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, 56, 65, 73, 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 it's Tranco rank[1]—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[2], 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:

[1]we use the version from October 15th, 2019
[2]https://www.alexa.com/siteinfo

Exercise	Rank[1]	Exercise	Rank[1]
gCTF† [32]	1.4	Backdoor [79]	*949.1
Infosec Institute [43]	14.1	Crackmes.one† [76]	*1011.4
HackTheBox† [10]	97.3	CSAW365 [49]	*1228.1
picoCTF† [62]	149.8	HackerTest [37]	*1254.5
HackthisSite [38]	105.1	CTFlearn [21]	*1267.0
OverTheWire [64]	151.3	HackEDU [34]	*2014.2
Root-me.org† [54]	172.7	Pwnadventure† [2]	*2364.7
Vulnhub† [30]	175.8	Mr. Code† [46]	*4570.2
Hacker101 [35]	330.4	IO Wargame [58]	*7168.8
Hellbound Hackers [36]	432.8	PACTF [66]	*9156.0
Smash the Stack† [86]	966.1	Angstrom† [4]	*11708.9
Microcorruption [33]	*378.8	HXP CTF [42]	–
Pwnable [83]	*515.4	BIBIFI† [14]	–
Cyber Talents [23]	*528.0	Pwn College† [81]	–
XSS-Game† [31]	*626.1	GirlsGo CyberStart† [44]	–

[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 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 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

[1] Binary ninja cloud, 2020. (Accessed 06-03-2020).

[2] Vector 35. Pwnadventure sourcery. (Accessed 05-27-2020).

[3] Susan A Ambrose, Michael W Bridges, Michele DiPietro, Marsha C Lovett, and Marie K Norman. *How learning works: Seven research-based principles for smart teaching*. John Wiley & Sons, 2010.

[4] angstromCTF. angstromctf. (Accessed 05-27-2020).

[5] Nuno Antunes and Marco Vieira. Comparing the effectiveness of penetration testing and static code analysis on the detection of sql injection vulnerabilities in web services. In *Proceedings of the 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing*, PRDC '09, pages 301–306, Washington, DC, USA, 2009. IEEE Computer Society.

[6] Andrew Austin and Laurie Williams. One technique is not enough: A comparison of vulnerability discovery techniques. In *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement*, ESEM '11, pages 97–106, Washington, DC, USA, 2011. IEEE Computer Society.

[7] Dejan Baca, Bengt Carlsson, Kai Petersen, and Lars Lundberg. Improving software security with static automated code analysis in an industry setting. *Software: Practice and Experience*, 43(3):259–279, 2013.

[8] Nathan Backman. Facilitating a battle between hackers: Computer security outside of the classroom. In *In Proc. of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE ?16, page 603?608, New York, NY, USA, 2016. ACM.

[9] Kevin Bock, George Hughey, and Dave Levin. King of the hill: A novel cybersecurity competition for teaching penetration testing. In *2018 USENIX Workshop on Advances in Security Education (ASE 18)*, Baltimore, MD, August 2018. USENIX Association.

[10] Hack The Box. Hack the box. (Accessed 05-27-2020).

[11] John D. Bransford, Ann L. Brown, and Rodney R. Cocking. *How people learn: Brain, mind, experience, and school: Expanded edition*. National Academies Press, 2000.

[12] Ann L. Brown. The development of memory: Knowing, knowing about knowing, and knowing how to know. volume 10 of *Advances in Child Development and Behavior*, pages 103 – 152. JAI, 1975.

[13] Tanner J. Burns, Samuel C. Rios, Thomas K. Jordan, Qijun Gu, and Trevor Underwood. Analysis and exercises for engaging beginners in online CTF competitions for security education. In *2017 USENIX Workshop on Advances in Security Education (ASE 17)*, Vancouver, BC, August 2017. USENIX Association.

[14] Maryland Cybersecurity Center. Build it break it fix it. (Accessed 05-27-2020).

[15] Yung-Yu Chang, Pavol Zavarsky, Ron Ruhl, and Dale Lindskog. Trend analysis of the cve for software vulnerability management. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 1290–1293. IEEE, 2011.

[16] William G Chase and Herbert A Simon. Perception in chess. *Cognitive psychology*, 4(1):55–81, 1973.

[17] Steve Christey and Robert A Martin. Vulnerability type distributions in cve. https://cwe.mitre.org/documents/vuln-trends/index.html, 2007.

[18] Kevin Chung and Julian Cohen. Learning obstacles in the capture the flag model. In *Proceedings of the 1st USENIX Summit on Gaming, Games, and Gamification in Security Education*, 3GSE '14, San Diego, CA, 2014. USENIX Association.

[19] Paul Cobb, Erna Yackel, and Terry Wood. A constructivist alternative to the representational view of mind in mathematics education. *Journal for research in mathematics education*, pages 2–33, 1992.

[20] Catherine Crouch, Adam P. Fagen, J. Paul Callan, and Eric Mazur. Classroom demonstrations: Learning tools or entertainment? *American Journal of Physics*, 72(6):835–838, 2004.

[21] CTFLearn. Ctflearn. (Accessed 05-27-2020).

[22] CTFTime. Ctftime.org / all about ctf (capture-the-flag), 2017. (Accessed 06-08-2017).

[23] CyberTalents. Cyber talents practice. (Accessed 05-27-2020).

[24] Adam Doupé, Marco Cova, and Giovanni Vigna. Why johnny can't pentest: An analysis of black-box web vulnerability scanners. In *Proceedings of the 7th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA'10, pages 111–131, Berlin, Heidelberg, 2010. Springer-Verlag.

[25] Adam Doupé, Manuel Egele, Benjamin Caillat, Gianluca Stringhini, Gorkem Yakin, Ali Zand, Ludovico

Cavedon, and Giovanni Vigna. Hit 'em where it hurts: A live security exercise on cyber situational awareness. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC ?11, page 51?61, New York, NY, USA, 2011. Association for Computing Machinery.

[26] W. Du. Seed: Hands-on lab exercises for computer security education. *IEEE Security Privacy*, 9(5):70–73, 2011.

[27] K Anders Ericsson and Neil Charness. Expert performance: Its structure and acquisition. *American psychologist*, 49(8):725, 1994.

[28] John H. Flavell. Metacognitive aspects of problem solving. In Lauren B. Resnick, editor, *The nature of intelligence*. Lawrence Erlbaum Associates, 1976.

[29] Gordon Fraser, Alessio Gambi, Marvin Kreis, and José Miguel Rojas. Gamifying a software testing course with code defenders. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE ?19, page 571?577, New York, NY, USA, 2019. Association for Computing Machinery.

[30] g0tmi1k. Vulnhub. (Accessed 05-27-2020).

[31] Google. Xss game. (Accessed 05-27-2020).

[32] Google. Google ctf 2019, 2019. (Accessed 05-27-2020).

[33] NCC Group. Embedded security ctf. (Accessed 05-27-2020).

[34] HackEDU. Hackedu. (Accessed 05-27-2020).

[35] HackerOne. Home | hacker 101. (Accessed 05-21-2020).

[36] HellBound Hackers. Hellbound hackers. (Accessed 05-27-2020).

[37] HackerTest. Hacker test. (Accessed 05-27-2020).

[38] HackThisSite. Hackthissite. (Accessed 05-27-2020).

[39] Andrew F Hayes and Klaus Krippendorff. Answering the call for a standard reliability measure for coding data. *Communication methods and measures*, 1(1):77–89, 2007.

[40] Sam Houston. Researcher resources - how to become a bug bounty hunter, 2016. (Accessed 05-21-2020).

[41] Sylvia Hurtado, Jeffrey Milem, Alma Clayton-Pedersen, and Walter Allen. Enacting diverse learning environments: Improving the climate for racial/ethnic diversity in higher education. *ASHE-ERIC Higher Education Report*, 26(8), 1999.

[42] HXP. Hxp ctf. (Accessed 05-27-2020).

[43] InfoSec Institute. n00bs ctf labs. (Accessed 05-27-2020).

[44] SANS Cybersecurity Institute. Girls go cyberstart. (Accessed 05-27-2020).

[45] Trevor Jim, J Gregory Morrisett, Dan Grossman, Michael W Hicks, James Cheney, and Yanling Wang. Cyclone: A safe dialect of c. In *USENIX Annual Technical Conference*, ATC 02, pages 275–288, 2002.

[46] David Keller. Mr code's wild ride. (Accessed 05-27-2020).

[47] Ada S. Kim and Andrew J. Ko. A pedagogical analysis of online coding tutorials. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, SIGCSE ?17, page 321?326, New York, NY, USA, 2017. Association for Computing Machinery.

[48] Gary A Klein. *Sources of power: How people make decisions*. MIT press, 2017.

[49] NYU OSIRIS Lab. Csaw365. (Accessed 05-27-2020).

[50] Sjoerd Langkemper. Practice your hacking skills with these ctfs, December 2018. (Accessed 05-22-2020).

[51] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, NDSS 2019, February 2019.

[52] Marsh. Global cyber risk perception survey report 2019. Technical report, Marsh, 2019.

[53] Gary McGraw and John Steven. Software [in]security: Comparing apples, oranges, and aardvarks (or, all static analysis tools are not created equal. http://www.informit.com/articles/article.aspx?p=1680863, 2011. (Accessed 02-26-2017).

[54] Root Me. Root me. (Accessed 05-27-2020).

[55] Jelena Mirkovic and Peter A. H. Peterson. Class capture-the-flag exercises. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, San Diego, CA, August 2014. USENIX Association.

[56] Jelena Mirkovic, Aimee Tabor, Simon Woo, and Portia Pusey. Engaging novices in cybersecurity competitions: A vision and lessons learned at ACM tapia 2015. In *2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15)*, Washington, D.C., August 2015. USENIX Association.

[57] Mitre. Cve. https://cve.mitre.org/, 2019.

[58] Netgarage. Io wargame. (Accessed 05-27-2020).

[59] Linda B Nilson. The graphic syllabus: A demonstration workshop on how to visually represent a course. In *Proceedings of the Professional and Organizational Development Network in Higher Education*, POD '00, 2000.

[60] Linda B. Nilson. The graphic syllabus: Shedding a visual light on course organization. *To Improve the Academy*, 20(1):238–259, 2002.

[61] Angela M. O'Donnell and King Alison. *Cognitive Perspectives on Peer Learning*. Routledge, 1st edition, 1999.

[62] Plaid Parliament of Pwning. picoctf. (Accessed 05-27-2020).

[63] Charlie Osborne. Hey google: What we search for most in cybersecurity .. cyber security?, September 2019. (Accessed 05-21-2020).

[64] OverTheWire. Overthewire. (Accessed 05-27-2020).

[65] Kentrell Owens, Alexander Fulton, Luke Jones, and Martin Carlisle. pico-boo!: How to avoid scaring students away in a ctf competition. 2019.

[66] PACTF. Pactf. (Accessed 05-27-2020).

[67] Annemarie Sullivan Palincsar and Ann L Brown. Reciprocal teaching of comprehension-monitoring activities. *Center for the Study of Reading Technical Report; no. 269*, 1983.

[68] Ernest T Pascarella and Patrick T Terenzini. *How college affects students: Findings and insights from twenty years of research.* ERIC, 1991.

[69] Jean Piaget. *Success and understanding*. Routledge, 1978.

[70] Jean Piaget and Margaret Cook. *The origins of intelligence in children*, volume 8. International Universities Press New York, 1952.

[71] K. Qian, D. Lo, H. Shahriar, L. Li, F. Wu, and P. Bhattacharya. Learning database security with hands-on mobile labs. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–6, 2017.

[72] Ernst Z Rothkopf and MJ Billington. Goal-guided learning from text: inferring a descriptive processing model from inspection times and eye movements. *Journal of educational psychology*, 71(3):310, 1979.

[73] Dale C. Rowe, Barry M. Lunt, and Joseph J. Ekstrom. The role of cyber-security in information technology education. In *Proceedings of the 2011 Conference on Information Technology Education*, SIGITE ?11, page 113?122, New York, NY, USA, 2011. Association for Computing Machinery.

[74] Andrew Ruef, Evan Jensen, Nick Anderson, Alex Sotirov, Jay Little, Brandon Edwards, Marcin W, Dino Dai Zovi, and Mike Myers. Ctf field guide. (Accessed 05-21-2020).

[75] Nick Rutar, Christian B. Almazan, and Jeffrey S. Foster. A comparison of bug finding tools for java. In *Proceedings of the 15th International Symposium on Software Reliability Engineering*, ISSRE '04, pages 245–256, Washington, DC, USA, 2004. IEEE Computer Society.

[76] s4r. crackmes. (Accessed 05-27-2020).

[77] Marlene Scardamalia, Carl Bereiter, and Rosanne Steinbach. Teachability of reflective processes in written composition. *Cognitive Science*, 8(2):173 – 190, 1984.

[78] Alan H Schoenfeld. Problem solving in the mathematics curriculum. a report, recommendations, and an annotated bibliography. maa notes, number 1. 1983.

[79] SDSLabs. backdoor. (Accessed 05-27-2020).

[80] Yan Shoshitaishvili. zardus/wargame-nexus: A sorted and updated list of security wargame sites, April 2020. (Accessed 05-22-2020).

[81] Yan Shoshitaishvili and Connor) Nelson. pwn.college. (Accessed 05-27-2020).

[82] Yan Shoshitaishvili, Michael Weissbacher, Lukas Dresel, Christopher Salls, Ruoyu Wang, Christopher Kruegel, and Giovanni Vigna. Rise of the hacrs: Augmenting autonomous cyber reasoning systems with human assistance. In *Proc. of the 24th ACM SIGSAC Conference on Computer and Communications Security*, CCS '17. ACM, 2017.

[83] GaTech SSLab. pwnable.kr. (Accessed 05-27-2020).

[84] Larry Suto. Analyzing the effectiveness and coverage of web application security scanners. Technical report, BeyondTrust, Inc, 2007.

[85] Larry Suto. Analyzing the accuracy and time costs of web application security scanners. Technical report, BeyondTrust, Inc, 2010.

[86] Smash the Stack Wargaming Network. Smash the stack. (Accessed 05-27-2020).

[87] David H. Tobey, Portia Pusey, and Diana L. Burley. Engaging learners in cybersecurity careers: Lessons from the launch of the national cyber league. *ACM Inroads*, 5(1):53?56, March 2014.

[88] Daniel Votipka, Rock Stevens, Elissa M Redmiles, Jeremy Hu, and Michelle L Mazurek. Hackers vs. testers: A comparison of software vulnerability discovery processes. *Proc. of the IEEE*, 2018.

[89] L.S. Vygotsky. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, 1980.

[90] L.S. Vygotsky, E. Hanfmann, G. Vakar, and A. Kozulin. *Thought and Language*. Mit Press. MIT Press, 2012.

[91] Jan Vykopal and Miloš Barták. On the design of security games: From frustrating to engaging learning. In *2016 USENIX Workshop on Advances in Security Education (ASE 16)*, Austin, TX, August 2016. USENIX Association.

[92] L.W. Watson. *How Minority Students Experience College: Implications for Planning and Policy*. Stylus, 2002.

[93] Elizabeth J Whitt, Marcia I Edison, Ernest T Pascarella, Amaury Nora, and Patrick T Terenzini. Women's perceptions of a" chilly climate" and cognitive outcomes in college: Additional evidence. *Journal of College Student Development*, 1999.

[94] Jordan Wiens. Practice ctf list, May 2019. (Accessed 05-21-2020).