Developing Effective Cybersecurity Labs: Initiating The Conversation

Glen Sagers
Computer Science and Information Security
Southern Utah University
Cedar City, USA
0000-0001-6294-3619

***Abstract*—Hands-on labs are critical to cybersecurity education, but are often based on one instructor's preferences, knowledge, and skills. The chosen topics may not align with established curriculum guidelines, and even when they do, labs may not be maximally effective in teaching the topics, partly because the professor is not an instructional designer. This paper represents a first attempt to draw up guidelines for cybersecurity labs, based on pedagogical principles and best practices in related fields.**

Keywords—cybersecurity labs, cybersecurity assessment, labs, authentic assessment

I. Introduction

It is widely agreed that hands-on laboratory exercises are needed for effective student education in the sciences and other technical fields such as engineering [25, 13]. This includes all aspects of computing, from programming, through IT and system administration, to cybersecurity.

Labs are so important to properly teaching the concepts of cybersecurity that both the joint IEEE and ACM 2017 curriculum guidelines [11] and the National Centers of Academic Excellence (NCAE, formerly CAE, administered jointly by the NSA and DHS), curriculum requirements [18] mandate the use of laboratory environments that allow students to practice skills that they learn in classes. Such labs should be realistic, and contain tools that are at least similar to the tools students will be called upon to use in their careers.

Some of the main obstacles to creating these labs are the expense, time, technical knowledge, and effort required to develop and maintain these lab environments and exercises as hardware, software, and threat environments change. Much has been written about ways to automate lab development and operation [14, 16, 22, 24], but very little literature is dedicated to the development of labs that are pedagogically effective.

This paper represents an attempt to bring together the pedagogical pillars that would help cybersecurity labs be as effective of a learning tool as possible, while preserving efficiency in maintaining, deploying, and grading the exercises.

II. Problem statements

This paper is in no way intended to minimize the difficulties in maintaining, automating, and deploying lab environments. The author has spent many hours on all of these tasks, and the automations that others have provided have relieved many of the burdens of those tasks.

All the automation in the world, however, cannot change the content of the lab to make it cover the topics that need to be covered, in a way that cements the topics in the students' minds. The only way to ensure that is to use proper pedagogical underpinnings to support the learner's comprehension and absorption of materials.

Beyond the difficulties of automation, a number of other issues have been identified with lab exercises. In no particular order, they are:

- Labs tend to be written by one instructor, based on their own, limited, experience. This may impact both topic choices and technical quality of the labs.
- Labs may not be engaging for the student.
- Labs may not be appropriate for the level of the student or course, even if they are engaging.
- Labs may be either too detailed, or too sparse in terms of instructions.
- Labs may use tools that are non-standard, or otherwise proprietary, limiting their applicability in other educational settings or in jobs.
- Labs, like anything in computing, may contain bugs.
- Exercise scope may be inappropriate for the student skill or course levels.
- Students may be able to collude or cheat on the lab assignments, leading to reduced learning of material.
- Grading labs can be difficult compared to other types of assessments.

When one instructor writes a lab, it must necessarily be based on that instructor's experience. Given the wide range of cybersecurity topics, both defensive and offensive, it is unlikely that any one instructor could develop assignments for a full cybersecurity curriculum. This necessitates sharing of materials, which is a great way of both dividing labor and taking advantage of the expertise of specialists. However, if the labs by different instructors are not written with pedagogical principles in mind, the student outcomes between classes will be very different. Additionally, when sharing labs between

979-8-3503-2028-2/22/$31.00 ©2022 IEEE
DOI 10.1109/CSCI58124.2022.00066

instructors that are written with different teaching and writing styles, it may be hard for the recipient to interpret the intent of the lab and skills it is supposed to teach.

An exercise which is technically sound and based on pedagogical principles may not hold the interest of the student. There is, of course, nothing that dictates that a highly technical lab cannot be interesting to students. But, if elements designed to hold the learner's attention are not consciously introduced, the overall effect may be a lack of attention on the part of the student, which will negatively impact learning.

Even an engaging lab may not be enough to impart the knowledge that was intended. If written at a level which is conceptually beyond the student, they may be able to complete the tasks, but not understand the implications. That situation is probably less common than the converse; it is more likely that a lab is engaging, but only gives basic understanding of the topic.

It may be very difficult to decide on a proper balance between enough instructions to prevent common problems faced by students on one hand, and making the lab too easy or rote on the other. Asking open-ended questions can help, but often it takes multiple iterations over many semesters to polish a lab to the point that students are able to apply their knowledge rather than simply following instructions.

Tool choice in labs is both difficult and vital. The limiting factor in tool choice may come down to budget; many commercial cybersecurity tools are quite expensive. While various educational discounts exist, the cost may still be too high to justify. This leaves open-source tools as a very cost-effective option. In some cases, these represent truly best of breed software that is used widely in industry. In others, there are no open equivalents to commercial software, or the freely available equivalents are substandard. If the tool chosen is extremely difficult to learn or use, requires too many resources to make it practical in the common virtual lab environments, or simply does not have enough useful features, student learning will be reduced due to frustration.

Bugs and other errors are common in computing. Whether in the lab infrastructure the software the student is using to perform the exercise, or mistakes in the lab instructions, these bugs can be a source of incredible frustration for students. Once a student becomes frustrated with the lab, or worse, with the instructor, they tend to simply follow rote instructions, and not go beyond that to comprehending, limiting learning.

A cybersecurity lab could cover many levels of a specific topic. For example, when teaching a student how to break into a vulnerable system in a penetration testing course, a single exercise could cover the initial compromise, or expand in scope to cover reconnaissance before the compromise and privilege escalation, scanning of other systems, and pivoting. There is not necessarily a correct scope or level for any given exercise, but just as in systems development, scope creep may easily occur in writing lab assignments.

Cheating and other forms of collusion continue to be a problem in academia. Any time students go beyond the bounds set by an instructor, their learning may be reduced. Conversely, when students legitimately and properly collaborate on an exercise, they learn from each other, enhancing comprehension [15]. The border between collaboration and cheating is blurry, and care must be taken to show and tell students what is acceptable and what is not.

Grading of labs is usually more time consuming than evaluating some other forms of assessment, such as exams or quizzes. This is not a reason to avoid lab exercises, but rather to develop labs with not only student outcomes but grading efficiency in mind.

III. Solutions

Solutions do exist to the difficulties just presented. Just as in cybersecurity, some are technical in nature, while others are human-based or procedural in nature.

As early as 1967, engineering faculty recognized the need for laboratory instruction in their field [13]. As in other science fields, especially applied sciences, computing has always been a discipline of practicing. Computer science students practice coding to learn a new language, those studying to become IT personnel spend hands-on time with servers, and budding cybersecurity practitioners need to actually work with firewalls.

Many studies have investigated the value of labs, including both physical and virtual setups. Findings have been mixed as to which is better [8], but suffice it to say, given the prevalence and ease of provisioning of cloud solutions for labs, coupled with the widespread use of cloud technologies in industry, cloud labs of various kinds are here to stay. This paper assumes no particular infrastructure; of course, labs need to be adapted per-technology, but the fundamentals of good instructional design hold across platforms.

In engineering, the question of "How many engineering colleges or individual disciplinary programs have taken a comprehensive look at their laboratory experience?" (p. 368) was asked in 2002 [9]. While this question may not have been explicitly asked in cybersecurity, it is time to do so. Simply having lab exercises is not enough, they must be effective in teaching the desired, and in some cases, mandated, content.

Educational theory tells us there are "Nine Events of Instruction". These were adapted by Zvacek and Restivo to read (p. 1442-1143) [25]:

- Learner Attention: *What elements have been included to gain (and hold) the student's attention?*
- Objectives: *How do students know what is expected of them when they do this lab assignment?*
- Recall of Prior Learning: *Are students reminded of what they already know, to facilitate the integration of new learning?*
- Supporting Materials: *What has been provided to help students navigate the learning activity, use equipment appropriately, and understand how their work will be assessed and graded?*
- Tools and Learning Resources: *What is available to enable students to complete the lab successfully?*
- Practice and Feedback: *What will students be doing as they engage in the activity, how have these tasks been*

organized, and what types of feedback and support are available?

- Assessment: *How will student performance be assessed?*
- Knowledge Transfer: *Will there be opportunities (at some point) for students to apply their new skills to unfamiliar problems?*

These eight components were then matched to ABET educational outcomes, as well as EUR-ACE and ABE/EAC colloquy objectives to create a set of guidelines for engineering students. The reader is encouraged to see the full guidelines in their paper. While most, if not all of these guidelines apply in any computing course, rather than simply applying all of these engineering-focused guidelines to a cybersecurity context, this paper develops focused guidelines for our discipline.

Along with the previous eight components, other aspects of curriculum design and pedagogy must be incorporated into the design of effective labs. These pedagogical pillars include aspects of active learning, project-based learning (PBL) [22], authentic assessment [2], and Piaget's learning-by-doing [20]. These hands-on learning elements are a natural fit to lab exercises, and are incorporated in the solutions and guidelines developed herein. The rest of this section will present solutions to each of the previously-introduced difficulties.

Topics covered in labs need to match the needs of industry. There are multiple ways to ensure this occurs. First, one could ask industry members what skills are needed in a graduate. This certainly occurs within our discipline, via industry advisory boards. This approach will result in fairly good information, for those companies, but will not represent all cybersecurity needs in such a broad field. Fortunately, there is no need to reinvent the wheel, there are several developed lists of topics that should be covered in a cybersecurity curriculum. Two such lists are the guidelines jointly developed by the IEEE & ACM in 2017 and the CAE requirements developed by the Centers of Academic Excellence. There is a great deal of overlap between these lists, and the choice of which to use may be mostly based on whether the institution is seeking CAE designation or simply wants a solid set of topics for their students. Regardless of which set is chosen, instructors must realize that the speed of change in the field requires changing topics for labs frequently. Because of the rapid evolution, the selection process often devolves to being based on the professor's personal interests and skills, ignoring the carefully curated topics that match what industry wants [3].

Beyond topic choice, the technical quality of the labs will be influenced by the expertise of the instructor. Technical quality includes both expertise in cybersecurity topics and the instructor's capabilities in instructional design. These disparate skillsets may not be found in the same individual, and even if they are, no one instructor can reasonably be expected to master all aspects of both. Luckily, most universities have some sort of teaching and learning improvement center which employs instructional designers. This resource can greatly assist in creating good labs. By coupling this with a student worker in cybersecurity who can beta-test labs, an instructor can be reasonably assured of a lab that students will be able to follow through. For technical cybersecurity content, there is simply no way for one individual to be able to master all of it. Some approaches to overcoming a lack of technical knowledge in select areas include collaborating with other faculty, exchanging labs with colleagues in other schools, and receiving technical training in those topics.

In summary, following existing frameworks will help guide topic choices, and collaborating with instructional designers and other faculty will alleviate some of the issues of limited knowledge in any one faculty member.

How can a professor write labs that are engaging to the student? Instructional theory provides many mechanisms for gaining and holding student attention. Some of the most relevant in this context include introducing topics within a relevant context and including motivational elements [25]. One way of making things relevant to the learner is to use the principles of authentic assessment. Authentic assessment is defined as assessment that incorporates a connection to a real-life context.

Other methods of keeping the pupil engaged include writing clear, concise objectives that show what they will know and be able to do at the conclusion of the exercise. Especially if the objective can be tied to specific needs of the pupil in a context of their personal life outside the classroom, they will be more engaged.

Gamification, defined as bringing game design elements into non-game contexts, [10] has been tried as a method of keeping students involved in lab exercises. Common gamification gambits in cybersecurity include capture the flag (CTF) exercises, forensic challenges, red-team/blue-team competitions, and so on. However, many of the exercises developed so far do not align well with curricular outcomes [10]. Aligning games with outcomes, such as those specified by the NCAE or IEEE/ACM guidelines shows promise in improving student learning

Good supporting materials are materials outside the assignment that provide directions for specialized equipment use and troubleshooting [25]. These materials are not necessarily developed by the instructor, but should be curated by them for relevancy and currency. As websites change, this will require professors to review the links and content every semester, but doing so will greatly reduce student frustration.

Milestones in assignments refer to the fact that many labs are sequential in nature, both in the individual exercise and between exercises. Because of this, labs can be designed that have natural stopping points where a deliverable or answer can be inserted. Subsequent answers may depend on this step being completed [25]. Nunez et al. show that especially when immediate feedback can be provided to the student, via automatic grading of submissions, the student will remain engaged and know their progress at all times [19]. This approach of automatic grading also reduces load on the instructor, but of course, it is not possible for all types of exercises.

In summary, keeping a student engaged has many facets, but some of the key elements include making the topic relevant to the student and their personal life; gamification of activities as long as they align with educational objectives, providing good supporting materials, and giving milestones of completion or

partial completion, especially when coupled with automatic assessment.

Even labs which are engaging to the student may not be at the right level for the course or the student. Initiating recall of prior learning by giving pre-lab tasks and assessing learner readiness before starting the assignment tasks aids in being sure students return to their previous levels [25]. Making explicit connections to previous assignments and labs also aids in student recall. Following established frameworks and model curricula can aid in presenting the material at the right level, as can utilizing labs others have created for similar courses. There are many online resources for such labs, which can be adapted for local use. Clark Center and DETER are two such resources [5, 7]

In summary, being sure the task assigned is aligned to course and overall curriculum objectives and adapting tested lab tasks to local needs will ensure that the assignment matches class level. Initiating recall of prior learning and assessing learner readiness for new topics can also ensure that materials are at the right level for students.

Lab instructions must be at an appropriate level of detail. Too much detail, specifying every single step a student must take, results in rote repetition, which only helps at lower levels of Bloom's Taxonomy [1]. On the other hand, skeletal instructions, especially if the lab content is at a level that already makes the student stretch past their comfort level, will likely result in frustration. Instructional design theory provides a few clues as to how to avoid these pitfalls. First, objectives need to be understandable to even novice learners. Second, give examples of what output should look like, and explicitly call out what should be contained in a good answer, whether that is a description of what should be in a good screenshot, or a formal rubric. Third, as is almost always the case in education, yes or no, true or false answers should be avoided. Questions should tap into higher levels of Bloom's taxonomy, and ask for justification of why a result occurred, or ask the student to draw conclusions based on output from a tool [25]. Further, the level of lab detail in terms of description and background information influences the student learning process, specifically between phases of storing information and integrating what was learned [23].

In summary, an effective assignment needs to have sufficient instructions without giving every step, and needs to go beyond simplistic comprehension and recall answers. Explicitly call out what should be contained in a good answer, and give examples. By drawing on established instructional design and educational theories, such as Bloom's taxonomy, as well as adapting previously-tested labs to local needs, a teacher may ensure that lab detail level is appropriate.

Many thousands of cybersecurity tools exist. Which is right for a particular job? There is no generalizable answer, other than one hated by students; "it depends…". Often, in an educational context, the right tool is the tool that budgets can afford. This may mean using tools for which an educational discount is available or open-source tools. What should be avoided, whether the tool is free via a donation or openly available, or simply reduced price, is the use of non-standard tools. In this context, non-standard means simply that the tool is not used in industry, or does not follow established standards in areas like file formats, operating systems, or conventions of command use.

Learning non-standard tools will be of less use to students in their careers. Although it means extra work, instructors should be constantly looking for newer, better tools to educate about a topic. One way of doing this is to have students research tools as part of class, and give presentations on these tools. In this author's experience, students will often go above and beyond the level required simply for the enjoyment of learning a novel program. Another source of information on standardization is industry advisory boards.

In summary, the right tool for the job can be hard to pin down, but by using tools that are used in industry, and keeping labs updated with newer tools as they emerge, an instructor's labs can be both current and relevant.

Bugs come in two principal categories as far as they relate to laboratory exercises. First, there are bugs in software. Software bugs may exist in infrastructure or the tools used to teach the concepts. Either may be frustrating for students. Unfortunately, such bugs are beyond the control of the instructor. For software bugs, good supporting materials help alleviate student issues [25]. To help alleviate bugs in infrastructure, support from IT personnel at the university or a cloud provider will likely be necessary.

The second type of bug of concern to this paper are those in lab instructions. These are probably the fault of the teacher, and as such, their own direct responsibility. Stamping out such bugs is as easy, and as hard, as revising any paper or written material. Three approaches to clearing up bugs are:

- Creating labs with the aid of an instructional designer. The practicality of this is limited by both the availability of designers at the university, and their technical knowledge levels. A lack of technical knowledge can be somewhat compensated for by having technically skilled students as teaching assistants, and letting them beta-test the instructions, noting difficult spots and possibly even suggesting workarounds.
- Revisions each semester. In the short run, this may be frustrating to students, but it will eventually lead to labs that run smoothly. To help avoid student frustration, open and immediate communication is key. As students see that the instructor is involved and available to answer their concerns, they'll implement any necessary changes and move on with subsequent steps.
- Bug bounties, with rewards such as a few extra points for students who properly report reproducible bugs. This approach requires dedication on the part of the instructor to check and verify the system daily, but has worked well in this author's experience.
- Proofreading is essential. While this seems easy, it is a skill that must be mastered. Two common approaches in education are having an outside entity read the instructions and printing out the document to proofread on paper [17]. A technically competent student can also be a great help here.

In summary, bugs and mistakes are part of life. To alleviate them, good supporting materials and IT support are critical to reduce load on the instructor. The mistakes introduced by our own errors are up to us to repair, but outside help is valuable here, too.

The scope of an exercise can be as difficult to determine as the tools used or the topic chosen. Much like with tools, the right level is "it depends…". It depends on the technical content and depth of the course; scope depends on the length of the assignment, a semester-long project vs. one of several lab assignments for example; whether labs are individual or group assignments; individual instruction style and class period length. Scope also depends on the infrastructure and support levels available to help students and instructors.

One interesting approach to scoping a lab is to use knowledge graphs. Knowledge graphs are diagrams of the relationships between an overarching topic and subtopics, and bear at least superficial resemblance to mind maps of topics [6]. Either of these tools can be useful to show relationships between topics and draw boundaries around topics that must be reinforced within a particular exercise.

Like bugs or tool choice, finding the right scope may be largely a matter of trial and error coupled with experience. Here again, a good student assistant can be invaluable. Adding mind maps and knowledge graphs to delineate the required vs. desirable topics can also be useful. Once appropriate scope is found, fortunately, it is one of the easier parts of a lab to transfer to subsequent exercises.

Student collusion and other forms of cheating are something all academics must deal with. Punishments may be dealt out after the fact, but this does not prevent the problem. There have been many studies on anti-cheating strategies [4], but not all apply to computing courses. Some strategies identified by other studies for cybersecurity labs include:

1. Creating customized, parameterized labs per-student. Thanks to various frameworks, this is not as difficult as it once was. Security Scenario Generator, FIND, and Tele-Lab represent some attempts to create individualized labs [21]. Two more recent, very promising approaches to generating individualized labs are Labtainers [12], which uses Docker containers within a virtual machine and Automatic Problem Generator (APG) [21]. When using Labtainers, the machines are customized via configuration file parameters. These parameters create machines which contain various levels of per-student watermarks, artifacts, and solutions, as well as introducing some randomness between student machines to avoid many forms of collusion, while still enabling the benefits that can come from student collaboration around a topic. The students' work products and output are bundled together and submitted to the instructor for evaluation. APG is conceptually similar, and utilizes Ansible, Python, Vagrant and Cyber Sandbox Creator [21]. For both of these, a student enters unique identifiers such as an email address that then is used as a random seed for other unique attributes. APG goes beyond Labtainers in that it automatically checks for similarities in student submissions that would indicate cheating by pupils. The only real problem with these two solutions is that they only generate Linux-based virtual machines. This is obviously enough for many or even the majority of lab exercises in cybersecurity, but there are both specialized distributions of Linux or BSD not covered by these tools, as well as Windows VMs that must be created for some scenarios.
2. Utilizing principles of authentic assessment can prevent many cheating issues. In order to be considered authentic assessment, a task must have some connection to real-life context [2]. In some cases, this can be accomplished by assigning tasks that the learner completes on their own IT systems, such as creating an encrypted volume, configuring backup solutions, or experimenting with wireless security protocols. The individual nature of each student's computing setup will lead to results that are unique for each learner. Besides preventing cheating, authentic assessment has been shown to lead to higher levels of student investment in the task [2]. When assigning tasks to be completed on the student's own machine is not possible, making tie-ins explicit between the lab environment and the outside world can be a substitute, but probably is better called Problem Based Learning, and doesn't necessarily fix any cheating issues, but still increases student investment in the task [23].
3. Perhaps the simplest technique for many types of labs is to require students to either create their own virtual machines or make their own accounts on pre-configured VMs. This strategy requires students to submit their own work. In some cases, such as with screenshots of output, a cheater could potentially modify the images to show their username rather than that of the student who actually did the work, but this may represent more effort than simply running the commands themselves. If students are instead required to submit the VM they've created or customized, this concern essentially vanishes. While not as robust as some of the other solutions, it is almost universally applicable without much additional effort.

In summary, preventing cheating is definitely a challenge, but by automating individualized machines, using authentic assessment principles, or simply requiring students to personalize accounts and other aspects of the assignment, the instructor can minimize the effort required to catch cheating.

The final issue with labs is that they tend to be more difficult to grade than other forms of assessment. Does grading effort, which is an instructor function, really affect the effectiveness of labs? Yes, because the effort and time expended by a teacher on grading cannot be put to developing further labs. There is no silver bullet, nor even a generalizable solution for all grading difficulties. At some level, as faculty, we must simply accept that the improved student outcomes enabled by labs are worth the tradeoff in our time. But there are solutions that can make that tradeoff more palatable. Automation of grading can greatly help, and solutions such as Labtainers and APG promise to reduce time required to grade assignments by minimizing and

automating materials that are submitted as well as detecting signs of collusion in the case of the latter.

Authentic assessment principles, as well as the concept of backward design (starting from the desired outcome and working back to learning activities) both indicate that labs are not the only way to enable student learning. In fact, for many topics at introductory levels, memorization, definition, and other items lower on Bloom's taxonomy are called for. This indicates assessment activities such as quizzes, exams, and group discussions, all of which may be more easily graded may be preferable in some cases. In other words, educators should not use a lab exercise to teach or assess every subject.

Some additional strategies to reduce the time needed to grade labs are to require very specific information, such as a screenshot that shows only the required inputs and the output that results, and phrasing questions in a way that leads the student to give specific answers, such as "Name ONE real-world threat which impacts YOUR chosen vulnerability", vs a more general question such as "name a vulnerability often found in software". Matching answers to these specific questions is much easier than reading an essay that results from students misunderstanding a question and trying to answer all possibilities.

In summary, while there is no single solution to the additional burdens of grading hands-on labs, strategies to reduce the time include using other forms of learning for some topics, using automated grading where possible, and asking questions that require specific answers tend to help.

IV. Resulting Guidelines

In this section, the guidelines above are summarized in list form for reference. These guidelines do not, and cannot, represent the last word in instructional design for cybersecurity labs, but they are theoretically sound, and practically useful in the author's experience.

1. Topic Choice and Technical Quality
    a. Follow existing frameworks such as the NCAE requirements and IEEE/ACM guidelines
    b. Work with instructional designers for theory-grounded writing
    c. Collaborate with other faculty to develop labs according to each faculty member's specialties
2. Student Engagement
    a. Make the topic as relevant to students as possible, applying authentic assessment principles can help with this
    b. Gamify aspects of exercises, being sure the finished lab aligns with curricular objectives
    c. Provide good supporting materials to give background and help students troubleshoot
    d. Identify milestones that must be completed before moving on, and when possible, give immediate feedback on those milestones
3. Task Alignment with Student and Course Level
    a. Be sure the lab fits in the right course within the curriculum
    b. Adapt labs develop by others for a certain level to local needs
    c. Initiate recall of prior learning and assess readiness with quizzes or pre-lab work
4. Instruction Detail Level
    a. Draw on established instructional design, and taxonomies such as Bloom's, to create questions that are more than simple yes or no answers
    b. Call out what should be in a good answer, and provide example output as needed
    c. Adapt validated labs from other instructors to local needs and platforms
5. Using The Right Tools
    a. Use tools that are used by industry, consult industry advisory boards for their choices
    b. Keep tools updated and actively seek new tools
6. Bugs and Errors
    a. Use good supporting materials and outside help and support resources to ameliorate bugs in platforms and software tools
    b. Make use of instructional designers and student workers to proofread and test exercises before issuing them to students
7. Scope
    a. Use knowledge graphs or mind maps to delineate required and desirable topics
    b. Utilize instructional designers and teaching assistants to beta test labs
    c. Once proper scope is established it can often be transferred between labs and even courses
8. Cheating and Collusion
    a. Require unique usernames, accounts and other customization that is difficult to fake
    b. Utilize principles of authentic assessment to individualize labs by having students perform them on their own machines
    c. Automate creation of customized machines with scripts and tool such as Labtainer or APG
9. Grading
    a. Use forms of learning and assessment other than labs when pedagogically appropriate
    b. Use automated grading where possible

c. Ask questions that require very specific, but not necessarily very detailed, answers

V. Limitations and Future research

Any list of guidelines to make labs better will be incomplete. There is no perfect list, but that should not stop faculty from trying to adopt best practices. Beyond simply adopting them, this paper represents a step much like [23] made to the engineering field, it is a contribution to the ongoing discussion about the role of labs in cybersecurity, and the best ways to develop them.

Further theoretical work needs to be done in applying instructional design principles to lab exercises. While many existing pedagogical designs have been developed for the humanities, some exist for technical fields. These can be used as-is or adapted for the specific needs of cybersecurity labs.

Practical advances in lab effectiveness will require testing labs to see which approaches work the best in given situations. This could be as simple as A/B testing, or more complex analyses.

VI. conclusion

Labs are vital to good cybersecurity education. There is no level of lecture or theory alone that is enough to teach a student the skills needed in industry. Whether the topics for the labs come from the IEEE/ACM guidelines, the CAE requirements, or are gleaned from discussions with industry advisory board members, they must be chosen carefully.

Once topics chosen, labs must be developed, updated, and tested. When this is done, and results applied to new iterations of the labs, continuous improvement is possible. Improving labs can lead to nothing more or less than better student outcomes.

It is time for cybersecurity programs to "take a comprehensive look at the laboratory experience". Experts in the field have agreed that labs are vital, let us as faculty make them maximally effective.

References

[1] L. W. Anderson & D. R. Krathwohl, A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. New York, USA, Longman, 2001.

[2] J. Biggs, & C. Tang, Teaching for Quality Learning at University (4th ed.). Berkshire, England: Open University Press, 2011.
DOI: https://doi.org/10.7275/ffb1-mm19

[3] Y. Cai & T. Arney. 2017. "Cybersecurity Should be Taught Top-Down and Case-Driven". In *Proceedings of the 18th Annual Conference on Information Technology Education* (SIGITE '17). Association for Computing Machinery, New York, NY, USA, 103–108. DOI: https://doi.org/10.1145/3125659.3125687

[4] G.J. Cizek & G.J. Cizek. "Detecting and Preventing Classroom Cheating: Promoting Integrity in Assessment". Corwin Press. 2003

[5] Clark Center Cybersecurity Labs. https://clark.center/home. 2022.

[6] Y. Deng, Z. Zeng, K. Jha, & D. Huang, "Problem-Based Cybersecurity Lab with Knowledge Graph as Guidance", *JAIT*, vol. 2, no. 2, pp. 55–61, Feb. 2022.

[7] DETER Project. https://deter-project.org/deterlab_education. 2022

[8] R. Dopplick. 2015. "Experiential cybersecurity learning". *ACM Inroads* 6, 2 (June 2015), 84. https://doi.org/10.1145/2743024

[9] L. Feisel and G. Peterson, "The challenge of the laboratory in engineering education," *J. Eng. Educ*, vol. 91 (4), pp. 367-368, 2002.

[10] G. Hugo, L. Rafael, & O. Francisco, "Cybersecurity teaching through gamification: Aligning training resources to our syllabus", *Res. Comput. Sci.*, 146 (2017), pp. 35-43

[11] IEEE Computer Society and ACM, "Cybersecurity curricula 2017: Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurty," 2017.

[12] C. Irvine, M. Thompson, & J. Khosalim, "Labtainers: A Framework for Parameterized Cybersecurity Labs Using Containers". *National Cyber Summit.* https://louis.uah.edu/cyber-summit/ncs2017/ncs2017papers/5 2017

[13] "New directions in laboratory instruction for engineering students," *J. Eng. Educ.*, vol. 58, pp. 191–195, Nov. 1967.

[14] Y. Kose, M. Ozer, M. Bastug, S. Varlioglu, O. Basibuyuk & H. P. Ponnakanti, "Developing Cybersecurity Workforce: Introducing CyberSec Labs for Industry Standard Cybersecurity Training," *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2021, pp. 716-721, doi: 10.1109/CSCI54926.2021.00184.

[15] E. Lai, "Motivation: A literature review", 2011. London, England. Pearson. http://images.pearsonassessments.com/images/tmrs/ Motivation_Review _final.pdf

[16] Y. Li, D. Nguyen, & M. Xie. 2017. EZSetup: A Novel Tool for Cybersecurity Practices Utilizing Cloud Resources. In *Proceedings of the 18th Annual Conference on Information Technology Education (SIGITE '17)*. Association for Computing Machinery, New York, NY, USA, 53–58. https://doi.org/10.1145/3125659.3125699

[17] J. Madraso. "Proofreading, the Skill We've Neglected to Teach", The Engilsh Journal. vol. 2 no. 2, pp. 32-41. Feb, 1993

[18] National Security Agency and Department of Homeland Security, "National Centers of Academic Excellence in Cyber Defense Education Program (CAE-CDE): Criteria for Measurement - Bachelor, Master, and Doctoral Level," 2020.

[19] D. Nunez, F. Moyano, A. Nieto, J. J. Ortega, I. Agudo-Ruiz, and J. López, "A Milestone-Driven Approach for Lab Assignments Evaluation in Information Security," in *International Conference on e-Learning 2014*, 2014

[20] J. Piaget, H. Gruber, and J. Vone`che, *The Essential Piaget*, ser. Harper colophon books. Basic Books, Incorporated, 1982. [Online]. Available: http://books.google.com/books?id=yRjeKgAACAAJ

[21] J. Vykopal, V. Švábenský, P. Seda, and P. Čeleda. 2022. Preventing Cheating in Hands-on Lab Assignments. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2022).* Association for Computing Machinery, New York, NY, USA, 78–84. https://doi.org/10.1145/3478431.3499420

[22] T. Yardley, S. Uludag, K. Nahrstedt & P. Sauer, "Developing a Smart Grid cybersecurity education platform and a preliminary assessment of its first application," *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, 2014, pp. 1-9, DOI: 10.1109/FIE.2014.7044273.

[23] Z. Zeng, Y. Deng, I. Hsiao, D. Huang & C. J. Chung, "Improving student learning performance in a virtual hands-on lab system in cybersecurity education," *2018 IEEE Frontiers in Education Conference (FIE)*, 2018, pp. 1-5, doi: 10.1109/FIE.2018.8658855.

[24] R. Zhang, C. Xu & M. Xie, "Powering Hands-on Cybersecurity Practices with Cloud Computing," *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, 2019, pp. 1-2, doi: 10.1109/ICNP.2019.8888060.

[25] S. M. Zvacek and M. T. Restivo, "Guidelines for effective online lab assignments: Contributions to the discussion," *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2018, pp. 1442-1446, DOI: 10.1109/EDUCON.2018.8363401.