This is Not a Game: Early Observations on Using Alternate Reality Games for Teaching Security Concepts to First-Year Undergraduates

Mark Gondree
Naval Postgraduate School

Zachary N. J. Peterson
Cal Poly, San Luis Obispo

Abstract

We describe a novel approach to delivering an introductory computer science course for first-year undergraduates, using computer security topics to explore core CS concepts. Our course is a first attempt at merging aspects of capture the flag-style challenges, puzzle-based learning, and alternate reality games (ARGs), with the goal of improving student engagement, increasing awareness of security as a discipline and professional opportunity, and providing context for the social relevance of security to our lives. Our challenges synthesize hands-on problem-solving, immediate feedback on incremental progress, scaffolded learning, a loosely-connective narrative, and a sense of intrigue to draw students into active engagement with course material. In this paper, we motivate the use of ARG characteristics to connect course tasks, we discuss our goals, course design, and a mixed-method evaluation of our objectives (using reflective journaling, cognitive interviews, and pre- and post-surveys using an adaptation of the Computer Attitude Scale instrument), and summarize our preliminary findings.

1 Introduction

Cybersecurity education programs are finding it difficult to meet new and projected workforce demands [9]. While specialized programs and professional certificates may be an appropriate short-term strategy to satisfy the immediate need for qualified professionals [14], future demand needs to be addressed by curbing more systemic trends affecting CS and computer security education. Indeed, these feelings are reflected in the US CyberSpace Policy Review, a 2009 report commissioned by President Obama (and a sentiment he repeated more recently at the 2015 Summit on Cybersecurity and Consumer Protection), which recommends the expansion of university educational programs for digital safety, ethics, and security.

Unfortunately, computer security is still underrepresented in many undergraduate CS curricula and curricular standards. As an example, the ACM Computer Science Curricula 2013 guidelines recommend a minimum of three contact hours of information assurance and security *throughout an entire undergraduate CS program* [4]. While security issues permeate nearly all aspects of our day-to-day lives, the technical complexities and mundane subtleties of computer security do not easily lend themselves to a lower-division college curricula [7, 22]. As a result, security is often left as upper-division electives, precluding all but the most advanced and self-selected students from enrolling, and limiting the time students have to explore the field. Thus, a significant challenge in computer security pedagogy is in identifying the appropriate topics and exercises that teach security concepts to a wider, lower-division audience (particularly students with no CS background) and that foster curiosity in security as a discipline, as well as develop a broader interest in CS and other STEM disciplines.

Similar deficiencies have been identified in the computer science curriculum, more broadly. There has been a recent acknowledgement by the computer science pedagogical community that the traditional approach to introductory CS courses—often a two or three course sequence in programming and architecture—may be failing to educate and retain students in the discipline, particularly students from populations underrepresented in CS. Common shortcomings have been identified as: (i) communicating the lack of real-world context and social relevance for the discipline, (ii) the reenforcement of preconceived notions of CS as a solitary and isolating pursuit, and (iii) limitations on creativity and individual expression in overly-constrained coursework [12, 13, 20, 21]. In fact, research shows that early exposure to engaging, social, and meaningful coursework is one of the strongest factors in CS undergraduate student retention [5].

Some universities have sought to re-design their introductory CS courseware, employing new methods and objectives to address these deficiencies. In particular,

CPE123 is a reinvented introduction to computer science offered at Cal Poly, San Luis Obispo [20, 21]. In this course, students engage with core CS principles through constructivist, open-ended assignments. The course is offered in a variety of "flavors" (including robotics, computational art, video game design, and digital music production), allowing students to explore CS concepts through pre-existing interests. Results from CPE123 and courses like it at other universities have shown decreased attrition rates, increased student performance in subsequent CS courses, and demonstrate a higher retention rates among minorities in CS.

In this paper, we report on our initial observations in developing a new and experimental "flavor" of CPE123 designed to engage first-year, undergraduate students in CS and computer security principles, through a series of thematically-connected, game-like coursework. The broad objectives of the course are: to attract undergraduates that have no prior experience in CS or security, using authentic problems demonstrating the relevance of security to the world around them; to highlight the role of computers in solving problems; and to challenge students to think "adversarially." Using puzzles, game play, and team-based projects, we have developed a collection of accessible and engaging exercises that explore security topics typically reserved for advanced computer science majors, despite their wide relevance to the public.

We present this content in a novel way, using techniques drawn from alternate-reality games (ARGs). ARGs are story-driven, trans-media art, designed to encourage players to collaboratively uncover and interpret fragments of a story, distributed across multiple forms of media, using the "real world" as its platform. First used for promotional and marketing purposes, ARGs have garnered attention in educational settings [8, 19, 26, 29], and have been shown to have valuable pedagogical characteristics, *e.g.*, being social and inclusive, stimulating counterfactual thinking, and supporting student autonomy—many of the same characteristics observed to be lacking in current CS curricula.

Our preliminary results show our course to be quite successful in achieving these goals. Evaluation based on pre- and post-surveys, reflective journal prompts, and cognitive interviews reveals that our approach improves student attitudes and beliefs about computer security as a discipline, their self-perception of their success in the field, and generally improves their awareness and behaviors with respect to security and privacy concepts.

2 Related Work

The use of games in communicating computer security concepts is not new. Indeed, "capture the flag" (CTF) events—a catch-all term used to describe a variety of full and partial simulation challenges—have exploded in both number and diversity. In addition to traditional red/blue team attack and defense exercises, there are CTF events that are purely defensive (*e.g.*, the National Collegiate Cyber Defense Competition), that focus on puzzles in a specific security sub-discipline (*e.g.*, Matasano Crypto Challenges [3]), and that target participation by high school students (*e.g.*, picoCTF [10]). Intercollegiate leagues for CTF games have even been designed, with brackets, divisions and sanctioned competitions [1, 2]. Some are even using competitive CTF-style game play directly in the classroom [28, 31].

We differentiate our work from classroom-based CTFs in many ways. Students do not engage in attack-defend scenarios, or competitively play against one another. Our game is long-form, taking weeks rather than days to play. While our approach appears similar in many respects to "Jeopardy-style" CTF challenges and puzzle-based learning [15], it deviates in its use of intrigue-building narratives, reinforcement through incremental progress, and scaffolded learning. Our ARG challenges are best compared to those of picoCTF [10], however, in their use of these features for pedagogical effect and evaluation.

3 Course Design & Methodology

At high level, our course aims to: (1) teach core CS principles, providing a foundation and context for more traditional, introductory CS coursework, (2) explore those core concepts through the lens of computer security, inspiring students to think adversarially about complex systems, and (3) remove many barriers commonly attributed to poor CS engagement, including isolation and exclusion, a lack of social relevance, and limitations on creativity. The reasons we elect to use a thematic, story-driven approach (as used in alternate reality games) are manifold: ARGs have been shown to be collaborative and inclusive [29]; to inspire counterfactual thinking [18]; to maintain engagement [17, 30]; and to provide an authentic context and purpose for presented material, both online and in the real world [8]. Specific goals of the course include:

Be inclusive and supportive of students new to CS. Our course provides students with a diverse range of challenges, enabling students to pick a start point appropriate for their own level of proficiency. As students increase their comfort level and skill set, they may independently increase the difficulty of challenges that they engage. By doing so, students develop a sense of autonomy and independent success in a new discipline. Further, our exercises feature a variety of puzzle types (requiring a combination of analytical, physical, and creative abilities to complete), providing for an opportunity to engage varying learning styles and skill development.

Maintain engagement. The course, offered as an ARG, maintains student engagement through an intriguing narrative and measurable progress, as well as provides a clear sense of purpose and shared experience among players. Students are empowered to influence outcomes and increase their own stake in the game. In the long term, our ARG is designed to develop intrinsic motivators, encouraging lifelong learning in computer science and security.

Be social. Our course is played and "won," not by individuals, but by teams working collaboratively. Creating teams helps to build a sense of community and camaraderie, providing an opportunity for students new to CS to feel supported by, not competitive with, their peers. Open dialogue and group decision-making create a space to demonstrate a democratic process where compromise, rather than competition, is critical to decision-making. We deliberately build-in incentives that encourage players to directly interact with their peers, collaboratively building useful skills, and helping one another critically think about and, ultimately, solve the challenges.

Course materials further leverage the counterfactual thinking required by ARGs [18], and strategic games more generally [6], to prompt students to engage in complex reasoning and computational thinking. By formulating problems, making abstractions, and expressing solutions in ways that can be satisfied computationally, students naturally draw upon concepts fundamental to computer science. Games provide a layer of abstraction, allowing students to conceptualize programmatic solutions to problems without necessitating prior programming skills. Further, the inherent social dynamics of ARGs encourage players to understand and interpret rules, and help others understand and apply those rules. By combing these ideas, our course has the potential to encourage students to learn to think through a language of computation, and then use it to articulate computational concepts with their peers. Using ARG techniques, our course attempts to:

Develop "adversarial thinking." ARGs provide unique opportunities to engage in the types of thinking beneficial to security researchers and practitioners. Namely, they inspire a counterfactual ("what-if") approach to problem solving, allowing students to explore the ways systems are intended to operate and, perhaps more interestingly, how they may fail. They are also supportive of thinking adversarially about a system. Exploring the ethical and moral boundaries of the use of computer systems may cause discomfort in students that consider themselves strictly law-abiding or professionally responsible. ARGs use a fictional narrative and role-playing as freedoms for thinking about a complex system from multiple perspectives without penalty.

Be relevant. ARGs can create an authentic and relatable context for exploring basic security principles and practices. ARGs adhere to a "this is not a game" philosophy [33], where students engage with the game in their own "reality," not in an isolated arena, such as those provided by capture the flag environments. For example, ARGs often use familiar and socially relevant technologies (*e.g.* SMS, Twitter, YouTube) as part of the storytelling. Using familiar technologies causes students to explore their understanding of those technologies in real life, affecting behavioral changes in how they use them (*e.g.*, strengthening password selections, weighing the consequences of posting embarrassing photos online).

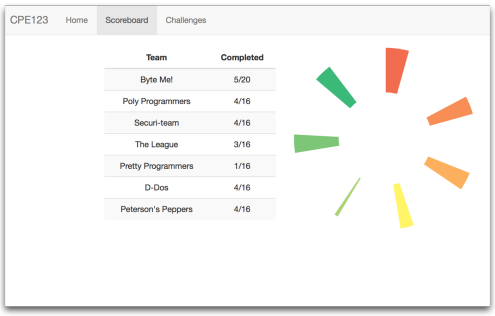
Reflect scientific accuracy. Of primary importance is ensuring that puzzles, behaviors, and relationships represented in the game reflect scientifically- and technically-accurate phenomena. Given the omnipresence of security terminology in society (passwords, hackers, malware, *etc.*), ARGs let players see these as terms in a meaningful context. Since the game's narrative takes place in "our reality," it can be suggestive of the true roles and limits of these technologies in the real world.

3.1 Course Organization & Delivery

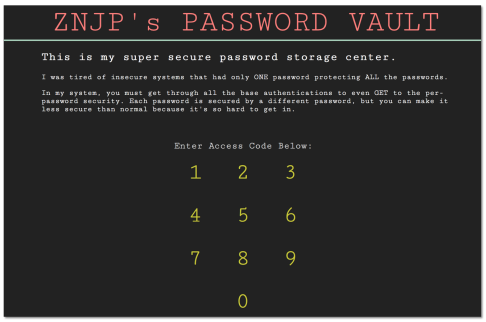
The course is organized into seven modules delivered over ten weeks. Each module explores a different security topic, supported by a set of core CS principles. Every module (or *level*) is divided into incrementally difficult exercises (or *challenges*) to scaffold instruction. Each challenge is connected thematically to security and objectively to the module's target objectives.

Students are organized into groups of four, where team members work together throughout the quarter. Our incoming CS students are fairly typical of national trends in that they reflect a broad range of prior computer science and programming experiences. Students range from those with top CS Advanced Placement scores, to those with prior programming experience from clubs or online courses, to those motivated to pursue the subject but with no prior experience and demonstrating low computer comfort. This vast differential in CS experience can pose a challenge in the classroom [12, 27]. Our strategy is to identify and evenly distribute experienced students amongst the teams, to achieve balance and create an inclusive environment. We conducted a brief, online survey on the first day of class, asking students to identify their confidence using a computer, to describe their experience in computer programming, and to describe any experience with computer security (casual interest, participation in CTFs, *etc.*).

During the course, students are tasked with solving all challenges within a level as a group. Modules are



(a) The scoreboard used to track group progress in solving challenges within a level.



(b) The first stage of the authentication challenge, requiring students to guess a single digit password.

Figure 1: Two screen captures from the CPE123 gaming infrastructure.

designed to be solved using a variety of approaches, including computer programing (*e.g.* port scanning for hidden web services), social engineering (*e.g.* fooling the instructor into entering a flag into a student-designed phishing website), and physical security (*e.g.* picking a lock, used on a locker containing a cell phone used for two-factor authentication).

While some ARGs blur the line between where the game ends and the "real world" begins, we endeavor to create a clearly-marked safe space in which students can explore concrete security concepts without repercussion. Our narrative is more like a campus treasure-hunt, where the boundary of the game is always clear and the freedom to play is unambiguous. We contend that if these lines are blurred, the ambiguity may lead students to discomfort when differentiating these spaces, *e.g.* when permission to attack is unclear or if attacking peers is acceptable. In contrast, our narratives always include permission and lead students to only attack systems clearly controlled by the instructor. Further, our approach provides many opportunities for discussions throughout the quarter on topics related to ethics, campus policies, responsible disclosure, and state and federal laws regarding hacking.

A live scoreboard allows teams to submit solutions (flags) and provides immediate, albeit coarse, feedback on every groups' progress (see Figure 1a). Rather than institute a point-award system tied directly to grades, the scoreboard is informational and tracks the progress of teams, and the class as a whole, toward solving challenges. The goal is to inspire friendly team competition and motivate students to engage in the material. When every team has solved all challenges in a level, the class is allowed to progress to the next level's set of exercises.

In addition to technical exercises, at the end of each module students are asked to reflect on their individual experiences, motivated by a prompt. These responses serve as a valuable pedagogical tool for identifying technical misconceptions, social problems within the group, degenerative cases in the challenges, and bugs in the infrastructure. Capturing responses weekly allowed us to tweak course content, infrastructure, and rules to improve gameplay and content delivery. Further, the journals yield valuable insight in measuring and evaluating classroom and project objectives (see Section 4.1).

3.2 Example Level: Authentication

In one of the early modules on identity and authentication, students are led to the instructor's personal password manager. Groups are challenged to pen-test this custom, web-based application, featuring obviously shoddy protection mechanisms: students can see a graphical interface for entering a single digit passcode (see Figure 1b). It is immediately obvious to students that they must correctly guess the passcode to unlock the next challenge. Because the passcode is a single digit, students can effectively guess it "by hand." Once solved, a flag is revealed and can be submitted to the scoreboard, unlocking the next challenge. The next challenge presents a similar interface, but requires a three digit code. As before, attempts may be entered using a JavaScript-driven keyboard or by recognizing that attempts are URL-encoded as part of the HTTP GET operation (*e.g.* http://cpe123/level1/stage2.php?pass=123) and are more efficiently solved by writing code to issue GET requests that exhaust the search space. As instructional scaffolding, students build on their prior work (incrementally modifying it to satisfy new restrictions and add features) and can test their programs against stages with previously cracked passwords. The final three challenges include cracking: a password chosen from a dictionary; a weak, four-word passphrase, where each word can be solved independently and in parallel by the group; and a random six-character password, where failed log-in attempts generate a debug message revealing the SHA1 hash of the target passcode. The final stage unlocks cre-

dentials for a Google account and a QR code revealing the GPS location of the mobile phone to which two-factor authentication codes are sent. Once found, students capture the flag stored in a Google Doc.

While these challenges are (intentionally) artificial, implementing a programmatic method of interacting with a web-server is authentic, as are the core mechanics of cracking passwords. These are tied to a playful and engaging puzzle-based narrative, to leverage student experiences to build new understandings. Abstractly, students develop intuitions that are at the core of complex security topics—like hardness amplification, password entropy, and the limits of password authentication—while avoiding much of the prerequisite knowledge required to explore these topics fully. These intuitions are built by engaging with and using a variety of fundamental CS concepts: programatic constructs like loops, conditionals, and logical operators; the use of an application program interface; client-server interactions; and data representation and encoding (*e.g.*, hex, base64, URL-encoding).

4 Course Assessment

We offered an inaugural version of the course in Fall 2014 with 29 students (24 male, 5 female) enrolled. Of these, 28 were "traditional" freshmen (recent high school graduates, 17–19 years old), and one was a transfer student with Junior standing but no formal CS coursework. Our pre-course survey revealed that no students claimed any prior experience with security (*e.g.* via clubs, CTFs, or independent inquiry) but for many it was a topic that piqued their interest.

During design and before enrollment, we partnered with an educational evaluator to guide assessment. Our course assessment methodology consists of a mixed-methods approach, including: surveys, reflective journal prompts, student interviews, and when possible, assessments embedded within the game itself. Our evaluation protocol and all assessment tools and strategies were approved by our campus institutional review board. This preliminary evaluation begins to address our course's goals identified in Section 3, including: (i) the use of game-based learning to engage first-year students in computer science and computer security, (ii) boosting confidence in learning and attitudes toward success in computer security, and (iii) improving students' awareness and behaviors about privacy and security.

4.1 Journal Observations

Through the use of reflective exercises—those that encourage students to make new connections between a variety of experiences through metacognition—students demonstrate better performance, greater retention, and stronger engagement with course material. Indeed, relfective exercises have grown in popularity across many STEM disciplines [23], including computer science [16]. Following this practice, at the end of each week we tasked students to reflect on specific aspects of their course experience, directed by a prompt [1]. Students were provided a private wiki space on the course website (accessible only to them, the instructor, and the evaluator) to use as a journal. Students were also welcomed to remark on any issues they considered relevant to course material, delivery, instruction modality (*e.g.* frustration with the game infrastructure), and group dynamics.

These open-ended journal responses were analyzed using the constant comparative method [32]. We used open and axial coding to generate descriptive categories from the data. Purposeful response samples were chosen to illustrate key findings from the data.

Game Play. Overall, our results show students were positively disposed to class exercises, and its narrative helped increase engagement and participation. Journals reflect that students felt the labs were "fun exercises" and that the class "wouldn't be as fun" without them. Students attested that their curiosities in the progress of other teams compelled them to work harder and be more engaged. Some students remarked that the "cloak and dagger" themes drew them into making connections to real-world computer security issues (*e.g.* the Target credit card breach), and characterized their intrigue by discussing the "problem solving" encouraged in these activities. The game play also fostered curiosity in the discipline, broadly. Students expressed "needing to continue learning," "wanting to stay in the major" and an acknowledgement that these activities "barely scratched the surface" of all that they need to know, opening up an "entirely new world." Additionally, at least two students wrote about their interest in a potential job in computer security under the premise that the job would entail work of this nature.

Group Dynamics. We hoped to leverage the many benefits associated with group work, fostering a sense of community within teams and leverage the varying skills and experiences students brought to the group. Thus, we are interested in the roles students perceived themselves and others playing during group work. When prompted, students characterized their teammates broadly, and directly related to a student's prior experience with programming. These roles can be generally categorized as "leaders/mentors" (those who knew how to code) and "learners" (those who didn't). We observe that "leaders"

[1] Journal prompts and the constructs they intend to measure are available at: `https://github.com/TableTopSecurity/cpe123-evaluation`.

were often described as taking a position of privilege by sitting at the keyboard during group work, but we did not observe any abuse of power to control or bully less experienced team members. Indeed, experienced students felt compelled to help their teammates, with many embracing the role, with far fewer perceiving it as a burden.

Interestingly, students' perception of their own role was more varied and nuanced but, again, largely dependent on prior programming experience. While some students lacking this experience considered themselves "needy" and dependent on group leaders with respect to the programming assignments, many viewed themselves as contributing to the group through positive support roles, including "quality assurance," "project manager," "researcher," and "facilitator of collaboration."

As observed previously [11, 27], prior experience in coding creates cultural division among students, but in our study, one that isn't invariably negative. While some students stated that their lack of coding experience led them to initially feel less confident in labs (with at least two students claiming it greatly impacted their participation in lab and their social positioning in the group), many more expressed that this lack of experience compelled them to participate more in lab and to "develop coping mechanisms to get around no code knowledge." We are encouraged by our observations and believe our exercises, which are expressly designed to emphasize a variety of skills and abilities, can be supportive of students with mixed experiences and have the potential to be as effective as multi-entry path curriculums.

Changes in Real World Behaviors. Students were prompted to reflect upon connections they made between class topics and their behaviors outside of class. The most frequent change cited was in creating more secure passwords, or acquiring a (good) password manager to handle more complex passwords. One student stated that he began to examine the source code of websites he visited out of curiosity. At least five students stated the course inspired independent and extracurricular investigation into security topics, including: best practices for securing websites, secure coding practices, and advanced techniques for cracking passwords. Some began to attend our university's computer security club meetings. Some students mentioned discussing course materials outside of class, typically with parents, family or friends. These conversations tended to involve encouraging people to adopt more sophisticated passwords (*e.g.*, length of the password used to protect a home network).

There were also quite a few students who did not change any behaviors. For some, their self-evaluation admitted insecure practices, but discussed possible future action (*e.g.*, changing passwords in the future). One student stated he did not think his information was valuable enough to merit any change in behavior. Thus, even among students whose actions went largely unchanged, we found the cognition and reasoning behind those actions demonstrate thoughtfulness and self-reflection.

Perceptions Toward Security. Throughout the course, we challenged students to think adversarially about many technologies they regularly use by asking them to attack or misuse these services. For example, in response to a challenge involving crafting a phishing e-mail directed against the instructor, some students expressed a degree of anxiety, feeling they were being asked to engage in nefarious activities or being graded on "being immoral." They found completing these deceptive tasks "requires strong moral values." Exercises instilled in students realizations that "the people on the other side of the screen [may not always be] good hearted, honest people," a greater inclination toward counterfactual problem solving and breaking systems down into their constituent parts, and a sense of empowerment, *i.e.*, that their "power can be used for the greater good."

4.2 Survey Results

To assess the effect of our course on student attitudes toward computer security, we built a survey instrument, administered in the first and last weeks of the course [2]. Our survey is based on a reliable and validated instrument called the Computer Attitude Scale (CAS) due to Loyd and Gressard [24] and Loyd and Loyd [25]. The CAS is a Likert-type instrument consisting of 40 items, designed to measure attitude toward working with computers, including anxiety, confidence, utility, and perceived success in the field. While intended for use in any field involving computer use, we seek to characterize the same attitudes among students new to computer security.

Specifically, students were asked questions that ascertained beliefs about their abilities to be successful in computer security through self-perceived aptitudes to handle challenging problems and to excel in coursework. Students were also asked questions that sought to measure their attitudes toward being successful in the discipline and to assess their feelings toward being recognized by peers as being good at computer security. To leverage the reliability and validity of the CAS, where appropriate we simply rephrased "computer science" to "computer security." We also introduced new instrument items related to attitudes towards women in computer security. These questions attempt to assess student beliefs about women's capabilities in handling security challenges, the appropriateness of the field for women, and the likelihood of a woman excelling in the discipline.

[2]Full survey instrument is available at: `https://github.com/TableTopSecurity/cpe123-evaluation`.

Variable	Pre-Survey	Post-Survey
Confidence in learning	3.64	4.41
Fit for women	4.43	4.45
Effective motivation	4.3	4.2
Secure computing behaviors	3.29	3.41

Table 1: Results from pre- and post-course surveys measuring students' attitudes toward variable aspects of computer security, presented as composite means (out of 5).

Upon collecting pre-survey data, evaluation was done to determine the reliability of the survey instrument. Cronbach's Alpha was used to measure the internal consistency of the questions with the measured constructs. Based on those results, questions were revised and/or omitted in order to ensure a more reliable instrument for the post-survey. Only questions not requiring revision are analyzed in this work. Additional measures were also taken to ensure the validity of the survey instrument. A construct validity check was completed by an expert in the field. In this capacity, constructs were attributed to questions to yield a 95% accuracy rate. Also, cognitive interviews with three CS majors (2 male, 1 female) were conducted prior to the pre-survey in order to have the survey questions interpreted by representative members of our sample.

Our initial survey results are positive (see Table 1). Prior to taking the course, students were neutral regarding their confidence in being able to learn computer security. Post-survey data show students' levels of confidence in learning computer security slightly increased after completing the course. We were also pleased to measure that pre-survey data show that students overwhelmingly believe that women fit in the field of computer security and students maintained these beliefs after taking the course. In addition, pre-survey data show students were favorably motivated by problems of a challenging nature and after participating in the course, students still attested to challenge as a motivator in the discipline. Lastly, students were asked a series of questions that ascertained their use of secure computing behaviors (*e.g.*, I change my computing habits when I use a computer that is not my own). Subsequently, students attested to using more secure behaviors, particularly their confidence in the ability to identify potentially malicious emails or websites.

5 Conclusions & Future Work

In our course, students had to think computationally and adversarially to solve challenges related to attacking or misusing systems. Toward this goal, students found programming to be an empowering skill, and prior coding experience had a major influence on their perceived role in their group. Programming, however, was recognized by most students as secondary to more core learning objectives. Some students unfamiliar with programming at the start of class attested that the course, rather than focusing on coding, emphasized problem-solving abilities and critical thinking skills. Students with prior programming experience especially recognized that code was "the least relevant skill required in this class;" the "challenge is the challenge itself, not the tools needed;" and the true requirement to solve each challenge is a "sense of logic." We consider this evidence that computer security concepts can be explored effectively in early undergraduate courses as more than brief diversions, without distracting students from core computational lessons or requiring specialized context.

Overall, our initial experiences are very positive. Using ARG-style narratives may help bridge class concepts to the real world, drawing students into fictional scenarios presenting authentic problems employing computation. This also provides rich fodder for discussions about professional and personal ethics, in a relevant, shared setting. Motivating computer science using security challenges, adversarial thinking and intriguing narrative appears to be a promising pedagogical approach, providing intrinsic motivators like those in courses exploring computers for expressive, creative tasks, *e.g.*, game design.

Over successive iterations, we will improve assessment and expand evaluation. Most immediately, we will continue to use, refine, and make publicly available our evaluation instruments. And will ultimately perform longitudinal evaluation, tracking students through their undergraduate careers and comparing their paths to those of students in course sections exploring other themes. Further, we hope to augment our assessments with other tools explored in pedagogical research involving ARGs. Indeed, the genesis of our project is based on a belief that there is enormous value in bridging two communities: those employing game-like security exercises, like CTFs, and those using pedagogical ARGs. We believe the evaluative frameworks developed for pervasive games and ARGs may be broadly useful to security competitions.

Many new research questions have emerged as a result of our initial inquiry. We view our project as having the ability to merge the best parts of a security competition with the best parts of educational ARGs and pervasive games. We find games played "in the real world," mixing puzzles and collaborative play to create fun and educational experiences, have high value warranting continued study: they provide authentic tasks requiring sophisticated problem-solving using real tools, in an environment that is made accessible, social and inclusive; use narrative and team play to stimulate "adversarial" thinking; provoke counterfactual reasoning; and, engage students leveraging the freedoms of play.

Acknowledgements

The authors would like to thank Brian Volk for his level design and web infrastructure development. This work was supported, in part, through an Intel-NSF-GTISC Security Education Micro-grant, by Google through a CS Engagement Small Award, and by the US National Science Foundation under awards #1140561 and #1419318.

References

[1] National Collegiate Cyber Defense Competition. `http://www.nationalccdc.org/`, last accessed Dec 2013.

[2] The National Cyber League. `http://www.nationalcyberleague.org/`, last accessed Dec 2013.

[3] You should do the Matasano crypto challenges. `http://www.matasano.com/articles/crypto-challenges/`, last accessed Dec 2013.

[4] ACM/IEEE JOINT TASK FORCE ON COMPUTING CURRICULA. Computer science curricula 2013 (ironman draft). `http://ai.stanford.edu/users/sahami/CS2013/`, Feb. 2013.

[5] BARKER, L., AND COHOON, J. M. Key practices for retaining undergraduates in computing. National Center for Women and Information Technology, `www.ncwit.org/retainundergrads`, Oct 2009.

[6] BERLAND, M., AND LEE, V. R. Collaborative strategic board games as a site for distributed computational thinking. *International Journal of Game-Based Learning 1*, 2 (2011).

[7] BISHOP, M. Computer security education: Training, scholarship, and research. *Security and Privacy 35*, 4 (2002).

[8] BONSIGNORE, E., HANSEN, D., KRAUS, K., VISCONTI, A., AHN, J., AND DRUIN, A. Playing for real: designing alternate reality games for teenagers in learning contexts. In *Proceedings of the International Conference on Interaction Design and Children* (2013).

[9] BOOZ ALLEN HAMILTON. Cyber IN-security: Strengthening the federal cybersecurity workforce. Whitepaper, July 2009.

[10] CHAPMAN, P., BURKET, J., AND BRUMLEY, D. PicoCTF: A game-based computer security competition for high school students. In *Proceedings of USENIX Summit on Gaming, Games, and Gamification in Security Education* (2014).

[11] COHOON, J. P., AND TYCHONIEVICH, L. A. Analysis of a CS1 approach for attracting diverse and inexperienced students to computing majors. In *Proceedings SIGCSE Technical Symposium on Computer Science Education* (2011).

[12] DODDS, Z., AND ALVARADO, C. Women in CS: an evaluation of three promising practices. In *Proceedings of the ACM Technical Symposium on Computer Science Education* (2010).

[13] DODDS, Z., LIBESKIND-HADAS, R., ALVARADO, C., AND KUENNING, G. Evaluating a Breadth-First CS 1 for Scientists. In *Proceedings of the ACM Technical Symposium on Computer Science Education* (2008).

[14] EVANS, K., AND REEDER, F. A human capital crisis in cybersecurity: A report of the CSIS commission on cybersecurity for the 44th presidency. Whitepaper, Center for Strategic & International Studies, November 2010.

[15] FALKNER, N., SOORIAMURTHI, R., AND MICHALEWICZ, Z. Puzzle-Based Learning for Engineering and Computer Science. *Computer 43*, 4 (2010), 20–28.

[16] FEKETE, A., KAY, J., KINGSTON, J., AND WIMALARATNE, K. Supporting reflection in introductory computer science. In *Proceedings SIGCSE Technical Symposium on Computer Science Education* (2000).

[17] GROBSTEIN, P. Revisiting science in culture: Science as story telling and story revising. *Journal of Research Practice 1*, 1 (2005).

[18] HANSEN, D., BONSIGNORE, E., RUPPEL, M., VISCONTI, A., AND KRAUS, K. Game design for promoting counterfactual thinking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (May 2012).

[19] HANSEN, D., BONSIGNORE, E., RUPPEL, M., VISCONTI, A., AND KRAUS, K. Designing reusable alternate reality games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2013).

[20] HAUNGS, M., CLARK, C., CLEMENTS, J., AND JANZEN, D. Improving first-year success and retention through interest-based CS0 courses. In *Proceedings of the ACM Technical Symposium on Computer Science Education* (2012).

[21] HAUNGS, M., CLEMENTS, J., AND JANZEN, D. Improving Engineering Education Through Creativity, Collaboration, and Context in a First Year Course. In *Proceedings of the American Society for Engineering Education Annual Conference* (2008).

[22] IRVINE, C. E., CHIN, S.-K., AND FRINCKE, D. Integrating security into the curriculum. *IEEE Computer 31*, 12 (1998), 25–30.

[23] KAVANAGH, L., AND OMOORE, L. Reflecting on reflection 10 years, engineering, and UQ. In *Proceedings of the AaaE Conference* (2008).

[24] LOYD, B. H., AND GRESSARD, C. Reliability and factorial validity of computer attitude scales. *Educational and Psychological Measurement 44*, 2 (1984).

[25] LOYD, B. H., AND LOYD, D. E. The reliability and validity of an instrument for the assessment of computer attitudes. *Educational and Psychological Measurement 45*, 4 (1985).

[26] MACVEAN, A., HAJARNIS, S., HEADRICK, B., FERGUSON, A., BARVE, C., KARNIK, D., AND RIEDL, M. O. WeQuest: scalable alternate reality games through end-user content authoring. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology* (2011).

[27] MARGOLIS, J., AND FISHER, A. *Unlocking the clubhouse : women in computing*. Massachusetts Institute of Technology, 2002.

[28] MIRKOVIC, J., AND PETERSON, P. A. H. Class capture-the-flag exercises. In *Proceedings of USENIX Summit on Gaming, Games, and Gamification in Security Education* (2014).

[29] MOSELEY, A. An Alternate Reality for Education? Lessons to be Learned from Online Immersive Games. *International Journal of Game-Based Learning* (2012).

[30] NIEMEYER, G., GARCIA, A., AND NAIMA, R. Black cloud: patterns towards da future. In *Proceedings of the ACM International Conference on Multimedia* (2009).

[31] OLANO, M., SHERMAN, A., OLIVA, L., COX, R., FIRESTONE, D., KUBIK, O., PATIL, M., SEYMOUR, J., SOHN, I., AND THOMAS, D. SecurityEmpire: Development and evaluation of a digital game to promote cybersecurity education. In *Proceedings of USENIX Summit on Gaming, Games, and Gamification in Security Education* (2014).

[32] STRAUSS, A., AND CORBIN, J. *Basics of qualitative research: Grounded theory procedures and techniques.* SAGE Publications, Inc., 1990.

[33] SZULBORSKI, D. *This is not a game: a guide to alternate reality gaming.* New Fiction Publishing, 2005.