Root the (Ballot) Box: Designing Security Engineering Courses with E-Voting

Tushar M. Jois
City College of New York
New York, NY, USA
tjois@ccny.cuny.edu

Atheer Almogbil
Johns Hopkins University
Baltimore, MD, USA
aalmogb1@jhu.edu

Logan Kostick
Johns Hopkins University
Baltimore, MD, USA
lkostic1@jhu.edu

ABSTRACT

Security courses often focus on individual attacks and defenses and lack practical experience with secure system design. Consequently, such courses rarely address the societal implications of security breaches. We propose an inductive teaching approach to address these challenges in teaching security engineering. To this end, we designed a course that focuses on electronic voting (e-voting) as a practical application of security engineering. Our course integrates relevant technical content through experiential learning, adversarial thinking skills through a group project, and societal factors with a student debate. Our approach has seen success in engaging students in both distance-learning and in-person teaching while preparing them for real-world security challenges.

1 BACKGROUND & RELATED WORK

Computer security courses teach a standard set of attacks and defenses, but these are often too abstract for students to understand. There has been some progress in this regard, with materials like SEED [3] providing hands-on learning activities on the attacks and defenses of computer security. These activities, however, do not encompass security *engineering*, the design and analysis of secure systems. A key component of security engineering is *adversarial thinking*, in which students attempt to identify potential threat actors in a system and their capabilities. Experiential introductions to security concepts, while important, miss this requirement. Moreover, security engineering necessitates understanding the *social context* of the system and the impacts of a security breach. These impacts are personal; all users – including students – are at risk.

Prior work contains examples of curricula that incorporate real-world applications where students feel like their learning is immediately relevant (e.g., [2, 5, 7, 10]). What is missing, however, is a *concrete* problem that encourages adversarial thinking while being realistic enough to motivate course material. Electronic voting (e-voting) system design is a security engineering problem that captures both dimensions. E-voting does away with the paper ballots traditionally associated with elections, replacing both vote casting and tabulation with a computer. Early systems were vulnerable to a litany of attacks, generating academic interest in analysis [6]. At the same time, the problem has major political implications [4].

We propose a new type of course that utilizes e-voting to teach security engineering, combining technical content, adversarial exercises, and societal connections to successfully prepare students for the future. As with a traditional course, we aim to give students an understanding of different attacks and defenses. Our course also has students engage with the social implications of computer security, as well as give them experience in how security analyses are done in practice. We do this by integrating two inductive teaching approaches [9]: project-based learning to build adversarial thinking, and case-based teaching to contextualize the course material.

Contributions. To our knowledge, this work represents the first investigation into security engineering education through such a method. Our contributions are as follows:

- We describe the content for a new type of security engineering course that utilizes e-voting as an application.
- We provide our experiences and lessons learned running the proposed course via both distance and in-person learning.
- We discuss future directions for this curriculum design.

2 APPROACH

In developing the course, we adhered to the following design principles: the course should (1) contain the required technical content for a security course, (2) build adversarial thinking skills, and (3) incorporate societal dimensions. Additionally, since we initially designed the course during the COVID-19 pandemic, we wanted the course flexible enough to support distance learning if required. As such, we ensured that none of these proposed course elements require a specific course delivery method.

Technical content. We aim to provide the same technical content as an equivalent traditional cybersecurity course. We use the SEED Labs [3], a suite of hands-on computer security labs, to provide this. The labs cover software, network, web, systems security, and an introduction to cryptography. Because the labs are performed on a virtual machine, the students (and instructors) do not have to worry about damaging a production system and can ensure all coursework is performed in a standard environment.

These labs have been successfully used in prior versions of the security and privacy course at Johns Hopkins University. But, since our new course must accommodate additional security engineering-related material, we must get through this technical content quicker than a usual course. To mitigate this increased pace, we add more support and guidance for students in the form of recitation sections. During these sections, teaching assistants guide students through the finer details of the labs and assist with any issues.

Adversarial thinking. To build adversarial thinking skills, we utilize project-based learning [9], in the form of a group project that allows students to creatively apply their learning to a problem.

First, in the *blue team* phase, groups of 2–3 students design an e-voting machine that is *intentionally* vulnerable to several of the attacks they learned in the course. Then, in the *red team* phase, each group receives a different voting machine and has to identify and exploit as many vulnerabilities as possible. This approach is similar to "root-the-box" or "capture-the-flag"-type games (e.g., [8]), but we add a voting machine application component that motivates the

assignment. At the end of the course, students present their (blue team) vulnerable machine design and the vulnerabilities identified in the given (red team) machine to the rest of the class.

We provide starter code to the students for the blue team phase that they may use for the project. Our starter implementation – a networked e-voting system with a web application interface – was set up to match the technical themes of the SEED Labs. This structure allows students to apply all of their knowledge in a structured environment. The friendly competition inherently creates adversarial thinking skills, with students taking the perspective of a malicious actor in developing the vulnerable voting machine. The students must understand what makes a system vulnerable before they are able to inject intentional vulnerabilities. Moreover, the project gives students practice in penetration testing, a highly sought-after skill in industry [1] where this blue-team–red-team approach to security engineering is common.

Societal dimensions. We also want students to get out of classroom experience and see how their security work impacts the real world. To this end, we split the class into two teams and had them debate and discuss the merits of e-voting and its impact on society. Because the debate is straight out of the headlines, students may have already been exposed to (and perhaps interested in) these topics. We introduce e-voting's impact on society through a lecture on its history, from the 2000 U.S. election to present. Then, we have each team collaborate on the points they wish to present and defend. Since e-voting has elicited strong opinions in the media, we encourage students to perform research using trustworthy sources and defend their claims with evidence. The debate is then held in class, with students voting on which team did the best.

3 INITIAL PILOTS

We first implemented our course as the introductory security & privacy course at Johns Hopkins University in Fall 2020. Due to the COVID-19 pandemic, we taught this course via distance learning. Students responded positively to the course, despite the challenges inherent to online learning, finding that the course structure kept them engaged throughout the semester. We also ran the course in Fall 2021 and Fall 2022 but in an in-person setting instead. Because of the flexibility of our curriculum design, the course translated well to in-person learning, and we received similar positive feedback.

We believe that the success of the course stems from its focus on project-based learning and case-based teaching, which allows students to take charge of their own learning [9]. Moreover, the collaborative nature of the project and debate encourages students to interact with each other. This type of collaboration is natural and common in-person, but can be challenging to foster online.

We aim to have other courses to apply our approach to a security engineering curriculum. To this end, all of our course materials are available at https://github.com/spacelab-ccny/rtbb.

4 CONCLUSION & FUTURE WORK

We present a new type of security engineering course centered around e-voting. It has the same technical content as a traditional course but also builds adversarial thinking skills in students and incorporates societal connections into its teaching. We describe its structure and provide our thoughts on running the course in distance and in-person learning environments.

We will continue to iterate on our course design, incorporating the lessons we learned over time. For example, our course could integrate more security engineering themes, such as the uniquely human elements of social engineering and physical security. To add this content, we plan to add a simulated "election day," in which some students act as poll workers and voters while others attempt to subvert the election. Students could then run their technical exploits on physical machines after bypassing human controls. This would crystallize the notion of an insider threat while being interactive and engaging. This could also be done in a distance learning setting, with local voting clients and a remote tabulation server. Additionally, our evidence so far is anecdotal. We desire to run a more quantitative study in the future, collecting feedback and running focus groups to review the design of the course.

ACKNOWLEDGMENTS

The authors would like to acknowledge support from the NSF under award 1955172. The views contained herein are those of the authors and should not be interpreted as those of the NSF. The authors would also like to thank Avi Rubin and Max Zinkus for their contributions to the development of the course.

REFERENCES

[1] Miriam E Armstrong, Keith S Jones, Akbar Siami Namin, and David C Newton. 2018. What vulnerability assessment and management cybersecurity professionals think their future colleagues need to know. In *SIGCSE '18*.

[2] Ruedi Arnold, Marc Langheinrich, and Werner Hartmann. 2007. InfoTraffic: Teaching Important Concepts of Computer Science and Math through Real-World Examples. *SIGCSE Bulletin* 39, 1 (mar 2007), 105–109.

[3] Wenliang Du and Ronghua Wang. 2008. SEED: A suite of instructional laboratories for computer security education. *Journal on Educational Resources in Computing (JERIC)* 8, 1 (2008), 1–24.

[4] Andrew C Eggers, Haritz Garro, and Justin Grimmer. 2021. No evidence for systematic voter fraud: A guide to statistical claims about the 2020 election. *Proceedings of the National Academy of Sciences* 118, 45 (2021), e2103619118.

[5] Mounib Khanafer and Tushar M. Jois. 2023. Towards Application-Driven IoT Education. In *EDUCON '23*. IEEE.

[6] Tadayoshi Kohno, Adam Stubblefield, Aviel D Rubin, and Dan S Wallach. 2004. Analysis of an electronic voting system. In *IEEE Symposium on Security & Privacy.*

[7] Daniel E. Krutz, Samuel A. Malachowsky, and Thomas Reichlmayr. 2014. Using a Real World Project in a Software Testing Course. In *SIGCSE '14*. 49–54.

[8] Jelena Mirkovic and Peter AH Peterson. 2014. Class Capture-the-Flag Exercises. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education.*

[9] Michael J Prince and Richard M Felder. 2006. Inductive teaching and learning methods: Definitions, comparisons, and research bases. *Journal of engineering education* 95, 2 (2006), 123–138.

[10] Sarah J. Van Wart, Sepehr Vakil, and Tapan S. Parikh. 2014. Apps for Social Justice: Motivating Computer Science Learning with Design and Real-World Problem Solving. In *ITiCSE '14*. 123–128.