Teaching Cybersecurity Analysis Skills in the Cloud [*]

Richard Weiss
The Evergreen State College
Olympia, WA 98505 U.S.A.
weissr@evergreen.edu

Stefan Boesen
Dartmouth College
Hanover, NH U.S.A.
stefan.boesen@gmail.com

James Sullivan,
Michael Locasto
The University of Calgary
Alberta, Ontario, CA
locasto@ucalgary.ca
sullivan.james.f@gmail.com

Jens Mache,
Erik Nilsen
Lewis & Clark College
Portland, OR U.S.A.
jmache@lclark.edu nilsen@lclark.edu

ABSTRACT

This paper reports on the experience of using the EDURange framework, a cloud-based resource for hosting on-demand interactive cybersecurity scenarios. Our framework is designed especially for the needs of teaching faculty. The scenarios we have implemented each are designed specifically to nurture the development of *analysis skills* in students as a complement to both theoretical security concepts and specific software tools.

Our infrastructure has two features that make it unique compared to other cybersecurity educational frameworks. First, EDURange is scalable because it is hosted on a commercial, large-scale cloud environment. Second, EDURange supplies instructors with the ability to dynamically change the parameters and characteristics of exercises so they can be replayed and adapted to multiple classes. Our framework has been used successfully in classes and workshops for students and faculty. We present our experiences building the system, testing it, and using feedback from surveys to improve the system and boost user interest.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and Protection*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Hacking*

General Terms

Security, Measurement

Keywords

offensive security; hacker curriculum; analysis skills; EDURange

1. INTRODUCTION

The US faces a major shortage of cybersecurity workers to defend our information infrastructure from attack [11]. In recognition of this need, security has been included as a core topic in the new ACM/IEEE Computer Science 2013 Curricula [16]. Cybersecurity is also mentioned in more than half of the other knowledge areas in this report. At educational conferences such as SIGCSE and regional CCSC conferences, we are also seeing a growing interest in cybersecurity among faculty who do not have expertise in this area. Given the tight constraints of the Computer Science curriculum, many schools do not have the luxury of offering a separate class in cybersecurity. We are seeing a growing concensus articulated in the CS 2013 Curricula and in the ACM report "Toward Curricular Guidelines for Cybersecurity" [14] to integrate cybersecurity into the Computer Science curriculum at multiple levels in multiple courses.

One of our primary goals for EDURange is to create exercises that nurture analysis skills. When speaking of analysis skills, we mean the ability to reason about large, complex, and opaque data and systems. Strong analytical skills enable people to impose structure and meaning on such artifacts, reason about these relationships, and draw meaningful conclusions or inferences. These are precisely the kinds of skills that we believe are useful in many cybersecurity scenarios from security policy design to reverse engineering to vulnerability analysis. Analysis skills work in conjunction with the security mindset, which is the ability to think about how systems can fail, and be made to fail in different ways. Questioning assumptions plays an important role in both defense and attack. In designing our exercises, we focus on the following analysis skills:

1. **Verifying assumptions** by checking network messages, protocols, file formats and other input data constraints to see if layers of abstraction are coherent and correct. Enumerating and checking if failure modes, exceptions, and errors are controlled, caught or anticipated.
2. **Gaining understanding** of program, network, or system behavior and semantics, network topology or organization, or a defense posture. Observing and enumerating how software components or network elements are actually composed.

[*] This work was partially supported by NSF grants 1141341 and 1141314

SIGCSE'15, March 04-07, 2015, Kansas City, MO, USA.
Copyright is held by the authors. Publication rights is licensed to ACM.
ACM 978-1-4503-2966-8/15/03 ...$15.00.
http://dx.doi.org/10.1145/2676723.2677290

3. **Extracting Information** from opaque artifacts. For example, analyzing a raw dump of network traffic or intrusion alerts or firewall logs and recognizing true anomalies.
4. **Creating Emergent Resilience** by understanding a system well enough to design and propose enhancements to reliability, fault tolerance, or availability.

One of the major obstacles to integrating the teaching of these skills and the security mindset into the curriculum is the amount of work required to create and set up new hands-on exercises that can be readily adapted to existing courses. There was a gap between what we wanted and what we could access, so we decided to build our own tools and exercises satisfying the following criteria:

1. *Flexibility* to use simple scripts to specify exercises at a high level and create variations. Many of the exercises written by others were good but not exactly what we wanted, and they were not easily modified. A big threat to creating *long-term teaching tools* with those approaches is that exercises become stale and answers become easy to Google.
2. *Ease-of-use for faculty*, which includes providing easy access to exercises, making them easy to create (not requiring configuration of VMs manually), easy to modify, and easy to share.
3. *Educational goals*: we wanted to implement scenarios that would teach analysis skills, the *security mindset,* and address the CS2013 guidelines.

This paper is about the design of our current set of exercises (Section 3.1) and our experience creating the framework and delivering the exercises to a variety of audiences (Section 4). We used surveys to evaluate whether students found the exercises interesting and considered them worthwhile. We also employed surveys and informal discussion to get feedback that we used to improve the exercises and the system. The main lessons learned were about the scaffolding required for the students, the interface required for the faculty, and the level of interaction required for the developers, which are discussed in Section 5. That section also discusses the mapping from exercises to the CS2013 curricula Security Knowledge Area. The next section describes related work and how our exercises and infrastructure extend previous work on hands-on exercises.

2. RELATED WORK

There is a growing pool of curriculum material, instructor/faculty training, and VM-based labs. Yet, it has been observed that most deployed exercises and hosted environments had several shortcomings that made them difficult to leverage in our classrooms [19]. For example, Towson's "Security Injections" [17] mainly focus on several important secure programming patterns, but do not emphasize analysis. The SEED [8] project presents a mature, well-documented set of exercises, which are not typically interactive or dynamic and require significant work to set up and run.

Our philosphy on information security education stems from our understanding and teaching of the hacker curriculum as described by Bratus [1]. This approach is predicated on the utility of understanding failure modes. Rather than teaching students the "success" cases, we attempt to deliver a culture shock that makes them disrespect API boundaries and adopt a cross-layer view of the CS discipline as described by Bratus et al. [3]. We also routinely encourage our students to adopt a dual frame of mind (attacker and defender) when solving problems to prevent artificial abstraction layers from becoming boundaries of competence [20]. The importance of analysis skills as explained by S. Bratus et al. [2] is based on linking expected behavior to actual behavior as seen in network traces, log files, program binaries, rules/policies, system call traces, network topologies, network interactions, unknown protocols, injected backdoor code, etc. All of our exercises are based on these skills. A tool that actually applies this type of analysis is NetCheck [21] which is used to debug network applications. Using a simplified model of normal network behavior, NetCheck collects information about network applications using strace.

Our work follows the tradition of creating cybersecurity games or exercises, which are known to engage students [18, 12]. This includes competitions such as CCDC[1], Plaid[2], notsosecure[3], iCTF[4] [7], CSAW[5] [12], TRACER FIRE[6], Packetwars[7], and many others. From our perspective, one problem with these competitions is that they require a significant amount of infrastructure and preparation by the organizers. For example, it took several grad students six months to create the exercises for iCTF [7]. Some competitions such as CCDC and Packetwars require the installation of physical hardware, and they require that students and their faculty travel to participate. There are also a number of non-technical games with the goal of interesting students with no technical background in cybersecurity. These include Control-Alt-Hack [6], [d0x3d!] [13], Security Cards[8], CyberCIEGE[9] [5] and Werewolves [9]. The last of these introduces players to the concept of covert channels in a non-technical context. Our exercises are intended to create scenarios that are closer technically to real-world situations that a security professional would face.

Cloud-based testbeds, such as DETERlab [15] and The RAVE were designed for large-scale security experiments and have also been used for teaching. However, they were not designed for the bursty demand for VMs produced by large classes. By using Amazon's EC2, our framework has a greater capacity for elasticity, hence scalability in terms of the number and size of classes that can access the testbed simultaneously as well as the total number of VMs. In addition, the former do not provide an easy-to-use framework for intructors to script the creation of exercises. The National Cyber Range is also of note, but its primary use is as a secure testbed for research. The Seattle Testbed[10] is a research environment with several security exercises including one on reference monitors [4]. The PacketWars game was a model for some of the EDURange exercises. It has provided students with access to "live" exercises on a small scale, and the overhead is high in terms of time and money. Table 1 shows the *strengths and weaknesses of existing cybersecurity labs, exercises, and curricula with respect to our goals.*

3. DESIGN OF THE EDURANGE FRAMEWORK

One of the main requirements was the ease of use in terms of accessibility and user interface. We address this accessibility requirement by deploying our framework on Amazon's AWS EC2 cloud. Students and faculty do not need to sign up in advance. The resources are always available, and students can work from

[1] http://nationalccdc.org/
[2] http://www.pwning.net/
[3] http://ctf.notsosecure.com/
[4] http://ictf.cs.ucsb.edu/
[5] https://csaw.isis.poly.edu/
[6] http://csr.lanl.gov/tf/
[7] http://packetwars.com/
[8] http://securitycards.cs.washington.edu/
[9] http://www.cisr.us/cyberciege/
[10] http://seattle.poly.edu/

Table 1: ***A Comparison of our Project and Other Projects.*** **EDURange focuses on developing cybersecurity analysis skills. This table is not a criticism of existing efforts, but rather meant to highlight the ways in which our project differs from the main characteristics of existing projects — note that these projects may have been built with different criteria in mind. Security Knitting Kit was not directly available from the website.**

Project	**Primary Weakness**	**Primary Strength**
CyberCIEGE	non-technical analysis	interactivity of training scenarios
SEED	lacks competitive interaction	comprehensive documentation
Security Injections	focus on defensive coding patterns	introductory; clear documentation; CS1, CS2
CCDC	requires travel; limited remote access	interactive and competitive
PacketWars	requires travel; limited availability	engaging, dynamic, competitive scenarios
ITSEED	minimal instructor support, distrib by flash drive	good documentation for students
Google Gruyere	narrow focus (web apps)	cloud-based; well-documented
The RAVE	limited scalability, complex to modify	cloud-based; existing lab manual
Seattle Testbed	limited in scope	easy-to-use; scalable; includes mobile devices
DETERlab	limited scalability	range of exercises
EDURange	limited number of exercises	flexible, easy-to-use, interactive, cloud-based

anywhere. The original interface was an ssh client, and we now have a browser interface. We decided against the alternative of running VMs on a local cluster because that would have been more expensive to create and more work to maintain. Running the Recon 1 exercise for a class of 30 students only costs $0.40 per hour.

An important way in which we achieved flexiblity is through the use of tools such as Chef that script the installation of software. The same Chef script can install packages for a wide range of operating systems. The base VMs on AWS don't change frequently, and the Chef scripts that install software can handle upgrades transparently.

Each EDURange exercise is specified by a YAML file. YAML is similar to XML but more concise. A small number of types of entities recur in all of our scenarios. Based on a Scenario Description Language [10], we chose our primitive types to be: networks, instances (host computers, nodes), software that is directly involved in the exercise, participants (users), groups (teams of users), artifacts (flags), and goals (scoring events). Some exercises clearly involve networks and subnets. For example, Recon 1 and Recon 2 have a subnet for the battlespace and a subnet for each team. The ELF Infection exercise has a simple network topology with a subnet for each team and a subnet for the infected VM. An "instance" is usually a VM with an operating system. The special software for Recon 1 includes nmap and tcpdump. The scoring events in this exercise would include the IP addeses of the instances in the battlespace.

Our scenario description language together with the interpreter provide flexiblity to modify exercises each time they are run. In our pedagogical model, students repeat exercises. For example, with Recon 1, a student may try a set of options for nmap and discover that it takes too long or is not stealthy enough. We want the student to have time to think about trying different options after experiencing the problem. Analysis takes time, which is a limitation of most competitions. They tend to reward speed and don't allow time for in-depth analysis. Repeating exercises is not viable if the network configuration is static. Scripting in EDURange makes network configuration dynamic. For example in Recon 1, instructors can change the IP addresses of hosts in the BattleSpace through the YAML file. In Elf Infection, an instructor with help from us was able to change which binaries were infected.

3.1 Scenarios

An educational goal of each scenario is the development of analysis skills; in other words, the student would come away from the experience with not only an appreciation for the content knowledge involved or a basic understanding of the tools, but also with *insight* and a *logical approach* for understanding the conceptual issues at play. EDURange is a work in progress, and we have a number of exercises under development. Each exercise has multiple levels, and except for Recon, we only describe the first level. Each exercise addresses information assurance and security (IAS) goals in the ACM/IEEE CS2013 Curricula [16], page numbers refer to that document.

- **Recon 1** is about mapping a network and understanding network protocols, such as TCP, UDP, ICMP. The learning outcome being able to diagram a network for security (p. 107)
- **Recon 2** includes intrusion detection and prevention. The student trades off speed with stealth, the attacker must be able to map a network without triggering the defenses. In the attacker/defender mode, background traffic is injected into the network. The defender must distinguish between the attack and background. This addresses network monitoring and intrusion detection (p. 106)
- **ELF Infection** is about forensics and reverse engineering. The student is given a VM, which has an infected utility. They must discover which utility is infected and where the malicious behavior is. This an example of malware (p. 105)
- **ScapyHunt** is a puzzle set in a software defined network. The players must find data on a target host that is behind a gateway by passively examining network traffic and crafting packets to reveal specific information. The style of play is similar to a text adventure game. This is an example of diagraming a network and network monitoring (p. 106).
- **Firewall** is about creating a set of rules to control traffic in and out of a network. More generally, it requires understanding how a complex set of rules implements an access conrol policy. (p. 106)
- **Fuzzing** In the simplest version, the defender is given the grammar for a calculator and must implement an interpreter for that grammar. The attacker tries to fuzz the interpreter to produce incorrect results or get it to reject a valid expression. This is an attacker/defender game. Fuzzing is a Core Tier2 elective (p. 104)
- **Process Records (strace)** involves dynamic analysis of bi-

> naries and poses the challenge of understanding what a process is doing based on its system calls. Students learn to filter large amounts of data to distinguish between normal and anomalous behavior indicative of malware. (p. 105)

4. TESTING EXERCISES

4.1 Methodology

For this initial study, we focused on two research questions:

1. Would our framework meet the needs of faculty?
2. Would our exercises be engaging for students?

We administered surveys and interviewed participants at several events, including SISMAT, classes in computer security, the two hackathons, and workshops for faculty at conferences such as SIGCSE and CCSC. *SISMAT is a summer program for college undergraduates* that includes a two-week intensive program in cyber security at Dartmouth College followed by a summer internship. Students do extensive lab work, and Recon 1 is one of the exercises used. A PacketWars competition is held on the last day, set up and run by a Packet Master, who is independent of the SISMAT program. A survey is given at the end of the two weeks to assess the exercises, including PacketWars, and how they affect student interest in security. The survey questions can be found in the Appendix. We also used Recon 1 in four security classes and we gave abbreviated surveys in those classes.

Hackathons lasting two days have been used twice to facilitate the development of exercises and infrastructure. The participants were faculty and students. The activities included design sessions for new scenarios and software architecture, implementation sessions for current scenarios; and testing sessions for implemented scenarios (Recon 1). An independent evaluator attended these hackathons, observed some sessions and interviewed participants.

Several workshops lasting 1.5 – 3.0 hours were offered to faculty. They were given a brief introduction and asked to play the Recon 1 exercise in pairs. The surveys focused on what faculty got from the workshop.

4.2 A Detailed View: The Recon 1 Exercise

The Recon 1 exercise was inspired by a largely similar execise from PacketWars. We chose it as the first exercise to implement because it is simple, we understood it, and it exhibits the characteristics of the types of scenarios we are interesting in deploying and supporting in our framework. This enabled us to concentrate on building the infrastructure without having to worry about specifying the scenario. A diagram of the network topology is shown in Figure 1.

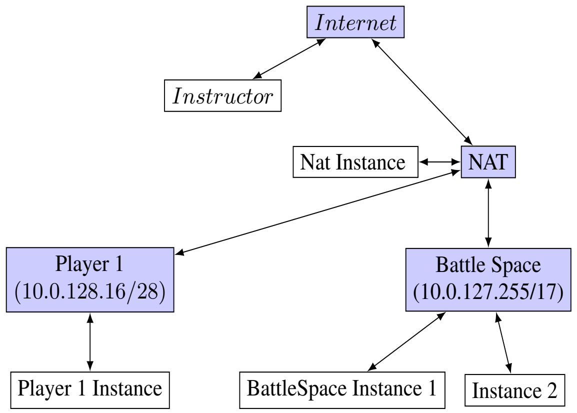

Figure 1: Conceptual diagram of the Recon I game. Note subnets are shaded (blue)

We have now run the Recon 1 exercise with over 120 students in 6 different settings spanning four different institutions as shown in the table below. The first roll out of the Recon 1 exercise was at the SISMAT 2013 program. Recon 1 was one of 6 lab experiences that the students participated in during the 10 day workshop. Student feedback indicated that there were problems. However, student responses on a post-class survey revealed that students felt that the exercise was worthwhile and that it increased their interest in cybersecurity education (one-tailed t-test found ratings well above neutral $p < .01$). This p-value indicates that the likelihood that we would have measured this result if the students had been selected at random from a population that was on average neutral is less than 1%. See Appendix for a list of the survey questions. Further investigation revealed what background information and skills the students needed. For example, a better understanding of network partitioning, TCP and UDP protocols, and more practice with using nmap. An updated version of Recon 1 was given again at SISMAT 2014, and qualitative results show that it was ranked second out of six compared with fourth out of six in 2013. The top-ranked exercise was still PacketWars, which was the only exercise run as a competitive game. The ranking of PacketWars over all other exercises was statistically significant using a Friedman test. Since the content was similar to Recon 1, this supports the claim that live competitions are more exciting than non-competitive exercises. However, we did not assess learning in this trial, and we did not control for instructor.

Eleven College students along with four professors spent two days in August 2013 in an intensive hackathon that included 2 trials of the Recon 1 exercise, once with advanced students with previous computer security coursework and once with students with no prior computer security training. This workshop uncovered several issues with remote configuration and set up of the infrastructure, that were subsequently fixed. We also learned that the prerequisite knowledge needed for students with no prior security training should include the OSI model for network layers.

The Recon 1 exercise has also been piloted with 85 students in four classes at three schools. In one of these courses (CS 495), student surveys conducted at the end of the semester indicate that students found the Recon 1 exercise worthwhile in their learning (M=5.25 on a 7 point likert scale, $p < .005$). **Conclusion: The Recon 1 exercise was engaging for students in a variety of settings.**

Along with testing the Recon 1 exercise with over 120 students, we also held workshops for 29 faculty at three different conferences. As you can see in the table below, we targeted a variety of institutions, including instructors at two year colleges, small liberal arts colleges and research universities. Our workshop attendees also differed widely in experience with respect to teaching Computer Security courses, some had no experience yet while others had taught several different courses at both undergraduate and graduate levels. At each conference, we made adjustments based on feedback and challenges experienced in the how to present the Recon 1 Scenario. We also gave post workshop surveys at each conference. At each conference participants felt that taking the workshop increased their interest in the topic of Cyber Security (all $p < .02$). At our SIGCSE workshop, some felt that Recon 1 was too challenging, while others said it was easy. **Conclusion: Recon 1 did not meet the needs of some faculty in that it was too challenging for them without additional preparation.**

Table 2: Classes and workshops. The cost of each event in terms of cloud usage was less than $20. Each VM costs $0.013 per hour. We ran 30 VMs for 48 hours for the hackathons.

Date	Site	Audience	Attendance	student or faculty
June 2013	Dartmouth	SISMAT	12	student
Aug 2013	Lewis & Clark	Hackathon	11	student
Oct 2013	CCSC-NW	workshop for faculty (Liberal Arts)	8	faculty
Nov 2013	Evergreen	Network Security class	40	student
Jan 2014	MPICT	faculty (2-year Colleges)	7	faculty
Feb 2014	Lewis & Clark	CS 495 Cybersecurity	19	student
Feb 2014	Univ. of Calgary	CPSC 601 Seminar: Security Analysis	4	grad student
March 2014	Univ. of Calgary	CPSC 525 Network Security	25	student
March 2014	SIGCSE	faculty workshop (Broad Scope)	14	faculty
May 2014	Lewis & Clark	Hackathon	15	student & faculty
June 2014	Dartmouth	SISMAT	17	student
Fall 2014	Wellesley	CS342	28	student
Fall 2014	Lewis & Clark	CS393 Networking	20	student
Oct 2014	CCSC-NW	workshop for faculty (Liberal Arts)	14	student & faculty

4.3 A Detailed View: ScapyHunt

ScapyHunt has been tested twice. Once with a small class of graduate students together with Recon 1, and a second time at SISMAT 2014. The exercise provides a login prompt and little else beyond the directive "find the hidden resource in this hidden network topology." Students found it to be significantly harder than Recon 1, and based on this feedback we are designing an introductory level. They also expressed a wish for a simple canned demo or hint to start off with. There is a constant stream of network data that students must analyze, and the software defined network (SDN) has a complex topology with multiple subnets, which students must discover and diagram. This addresses both analysis of complex data and a complex network topology.

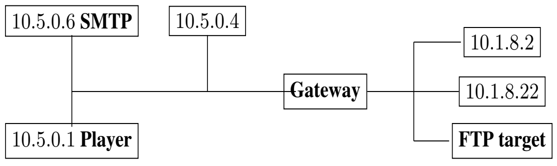

Figure 2: *The ScapyHunt Topology*. Students must discover this topology by crafting packets and observing how the network reacts.

The students required prompting during most of the interaction. For example, the students needed prompting to open a terminal and use standard network command line tools and utilities (e.g., Wireshark) to discover network information. They quickly fixated on nmap, although nmap is ultimately of little utility for this exercise. To us, this confirms that we need an exercise or demo to bridge the gap between Recon 1 and ScapyHunt. We prompted students to both "write" to the network (via ping, nmap, netcat, and some packet-crafting tools) and simultaneously "read" from the network to observe both their own actions and the actions of the entities in the software defined network (SDN). They also did not use any notes, drawings, or scripts. This kind of mistake is unlikely to be repeated when they repeat the exercise because they saw how "big" the task was in terms of the amount of information generated by the tools they eventually used.

5. DISCUSSION AND LESSONS LEARNED

Based on informal feedback from faculty, there is a demand for hands-on cybersecurity exercises, and they want them to be easy to use. This demand seems to come from a wide range of faculty, and they have expressed interest in our exercises.

We learned from surveys, using a 7-point likert scale with 4 as neutral, at CCSC-NW 2013 (CC) and SIGCSE 2014 (SI), that there was significant interest in using these exercises 1) as an introduction to a topic to stimulate further interest and lay the groundwork for class ($p(>4)=.01$ for both); 2) as a tool to use during lecture/instruction to help teach the key concepts ($p(>4)= .02$ CC, $p(>4)= .01$ SI); and 3) to test students' knowledge ($p(>4)= .002$ CC, $p(>4)= .004$ SI). Thus, faculty felt most strongly that EDURange could be used for assessment.

We learned from our classes and SISMAT that hands-on exercises provide a starting point for discussing important topics. For example, Recon 1 can be an opportunity to discuss the OSI model, subnet masking, broadcast addresses, even using the command line. Based on survey results, students were positive that the exercises were worthwhile and contributed to their learning. We conclude from this that they were able to use our exercises for formative self-assessment. Unlike with exams, where students do not want to admit what they don't know, with hands-on exercises such as Recon 1, students were able to reflect on what they didn't know in the context of what they wished they had known when trying the exercise. Since Recon 1 is a very focused exercise, students were able to identify at SISMAT and hackathons the need for more tutorials and canned demos on specific topics, such as TCP, ICMP and subnetworks. With exercises such as Recon 1, there is no way to fake the knowledge needed. On the other hand, we learned that the game must be at the appropriate level for the audience. If students haven't studied networking and don't have experience with the command line, they will have difficulty and may get frustrated. In response, we have started to create some tutorials and "level zero" versions.

6. CONCLUSION AND FUTURE WORK

Our framework provides a scalable infrastructure to an audience of instructors that have few local resources or capacity to set up complex systems. By using a public industry "best of breed" cloud, EDURange is unique and cost effective, and avoids some of the limitations associated with dedicated testbeds. Many of our student and instructor audiences were positive about its potential.

While our framework has significant advantages as a teaching platform over existing infrastructure, it is not intended to replace such environments (The RAVE, DETERlab). Our main focus is on providing dynamic, flexible cybersecurity scenarios that teach analysis skills (rather than toolsets or specific attacks). While the exercises we designed lead to the analysis of large, opaque artifacts, our framework can also incorporate exercises that teach background knowledge. We believe that support for customizing scenarios represents a natural evolution of cybersecurity education infrastructure. We have plans to add a scoring mechanism which will make it possible for the system to give continuous feedback to students and provide summative assessment data to faculty.

7. REFERENCES

[1] BRATUS, S. What hackers learn that the rest of us don't: Notes on hacker curriculum. *IEEE Security and Privacy 5* (2007), 72–75.

[2] BRATUS, S., D'CUNHA, N., SPARKS, E., AND SMITH, S. W. Toctou, traps, and trusted computing. In *Trusted Computing-Challenges and Applications*. Springer, 2008, pp. 14–32.

[3] BRATUS, S., SHUBINA, A., AND LOCASTO, M. E. Teaching the principles of the hacker curriculum to undergraduates. In *Proceedings of the 41st ACM technical symposium on Computer science education* (New York, NY, USA, 2010), SIGCSE '10, ACM, pp. 122–126.

[4] CAPPOS, J., AND WEISS, R. Teaching the security mindset with reference monitors. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2014), SIGCSE '14, ACM, pp. 523–528.

[5] CONE, B. D., IRVINE, C. E., THOMPSON, M. F., AND NGUYEN, T. D. A video game for cyber security training and awareness. *Computers & Security 26*, 1 (2007), 63 – 72.

[6] DENNING, T., LERNER, A., SHOSTACK, A., AND KOHNO, T. Control-alt-hack: the design and evaluation of a card game for computer security awareness and education. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (2013), ACM, pp. 915–928.

[7] DOUPÉ, A., EGELE, M., CAILLAT, B., STRINGHINI, G., YAKIN, G., ZAND, A., CAVEDON, L., AND VIGNA, G. Hit 'em where it hurts: A live security exercise on cyber situational awareness. In *Proceedings of the 27th Annual Computer Security Applications Conference* (New York, NY, USA, 2011), ACSAC '11, ACM, pp. 51–61.

[8] DU, W., AND WANG, R. Seed: A suite of instructional laboratories for computer security education. *J. Educ. Resour. Comput. 8* (March 2008), 3:1–3:24.

[9] ENSAFI, R., JACOBI, M., AND CRANDALL, J. R. Students Who Don't Understand Information Flow Should Be Eaten: An Experience Paper. In *Proceedings of the 5th USENIX conference on Cyber Security Experimentation and Test (CSET'12)* (2012), pp. 10–10.

[10] FITE, B. Simulating cyber operations: A cyber security training framework, 2014. http://www.sans.org/reading-room/whitepapers/bestprac/simulating-cyber-operations-cyber-security-training-framework-34510.

[11] FRYER-BIGGS, Z. DoD faces cyber expert talent shortage. *Computer 33*, 12 (2000), 52–59.

[12] GAVAS, E., MEMON, N., AND BRITTON, D. Winning cybersecurity one challenge at a time. *Security & Privacy, IEEE 10*, 4 (2012), 75–79.

[13] GONDREE, M., AND PETERSON, Z. N. Valuing security by getting [d0x3d!]: Experiences with a network security board game. In *Presented as part of the 6th Workshop on Cyber Security Experimentation and Test* (Berkeley, CA, 2013), USENIX.

[14] MCGETTRICK, A., CASSEL, L. N., DARK, M., HAWTHORNE, E. K., AND IMPAGLIAZZO, J. Toward curricular guidelines for cybersecurity. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2014), SIGCSE '14, ACM, pp. 81–82.

[15] PETERSON, P. A., AND REIHER, P. L. Security exercises for the online classroom with deter. *Proc. of the 3rd USENIX CSET* (2010).

[16] SAHAMI, M., GUZDIAL, M., MCGETTRICK, A., AND ROACH, S. Setting the stage for computing curricula 2013: computer science–report from the acm/ieee-cs joint task force. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (2011), ACM, pp. 161–162.

[17] TURNER, C. F., TAYLOR, B., AND KAZA, S. Security in computer literacy: a model for design, dissemination, and assessment. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (New York, NY, USA, 2011), SIGCSE '11, ACM, pp. 15–20.

[18] VIGNA, G. Teaching Network Security through Live Exercises. In *Proc. 3rd Ann. World Conf. Information Security Education (WISE 03)* (2003), Kluwer Academic, pp. 3–18.

[19] WEISS, R., MACHE, J., AND NILSEN, E. Top 10 hands-on cybersecurity exercises. *Journal of Computing Sciences in Colleges 29*, 1 (2013), 140–147.

[20] WHITE, G., AND NORDSTROM, G. Security across the Curriculum: Using Computer Security to Teach Computer Science Principles. In *Proceedings of the* 19^{th} *National Information Systems Security Conference* (1996), NIST, pp. 483–488.

[21] ZHUANG, Y., GESSIOU, E., PORTZER, S., FUND, F., MUHAMMAD, M., BESCHASTNIKH, I., AND CAPPOS, J. Netcheck: Network diagnoses from blackbox traces. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI'14), USENIX.*

APPENDIX

Survey given to SISMAT students

The following questions were given at the end of the two-week intensive training. A seven-point Likert scale was used, except for question 3 which asks for a ranking.

1. What was your level of interest in each exercise?
2. How many hours did you spend on each exercise?
3. Rank the activities from most interesting to least interesting
4. The time spent on the activity was worthwhile
5. The activity contributed to my overall understanding of the material
6. Preparation (reading, lecture) were sufficient for me to successfully understand the lab
7. What was the level of difficulty?