A Laboratory Based Capstone Course in Computer Security for Undergraduates

Mike O'Leary
Department of Computer and Information Science
Towson University
Towson, MD 21252
+1 410-704-4757
moleary@towson.edu

ABSTRACT

We describe a laboratory based capstone course in computer security for undergraduates. The course is based on a sequence of hands-on laboratory exercises for four teams of students. It emphasizes defensive tools and techniques at the expense of attacks; it also takes a network centered view where student teams set up and configure entire networks. In this paper, we describe the course, how it fits into the curriculum, and the laboratory facilities we have developed. We then present the details of some of our lab exercises, and discuss the lessons that we have learned.

Categories and Subject Descriptors

K.3.2 [**Computer & Information Science Education**]: Computer Science education, Curriculum

General Terms

Security, Experimentation

Keywords

Security, Cybersecurity, Laboratory, Exercise, Curriculum

1. INTRODUCTION

We have created a laboratory based capstone course in computer security for the undergraduate students in our computer security track. This is a hands-on course that emphasizes defense, detection, and administration. The course is arranged around a series of competitive team-based laboratory exercises for four different teams of students in our isolated test laboratory. In each exercise, students design and construct their own network of test machines using a variety of operating systems. They also set up and securely configure one or more remote services [e.g. SMB, NFS, SSH, Web]. Each team is then provided with authentication credentials to one or more of the services provided by some of the other teams, with the following conditions

- No team has root equivalent credentials on any other machine,
- No team has all of the non-root credentials for any other team, and
- No team knows which other teams have credentials for the services in their network.

During the exercise, each team must verify that the services provided by opposing teams for which they have authentication credentials are correctly functioning. They then try to gain access to all of the remaining services and the remote host itself, if possible. Once the live portion of the exercise has been completed, students review their logs to try to determine who accessed their network, and whether they did so legitimately or illegitimately.

Hands-on courses in computer security have been around for a long time; for example Huss [9] describes such a course that ran in Fall 1993. There are a number of different models for hands-on courses and exercises. Mattford and Whitman [12] survey current practices in the development of an information security and assurance laboratory while Mateti [11] describes such a course at Wright State University and Carlson [4] describes a course at a small college. Many of these hands-on courses are "cyberwar" courses [13, 23]; similar cyberwar courses are also offered at the graduate level [8, 21, 22]. One particular exercise often taught in a course with a cyberwar component is the Capture the Flag exercise [6, 24].

Our course focuses on defensive and administrative tools, primarily for practical reasons. Though there are those who feel that hands-on courses that describe attacks in detail are unethical [7], we limit the time we spend teaching attacks because we feel that the time can be better spent learning about defense. Our goal is to teach potential security officers, rather than penetration testers. In the course, we emphasize tools like logging systems, intrusion detection systems, and firewalls, and a thorough introduction to these topics takes quite a bit of time. Attacks are described, but in dramatically less detail; no effort is made, for example, to describe how to craft 0-day exploits, which are often at the core of Capture the Flag exercises [6].

We take a network-centric view of the course, and student teams are responsible for the creation and maintenance of entire networks. In particular, students design their own network topologies, and configure and place their own servers, firewalls and intrusion detection systems.

In this paper, we describe our course and how it fits into our curriculum. We also give the details of our isolated security lab, and describe some of the exercises we give in detail. Finally, we discuss the lessons we have learned from the past two times the course has been taught.

2. PLACE IN THE CURRICULUM

Our course is called Case Studies in Computer Security; students in our computer security track take this course in the spring semester of their senior year. The computer security track has the same requirements as our traditional computer science major, save that the usual upper-level electives have been replaced by specific computer security courses. The key components of the track are the following seven courses:

1. Computer Ethics,
2. Introduction to Information Security,
3. Introduction to Cryptography,
4. Network Security,
5. Application Software Security,
6. Operating Systems Security, and
7. Case Studies in Computer Security.

The track was launched in Fall 2002, and the case studies course has been offered in both Spring 2004 and Spring 2005. A more detailed description of our track can be found in [1, 2]. Other approaches to computer security curricula include [3, 5, 14, 18, 19].

Our students are well prepared for the Case Studies course. Both Operating Systems Security and Network Security are prerequisites for the course, and most students are concurrently taking Application Software Security. These courses focus on the foundations; for example the Network Security course uses Stalling's text [17] while the Application Software Security course uses the book of Viega and McGraw [20]. As a consequence, the students come into our Case Studies course with a solid understanding of security principles and network protocols. This leaves us free in the Case Studies course to focus on the practical aspects of security. In particular, though most of our students understand the operation of Windows based machines and are familiar with Linux, they have little experience administering a heterogeneous network replete with services.

We also have a graduate level version of the Case Studies course which is part of the Computer Security track for our M.S. program. This course has the same purpose and scope as the undergraduate course, and the courses have been offered in a combined section.

3. FACILITIES

All of the courses in the computer security track use our dedicated computer security laboratory for their laboratory exercises; the Case Studies course always meets in the lab. The lab contains 28 workstations on an isolated network; these machines are grouped into four tables of 6-8 machines, each with its own local network switch. The physical arrangement allows us to comfortably separate our four teams so that they do not interfere with one another during the live portion of our exercises. Students can use the laboratory when class is not in session to prepare for the exercises; however the room is secured, and only students who are enrolled in one of our security courses can use the room.

There are a number of different approaches one can take to the design of an isolated laboratory for security exercises; we mention [8, 10, 15, 16, 21, 22, 23].

Each of our 28 host machines runs VMWare Workstation and students do all of their class work in virtual machines. This has a number of advantages for us. First, students in different courses can use different virtual machines, which enables us to use the room for multiple courses. It also gives the students the opportunity to freely experiment with different operating systems; we have used Windows 2000 Professional, Windows XP Professional, Red Hat 7.0, 7.3, and 9.0, SUSE 9.3, CentOS 4.0, and FreeBSD. Because the entire virtual machine is just a collection of files, they can be easily shared between students on a team; they also let students experiment with different configurations, as they can return to a previously saved state by simply restoring the saved files. Further, VMWare allows a user to run more than one virtual machine on a host at one time; we have run as many as six virtual machines on the same host. Because VMWare has the ability to set up and configure virtual networks for the virtual machines on a single host, students have the ability to set up their own network topologies within these limitations without the need for additional hardware.

This flexible approach has some disadvantages however. In exchange for the flexibility of a dynamically configured general purpose laboratory, we have not been able to construct complex network topologies with separate attack, defense, and monitoring networks like you find in, for example, [10, 16]; instead we use a simple flat network with hosts on one subnet and the virtual machines on a second subnet. We also have not implemented DNS services in the laboratory; during live exercises we often use "whiteboard DNS" where students simply write on the whiteboard the names of their hosts that are offering classroom services, together with their current IP address. Finally, we are limited to Intel hardware; we do not have virtual machines that simulate Macs or various networking hardware. Despite these disadvantages, we have found this approach has produced a very effective multi-purpose computer security laboratory.

The most significant limiting factor on the number of virtual machines that can run on a single host is memory; the memory for a virtual machine must be taken from the memory of the host. As a consequence, when the laboratory was designed we chose machines that each have 2 GB of RAM; this lets us allocate 256 MB for each of six virtual machines and still have 512 MB for the host.

The laboratory is isolated from the campus network and from the Internet; however we do have a file server for the laboratory. The server contains preconfigured virtual machines for a number of common operating systems; it also contains `.iso` images of the installation media so that students can install their own machines from scratch should they want a custom build. Finally, it also hosts an extensive collection of software and documentation, running the gamut from servers like Apache to security tools like Nmap and Snort, to attack oriented programs like Metasploit and Cain & Abel, together with relevant documentation.

4. THE COURSE

The course is arranged around a sequence of 5 or 6 lab exercises; the last exercise serves as the course final and is more extensive. Each exercise other than the final takes roughly two weeks. This includes the time to introduce the material for the exercise, time for the students to prepare their network, time for the actual live exercise, and some time for the student analysis to begin. The live portion of each regular exercise is usually accomplished in one 75 minute session, though the live portion of the final project usually lasts for two or three 75 minute sessions. After each live exercise students prepare a detailed report which is submitted the following week.

Before each exercise, we introduce a collection of related ideas and tools that students use in the exercise. Topics covered in the course include:

- Account and password management. PAM, password cracking.
- Logging and Auditing. Setting up a log server.
- Simple reconnaissance techniques; Ping, Nmap.
- Packet sniffers; Ethereal.
- Intrusion detection systems; Snort.
- Configuring common services: IIS, Apache, OpenSSH, WU-FTP.
- Advanced reconnaissance: Null connections and NetBIOS enumeration, SNMP walking.
- Backdoors: netcat, vnc.
- Firewalls. Iptables.
- Security analysis tools: Nessus, Microsoft baseline security analyzer.
- Security configuration tools: Bastille, Microsoft IIS lockdown tool.

We describe a much smaller collection of attacks and attack tools. For example, we discuss the IIS Double Decode vulnerability and show students how to use the vulnerability together with netcat to obtain a shell. We introduce Cain & Abel, which is a combination packet sniffer and password cracker; we then discuss different methods to convince users to expose their SMB logon credentials. We also present the metasploit framework, and demonstrate some of its more potent exploits.

We will describe three of our lab exercises in more detail.

4.1 Logging Lab

The first competitive hands-on exercise is our logging lab which takes place in the third week of the semester. At this point, we have described how to manage users and configure password policies in both Windows and Linux, including introducing PAM and using it to set password policies. We have also described Syslog and the Windows auditing system, and discussed how to set up a centralized logging server using NTSyslog. Finally we discuss some common services (IIS Web, IIS FTP, Apache, OpenSSH) and describe how their logs are kept.

In the exercise, teams begin by setting up a network of six machines with a mixture of operating systems. Students are required to add a number of users to each machine and to configure them to offer particular common services. They must also set up and configure logging on each machine, and set up a centralized logging server.

Before the live portion of the exercise begins, the names and IP addresses of each machine that is offering a public service is recorded on the laboratory whiteboard. Each team is then provided with a set of non-root equivalent credentials for some of the machines and services offered by the other teams; however the team does not know who has credentials to the machines on their network.

During the live portion of the exercise, students begin by accessing the available services provided by other teams; they then try to gain access to any available service. They are instructed to try to cover their tracks as best as they can. They are also free to use any of the attack tools from our software server; however because this is the first exercise of the course and because attack tools have yet to be covered, this is rarely attempted.

When the live portion of the exercise is completed, students complete a report that describes their results. The report is divided into two portions; first students must describe their reconnaissance activities. In particular they must detail what services they were able to access, and how they did so. In the second, and more significant portion of the report, they must describe how other teams accessed the services the team provided. In particular they need to provide details of which user on which machine accessed which service. Only one-third of the report grade is based on what the team did to others; two-thirds of the grade is based on the correctness and completeness of their log analysis.

Exercises like this are very beneficial for the students, because the students do not know the state of the network, and they do not know *a priori* which connections are legitimate and which ones are not. Thus, they must proceed much as they would in the real world. On the other hand, exercises like this are very difficult to grade, because even the instructor does not know the precise state of the network, or which connections are legitimate and which are not.

Prior to the start of the exercise, students are required to fill out a *Machine Information Sheet* for each machine that they plan to have running during the live portion of the exercise. This is a form where students record all of the pertinent information for the machine, including

- Hostname, IP address, and MAC address,
- A list of all accounts and passwords,
- A list of all provided services, complete with version information; and
- For web servers, a description of the web page; for FTP servers and file shares, a list of the files available for download.

Further, during the live portion of the exercise students are required to keep a written record of each command that they execute on log forms provided by the instructor. On these forms, students note host on which the command was executed, together with the command itself and the results.

Thus, although neither the instructor nor the students knows the precise state of the network during the exercise, it is relatively simple to use the provided information to reconstruct the actual network state while grading takes place. Of course, students being students, there are often some errors in the machine information sheets. However, these have been correctable- for example if three teams discuss a server from another team, one can assume that the fourth team simply made one or more mistakes in their machine information sheets.

4.2 Reconnaissance Lab

The reconnaissance lab is our third lab, and it takes place around week 7. At this point in the course, we have covered a number of reconnaissance techniques, including ping sweeps and other ICMP methods, Nmap, banner grabbing, and NetBIOS enumeration. We also have introduced packet sniffers, including Ethereal. The last major topic before we begin the lab is intrusion detection systems, and students have learned how to set up and configure Snort on their networks.

Before the live portion of the exercise begins, students are given a list of services that they need to provide. Each team decides the number and type of machines that they will use to provide the required services; they also have many more choices than the first lab. For example, they can use Apache 1.x or 2.x on Windows or Linux; they can use IIS FTP or WU-FTPD or various other FTP servers; they can use OpenSSH on Windows or Linux. Each team also sets up a log server and decides how to implement intrusion detection. Students can also start additional machines; these are usually used as attack machines.

During the live portion of the exercise, students are directed to try to make as complete a reconnaissance of the laboratory network as possible. In particular, students try to determine which machines are active, what operating system they are running, what services they are offering, and what software and version is being used to provide the service.

As in the Logging laboratory, students will be provided with non-root access credentials to some opposing teams services, but they will not know who has access to their own systems.

Once the lab has been completed, students again write a report and, like the Logging lab report, it contains a complete description of their reconnaissance activities. They also analyze their logs and intrusion detection system records to determine who accessed their services, and whether they did so legitimately or not.

4.3 System Lab

In this, the final lab exercise, each team is told that they are the IT department for a hypothetical company. The company has offices in three different physical locations and they need to be able to work collaboratively on projects. The company also needs a simple web presence, and the ability to deliver documents to the public. Finally the company has a corporate partner that should have access to additional information not available to the public, as well as the ability to work collaboratively on projects. The role of the partner company will be played by one of the other three teams. Within these general guidelines, student teams are free to construct their own network and choose what services they will provide to meet these business requirements.

At this point of the course, all of the material described in section 4 has been presented. In particular, students have learned how to use Iptables to create their own firewalls, and can now fully utilize the virtual networks of VMWare to create the network of this hypothetical company. Students create firewalled subnetworks for each of the corporate locations, together with a DMZ that contains their servers. Students have also used the security analysis tools like Nessus to analyze their machines for vulnerabilities, and used tools like Bastille to harden them, especially those machines that will be offering services.

During the live portion of the exercise, students first must verify that the public facing services of the other teams are correctly functioning, as well as checking that they have the correct level of access to the services of their corporate partner. They then try to use the attack tools that have been described to compromise the machines of their opponents, or even their corporate partners.

The exercise concludes with a written report detailing the results of their analysis of the public and partner services and describing their offensive actions. As before, they also must determine who accessed the systems that were under their care and whether it was done legitimately or not; this still counts for two-thirds of the project grade.

5. LESSONS LEARNED

We have learned a number of valuable lessons as we have taught this course. We found that students have significant difficulties writing their own firewall scripts, especially in the complex networks with NAT that they find in the final lab exercise. Both times the course was taught it took students over four weeks to come up with firewall scripts that truly were effective. The difficulty seems to lie in the fact that students conflate problems with networking and NAT with troubles in the filtering rules for the firewall, and it takes them quite a bit of time to learn how to separate these issues.

Another issue is the fact that students rarely attempt sophisticated attacks, even in the final laboratory exercise. There is no doubt that this is due to the relative emphasis placed on attack and defense in the course, but it would be beneficial for students to learn how to detect and respond to more sophisticated attacks. However, even the simple attacks that we use force students to think carefully about how they configure their network, and student teams that make configuration errors usually get punished quite quickly by their opponents.

Finally, the added realism of the final exercise where the students acted as a simulated IT department for a fictional company was far more valuable than expected. In part this was due to the fact that students were given questions couched in the language of business and function, rather than as technical questions. Students had to think about the business implications of their networking and configuration decisions, which is a point of view to which they had not yet been exposed.

6. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under grant No. DUE-0113783

7. REFERENCES

[1] S. Azadegan, M. Lavine, M. O'Leary, A. Wijesinha, and M. Zimand. A dedicated undergraduate track in computer security education. In C. Irvine and H. Armstrong, editors, *Security education and critical infrastructures*, pages 319–331. Kluwer Academic Publishers, 2003.

[2] S. Azadegan, M. Lavine, M. O'Leary, A. Wijesinha, and M. Zimand. An undergraduate track in computer security. In *ITiCSE '03: Proceedings of the 8th annual conference on Innovation and technology in computer science education*, pages 207–210, New York, NY, USA, 2003. ACM Press.

[3] B. Bogolea and K. Wijekumar. Information security curriculum creation: a case study. In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*, pages 59–65, New York, NY, USA, 2004. ACM Press.

[4] D. Carlson. Teaching computer security. *SIGCSE Bull.*, 36(2):64–67, 2004.

[5] M. Dornseif, F. C. Gaertner, M. Mink, and L. Pimenidis. Teaching data security at university degree level. In *Proceedings of the WISE04 (to appear)*. Internet: http://md.hudora.de/publications/.

[6] C. Eagle and J. L. Clark. Capture the flag: Learning computer security under fire. In C. Irvine and M. Rose, editors, *Avoiding Fear, Uncertainty and Doubt Through Effective Security Education*, pages 17–21. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.

[7] J. Harris. Maintaining ethical standards for a computer security curriculum. In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*, pages 46–48, New York, NY, USA, 2004. ACM Press.

[8] J. M. D. Hill, J. Curtis A. Carver, J. W. Humphries, and U. W. Pooch. Using an isolated network laboratory to teach advanced networks and security. In *SIGCSE '01: Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*, pages 36–40, New York, NY, USA, 2001. ACM Press.

[9] J. E. Huss. Laboratory projects for promoting hands-on learning in a computer security course. *SIGCSE Bull.*, 27(2):2–6, 1995.

[10] S. D. Lathrop, G. J. Conti, and D. J. Ragsdale. Information warfare in the trenches. In *Security education and critical infrastructures*, pages 19–39. Kluwer Academic Publishers, 2003.

[11] P. Mateti. A laboratory-based course on internet security. In *SIGCSE '03: Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pages 252–256, New York, NY, USA, 2003. ACM Press.

[12] H. J. Mattord and M. E. Whitman. Planning, building and operating the information security and assurance laboratory. In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*, pages 8–14, New York, NY, USA, 2004. ACM Press.

[13] M. Micco and H. Rossman. Building a cyberwar lab: lessons learned: teaching cybersecurity principles to undergraduates. In *SIGCSE '02: Proceedings of the 33rd SIGCSE technical symposium on Computer science education*, pages 23–27, New York, NY, USA, 2002. ACM Press.

[14] L. F. Perrone, M. Aburdene, and X. Meng. Approaches to undergraduate instruction in computer security. Proceedings of the American Society for Engineering Education Annual Conference and Exhibition, ASEE 2005. Internet: http://www.ists.dartmouth.edu/library/116.pdf.

[15] G. W. Romney and B. R. Stevenson. An isolated, multi-platform network sandbox for teaching it security system engineers. In *CITC5 '04: Proceedings of the 5th conference on Information technology education*, pages 19–23, New York, NY, USA, 2004. ACM Press.

[16] J. Schafer, D. J. Ragsdale, J. R. Surdu, and C. A. Carver. The IWAR range: a laboratory for undergraduate information assurance education. In *CCSC '01: Proceedings of the sixth annual CCSC northeastern conference on The journal of computing in small colleges*, pages 223–232, , USA, 2001. Consortium for Computing Sciences in Colleges.

[17] W. Stallings. *Network Security Essentials: Applications and Standards*. Prentice Hall Professional Technical Reference, 2002.

[18] R. Tikekar and T. Bacon. The challenges of designing lab exercises for a curriculum in computer security. *J. Comput. Small Coll.*, 18(5):175–183, 2003.

[19] R. V. Tikekar. Teaching computer security to undergraduates. In C. Irvine and M. Rose, editors, *Avoiding Fear, Uncertainty and Doubt Through Effective Security Education*, pages 5–10. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.

[20] J. Viega and G. McGraw. *Building secure software: how to avoid security problems the right way.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[21] G. Vigna. Teaching Hands-On Network Security: Testbeds and Live Exercises. *Journal of Information Warfare*, 3(2):8–25, 2003.

[22] G. Vigna. Teaching Network Security Through Live Exercises. In C. Irvine and H. Armstrong, editors, *Proceedings of the Third Annual World Conference on Information Security Education (WISE 3)*, pages 3–18, Monterey, CA, June 2003. Kluwer Academic Publishers.

[23] P. J. Wagner and J. M. Wudi. Designing and implementing a cyberwar laboratory exercise for a computer security course. *SIGCSE Bull.*, 36(1):402–406, 2004.

[24] J. Walden. A real-time information warfare exercise on a virtual network. In *SIGCSE '05: Proceedings of the 36th SIGCSE technical symposium on Computer science education*, pages 86–90, New York, NY, USA, 2005. ACM Press.