Open Access License Notice

Hardware Hacking: An Approach to Trustable Computing Systems Security Education

John A. Chandy
University of Connecticut

Zhijie Shi
University of Connecticut

Mark Tehranipoor
University of Connecticut

Megan Welsh
University of California, Davis

Chujiao Ma
University of Connecticut

Ujjwal Guin
University of Connecticut

Qihang Shi
University of Connecticut

Abstract - Traditional approaches to teaching computer security have focused on understanding software and network security. However, computer systems comprise not only software and networks, but also include hardware components. The security of computer systems hardware has been typically ignored in most computer security curricula. In this paper, we describe a set of courses that can form a core of a hardware security curriculum. We pay particular emphasis to a "hardware hacking" class where students are exposed to a variety of hands-on exercises with hardware assurance. The class has shown that it not only introduce

students to the topics of hardware assurance but also improve their hardware and digital design skills as well.

1. INTRODUCTION

The focus of the curricular effort described in this paper is to help students develop skills in computer systems security. Computing devices are ever-present in our life whether in computers, cell phones, or in the abundance of microcontrollers that manage the various systems we interact with on a daily basis. In addition to our personal lives, computer systems are also integrated into every part of our infrastructure - whether it is financial systems, communication systems, transportation, or defense. Thus, these computer systems are an obvious target for those who intend to corrupt the systems for financial, political or other reasons. The need for practitioners who can address the security of these systems has been identified as a critical national need [1]. With the rapid proliferation of wireless networks and the Internet of Things, security has become increasingly important for networked computing devices. However, the security of networked computer systems is far from satisfactory and becomes even more challenging with the emergence of new attacks.

Traditionally, computer security courses have focused on software and networks, for example cryptographic algorithms, access control mechanisms, digital forensics, intrusion detection, etc. For the most part, these classes assume that the hardware itself is trustworthy. However, as recent research has shown, this assumption is no longer valid. For example, when running cryptographic algorithms, computer systems leak side-channel information, such as timing, power, electromagnetic radiation, visible light, error messages, etc. [2]. Side-channel attacks exploit this information to obtain secret data that is being processed by the computer system. Another possible attack is to place Trojans on integrated circuits that can be triggered at a later point. All aspects of computer systems and cyberspace are dependent on the hardware that underlies these systems. If the hardware is not secure, trustable, and reliable, no amount of cybersecurity algorithms at higher levels will protect the system.

While software system security is a mature field with abundant education and training programs, pedagogy exposing the vulnerabilities in hardware and ensuring its trustworthiness is not available anywhere. The Department of Defense has recently issued an instruction, a critical part of which is to manage supply chains securely including techniques that employ protections that manage risk in the supply chain for component products and services (e.g., integrated circuits, field-programmable gate arrays (FPGA), printed circuit boards) [3]. Thus, it is imperative that we train the next generation of security practitioners to be aware of these hardware security challenges.

At the University of Connecticut (UConn), we have developed a set of courses that expose students to various aspects of hardware security and assurance. The primary objectives of the curriculum effort were to: 1) Develop a curriculum requiring intense study of *computer systems hardware security* and the trust assumptions provided by the hardware; 2) Provide hands-on experience in hardware *reverse engineering*, *debugging*, and *analysis*; 3) Develop modules that are adaptable for use at other colleges; 4) Develop a culture of aggressive *hacking* while still emphasizing professionalism and ethics; and 5) Promote understanding of countermeasures and techniques to address non-traditional attack scenarios. The goal of this project is to improve and modernize computer engineering and computer science education by introducing an experimental curriculum in computer systems security. The curricular evolution proposed in this project builds on the momentum gained in recent initiatives such as the "Hacker Curriculum" [4].

A secondary goal of the curriculum improvement was to enhance recruitment of potential computer engineers. Our university, as well as other universities around the country, has seen enrollment in computer engineering remain flat or decrease slightly over recent years. There are many reasons, some particular to UConn, such as Connecticut's mechanical engineering industry base, and some that are national in nature, such as outsourcing and the general downturn in the high-tech job market since the dot-com boom. We see the introduction of new teaching strategies enabled by this curriculum as a way to increase awareness of the department and the promise of careers in computer engineering.

The learning materials and teaching strategies in this curriculum can be adapted to fit in an engineering curriculum at a 4-year college as well as a 2-year community college.

2. BACKGROUND

Our approach begins with the belief that computer security is best understood by not only understanding how to protect your systems security but also how to attack your systems. The military community has long understood this principle by using war game scenarios where a "red" team plays the role of an attacker and an independent "blue" team must then defend against the attack. By allowing the "red" team to independently strategize on attack modalities, the military can better identify weaknesses or holes in its defense systems.

The computer security community has adopted similar tactics with the employment of "white hat" hackers who are hired to penetrate computer systems and identify weak points in systems. Note that "white hat" hackers are distinct from "black hat" hackers who attack systems for personal gain or "grey hat" hackers who are independent security experts who attack systems to reveal system vulnerabilities - not only to the system owner but also to the public. Both "white" and "grey" hat hackers have been termed ethical hackers to distinguish them from hackers whose primary motivation is hostile.

There is very little training available to teach new computer scientists and information technologists the importance of "white hat" or "red" team hacking. There are a few professional certification programs on ethical hacking [5,6,7], and a handful of universities, such as Northumbria University and Dakota State University, have established programs in ethical hacking. However, very little on ethical hacking has appeared in the majority of undergraduate computer science or computer engineering curricula.

Sergey Bratus at Dartmouth College has advocated for a "hacker" curriculum to be incorporated into undergraduate computer security courses [4]. The intention is to inculcate students with an awareness of security issues by training them to think like attackers - i.e. questioning barriers, APIs, implementations, etc. Bratus has

observed that computer science students traditionally learn about programming languages, data structures, and algorithms without ever understanding the tools and frameworks that allows everything to work [8]. For example, most students focus on user applications, but do not know how linkers work, how operating system calls are made, how memory allocation is performed, the difference between stack data and heap data, etc. - even though all software depends on these systems to function properly. This lack of understanding means a lack of awareness of potential vulnerabilities in code that lie in the boundaries between user applications and system support frameworks.

Hackers, however, are well aware of these tools and frameworks and even more importantly, the vulnerabilities in these tools and frameworks. The notion behind the "hacker" curriculum is to train students to think like hackers and train them to poke and probe the weak points in a system. This training will allow them to become "red" team members working on attacks or "blue" team members working on countermeasures. Even if they do not continue as computer security professionals, the better understanding of vulnerabilities should produce software engineers who write more secure code. In addition, the deep analysis of software systems required in ethical hacking should also give students a better understanding of software systems and thus produce better engineers who can competently write, debug, and test their software.

We concur with Bratus' observation and feel that the approach not only applies to software security but also hardware security. Exploring hardware attack scenarios requires the same type of non-traditional thinking and a need to question hardware trust boundaries and also explore non-obvious side channel attacks. Understanding how to attack hardware and also secure hardware requires a set of skills not taught in traditional computer science security or computer engineering classes. Similar to the experiences with computer science students, computer engineering students do not learn how to reverse engineer hardware systems, how to debug hardware (other than through simulation), or the interactions of multiple components.

Thus, the primary focus of this proposal is to develop a set of courses that instills a hacker's mentality in computer engineering students with respect to hardware design.

3. NEW HARDWARE SECURITY CURRICULUM

The new hardware security curriculum is comprised of three components: 1) a new hands-on course on computer hardware security and trustable computing systems that would be approach the material from a hacker point of view as well as being very much hardware focused; 2) A lecture-based course to cover the theoretical aspects of hardware security that draws from the latest research; and 3) Computer systems security related projects as part of the existing senior design/capstone experience. What follows is a description of the three components.

3.1 ECE4095 - Trustable Computing Systems

Our *ECE4095 Trustable Computing Systems* class was taught for the first time in Spring 2012 and has been taught every subsequent spring. It is an elective class for computer engineering students, and they typically take the class in the spring of their junior year. This course takes the pedagogical approach that computer security is best taught from the hacker's perspective. Thus, the course is heavily project-based and attempts to inculcate in the students a mindset similar to what hackers would use. These include learning reverse-engineering principles, understanding cross-layer interactions, mistrusting API black-box rules, etc. Unlike other programs that have instituted a similar philosophy in their computer security classes, however, we are focusing on hardware security. This requires an understanding of the physical properties of computer hardware - e.g. power, electromagnetic emissions, acoustics, thermal signature, etc. The course, thus, not only introduces hardware security topics but also reinforces material from other engineering classes including electromagnetics, acoustics, circuits, statistics, spectral analysis, etc. The main corequisite for the class is a Digital Systems Design class, which is typically taken at the same time as this Trustable Computing Systems class.

The course starts with an introductory lecture on professional ethics with respect to hacking and reverse engineering. With any course like this, there is always the danger that we may be training "potential future hackers" and we do not take this lightly. As with any ABET-accredited school, we do require that our engineering students take a course on ethics. We expand on that in this course and remind students of their professional duties and responsibilities as engineers. The goal is to train these students to become participants in white-hat, gray-hat or legitimate "red" team activities.

The remainder of the course is organized around several stand-alone hands-on project modules as listed in Table 1. We have developed several module projects that explore various properties of computer hardware, including power, timing, electromagnetics, thermal, etc., and also cover a broad range of hardware components, including CPU, I/O, storage, memory, etc. modules. Each module consists of:

- An opening lecture that gives the background science behind the project module and introduces new measurement or lab techniques
- A hands-on project where the student works on a particular type of hardware attack. Each modules takes between 1-4 weeks depending on project complexity.
- Student report on the post-mortem analysis of the project and its ethical impacts, and a discussion of potential countermeasures and new attack strategies.

Module Title	Description
Differential power analysis for key recovery	Students learn how to retrieve keys exposed by cryptographic algorithms through side channel leakage in CPU power lines. The students must write algorithms to process power traces and do differential power analysis [9].

Timing analysis of CPUs for key recovery	Students learn timing analysis attacks that can retrieve secret data based on computation time required to perform cryptographic operations [10,11].
Acoustic keypad recovery	Students attempt to recover key presses on a keyboard by using neural network analysis on the acoustic emanations from a keyboard [12,13].
Data remanance in DRAM	Students recover data from DRAMs that remains readable after power has been removed due to data remanence. The technique requires an understanding of the boot process and memory allocation in modern computers [14].
HW Trojans	Students learn how to insert hardware Trojans into an integrated circuit by writing VHDL Trojans and an FPGA to emulate the effect of HW Trojans. See Figure 1.
Counterfeit Detection	Students learn electrical test and imaging techniques to detect counterfeit integrated circuits. See Figure 2.
Hard disk data	Students learn data forensics techniques to recover data from discarded hard disks that often have sensitive data [15].

Table 1: ECE4095: Trustable Computing Systems Project Modules

The class was team-taught by three faculty, each leading modules related to their own area of specialization. Graduate assistants were also assigned to the course and helped with running labs and conducting some lectures. Because of the highly specialized nature of each lecture, the graduate students rotated in and out based on the topic being covered that week, as did the faculty instructors. The course format

involves presenting students with the background knowledge with lectures and then having them apply the content they just learned with lab. Lab reports were the main form of assessment in the course.

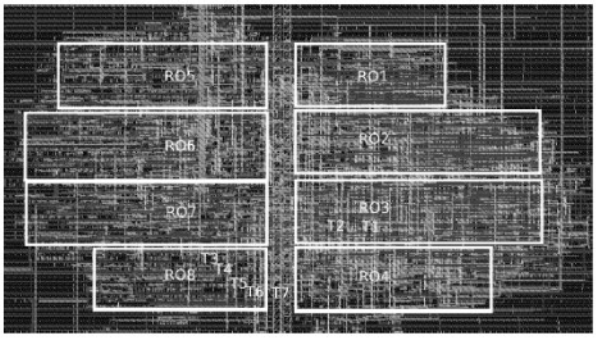

Figure 1. Example of ASIC circuit with hardware Trojans

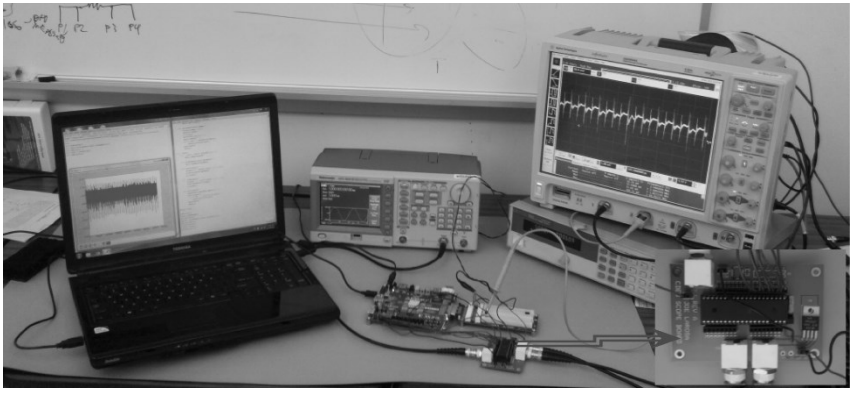

Figure 2. Counterfeit detection electrical test setup

3.2 ECE4451 - Hardware Security and Trust

This course was offered for the first time in Spring 2010 and has since been offered in Fall 2012 and every subsequent fall. The course is intended as a follow-on to students who have taken the previous *Trustable Computing* Systems class.

While the *Trustable Computing Systems* class covers a wide array of hardware security attacks and countermeasures, the ECE4451 course provide a more rigorous theoretical framework and expands on secure hardware design techniques, principles, and methods.

Both computer engineering graduate and senior undergraduate students are encouraged to take this course. The course covers the following topics: Cryptographic processor and processing overhead analysis, physical and invasive attacks, side-channel attacks, physically unclonable functions, hardware-based true random number generators, watermarking of Intellectual Property (IP) blocks, FPGA security, passive and active metering for prevention of piracy, access control, hardware Trojan detection and isolation in IP cores and integrated circuits. The main goals for this course are:

- Learning the state-of-the-art security methods and devices
- Integration of security as a design metric, not as an afterthought
- Protection of the design intellectual property against piracy and tampering
- Better understanding of attacks and providing countermeasures against them
- Detection and isolation of hardware Trojans

3.3 Capstone Projects

As with most engineering programs, we also require all of our graduates to complete a two-semester capstone course in which students work in teams on longer term projects that allows them to apply their learning on real world projects. Projects are typically sponsored by industry and the teams are advised by a faculty member and an industry liaison. Students who have taken one of the prior hardware security courses in their junior have been assigned to hardware security related design project in their senior year. While in the past, we have had security related senior design projects (Home Automation Security, Building Security System, etc.), they have not been integrated with the existing computer security curriculum.

With the introduction of the new hardware security courses, projects started in the *Trustable Computing Systems* or *Hardware Security and Trust* courses can develop into longer-term capstone projects. Taking class based projects and expanding them into capstone projects is a new approach that allows students to work on potentially 1.5 to 2-year projects that greatly increases the scope of what is possible in a traditional one-year senior design sequence. Over the past three years, we have had several hardware security senior design projects including Automated Hardware Trojan Insertion and Detection, Automated IC Counterfeit Detection, Physical Inspection of Counterfeits, Imaging Techniques to Detect Counterfeits, Virtual Laboratory for Hardware Security, and Counterfeit Defect Characterization.

4. PROJECT OUTCOMES AND ASSESSMENT

The goal of this project is to provide a novel approach to undergraduate computer systems security education by introducing an experimental course in hacking and countermeasures of computing systems and then extending that to a senior capstone project with entrepreneurial focus. Education objectives and students learning outcomes of the new curriculum have been regularly assessed in relation to ABET standards (a-k outcomes). A relevant point is that the assessment of the proposed project outcomes and results will have a natural reference against the regular programs. By using the current university-wide course evaluation criteria, the new computer systems security course has a firm reference against the regular computer science security course for comparison and success assessment. Quantitative and qualitative improvements are therefore easily established for the new course.

The evaluation collected data in multiple forms from the students enrolled in ECE 4095, with the input of faculty teaching the course. These data included a basic demographic background sheet, focus groups with all students enrolled in ECE 4095, and a problem review activity administered both to ECE 4095 students and to Computer Engineering majors who did not enroll in ECE 4095.

4.1 Focus Groups

The evaluation held two in-class focus groups with students at the end of each semester in which ECE 4095 was offered (Spring 2012 and Spring 2013). The interview spanned the same topics as the focus group, namely:

1) A general description of the course, including: topics covered, modules, activities, assignments, and general attitudes about the class.

2) The instructional units or modules students learned the most from and what made them effective.

3) Aspects of the class that could be strengthened and suggestions for improvement.

4) Recommendations for improving the class in future years.

The evaluation also intended to administer the Student Assessment of Learning Gains (SALG), and developed a SALG instrument (also included in Appendix A). However, no students completed the SALG even with multiple reminders and the offer of a small incentive for completing it (students were offered a $5 Amazon gift card). Because a relatively small number of students enrolled in ECE 4095—six students in Spring 2012, five students in Spring 2013—and because the same information was addressed in the focus group as on the SALG, SALG results were not deemed crucial to gathering useful information about students' experiences in the ECE 4095.

4.2 Problem Review

One of the goals of the "Hardware Hacking" class was to determine if the course helped improve the students' computer engineering skills. In other words, we wanted to know if students left the course with a distinct set of skills that other Computer Engineering students lacked. However, it was not possible to administer an exam to all Computer Engineering students and it would be difficult to establish that the exam adequately gauged a wide array of skills to make any definitive judgments. Sample size is also of concern since 11 students enrolled in ECE 4095

across the two semesters it was offered and 25-30 Computer Engineering majors graduate in a given year.

We decided to attempt to get at differences in the skills that the students acquired by generating a set of problems for students to review, but did not have to solve. The problems, generated by the faculty members teaching ECE 4095, spanned five topics, some of which all Computer Engineering majors should be able to answer (Software and Operating Systems) and also topics that were only addressed in ECE 4095 (Digital Systems, Embedded Systems, and Security). Students were asked to read the problems, rank order them from easiest to most difficult, and then to indicate with problems were attainable by all students, only by C students, only by B students, and only by A students.

The rank order responses were analyzed using a method proposed by Thurstone [16]; item preferences were quantified based on their comparison with the difficulty of all other items. These relative difficulty ratings were when converted to *z*-scores (units reflecting standard deviations from the mean) based on the proportion of items they were deemed to be more difficult than. The *z*-scores were then examined in relation to cut scores established for each attainability ranking (anyone can answer this, B students can answer this, only A students can answer this, etc.) to describe the general level of difficulty. Patterns of item difficulty were examined for ECE 4095 students and for all participating students.

4.3 Participants

Eleven students enrolled in ECE 4095, six in Spring 2012 and five in Spring 2013. All students were male upperclassmen. Across the two sections, five students were seniors and six were juniors. The students had attended college for 3 to 5 years and were between 20 and 24 years of age (*Mdn* = 21) when enrolled in ECE 4095. Seven students were Caucasian, two were Asian, and one was African American. One student declined to report their race. Of those reporting their race/ethnicity, none were of Hispanic origin.

The five Computer Engineering majors who reviewed problems, but did not enroll in ECE 4095 were also all male and were all seniors. They had all attended

college for four years (and participated in the evaluation in their last semester at UConn). Four were Caucasian and one was Asian. One student was of Hispanic origin.

4.4 Observations

Students liked the course. Students were quite enthusiastic about the course. They seemed to find the topic of hacking very interesting and also greatly appreciated having the opportunity to learn about hacking from some of the world's leading experts on the topic. When asked to explain the course to a layperson, one student said, "Sensitive information is stored in a computer's hardware, not its software, its hardware, and can be accessed if you know where to look for it and get to things quickly enough. This class was all about learning how people hack hardware and how to prevent it."

They also indicated that the structure of the class (which was more of a seminar format than other classes they had taken), and the small class size was of great benefit. When asked about particular activities or modules that they found most beneficial, students mentioned the acoustic emanations project module, where they determined what someone had typed by recording the sounds that emanated from the computer keyboard and analyzing that recording, as well as the data remanance module where data can be hacked as a computer powers down. When explaining the concepts covered by the class, students became very animated in discussing both the variety of ways in which hacking can occur, and how difficult it is to detect and prevent. More than anything, they seemed to appreciate the problem and to respect the work that faculty are doing to improve the field.

Students found the lecture-lab format effective. Students said that the general instructional approach of learning about a topic in a classroom setting and then applying their knowledge with a lab worked for them. As is common the first time a class is taught, students reported a few hiccups in some of the labs (e.g., it took very long to run some programs in Matlab), but this was corrected by the second time the course was offered. Some activities required programming in C instead of in Matlab to shorten the time it took to run certain activities. Students said that

lectures were clear and easy to understand. When asked what worked best about this format, students pointed to instructor's skill in lecturing, their expertise in general, and in the clarity of directions in labs. As one student put it, "We just need to know what we need to do and how things should look when we are done."

Proper staffing is essential. While generally quite happy with the course, when asked how the course could be improved, students mentioned how essential good graduate assistants are to the quality of this course in particular. Students in the 2012 class seemed more concerned about the quality of graduate assistant support than did students in the 2013 class. In the first class, students mentioned that labs were not always graded in a timely manner and that some graduate assistants were more knowledgeable and more available for questions than others. Some students in that year recommended that one graduate assistant be assigned to support the class throughout while others acknowledged that that would not be reasonable given the specialized knowledge required for each lab. By 2013, students seemed generally happy with the graduate assistant support they received. Although students did mention that one graduate assistant did not know how to program in C and had difficulty provided support and grading assignments because of their lack of expertise in this area.

Topic	All Students	ECE4095 Students
Digital Systems	-0.55	-1.00
Software	-0.15	-0.12
Embedded Systems	-0.15	-0.08
Security	+0.28	+0.25
Operating Systems	+0.98	+1.50

Table 2: Average z-scores for various computer engineering topics

Differences between ECE 4095 students and the sample as a whole in terms of perception of problem difficulty. The results of the problem set review are summarized in Table 2 with a comparison of the average *z*-scores for all computer engineering students and just ECE 4095 students. Negative *z*-scores indicate that the students found the problems easier, and positive scores indicate the problems were harder. Overall, both groups of students identified similar numbers of problems in each difficulty category (attainable by A students, B students, C students, etc.). Students also seemed to have similar opinions of the difficulty of different topics, with Digital Systems problems rated the easiest and Operating Systems and Security problems rated most difficult, and Software and Embedded Systems problems ran the full gamut of difficulty for both groups of students.

There were specific problems that one group found easier or more difficult than the other. The only real differences that could be discerned is that the ECE 4095 students as a whole found the digital systems problems easier and the operating systems problems harder. As we would have expected by our initial premise, the hands-on experience in ECE4095 improved students' ability in hardware-related problems. The difference in operating systems is most likely because most ECE4095 students were juniors, and computer engineering students typically take the operating systems class in their senior year. What surprised us, however, was the lack of difference in security related problems. On the other hand, when examining each security problem individually, we found distinct differences. Specifically, ECE4095 students found problems on analysis of security properties of a system easier, but they found software implementations of cryptographic primitives more difficult. Since we focused on hacking, analyzing security properties would be a skill that students would become better at. However, we did not spend us much time on the basic cryptographic security primitives such as encryption, hashing, etc. Much of this material is covered in the follow-on ECE4451 class.

5. CONCLUSIONS

Overall, students reported that they greatly enjoyed the new courses and got a lot out of them. The enrollment is somewhat low, but this is the result of the

relatively small number of Computer Engineering majors and their limited ability to enroll in electives given their heavy course loads. The importance of the topic itself, and the need to both recruit people into the field and to train them outweighs concerns about class size. The class size may also be beneficial as it allows for the kinds of individualized, in-depth instruction that is required to develop expertise in the field. We also need to do a better job with respect to preparation of students – specifically learning C for some of the programming assignments.

The "hardware hacking" course demonstrated the beneficial effect of teaching hacking to students in improving their ability to solve hardware-related problems. We had hoped for a broader effect across all computer engineering topics, and we will work on adjusting our project modules to give students a breadth of activities, while still retaining the hardware focus. The project modules are available to any university that wishes to adapt them. In addition, we are working on building a virtual laboratory environment to allow students to work on these problems remotely.

6. ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation grants (award numbers DUE-1043313 and DGE-1318964). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation.

REFERENCES

[1] Executive Office of the President National Science and Technology Council, "Trustworthy Cyberspace: Strategic Plan for the Federal Cyber-Security Research and Development Program," December 2011.

[2] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Side channel cryptanalysis of product ciphers," *Journal of Computer Security*, vol. 8, no. 2-3, pp. 141–158, 2000.

[3] Department of Defense, "Protection of Mission Critical Functions to Achieve Trusted Systems and Networks," Instruction 5200.44, November 5, 2012.

[4] S. Bratus, A. Shubina, and M. E. Locasto, "Teaching the principles of the hacker curriculum to undergraduates," in Proceedings of the ACM Technical Symposium on Computer Science Education (SIGCSE), (Milwaukee, WI), pp. 122–126, March 2010.

[5] EC-Council, "Certified ethical hacker." http://www.eccouncil.org/certification/ certified ethical hacker.aspx.

[6] Offensive Security, "Offensive security certifications: OSCP, OSCE, OSWP." http://www.offensive-security.com/information-security-certifications.

[7] 7Safe Limited, "CSTA ethical hacking training course: Hands-on." http://www.7safe.com/ ethical hacking course-technical hands-on.htm.

[8] S. Bratus, "What hackers learn that the rest of us don't," *IEEE Security & Privacy*, pp. 72–75, July/August 2007.

[9] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. 19th International Advances in Cryptology Conference* – CRYPTO '99, vol. 1666 of *Lecture Notes in Computer Science*, pp. 388–397, 1999.

[10] P. C. Kocher, "Timing attacks on implementations of Diffe-Hellman, RSA, DSS, and other systems," in *Proc. 16th International Advances in Cryptology* Conference – CRYPTO '96, vol. 1109 of *Lecture Notes in Computer Science*, pp. 104–113, August 1996.

[11] Z. J. Shi and F. Zhang, "New attacks on randomized ECC algorithms," in *Proceedings of EITC 2006*, pp. 22–25, August 2006.

[12] D. Asonov and R. Agrawal, "Keyboard acoustic emanations," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 3–11, 2004.

[13] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 373–382, November 2005.

[14] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten, "Lest we remember: Cold boot attacks on encryption keys," in *Proceedings of USENIX Security Symposium*, 2008.

[15] S. L. Garfinkel and A. Shelat, "Remembrance of data passed: A study of disk sanitization practices," *IEEE Security and Privacy*, vol. 1, pp. 17–27, 2003.

[16] L.L. Thurstone, "The method of paired comparisons for social values." *The Journal of Abnormal and Social Psychology*, 21(4), 384-400, 1927.