Engaged Student Learning with Gamified Labs: A New Approach to Hardware Security Education

Robert Karam, Srinivas Katkoori, Mehran Mozaffari-Kermani
University of South Florida, Tampa, FL, USA
{rkaram, katkoori, mehran2}@usf.edu

Abstract—**Hardware security is an emerging field with far-ranging impacts on the design and implementation of the devices we use in our everyday lives – from wearable and implantable medical devices, to personal mobile devices, and even cloud devices powering the software services that drive our society forward. Practical, hands-on experience is vital to the training of students in this and other security-related fields. We are developing a new model for hardware security education using readily available, cost-efficient, off-the-shelf development boards, with hands-on experiments that offer new learning opportunities for students. Beyond this, we are experimenting with different pedagogical methods to improve student engagement. In particular, we aim to *gamify* a subset of the experiments and evaluate the impact on student engagement and learning. This work-in-progress paper describes our initial approach to the gamification of hardware security labs and reports on baseline results from our control study using a more traditional, non-gamified approach.**

I. Introduction

Hardware security is a nascent field that considers the security of the entire hardware development life cycle, including design, implementation, fabrication, deployment, and in-field lifetime. Numerous academic programs now feature one or more elective courses on this topic, including surveys of the field and experiential/hands-on courses, both as elective courses fulfilling degree requirements, as well as part of certificate programs. These courses introduce students to concepts like test and verification, supply chain security, and various hardware-level attacks and countermeasures. Hardware security primitives, including cryptographic hardware or structures like Physical Unclonable Functions (PUFs) and True Random Number Generators (TRNGs) may also be covered. Survey courses present material in a high-level, theoretic approach with limited practical or hands-on work. Some assignments require students to analyze datasets that contain premeasured values from victim devices, such as the power consumption over time during an encryption operation, and students are asked to analyze the data through various means. More recent efforts have sought to introduce laboratory experiments involving real hardware, further concretizing the theoretical underpinnings of the course material, and giving students an opportunity to address real-world issues like measurement noise or hardware/software interfacing.

We have developed a new hybrid lecture/laboratory course for teaching students the basics of hardware security with a hands-on approach. It is "hybrid" as students learn the theory during the lecture portion of the class session and later receive a demonstration of a particular technique that puts the theory into practice. In some instances, the lecture takes place one day, and the demonstration and hands-on lab activity take place the following class session. We have had success with this format in the two semesters in which it was offered. In particular, as a result of successfully completing this course, a student will be able to:

1) utilize hardware test and debug tools to evaluate the physical and side channel security of a computing platform against state-of-the-art hardware-based attacks
2) recognize and understand the trade-offs in area, power, delay, and security that exist for typical countermeasures
3) interpret experimental data and learn to draw meaningful conclusions from appropriate statistical measures
4) demonstrate proficiency in hardware and software tools for the design, development, and testing of hardware and embedded software
5) develop strategies for successful teamwork

These highlight the development of strong technical and professional skills including proficiency with software tools for computer-aided design (CAD) and a variety of programming languages (Python and C, in particular), as well as the use of hardware measurement and debug tools, all of which are transferable to other computer engineering subdomains.

We have worked to debug and refine the various course modules and associated experiments during the first semester it was offered. We began with data collection via pre- and post-surveys during the second semester, after the course had been more established and a larger cohort of students was registered. Now, in the current (third) semester offering, we are implementing a plan to investigate the impact of *gamification* on these student learning outcomes (SLOs). We hypothesize that game-based learning in a laboratory setting will improve student learning as measured through standard formative and summative assessments, compared to prior course offerings.

To test this hypothesis, we added certain game mechanics to the course. We expect that, with the integration of these game mechanics, students may be more likely to engage in the course, which will in turn have a positive impact on student learning of this complex subject area and increased proficiency in the languages and tools. In this paper, we will briefly overview the project, describe our approach to gamification, and discuss our evaluation plan.

This material is based upon work supported by the National Science Foundation under Grant No. DGE-1954259. Contact: rkaram@usf.edu

TABLE I
OVERVIEW OF THE HARDWARE SECURITY LABORATORY EXPERIMENTS AND RELEVANT LANGUAGES/TOOLS.

Lab	Name	Tools and Components	Badges
1	Physical Unclonable Functions (PUFs) Implementation and Evaluation	FPGA, Python	FPGA1, COM1, FILE1, STAT1, PLOT1
2	Deep Learning and Modeling Attacks on Arbiter PUFs	Python	ML1, FILE2, STAT2, PLOT2
3	Implementation and Evaluation of Pseudo- and True Randomness	FPGA, Verilog, Python	FPGA2, FILE3, STAT3, PLOT3
4	Introduction to Side Channel Analysis and Leakage Assessment	ChipWhisperer, Python, C	C1, COM2, STAT4
5	Correlation Power Analysis Attack and Countermeasure for AES	ChipWhisperer, Python, C	CRYPTO1, C2, PLOT4, STAT5
6	Timing Attacks and Countermeasures for Embedded Systems	ChipWhisperer, Python, C	ML2, CRYPTO2, C3, PLOT5
7	Hardware Trojan Design and Countermeasures in a Soft CPU	Verilog, RISC-V Assembly	HDL1, FPGA3, COM3
8	Security for the Internet of Things: Attack and Countermeasure for RSA	ChipWhisperer, Python, C	CRYPTO3, C4, STAT6
9	Post-Quantum Cryptographic Hardware and Embedded Systems	Verilog	HDL2, FPGA4, CRYPTO4

II. BACKGROUND

Gamification refers to introducing game-like features into non-game contexts [1]. For example, badges or awards can be given for reaching a milestone or completing a task during an experiment or demonstrating mastery of a certain technical skill. Gamification research at the university level is relatively new and coincides with the adoption and spread of video games [2]–[4]. A recent meta-analysis found a generally positive relationship between course module gamification and student engagement and learning [2]. Scholarship in teaching and learning suggests that gamification can result in higher performance in hands-on exercises, increased motivation, and enhanced engagement [5]–[7]. Gamification has been investigated in a number of contexts related to computing, including introductory computer programming classes [8], IT compliance training [9], and cybersecurity competitions [10]. A "skills" type lab course, particularly one where students learn to use benchtop laboratory equipment, are common targets for gamification [11]–[13]. In a traditional classroom setting, a "skills" course may be perceived by students as a "task list" of unrelated assignments they must complete for a grade [12], but alternative methods – such as gamification – may provide a more coherent framework for increasing student engagement. To our knowledge, this has not been studied in the context of a hardware laboratory course, particularly not an advanced undergraduate course like Hardware Security. This has motivated us to begin studying this topic and address this research gap in order to improve student learning in this critical area.

III. COURSE OVERVIEW

A. Course Description

Our course includes nine hands-on experiments, each involving software only, hardware only, or a combination of both hardware and software. Table I summarizes the topics covered. The topics are suitable for junior and senior-level computer engineering, electrical engineering, and computer science students who have previously taken hardware courses such as Computer Logic Design and Computer Architecture, as well as programming courses such as Data Structures. The topics expose students to many different facets of secure hardware design, including FPGA development, scripting languages like Python, and embedded software development in C/C++. Due to time limitations, certain complex hardware primitive functions, such as PUFs and TRNGs, are provided to students as pre-compiled configuration files (bitstreams). We utilize two development boards, the Digilent Cmod S7 (FPGA) and the ChipWhisperer (CW) Nano. The CW nano includes two microcontrollers, one that can be programmed in C/C++ (the "victim" or "target" board) and one that interfaces with a computer and can relay side channel information (like power consumption) from the target chip. Every lab has an associated in-class activity serving as a tutorial or walk-through for the related experiment. This helps tie the theory portion of the course with the specific implementation steps needed for the experimental portion of the assignment. We have previously had success with this format, and we have now added game mechanics in an effort to further enhance student engagement.

B. Approach to Gamification

We considered a number of potential game mechanics for the course, including race-against-the-clock, leaderboards, or writing a backstory to the labs. Ultimately, we decided to do so through badges and achievements. *Badges* represent the acquisition of certain technical skills that comprise portions of the labs and are earned through the normal course of completing an assignment. A student can earn an "A" on the assignment per the rubric, and acquire the badges with no additional work. *Achievements* are not tied to grades but rather focus on proficiency with a certain tool or technique, the use of which is not required for earning an "A" on the assignment.

1) Badges: Each lab involves several tasks, and completing each task requires a certain skill or technique. These skills are then built up over the course of the semester. To summarize, from Table I, the first lab utilizes FPGAs and Python, but the actual experiment involves programming an FPGA with a provided configuration file, gathering data from the board via serial communication, and processing and visualizing the data. The students learn to set up a serial communication link in Python, store data to a compressed file using NumPy (npz file), read from that file, and perform certain data processing and visualization tasks (computing certain metrics, basic statistics, plotting a histogram). These are all "Level 1" badges (denoted in the table by the "1" following the badge name). The second lab involves training a machine learning (ML) model (ML1) using data obtained from the first lab, with more involved data processing and visualization (FILE2, STAT2, PLOT2) which build on the techniques from Lab 1. The third lab involves more advanced FPGA topics – random number generation

(RNG) and postprocessing – reading and processing of binary files (FILE3), performing statistical hypothesis testing using a RNG test suite (STAT3) and generating multiple plots using the subfigure environment (PLOT3). Labs 4, 5, and 6 introduce badges for C programming, as we switch to the ChipWhisperer platform for power and timing side-channel analysis, building on the statistics and data visualization techniques from previous labs, and adding the topic of Cryptography. This approach continues through the final three labs, building on skills and techniques previously mastered in earlier labs. Not all lab *topics* build on each other, but the *skills* in the labs do. This approach may help to remind students how much they have learned over the course of the semester by categorizing the techniques as, e.g., "FPGA" or "STAT" badges.

2) Achievements: Every student who meets expectations (per the lab rubric) will automatically receive the "bronze" achievement for a particular badge. Taking it one step further, a student may achieve a "silver" or "gold" by utilizing certain advanced techniques or doing additional work beyond what is expected for the course. For example, when plotting the histogram (PLOT1 badge) it is expected that the student will include the standard plot features (title, axis labels, etc.). Adding certain annotations to the plot is not expected at this point, so students who do this will earn the "silver" achievement. Fitting a curve to the histogram (in this case, the data is expected to be normally distributed) is also not expected at this stage and would earn a "gold" achievement. Similarly, the way in which code is written can differentiate between different levels of achievement. For example, a basic for loop can be used to complete a certain task, but it may be more efficient to use list comprehension in a particular situation. This is noted in the lab document, and students who opt to use the more optimized approach will earn a higher tier achievement.

IV. Methodology

The project aims include 1) the development of a new hardware security course that stresses experiential learning, and 2) experiments with different pedagogical techniques in hardware security, which is ongoing at this time. The education research is conducted as an ABAB study, in which the course is taught two ways: 1) "A" semesters are comprised of a hybrid lecture/lab course, and 2) "B" semesters with a *gamified* hybrid lecture/lab course. Each semester, we administer a pre- and post-survey and gather metrics from assessments.

The pre-survey consists of basic demographic data, a self-assessment of computer engineering skills, and questions regarding students' preferences and habits with respect to video games. The demographic data helps us to evaluate/identify similarities between cohorts in different semesters. For the self-assessment, we define a *Hardware Skills Inventory* that asks students to rate their background/familiarity and competence in a wide variety of topics related to computer engineering. This list includes topics that we would expect upper-division students to have taken in prior coursework, such as digital design, reconfigurable computing, and computer architecture, electronic circuits. In addition, we include topics they may not yet have learned about (perhaps an elective, or studied outside of class as a hobby), such as secure coding, cryptography, and reverse engineering of hardware or software, among others. We then ask students to rate their own skill levels in using certain electronic bench equipment (oscilloscope, function generator, etc.) and certain programming languages (Python, C, Java, etc.) on a scale from 1 to 5, with 1 being no knowledge of or experience in the topic, and 5 being an expert on the topic. Finally, we ask about student's habits and preferences regarding video games – whether or not they play video games, what types of games, and how often.

As the course touches on a number of the topics listed in the Hardware Skills Inventory, we expect that the self-reported experience in those topics will increase in the post-survey compared to the pre-survey. We expect that the level of increase will be significantly higher in "B" semesters relative to "A" semesters due to the gamification. We are also interested in determining whether a relationship exists between the amount of improvement in the self-assessment in "B" semesters for students who routinely play video games compared to those who do not. However, this will require a larger sample size, since most students (in the first "A" semester) reported playing games (n=32) compared with students who reported not playing games at all (n=10).

V. Preliminary Results

The course has been offered twice, in Fall 2021 and again in Fall 2022. It is currently in its third iteration in Fall 2023. The first time it was offered was considered a trial run of the course content in which we addressed any bugs or any remaining hardware/software issues not found in prior testing. This first offering was also complicated by being a "hyflex" course, taught synchronously in-person and online, as this was during the COVID-19 pandemic. Our approach to this hyflex laboratory course was previously documented in the literature [14]. Thus, data collection began in Fall 2022, at which time 42 students were enrolled in the course. Another 40 students are enrolled this semester, with roughly the same demographics, enabling us to compare the impact of gamifying the course modules during the current semester. The initial cohort (Fall 2022) was comprised of 62% seniors, 29% Master's, 5% juniors, and 5% PhD students. Most undergraduate and Master's students were pursuing a degree in Computer Engineering (62%), with the majority of other students studying Computer Science (26%) and Electrical Engineering (12%). About 80% of students identified as male, and 20% as female. The cohort (Fall 2023) is comprised of 30 students. Of these, 67% are seniors and 33% are Master's students. Undergraduate and Master's students pursuing a Computer Engineering degree comprise 50% of the class, with the remainder pursuing a degree in Computer Science (50%). For the first cohort, survey results from the self-assessment indicated a statistically significant increase in students' confidence in topics and skills covered in the class, as expected. Further analysis of these results is available in [15]. Results from the self-assessment data in the Hardware Skills Inventory from the gamified course are not yet available but will be in the coming months. In the future, we hope to expand the study to other universities to gather additional sample data, and potentially evaluate the impact of gamification in larger, less specialized hardware lab courses such as Computer Logic Design Lab.

VI. Conclusion

In this paper, we have described an ongoing education research project in which we evaluate the impact of gamification in a hands-on hardware course. This is an advanced course targeting senior undergraduates or graduate students studying Computer Engineering, Electrical Engineering, or Computer Science. The standard course is a "hybrid" lecture/lab, which we have gamified through the addition of badges and achievements. We expect that our approach to gamifying this hardware course will improve student learning due to increased student engagement with the course material. We are collecting pre and post surveys which include demographic information, a hardware skills inventory we have developed, and information on student's habits with respect to video games. The initial self-assessment survey has some interesting findings, and we look forward to continuing the work and expanding the project in the future.

References

[1] D. D. Burkey, M. D. D. Anastasio, and A. Suresh, "Improving student attitudes toward the capstone laboratory course using gamification," *Age*, vol. 23, p. 1, 2013.

[2] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work?-a literature review of empirical studies on gamification." in *HICSS*, vol. 14, no. 2014, 2014, pp. 3025–3034.

[3] S. de Sousa Borges, V. H. Durelli, H. M. Reis, and S. Isotani, "A systematic mapping on gamification applied to education," in *Proceedings of the 29th annual ACM symposium on applied computing*. ACM, 2014, pp. 216–222.

[4] D. Dicheva, C. Dichev, G. Agre, and G. Angelova, "Gamification in education: A systematic mapping study," *Educational Technology & Society*, vol. 18, no. 3, pp. 75–88, 2015.

[5] A. Domínguez, J. Saenz-de Navarrete, L. De-Marcos, L. Fernández-Sanz, C. Pagés, and J.-J. Martínez-Herráiz, "Gamifying learning experiences: Practical implications and outcomes," *Computers & education*, vol. 63, pp. 380–392, 2013.

[6] R. Garris, R. Ahlers, and J. E. Driskell, "Games, motivation, and learning: A research and practice model," *Simulation & gaming*, vol. 33, no. 4, pp. 441–467, 2002.

[7] D. Giannetto, J. Chao, and A. Fontana, "Gamification in a social learning environment," in *Proceedings of the Informing Science and Information Technology Education Conference*. Informing Science Institute, 2013, pp. 195–207.

[8] F. Panagiotis, M. Theodoros, R. Leinfellner, and R. Yasmine, "Climbing up the leaderboard: An empirical study of applying gamification techniques to a computer programming class," *Electronic Journal of e-learning*, vol. 14, no. 2, pp. 94–110, 2016.

[9] R. J. Baxter, D. K. Holderness Jr, and D. A. Wood, "Applying basic gamification techniques to it compliance training: Evidence from the lab and field," *Journal of information systems*, vol. 30, no. 3, pp. 119–133, 2015.

[10] G. Fink, D. Best, D. Manz, V. Popovsky, and B. Endicott-Popovsky, "Gamification for measuring cyber security situational awareness," in *International Conference on Augmented Cognition*. Springer, 2013, pp. 656–665.

[11] K. Fleischman and E. Ariel, "Gamification in science education: Gamifying learning of microscopic processes in the laboratory," *Contemporary Educational Technology*, vol. 7, no. 2, pp. 138–159, 2016.

[12] K. Drace, "Gamification of the laboratory experience to encourage student engagement," *Journal of Microbiology & Biology Education: JMBE*, vol. 14, no. 2, p. 273, 2013.

[13] M. T. Bonde, G. Makransky, J. Wandall, M. V. Larsen, M. Morsing, H. Jarmer, and M. O. Sommer, "Improving biotech education through gamified laboratory simulations," *Nature biotechnology*, vol. 32, no. 7, p. 694, 2014.

[14] R. A. Karam, S. Katkoori, and M. M. Kermani, "Work-in-progress: Hyflex hands-on hardware security education during covid-19," in *2022 IEEE World Engineering Education Conference (EDUNINE)*. IEEE, 2022, pp. 1–4.

[15] B. Olney, M. A. F. Amador, and R. Karam, "Development of course modules in python for hardware security education," in *SoutheastCon 2023*. IEEE, 2023, pp. 912–919.