical dimension. Additionally, we interview the organizers of 14 exercises to identify potential roadblocks for each pedagogical dimension.

We found hacking exercises generally were tailored to students’ prior security experience and support learning by limiting extraneous load and establishing helpful online communities. However, few exercises provide necessary context, structure, or direct support for metacognition to help students transfer learned knowledge to new contexts. Additionally, immediate and tailored feedback and secure development practice were uncommon. Based on our results, we discuss hacking exercises’ strengths and weaknesses and make suggestions for improvement.

1 Introduction

Developing secure software is challenging. Even as awareness of the problem has grown [52, 63], vulnerabilities are regularly found in code running in the wild [15, 17, 57]. Ideally, these problems could be avoided completely through the use of secure languages and libraries or identified and mitigated through automated analysis. Unfortunately, most organizations rely on legacy code, which would be difficult to transition to newer, more secure technologies [45]. Further, while significant advances have been made toward automatically identifying — and in some cases remediating — vulnerabilities prior to code release [5–7, 24, 53, 75, 82, 84, 85], these results are currently limited. Human intelligence remains necessary at least for the foreseeable future, making it important to teach developers how to find and fix vulnerabilities.

Historically, the security community has used online hacking exercises to provide practical education, exposing participants to a variety of vulnerabilities and security concepts. In these exercises, participants demonstrate their understanding of security concepts by finding, exploiting, and fixing vulnerabilities in programs. They offer—in contrast to more traditional project-based learning—discrete practice sets that can be undertaken in a modular fashion, similarly to the exercises commonly included at the end of each chapter in mathematics textbooks. In fact, hacking exercises are commonly considered a very useful educational tool, with security experts often reporting that they rely on these exercises for their education [88], bug bounty platforms directing those interested in security to start with these exercises [35, 40], and a significant amount of recent security education work focused on creating new hacking exercises [8, 9, 25, 26, 55, 65, 91].

Further, prior work has provided some evidence that hacking exercises can provide valuable immediate feedback to learners in academic settings [13, 29, 65, 71, 73].

However, this evidence is limited for several reasons. First, these studies only consider a few exercises [8, 9, 18, 25, 26, 55, 65, 71, 91], producing sparse results and providing limited understanding of the broad set of popular exercises students participate in. Next, this work only focuses on a few measures of learning and engagement [25, 55, 65, 71, 87, 91], making the evidence narrow. They do not consider significant learning factors which are difficult to control for and measure. Therefore, exercise organizers have little guidance for building effective exercises, educators can not know which exercises provide the most effective learning, and researchers do not have a broad view of the landscape of current exercises.

In this study, we review of online hacking exercises to provide perspective on the current landscape of educational interventions. Specifically, we set out to answer two main research questions:

- **RQ1**: Do currently available exercises apply pedagogical principles suggested by the learning sciences literature? If so, how are these principles implemented?
- **RQ2**: What challenges do exercise organizers face when applying these principles?

To answer these questions we performed a qualitative review of 30 popular online hacking exercises. We evaluated each exercise against recommended pedagogical principles grounded in learning theory [3, 11]. We base our approach on previous curriculum evaluation efforts [47], tailoring the specific pedagogical principles we examine for applicability to hacking exercises. Further, we interview the organizers of 14 exercises to understand the challenges they face.

We found that no exercise implemented every pedagogical principle, but most were implemented by some exercises, some in thoughtful and creative ways. Notable shortcomings include that many exercises lack sufficient structure to help students organize knowledge, and enough feedback to guide
learning progress. Few organizers had considered metacognition, or helping students understand what and how much they have learned. From these results, we distill recommendations for improving exercises’ educational value.

2 Methods

To understand the current landscape of online hacking exercises, we performed a two-phase study: a qualitative review of exercises and interviews with exercise organizers.

2.1 Exercise Selection

We chose to focus on popular online educational exercises, based on prior work suggesting this intervention type is preferred by security experts [88]. Specifically, we consider exercises meeting the following criteria:

- **Educational** - We only include exercises explicitly stating education as a goal.
- **Hands-on** - The exercise included a component requiring students to actively practice security concepts.
- **Online and publicly accessible** - For practicality, we focus on online exercises so we could perform a full and fair analysis of all exercises through participation.
- **Popular** - Our goal is to understand the exercises students are most likely to participate in. To estimate a site’s popularity, we used its Tranco rank—a secure method for ranking sites based on user visits [51]. We used Alexa rankings whenever no Tranco ranking was available.

To identify eligible exercises, we used the recommendations of eight security education experts, relevant Google searches, and curated exercise lists [22, 50, 74, 80, 94]. Additionally, we included the top three similar sites provided by Alexa.com, for each exercise identified. We repeated this process until no new exercises were identified. We completed this search in October 2019.

We identified 45 exercises meeting our criteria. Due to the significant time required for each review (2.5 hrs each), we performed a random weighted sampling of 30 exercises, prioritizing the most popular exercises. The full list of reviewed exercises is given in Table 1.

2.2 Pedagogical Review (RQ1)

To determine the set of pedagogical principles, we drew on previous efforts to synthesize major theoretical and empirical learning sciences and education research findings [11]. This led us to five core pedagogical principles:

<table>
<thead>
<tr>
<th>Exercise</th>
<th>Rank</th>
<th>Exercise</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>gCTF† [32]</td>
<td>1.4</td>
<td>Backdoor [79]</td>
<td>*949.1</td>
</tr>
<tr>
<td>Infosec Institute [43]</td>
<td>14.1</td>
<td>Crackmes.one† [76]</td>
<td>*1011.4</td>
</tr>
<tr>
<td>HackTheBox† [10]</td>
<td>97.3</td>
<td>CSAW365 [49]</td>
<td>*1228.1</td>
</tr>
<tr>
<td>picoCTF† [62]</td>
<td>149.8</td>
<td>HackerTest [37]</td>
<td>*1254.5</td>
</tr>
<tr>
<td>HackthisSite [38]</td>
<td>105.1</td>
<td>CTFlearn [21]</td>
<td>*1267.0</td>
</tr>
<tr>
<td>OverTheWire [64]</td>
<td>151.3</td>
<td>HackEDU [34]</td>
<td>*2041.2</td>
</tr>
<tr>
<td>Root-me.org† [54]</td>
<td>172.7</td>
<td>Pwnadventure† [2]</td>
<td>*2364.7</td>
</tr>
<tr>
<td>Vulnhub† [30]</td>
<td>175.8</td>
<td>Mr. Code† [46]</td>
<td>*4570.2</td>
</tr>
<tr>
<td>Hacker101 [35]</td>
<td>330.4</td>
<td>IO Wargame [58]</td>
<td>*7168.8</td>
</tr>
<tr>
<td>Hellbound Hackers [36]</td>
<td>432.8</td>
<td>PACTF [66]</td>
<td>*9156.0</td>
</tr>
<tr>
<td>Smash the Stack† [86]</td>
<td>966.1</td>
<td>HXP CTF [42]</td>
<td>–</td>
</tr>
<tr>
<td>Microcorruption [33]</td>
<td>*378.8</td>
<td>BIBIFI† [14]</td>
<td>–</td>
</tr>
<tr>
<td>Pwnable [83]</td>
<td>*515.4</td>
<td>Pwn College† [81]</td>
<td>–</td>
</tr>
<tr>
<td>Cyber Talents [23]</td>
<td>*528.0</td>
<td>GirlsGo</td>
<td>–</td>
</tr>
<tr>
<td>XSS-Game† [31]</td>
<td>*626.1</td>
<td>CyberStart† [44]</td>
<td>–</td>
</tr>
</tbody>
</table>

1 Visit rank for the website, in thousands - Alexa if *, otherwise, using Tranco ranking which is less prone to tampering [51].

† An organizer from this exercise was interviewed or responded via email to our review.

Table 1: Exercises reviewed and their popularity.

- **Connecting to learners’ prior knowledge.** People develop new knowledge based on their pre-existing knowledge and beliefs [19, 69, 70, 89, 90]. This includes facts, perceptions, beliefs, values, and attitudes [19, 69]. Students interpret new information based on their current view of the world. Therefore, exercises should consider these to provide effective education.

- **Organizing declarative knowledge.** Another key to effective learning comes in students’ ability to transform facts into robust declarative knowledge [11]. To achieve mastery, students must go beyond memorizing tricks to solve challenges, but also organize disconnected facts based on underlying abstract concepts [3, 70, 89, 90].

- **Active practice and feedback.** Students must perform a task to achieve domain mastery [27, 48, 72]. Through deliberate, active practice, students can translate abstract concepts into practical knowledge. Students must also receive tailored feedback to guide their learning to specific goals [3]. Without feedback, students may become stuck or misunderstand the challenge [3, 11, 16].

- **Encouraging metacognitive learning.** Metacognitive learning has two main components: students’ ability to predict learning task outcomes, and their ability to gauge their own grasp of concepts [12, 16]. Guiding students to reflect on which solutions worked and why helps students develop a deeper conceptual understanding, supporting knowledge transfer [28, 67, 77, 78]. It also helps
students target further learning [28].

- Establishing a supportive and collaborative learning environment. A negative environment can hamper student progress, while a positive environment can excite and engage students [3, 68]. By participating in a group setting, students receive mentoring from more senior students, brainstorm possible solutions with peers, and get support and encouragement when stuck [61]. Additionally, the exercise framing can have a significant impact on whether students feel “good enough” to participate [68]. If the perceived barrier to entry is high, students may choose not to try. This is especially true for commonly underrepresented populations [41, 92, 93].

To identify actionable dimensions of each core principle, we started with the 24 dimensions used by Kim and Ko [47] in their similar review of online coding exercises. Two of the authors then updated these dimensions specifically for hacking exercises through collaborative open coding of five exercises. This process resulted in 36 total pedagogical dimensions, across the 5 core principles.

For each exercise, we performed a qualitative coding where two researchers evaluated each exercise independently according to the pedagogical dimensions. Each researcher completed at least one logical unit of the exercise (e.g., all questions in a category or a single specified path through the exercise), or five challenges if no logical relationship was present. We completed ten challenges on average per exercise (306 total).

After establishing our initial codebook, two researchers independently reviewed 20 exercises, comparing dimension codes after every five exercises for inter-coder reliability. After each round, the researchers resolved coding differences, modified the codebook when necessary, and re-coded previously reviewed exercises. This process was repeated until an Krippendorff’s Alpha (α) of at least 0.8— the recommended threshold for result reliability [39]— was achieved. The remaining exercises were divided evenly between the two researchers and coded independently.

2.3 Organizer Interviews (RQ2)

Next, we needed additional context from the organizers about their decision-making process. We reached out to all 30 organizers, inviting them to participate in a 45-minute structured interview. Note, throughout our interviews, we were careful to ensure organizers understood our goal was to understand their decision-making, not critique it.

In our interviews, we walked organizers through our review and asked whether they agreed with our assessment. For dimensions not implemented, we asked organizers whether they considered the dimension during exercise design and if so, why they did not implement it. Since this part of our study constituted human-subjects research, it was reviewed and approved by our organization’s ethics review board.

2.4 Limitations

Our study has several limitations, some related to our sampling method and some common to exploratory qualitative research. First, it is likely that we did not identify all exercises meeting our stated criteria through our review. Additionally, because we only perform our review on a sample of exercises, we may have missed a particularly good implementation of one of our educational interventions. However, because of our thorough search process and by weighting our sample toward more popular exercises, our results are likely representative of most students’ experience.

In our pedagogical review, we adopt a conservative approach, checking whether the dimension is implemented, but not whether it is implemented well. We did this so to broadly evaluate the pedagogy considered and establish an initial understanding of the current landscape. However, this broad view does not allow us to make statements about specific approaches’ efficacy. We encourage future work to build on our established roadmap through more focused review.

3 Results

For brevity, we only present highlights of our findings regarding each of our five core principles. Throughout, we use N to indicate the number of exercises demonstrating the given theme and O to indicate the number of organizers who mentioned a given theme when interviewed.

3.1 Connecting to students’ prior knowledge

Experience-based personalization was common. Most (N=22) exercises allow some personalization by experience. These exercises used a mix of difficulty indicators, including difficulty labels (e.g., Easy, Medium, Hard) (N=10), the number of students who have solved the challenge (N=14), and point values (i.e., more points indicate increased difficulty) (N=18). This lets participants attempt skill-level-appropriate problems, avoiding burnout on problems beyond their reach or boredom with too many easily solvable challenges.

Exercise designers build clear challenge concept progressions. Almost all exercises (N=29) include some challenges whose concepts build one on top of the others where appropriate. As an example, Microcorruption offers a progression across several challenges to teach buffer overflow concepts in a binary exploitation challenge. It begins with a program that requires the student to disassemble the program and read a hardcoded password string. The next challenge, forces the student to actually read the assembly code and understand the stack to reconstruct the password from a set of characters. Then, the student must exploit a simple buffer overflow with no mitigations in place to force execution down a successful path. The progression then continues by adding further mitigations to complicate the exploitation process.
3.2 Organizing Declarative Knowledge

Many exercises lacked clear structure. Providing explicit cues, such as challenge names that indicate a hierarchical concept structure or suggesting a progression through conceptually-related problems, can help students associate individual facts [3, 16]. Unfortunately, a majority (N=17) of exercises did not clearly group challenges with related concepts. Similarly, several exercises did not provide students a path to follow through more than two to three challenges as a conceptual organizing guide (N=11).

Few exercises included realistic challenges. Few exercises included any real-world-scale challenge programs (N=8). This potentially prevents students from learning practical skills necessary for scaling analyses to larger programs. Many organizers said they chose to avoid realistic challenges because they believed focusing students on specific concepts was more important (O=9) and developing challenges with this complexity is difficult (O=1). Others chose not to include complexity—and therefore require the student to perform extraneous tasks—because they wanted to make sure their exercise was fun and engaging (O=5).

3.3 Practice and Feedback

Secure development practice was uncommon. Very few exercises (N=4) included any challenges asking students to write secure code and two of those (Hellbound Hackers and Pwnable) only included a few. Instead, students are left to make the logical jump from identifying and exploiting to preventing a vulnerability without educational support. In many cases, organizers simply felt that including secure development practice was difficult to evaluate (O=7).

Some challenges provide “correct path” markers. In some challenge developers included checks in the target program’s execution to update their output if the student’s exploit was following the correct path, even if the exploit was not yet fully successful (N=10).

Many organizers said providing this type of tailored feedback was difficult because this feedback had to be specifically tailored for each challenge (O=5). Instead of providing automated feedback, many organizers opted to provide tailored information in the exercise’s forum based on student demand (O=5) or through publicly available walkthroughs (N=28).

3.4 Encouraging Metacognitive Learning

Few exercises guided transfer beyond the challenge. While the exercises almost all taught how to use concepts through hands-on exercises (N=29), few explained when (N=6) or why (N=5) to use the concept in other settings.

Interestingly, this was the dimension group organizers most often reported not considering (O=9). As an example, when we explained metacognition to the picoCTF organizer, they said “I don’t know if I ever heard of metacognition before… that could really guide us in developing problems that can guide our learners even better.”

3.5 Establishing an Environment Conducive to Learning

Exercises help students find community through online forums. Most exercises provided IRC, Slack, or Discord channels or online forums, where students could post questions and share their experiences with other competitors (N=17). The HackTheBox organizers explained they have “a vocal community that everyone chats… in order to help each other to understand challenges and learn.”

Many exercises reduced extraneous load. This included providing browser-based tool support (e.g., wireshark, command line, disassembler) (N=6) or an ssh server with required tools (N=4). The best examples were Microcorruption, which allowed students to perform required tasks with a browser-based disassembler and debugger, and Pwn College, which links to binaries in the BinaryNinja cloud service [1] for advanced reverse engineering support.

Most exercises used supportive terminology, but a few marginalized beginners. A plurality of exercises included language throughout offering encouragement (N=17). Unfortunately, some exercises chose terminology marginalizing newer students who might struggle with basic concepts (N=5). For example, HackthisSite called their first challenge the “idiot” challenge and saying “if you can’t solve it, don’t go crying to anyone because they’ll just make fun of you.”

4 Recommendations

With these findings in mind, we suggest recommendations for exercise organizers and directions for future work.

Support active student engagement in metacognition. Because many exercises simply did not consider metacognition, the first step should be to apply best practices from the learning sciences and education literature. For example, students could be prompted to predict the outcome of an exploitation attempt prior to its execution and subsequent success or failure. This foregrounds the student’s current mental model of system they are attempting to exploit and the exploit itself’s function. This technique has proved effective in other domains [20].

Use a graphical syllabus to provide concept structure. A graphical syllabus is a visual representation (e.g., flow chart or diagram) of concepts covered in a course and their relationships [59, 60]. These visualizations help students process and organize information.

In addition to these recommendations, future work should explore the pedagogical dimensions organizers reported as difficult to implement, namely secure development practice and tailored feedback.
References


