T1B2 1555

Submitted to the 2004 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 11–12 June 2004

Developing and Delivering Hands-On Information Assurance Exercises: Experiences with the Cyber Defense Lab at UMBC

Alan T. Sherman, Brian O. Roberts, William E. Byrd, Matthew R. Baker, John Simmons

Abstract—

In summer 2003, we developed four new hands-on information assurance educational exercises for use in the UMBC undergraduate and graduate curricula. Exercise topics comprise buffer overflow attacks, vulnerability scanning, password security and policy, and flaws in the Wired Equivalent Privacy (WEP) protocol. During each exercise, each student carries out structured activities using a laptop from a mobile cart that can be rolled into any classroom. These dedicated, isolated machines permit a student to make mistakes safely, even while acting as the system administrator, without adversely affecting any other user. Each exercise is organized in a modular fashion to facilitate varied use for different courses, levels, and available time. Our experiences delivering these exercises show that practical hands-on activities motivate students and enhance learning. In this paper we describe our exercises and share lessons learned, including the importance of careful planning, ethical considerations, the rapid obsolescence of tools, and the difficulty of including exercises in already busy courses.

Keywords—

Buffer overflow, computer security education, cyber defense exercises, information assurance education, password security, scanning vulnerabilities, UMBC Cyber Defense Lab, Wired Equivalent Privacy (WEP).

I. Introduction

Many students learn more effectively and efficiently, and become more motivated, when they engage actively in hands-on problem solving. Building on this fact, in summer 2003, we designed and delivered four new hands-on educational exercises in *Information Assurance (IA)* for use in the undergraduate and graduate curricula at the *University of Maryland, Baltimore County (UMBC)*. Exercise topics comprise protection against buffer overflow attacks, vulnerability scanning, password security and policy, and insecurity of the *Wired Equivalent Privacy (WEP)* protocol. To carry out each exercise, each student uses a laptop from a mobile cart that can be pushed into any classroom. In this paper, we describe our exercises and our experiences developing and delivering them.

We designed each exercise to be self-contained, modular, flexible, and to fit into one or more courses within our curricula. Each exercise includes background material, problem-solving activities, discussion questions, and supporting software and instructions for the instructor. The flexible modular nature of the exercises permits their use in class periods of various lengths, and with students at various experience levels. Typically, each exercise can be delivered in one to four class meetings.

Although the idea of hands-on laboratory exercises is not new, relatively few such exercises have been developed for IA. Our goal is to enhance, promote, and facilitate IA education by contributing some additional useful exercises.

IA exercises present some special challenges stemming from the essential distinguishing characteristics of IA: adversarial thinking, attention to detail, security paranoia, dual offensive and defensive uses of technologies, and a holistic practical view toward the design, development, and management of enterprise security, safety, and reliability. To prevent inadvertent damage to other systems, exercises that involve dangerous programs (*e.g.*, worms, viruses, and attack tools) must be safely isolated. It is extremely beneficial to enable students to learn how to manage systems by direct experience, including making mistakes, and recovering from them, while acting as administrator of an actual system. Therefore, each student needs an isolated system on which he or she can safely operate without adversely affecting others. Also, through education, promotion of ethical considerations, and appropriate selection of material, the exercises must balance the benefits of offensive knowledge in strengthening defense with the dangers of facilitating antisocial behavior.

Our informal experiences demonstrate that our exercises strongly attract student attention, help students learn important concepts and practical tools, and motivate students to pursue the topics further. Our experiences also show that significant effort is required to prepare and deliver such exercises. We hope that others may benefit from our

Alan Sherman is Associate Professor in the Department of Computer Science and Electrical Engineering, and Director of the UMBC Center for Information Security and Assurance at the University of Maryland, Baltimore County, Baltimore, MD. Email: sherman@umbc.edu

Brian Roberts, Matt Baker, and John Simmons are graduate students in the Department of Computer Science and Electrical Engineering at the University of Maryland, Baltimore County, Baltimore, MD. Email: {roberts2, mbaker5, js5}@umbc.edu

William Byrd, formerly of the University of Maryland, Baltimore County, is a graduate student in the Computer Science Department at Indiana University, Bloomington, IN. Email: webyrd@indiana.edu

Support for this research was provided in part by the Department of Defense under Contracts MDA904-02-1-0207 and MDA904-03-1-0208.

ISBN 0-7803-9814-9/$10.00

exercises and lessons learned.

The rest of this paper is organized in six sections. Section II reviews previous related work. Section III describes the history and purpose of UMBC's Cyber Defense Lab, and provides additional context for the exercises. Section IV describes our four new exercises. Section V reviews how we tested the exercises internally, and describes our initial deployment of the vulnerability scanning exercise. Section VI discusses how well our exercises worked, along with the lessons we learned. Finally, Section VII summarizes our conclusions.

II. Previous Work

Previous relevant research on information assurance education includes other IA exercises, along with suggestions for IA curricula, configuring cyber defense labs, and conducting cyber defense competitions. In addition, there is a large body of research and experience not associated with IA on conducting educational labs, carrying out defense training, and evaluating the benefits of hands-on learning activities. We now selectively review some of this previous work.

Hoffman, *et al.* [1] emphasize the importance of applying knowledge learned in the classroom in hands-on exercises at The George Washington University, where they too have a mobile cart of laptops.

Conti, *et al.* [2] offer suggestions for a comprehensive undergraduate program in IA, including labs.

Schepens, Ragsdale, *et al.* [3], [4] document their experiences at the United States Military Academy with the cyber defense competitions among the military academies, held for the past three years. These Cyber Defense Exercises, and their associated labs and undergraduate courses, stand as an excellent complex model example for how to promote learning through intercollegiate competition [5], [6], [7], [8].

Moore, Williams, and McCain [9] show how traditional defense intelligence planning crosses over into the realm of cyber defense planning.

The *National Information Assurance Training and Education Center (NIATEC)* organizes a repository of shared courseware and modules [10].

While developing our four exercises, we were heavily influenced by existing research on buffer overflow attacks [11], [12], [13], [14], [15], vulnerability scanning, password security [16], and security of the WEP protocol [17].

III. UMBC Cyber Defense Lab

The UMBC *Center for Information Security and Assurance (CISA)* has a three-fold mission of promoting research, education, and best practices in IA. A highlight of the center is its *Cyber Defense Lab (CDL)* which includes an isolated stationary network of desktop computers, switches, and routers, in addition to a mobile cart of laptops. UMBC is a Center of Academic Excellence in IA, as designated by the *Department of Defense (DoD)*; CISA is supported by DoD and Cisco.

The mobile component of the lab consists of thirty Dell Latitude D600 laptops loaded with both the Windows XP and Redhat Linux 9 operating systems, and equipped with Pentium M processors, 256MB of RAM, 30GB hard drives and Dell TruMobile 1400 wireless cards. In addition, Orinoco Classic Gold 802.11b cards are used to provide wireless connectivity in GNU/Linux. The mobile lab is housed in a secure rolling containment cart designed to store and power thirty laptops. The mobile CDL permits cyber defense exercises to be run in any classroom.

IV. Four New Exercises

Our four exercises deal with buffer overflow attacks, password security and policy, vulnerability scanning, and insecurities in the WEP protocol. These important, timely, diverse topics cover the essential triad of people, policies and procedures, and technology, and match interests of the four student lab assistants who developed the exercises.

In the rest of this section we describe our four exercises in terms of their learning objectives, intended audience, exercise requirements, background material, instructional activities, instructor preparation, and implementation notes. For more detailed descriptions of these exercises, see Baker [18], Byrd [19], Roberts [20], and Simmons [21].

To place the lab exercises in context, each exercise should be preceded by a classroom lecture on the IA topic to be explored. After the exercise, each student completes a detailed lab report. If time permits, the instructor should engage participants in a post-lab discussion, to provide students with additional opportunities to ask questions and examine ideas presented during the exercise.

A. Exercise 1: Buffer Overflow Defense

According to Bartaloo [15], recent statistics from Carnegie Mellon's Computer Emergency Response Team show that buffer overflow vulnerabilities account for approximately 50% of all reported security vulnerabilities. Most buffer overflow attacks compromise the victim application by overwriting the return address of the currently executing function with an address that points to malicious code stored at the beginning of the buffer. Typically the malicious code spawns a command shell with `root`-level access. Other types of buffer overflow attacks are possible, such as application-specific attacks that overwrite local variables located higher up on the stack, corrupting the data used by the victim application.

A.1 Learning Objectives

Students will be able to recognize buffer overflow vulnerabilities in their programs, and will be able to identify

ISBN 0-7803-9814-9/$10.00

the advantages and disadvantages of both canary based and non-canary based buffer overflow defense techniques, as discussed in Cowan, *et al.* [13]. After completing this exercise, students will be able to describe how a stack smashing attack occurs. Students will gain hands-on experience using IBM's *Stack Smashing Protector (SSP)* [12], a modified version of the GCC compiler, to protect a C program against a stack smashing attack. After completing the exercise, students will write a two- to three-page report answering questions about their experiences.

A.2 Intended Audience

This exercise is intended primarily for undergraduate computer science students with intermediate programming experience, such as those in UMBC's *Software Design and Development*, *Computer Organization and Assembly Language*, *Principles of Programming Languages*, and *Data and Network Security* courses. Some experience with assembly language programming would be beneficial.

This exercise also can be modified for use at the graduate level by requiring each student to design and implement a canary-based stack protection mechanism.

A.3 Exercise Requirements

This exercise is broken into three parts, each of which can be completed in approximately forty-five minutes. When working under strict time constraints, the instructor may wish to conduct only the first one or two parts of the exercise. This exercise should be preceded by a one-hour lecture on buffer overflow exploits and canary based defense mechanisms.

Each student conducting the exercise will require one Dell Latitude D600 laptop with the exercise image preloaded.

A.4 Background Material

An introduction to the concepts covered in the lab component of the exercise will be presented during the lecture so it is not necessary for students to have in-depth knowledge of stack smashing attacks. Students should have basic knowledge of canary based defenses, as presented in lecture, and should be familiar with the C programming language and both a command shell and a text editor available on GNU/Linux.

A.5 Instructional Activities

Students first examine a C program, `withdraw.c`, and explain how the program behaves. Then they compile and run the program without compiler-based stack protection, and observe the effect of a successful canned stack smashing attack on the program. Next, the students predict the effect that compiler-based stack protection will have on their program. They recompile the program using SSP, launch the same stack smashing attack, and compare the actual behavior of the program with their predictions.

Afterward, students examine the source code of another C program, `easy_grader.c`, and describe how this program works. They predict how the program will behave when passed a certain malicious input string, and then compare their prediction against the actual behavior of the program. Next the students recompile the C program with various compiler optimization flags, and predict the effects these optimizations will have when the program is given the malicious input string. Once again, the students compare the actual behavior of the program with their predictions. Finally the students recompile the C program using both the optimization and stack protection options provided by SSP, and once more compare the predicted and actual behavior of the program when attacked. The students then explain how these experiments illustrate the limitations of canary based protection mechanisms.

A.6 Instructor Preparation

The instructor prepares the laptops for use in this exercise by loading a preconfigured exercise image. The instructor must also lecture on buffer overflow attack and defense prior to holding the exercise. The instructor is responsible for collecting and grading the lab reports.

A.7 Implementation Notes

Due to frequent changes in operating systems and compilers, it is necessary to review this exercise once a semester, and to update the exercise and associated software when appropriate.

B. Exercise 2: Vulnerability Scanning

Vulnerability scanning is an important part of defensive systems administration and security research because it automates and facilitates the discovery and patching of security holes. Exercises that provide hands-on experience with vulnerability analysis give students the opportunity to gain familiarity with common system weaknesses as well as to develop useful analytical skills.

B.1 Learning Objectives

This exercise focuses on student discovery and remediation of vulnerabilities commonly found in default installations of two popular operating systems (Microsoft Windows XP and Redhat Linux 9) by introducing real world applications designed to scan for vulnerabilities (*nmap*, *nessus*, *SARA*, and *Microsoft Baseline Security Analyzer*). Exercises are run individually or in small groups to maximize hands-on experience. After completing the exercise, the students explain results of their scans and how they fixed the vulnerabilities. The students also provide written answers to questions about the vulnerability scanning

process. After conducting this exercise, students will have a familiarity with scanning tools and procedures.

B.2 Intended Audience

This exercise is designed for computer science students with beginning to intermediate experience with operating systems and little or no experience with systems administration. Students should also have a basic understanding of programming concepts in order to appreciate how improper programming can lead to vulnerabilities. Appropriate UMBC undergraduate courses are *Computer Networks* and *Data and Network Security*, while appropriate graduate courses include *Networking Technologies* and *Information Assurance*.

B.3 Exercise Requirements

This exercise can be completed in its entirety in approximately six hours, while an abbreviated version of the exercise can be conducted in less than one hour. This exercise should be preceded by one or two one-hour lectures on system vulnerabilities and vulnerability scanning.

The original hardware requirement for this exercise is one Dell Latitude D600 laptop for each student or student group participating. It is possible to use different hardware, but we cannot guarantee that our preconfigured images will work in any other machine than the D600. Additionally, a license is required for Windows XP and any proprietary software included with the Dell D600.

B.4 Background Material

Students should be familiar with the basic concepts of security vulnerabilities and how they can be exploited to gain unauthorized or elevated access to a system.

B.5 Instructional Activities

Students conducting this exercise boot their machines into the GNU/Linux operating system and perform vulnerability scans using the *nessus* and *SARA* scanning applications. Students then develop strategies to resolve identified vulnerabilities; these strategies might include updating or reconfiguring system software and disabling unused services. Once students have implemented their strategies, they rescan their systems to test the effectiveness of their solutions. After they have eliminated security holes in GNU/Linux, students boot into the Windows XP partition and repeat the process using the *Microsoft Baseline Security Analyzer*.

B.6 Instructor Preparation

The instructor loads each machine with the exercise image. The instructor also gives each student participating in the exercise a copy of the instructions, requirements and questions. After the students complete the exercise the instructor compares student results against the original list of vulnerabilities. The instructor also collects and grades the lab reports.

B.7 Implementation Notes

This exercise is designed to simulate a real world environment and to present students with common tasks that systems administrators must perform when checking for vulnerabilities. The configuration process for this exercise consists of installing both Windows XP and Redhat Linux 9 on a laptop and running the analysis programs to develop a list of vulnerabilities that exist upon initial installation. After this installation is complete, the drive is imaged and prepared for distribution with the exercise.

C. Exercise 3: Password Security

The topic of password security encompasses the IA themes of people, policies and procedures, and technology. Passwords are easily understood by even beginning computer science students, and indeed practically anyone in modern society, yet most people use passwords poorly. Moreover, passwords are an important security topic as they are the primary user authentication mechanism in many information systems today.

C.1 Learning Objectives

Students will become familiar with selected current best practices, policy design issues, and available tools in password security, including strategies for generating strong passwords.

This exercise comprises three parts. In the *best practices* portion of the exercise, students explore issues in restricting access to the password file, controlling password aging, and locking accounts due to login failures. In the *policy design* portion, students investigate password length and composition rules, and human factors that complicate these issues. In the *tools* portion of the exercise, students are exposed to the *Linux-PAM (Pluggable Authentication Modules)* environment as well as *John the Ripper*, a dictionary attack tool for security administrators.

C.2 Intended Audience

This exercise is intended for beginning to advanced computer science students such as those in UMBC undergraduate courses *Computer Science I for Majors*, *Computer Science II for Majors*, *Computer Networks*, *Data and Network Security*, and in the graduate *Network Technologies* course. While not specifically required, students who have completed basic programming and operating systems coursework will be more able to grasp some of the issues in security administration. Previous experience with system administration is not necessary.

C.3 Exercise Requirements

This exercise can be completed in approximately two hours, and should be preceded by a one-hour lecture on password security.

C.4 Background Material

Since students implement and enforce password security policies in the lab, it will be beneficial for them to be exposed to current best practices in a prior lecture. Modern alternatives to traditional password authorization should also be discussed, including time-varying passwords, biometric authentication, physical tokens, and zero-knowledge proofs of authentication.

C.5 Instructional Activities

The instructor divides the class into small teams of one to three students each, and each team is given one computer to administer. Each team designs and implements a password policy for their system. Some aspects of the password policy may be enforced by the Linux-PAM, while others may be communicated to users via written instructions (*e.g.*, advice on choosing new passwords).

Next, teams swap computers and assume the role of users on another team's system. They create a number of account passwords on the other team's system, and provide feedback on the friendliness of the user experience.

Finally, the strength of each newly created password is evaluated using a dictionary attack. Students then discuss user feedback and attack results in a lab report, and evaluate the success of their password policies.

C.6 Instructor Preparation

Before the exercise, the instructor loads the laptop images, and provides students with the lab instructions and background lecture. We provide lab instructions for students and instructors, as well as a set of lecture slides and references. After the lab, the instructor collects and grades student lab reports. We also provide suggestions for grading criteria.

C.7 Implementation Notes

This exercise is designed to give students experience with a modern password security system from both the user and administrator perspectives. Instructions are tailored to a specific environment, and will need to be updated as password systems evolve.

D. Exercise 4: Breaking Wired Equivalent Privacy

With the profusion of wireless networks, wireless security is an integral part of defensive systems administration and security research. This exercise provides hands-on experience with weaknesses of the *Wired Equivalent Privacy* (WEP) protocol and strategies for addressing these weaknesses.

D.1 Learning Objectives

This exercise focuses on student discovery and remediation of vulnerabilities commonly found in the 802.11b protocol by introducing a real world packet analyzer (*AirSnort*) designed to exploit the vulnerabilities of the WEP protocol. Exercises are run individually or in small groups to maximize hands-on experience. After completing the exercise, each student writes a lab report describing how AirSnort exploits the flaws of WEP, analyzing results of the attack, and explaining how to modify the WEP protocol in order to make it more secure. Each student also provides written answers to questions about the vulnerabilities of WEP. After conducting this exercise, students are familiar with WEP, WEP's flaws, and alternatives as well as additions to WEP that provide greater security than does WEP alone.

D.2 Intended Audience

This exercise is designed for computer science students with beginning to intermediate experience with networking and little or no experience with operating systems, such as students in UMBC's undergraduate *Computer Networks* course. Students should also have a basic understanding of cryptography in order to appreciate how improper use of cryptographic schemes can lead to security flaws.

D.3 Exercise Requirements

This exercise can be completed in approximately three hours, and should be preceded by a one-hour lecture on WEP and wireless security.

The hardware required for this exercise is one Dell Latitude D600 laptop and a Proxim ORiNOCO Classic Gold PC Card for each participating student or student group. It is possible to use different hardware, but we cannot guarantee that our preconfigured images will work in any machine other than the D600 utilizing the ORiNOCO Classic Gold Card. Also required is a specific ORiNOCO driver that provides monitor mode functionality and a WEP-encrypted 802.11b access point over which the instructor has control.

D.4 Background Material

Students should be familiar with the basic concepts of the 802.11b protocol and how it can be exploited, violating authenticity, confidentiality, or integrity of data.

D.5 Instructional Activities

Students use the AirSnort wireless packet sniffer to collect, and then analyze, packets from a WEP-enabled 802.11b network. After analyzing the collected packets, and recovering the WEP key, students record their results in their lab reports. Each report also contains answers to written questions about WEP, the student's description of why WEP is insecure, and the student's explanation of how the security of WEP can be improved.

ISBN 0-7803-9814-9/$10.00

The exercise can be shortened by having each student examine only a small number of intercepted packets and then complete the key recovery using a large file of intercepted packets provided by the instructor.

D.6 Instructor Preparation

The instructor loads each machine with the exercise image. The instructor also gives each student participating in the exercise a copy of the instructions, requirements and questions. After each student completes the exercise, the instructor compares the student's results against the original list of WEP keys. The instructor also collects and grades the students' lab reports and answers to the WEP-related questions.

D.7 Implementation Notes

This exercise is designed to simulate a real world environment and to present students with some of the issues system administrators face when deploying wireless networks. The configuration process for this exercise consists of installing both Windows XP and Redhat Linux 9 on a laptop, configuring the wireless card, and modifying the wireless card to operate in monitor mode. After this initialization is completed, the AirSnort program is tested to determine if it can, in fact, retrieve a WEP key. Finally, the drive is imaged and prepared for distribution with the exercise.

As we learned from experience, care must be taken not to use new wireless components that correct weaknesses of WEP.

V. Delivering the Exercises

During summer 2003, each CDL lab assistant performed a trial run of his exercise, with the other three lab assistants assuming the role of undergraduate students. Each exercise was then improved, based on the results of this internal testing. For example, the buffer overflow exercise was revised to make more explicit the effects of compiler optimizations on code protected by the SSP compiler.

The first classroom use of one of our exercises occurred in fall 2003, when Sherman and Roberts ran the vulnerability scanning exercise with a group of students enrolled in CMSC-652, *Cryptology and Data Security.* Sherman and Roberts again deployed this exercise in spring 2004 in CMSC-491/691, *Information Assurance.*

A. Deployment in CMSC-652

In CMSC-652, students read and present recent research papers and complete an original research project of their choice. Historically, CMSC-652 had not included hands-on projects or exercises.

All ten students were graduate students, with seven students enrolled in Master's programs, and three student's enrolled in PhD programs. Nine of the ten students were computer science students, while the tenth was a mathematics student with little formal training in computer science.

Sherman was concerned that the seventy-five minute class period would not be sufficient for completing the vulnerability scanning exercise. Especially for such a mathematical course, he was reluctant to hold the exercise during the standard lecture period, since doing so would displace important material. Sherman therefore decided to schedule the exercise as an optional activity to be held on a Friday afternoon, outside of the scheduled lecture time.

To provide additional incentive for students to participate in the optional exercise, Sherman offered a free copy of Bishop's new textbook [22],[1] as a prize for the student with the best exercise solution.

All ten students arrived on the afternoon of the exercise, and all ten students participated until the end of the exercise. Each student worked individually, and was issued a preconfigured laptop. Unfortunately, only nine laptops had been configured in advance, and Roberts had to spend several minutes setting up a "fresh" laptop for the tenth student. All ten students then had to finish the Windows XP setup process and assign unique hostnames to each laptop, which required another ten minutes. The start of the exercise was thereby delayed by approximately twenty minutes.

The students required approximately seventy-five minutes to complete the vulnerability scanning exercise, beyond the initial twenty minute delay. Several students finished the exercise early and spent the remainder of the time discussing with one another the scanning techniques they had used.

Roberts and Sherman were pleasantly surprised by the detailed knowledge of vulnerability scanning displayed by the computer science students during the exercise. Although the lone mathematics student was not as knowledgeable about the subject as his computer science peers, he also was able to complete the exercise. In fact, the most common suggestion offered by the students was to make future versions of the exercise more difficult.

Sherman and Roberts found the students to be highly engaged, inquisitive and attentive throughout the exercise. Students found the exercise a welcome relief from analyzing papers on Elliptic Curve cryptography and other mathematical subjects.

B. Deployment in CMSC-491/691

In spring 2004, Sherman scheduled four regular class periods for hands-on exercises in his information assurance course, CMSC-491/691. As of this writing, the scanning vulnerability exercise has been conducted. Ten students participated, including five undergraduates, three Master's students, and two PhD students; all were computer science or computer engineering majors. At the end of the

[1] Suggested retail price, $79.99.

exercise, each student filled out an anonymous two-page feedback form that we had developed.

Students were highly interested in the exercise, as evidenced by their enthusiastic expressions, concentration, comments on the feedback form, and spirited discussion after class. For example, on a scale from one (low) to five (high), seven of the ten respondents circled 'five' when asked if they would like to participate in similar exercises in the future (two students circled 'four', and one student circled 'three').

VI. Discussion & Lessons Learned

In this section we reflect on several issues that were significant during the development and delivery of our exercises. Although our observations focus on IA education at UMBC, we hope our experiences will prove helpful to people at other institutions.

Because of the dual-use nature of IA-related tools and techniques, the creation and deployment of IA exercises raise various ethical and legal issues. When we ran the vulnerability scanning exercise in fall 2003, many of the students worked for the Federal Government, or were employed in the information technology departments of corporations. These students were already aware of the legal and ethical considerations of vulnerability scanning. Even so, we informed the class of UMBC's zero tolerance policy toward students who complete an IA course and then abuse their network privileges. We were even more more explicit when discussing ethical issues with students in CMSC-491/691, many of whom were undergraduates. In the future we will require students to read and sign an ethics form, similar to the form used at the United States Military Academy, before participating in any IA exercise.

Perhaps the greatest obstacle to the adoption of our exercises at UMBC and other institutions is the overwhelming amount of material most instructors choose to present each semester. Many of the computer science instructors we have approached would like to include IA exercises in their courses, but are reluctant to eliminate existing lecture material in favor of these exercises. We feel, however, that the quality of many courses can be enhanced by covering less material more thoroughly and including hands-on problem solving activities. We also believe that many computer science courses ought to include regularly scheduled required labs, similar to the labs held in conjunction with physics and chemistry courses.

For classes that do not include lab sections, scheduling IA exercises as optional class activities outside of the standard lecture times may appeal to instructors who want to include exercises in their courses without displacing other material. Sherman took this approach when scheduling the vulnerability scanning exercise in CMSC-652.

One of the difficulties of conducting labs is that it is very easy to loose precious instructional time due to technical difficulties. During the vulnerability scanning exercise, for example, both students and instructors spent the first twenty minutes of the lab configuring their laptops. Many technical problems can be avoided through careful testing of the exercise in the exact environment in which the exercise will be held. Careful planning, along with preconfiguration of extra laptops, also can reduce the danger of technical difficulties. Finally, it is important for instructors always to have a backup plan, in case it is not possible to complete the exercise for any reason.

The main purpose of our exercises is to help develop the high-level critical thinking skills our our students. It is a fundamental difficulty of concrete exercises, however, that the particular programs and tools used in an exercise eventually become obsolete. The high-level topics of our exercises are fundamental IA concepts, but our exercises still are not immune to the effects of time. Given the rapid rate at which information technology changes, we are required to update the tools used in our exercises on an ongoing basis.

Despite our attempts at care, and despite knowing of similar problems at other universities, we too were victims of equipment theft during our move to our new building. Whenever a lab has valuable computer equipment, significant attention must be paid to physical security, access controls, delivery procedures, and protocols for emergency situations such as fire alarms.

VII. Conclusions

The four exercises we developed for the UMBC Cyber Defense Lab cover a variety of important and timely IA topics. The vulnerability scanning exercise, the first of our exercises to be used in the classroom, received overwhelmingly positive reactions from students, who appreciated the practical, hands-on learning activities related to a useful and interesting topic. We look forward to deploying our other hands-on exercises, which we anticipate will similarly motivate students and help them develop a healthy concern for security details and adversarial threats. The mobile laptop cart will continue to facilitate use of our exercises, in a variety of different courses and physical locations.

Developing effective exercises, however, is nontrivial. To use class time efficiently, careful preparation is needed. To enable the exercises to be used by other instructors in different universities, the exercises must be sufficiently polished, debugged, robust, and well documented. Exercises must be maintained to keep current with changing technologies and best security practices. Care must be taken to deter theft of expensive equipment

Our future plans include deploying all four exercises at UMBC this spring, developing additional exercises and further incorporating them into the UMBC curriculum. Currently under development are three new exercises on firewalls, intrusion detection, and forensics. Future topics

might include spyware, worms and viruses, and recovery and response. Eventually, we hope to conduct cyber defense competitions similar to those carried out among the military academies, though perhaps on a smaller scale.

Given the high costs and difficulties of maintaining physical cyber defense labs and of developing quality educational software, there would be value in amortizing such costs over large populations of users through shared software, common minimal standards for such software, shared virtual cyber defense labs accessed via the Internet, and the commercialization of educational exercises.

Hands-on exercises, such as the ones we have developed, help students learn important IA concepts and techniques in a practical, motivational setting in which each student can safely manage his or her own computer system.

Acknowledgments

For helpful comments, we thank John Pinkston, Anupam Joshi, Jack Suess, Daniel Ragsdale, and the students who performed our exercises. All computer work was carried out at UMBC.

References

[1] L. J. Hoffman, R. Dodge, T. Rosenberg, and D. Ragsdale, "Information assurance laboratory innovations." http://www.itoc.usma.edu/documents/IALabInnovations.pdf.

[2] G. Conti, J. Hill, S. Lathrop, K. Alford, and D. Ragsdale, "A comprehensive undergraduate information assurance program," in *Security Education and Critical Infrastructures, IFIP TC11 / WG11.8 Third Annual World Conference on Information Security Education (WISE3), June 26-28, 2003, Monterey, California, USA* (C. E. Irvine and H. Armstrong, eds.), vol. 253 of *IFIP Conference Proceedings*, pp. 243–260, Kluwer, June 2003.

[3] W. J. Schepens, D. J. Ragsdale, and J. R. Surdu, "The Cyber Defense Exercise: An evaluation of the effectiveness of information assurance education," *The Journal of Information Security*, vol. 1, July 2002.

[4] W. Schepens, D. Ragsdale, J. Surdu, and J. Schafer, "The Cyber Defense Exercise," in *Proceedings of the 5th National Colloquium for Information System Security Education*, (Fairfax, VA), May 2001.

[5] D. Welch, D. Ragsdale, and W. Schepens, "Training for information assurance," *Computer*, vol. 35, pp. 30–37, Apr. 2002.

[6] W. Schepens, D. Welch, and D. Ragsdale, "A lesson in cyber defense," *Defense Systems International: Critical Information Systems*, June 2002.

[7] J. H. Schafer, D. J. Ragsdale, J. R. Surdu, and J. C.A. Carver, "The IWAR range: A laboratory for undergraduate information assurance education," *The Journal of Computing in Small Colleges*, vol. 16, no. 1, pp. 223–232, 2001.

[8] "CDX publications." Information Technology and Operations Center, United States Military Academy. http://www.itoc.usma.edu/CDX/publication.htm.

[9] R. A. Moore, J. K. Williams, and C. McCain, "Intelligence preparation of the information battlespace–A methodical approach to cyber defense planning," in *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, pp. 134–141, United States Military Academy, West Point, NY, June 2001.

[10] "IA course modules." National Information Assurance Training and Education Center. http://niatec.info/coursemodules.htm.

[11] Aleph One, "Smashing the stack for fun and profit," *Phrack Magazine*, vol. 7, no. 49, File 14, 1996. http://www.securityfocus.com/templates/archive.pike?list=1\&;date=1996-11-08\&msg=Pine.LNX.3.91.961109134601.15637b-100000@underground.org.

[12] H. Etoh, "GCC extension for protecting applications from stack-smashing attacks." http://www.trl.ibm.com/projects/security/ssp/.

[13] C. Cowan, P. Wagle, C. Pu, S. Beattie, and J. Walpole, "Buffer overflows: Attacks and defenses for the vulnerability of the decade," in *Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX 2000)*, IEEE Computer Society Press, Jan. 2000.

[14] C. Cowan, S. Beattie, J. Johansen, and P. Wagle, "Pointguard: Protecting pointers from buffer overflow vulnerabilities," in *Proceedings of the 12th USENIX Security Symposium 2003, August 4–8, 2003, Washington, DC* (USENIX, ed.), (Berkeley, CA, USA), pp. 91–104, USENIX, 2003. http://www.cse.ogi.edu/~crispin/pointguard_usenix_security2003.pdf.

[15] A. Baratloo, N. Singh, and T. Tsai, "Transparent run-time defense against stack-smashing attacks," in *Proceedings of the 2000 USENIX Annual Technical Conference (USENIX-00)*, (Berkeley, CA), pp. 251–262, USENIX Ass., June 18–23 2000.

[16] D. V. Klein, " 'Foiling the cracker'–A survey of, and improvements to, password security," in *Proceedings of the second USENIX Workshop on Security*, pp. 5–14, Summer 1990.

[17] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting mobile communications: The insecurity of 802.11," in *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)*, (New York), pp. 180–188, ACM Press, July 16–21 2001.

[18] M. Baker, "Breaking WEP: Cyber defense lab exercise." Dept. of CSEE, University of Maryland, Baltimore County, June 2003. http://www.cisa.umbc.edu/.

[19] W. E. Byrd, "Stack smashing defense: A buffer overflow lab exercise." Dept. of CSEE, University of Maryland, Baltimore County, Feb. 2004. http://www.cisa.umbc.edu/.

[20] B. O. Roberts, "An exercise in vulnerability scanning." Dept. of CSEE, University of Maryland, Baltimore County, July 2003. http://www.cisa.umbc.edu/.

[21] J. Simmons, "Cracker attack: A password security lab exercise." Dept. of CSEE, University of Maryland, Baltimore County, July 2003. http://www.cisa.umbc.edu/.

[22] M. Bishop, *Computer Security: Art and Science*. Reading, MA, USA: Addison-Wesley, 2003.