The SEED Project: Providing Hands-on Lab Exercises for Computer Security Education[*]

Wenliang Du
Department of Electrical Engineering & Computer Science
Syracuse University, Syracuse, New York, USA
Email: wedu@syr.edu Tel: +1 315 443 9180

1. INTRODUCTION

"I hear and I forget. I see and I remember. I do and I understand". This famous saying, by Chinese philosopher Confucius (551 BC – 479 BC), has been a motto for many educators, who firmly believe that learning must be grounded in experience.

When I started teaching computer security courses nine years ago, I could find only a few laboratory exercises that engaged the students in experiential learning, e.g. [1–5], but they were incoherent; their coverage of security topics was narrow, even jointly, and their lab environments were often difficult and costly to set up.

To address this problem, I decided to develop a set of hands-on exercises (labs) covering a wide spectrum of security topics and that could be shared with other instructors. All the labs would be based on one unified environment, so students would not need to spend too much time learning new environments for different labs. Moreover, the lab environment was to be easy and inexpensive to set up.

Given these goals, the SEED (SEcurity EDucation) project started in 2002 with NSF support. Nine years later, with help from over 20 students, we have developed 30 SEED labs, covering many security topics: vulnerabilities, attacks, software security, system security, network security, web security, access control, authentication, cryptography, etc. Most SEED labs have gone through multiple development-trial cycles in actual university courses. So far, more than 80 universities have requested the instructor's manual and have adopted a selection of SEED labs for their courses. All SEED labs, the related materials, and the lab environment are available online at: `http://www.cis.syr.edu/~wedu/seed/`. This article describes the SEED labs and the lab environment.

2. LAB ENVIRONMENT

Two operating systems are used in the lab environment: `UbuntuLinux` and `Minix`. The majority of the SEED labs use `Ubuntu`, but a few labs require a significant amount of effort in kernel-level coding, and for them, `Minix`, an instructional operating system, was chosen.

The SEED lab exercises are designed so that no dedicated physical laboratory is needed. All SEED labs can be carried out on students' personal computers or in a general computing laboratory. This is made possible by virtual machine software. We have created a pre-built virtual machine (VM) image of `Ubuntu`, and have installed all the necessary software needed by the SEED labs. Students simply download this image to their computers, and use virtual machine software to run the image. Immediately, their lab environment is ready to use. The image is fully tested and guaranteed to work for all the SEED labs. All of the labs are reimaged and retested bianually. The image can be downloaded from our SEED web site.

For students without personal computers, instructors can ask their system administrators to install the virtual machine software on the machines in public laboratories. Students only need to buy a portable harddrive to store their VM images. Most SEED labs require only one to three virtual machines; a physical computer with 4GB of RAM is sufficient. However, some labs, such as the VPN lab, require at least six VMs. We let students partner, so they have six VMs with two physical computers.

We have tried various ways to set up the lab environment, including using students' personal computers, using a general computing lab, and using a cloud computing infrastructure. We found that for convenience students prefer to use their own computers for SEED labs.

3. THE SEED LABS

In this section, we provide an overview of the SEED labs. Readers can find the detailed lab descriptions from our SEED web site. All SEED labs are listed in Table 1.

3.1 Vulnerability and Attacks Labs

Not many students can gain a full understanding of vulnerabilities and attacks without going through some attacks first hand. We developed 12 labs, covering many common vulnerabilities and attacks. In each lab, students are given a system (or program) with hidden vulnerabilities. Based upon the hints provided, students must find these vulnerabilities, and then devise strategies to exploit them. Furthermore, students need to demonstrate ways to defend against the attacks or comment on the prevailing mitigating meth-

[*]This project was supported by Awards No. 0618680 and No. 0231122 from the TUES/CCLI program of the US National Science Foundation.

ods and their effectiveness. Each lab takes one to two weeks—10 hours per week—for average students.

Three categories of vulnerabilities are covered: general software vulnerabilities, network protocol vulnerabilities, and web application vulnerabilities. All these labs have gone through several development-trial cycles. The initial versions of the labs were usually unsatisfactory for various reasons: the lab was either too difficult or too simple, the description was too vague, or too much time was spent on tangential tasks. Most of the reasons were hard to anticipate prior to field tests.

We use the buffer-overflow attack lab to illustrate how a lab evolved. Initially, we gave students a flawed program, and asked them to exploit its buffer-overflow vulnerability. This seemed so simple conceptually, yet turned out costing students a month of fruitless effort. After close investigation, we realized that the lab environment—Linux operating system—had four built-in defenses against buffer-overflow attacks: a non-executable stack, memory randomization, stack-guard, and a protection mechanism in the Bash shell. Not knowing these defenses, students could have done everything correctly, but still could not get the root shell. Lacking appropriate guidance, most students became frustrated, and eventually gave up.

In the next lab offering, we told students about these defenses, explained how they worked, and provided concrete instructions for disabling them. The lab became doable within 10 hours or less. We incorporated the four defenses into the exercises. Two were easy to defeat: memory randomization in 32-bit machines and Bash's protection mechanism. We asked students to turn these defenses back on, and then modify their attacks to defeat the defenses. Defeating the non-executable stack defense was quite sophisticated, so we designed another lab, the return-to-libc attack lab. It took several more years of further improvement and fine tuning to stabilize these two labs.

3.2 Design Labs

In security education, students should also be given opportunities to apply security principles in *designing and implementing* systems. However, building a meaningful system usually takes longer than possible in a semester long course.

We identified several systems suitable for secure systems education, including firewalls, IPSec, Virtual Private Networks (VPN), access control systems in operating systems, and encrypted file systems. Each system covers a broad scope of security principles, making them suitable for projects. To make projects achievable within 4-6 weeks, we reduced the system functionalities, but we kept those essential for security. We also provided supporting materials to help students reduce time expended on necessary but non-essential—to security—functionalities.

There are 9 design/implementation labs: five on systems and four on networking. The Virtual Private Network (VPN) Lab illustrates this category of labs. VPNs exemplify a number of security principles and technologies, including cryptography, integrity, authentication, key management, key exchange, and Public-Key Infrastructure (PKI). It serves as a platform to enhance students' security learning; it gives students an opportunity to incorporate security technologies into a real and complex system.

Implementing a fully-functional VPN in `Linux` is not easy. To make the task achievable, we used `OpenVPN` as the basis, stripped away its many functionalities, leaving only those essential to security. We provided detailed instructions on how to use the TUN/TAP and SSL to build VPNs, how to create and verify certificates, and how to use PKI to authenticate VPN servers and clients. Moreover, we used diagrams to illustrate the packet flow within the network stack in a typical VPN system. Most students were able to finish this project within five weeks. Students were very positive on their experience, noting a great sense of accomplishment.

3.3 Exploration Labs

The objective of the exploration labs are intended to enhance students' learning via observation, playing and exploration, so they can understand what security principles "feel" like in a real system; and to provide students with opportunities to apply security principles in *analyzing and evaluating* systems.

We have 8 exploration labs: two on systems, three on networking, and three on cryptography. The PKI lab illustrates this category of labs. Its learning objective is for students to get experience on the PKI technology. Students go through the process that a typical web server takes to secure web browsing with PKI. Students launch a web server called `PKILabServer.com`. Then they must get a digital certificate for this server. To avoid paying a commercial Certificate Authority (CA), we ask students to become a root CA themselves. Students then browse the web site `PKILabServer.com` using `Firefox`, and figure out what it takes for the browser and the server to use the certificate for secure web browsing. Detailed instructions are provided to guide students through the exercise.

4. DISSEMINATION

Based on instructor manual requests, we know that over 80 universities have adopted or are in the process of adopting the SEED labs. Since all the SEED labs are put online, the actual number of adoptions is probably higher. Instructors' feedback helped us improve our labs and also our dissemination efforts, which are summarized below.

Mapping to Textbook Chapters. Although independently developed, the SEED labs work well with most textbooks. We have mapped our SEED labs to the chapters of several security textbooks. The mappings, summarized in Table 1, were conducted on the following textbooks:

- *Introduction to Computer Security (2004)*, by Matt Bishop. We refer to this book as *MB1*.
- *Computer Security: Art and Science (2002)*, by Matt Bishop. We refer to this book as *MB2*.
- *Security in Computing (3rd Edition, 2003)*, by Charles P. Pfleeger and Shari Lawrence Pfleeger. We refer to this book as *PP*.
- *Network Security: Private Communication in a Public World (2nd Edition, 2002)*, by Charlie Kaufman, Radia Perlman, and Mike Speciner. We refer to this book as *KPS*.
- *Introduction to Computer Security (2011)*, by Michael T. Goodrich & Roberto Tamassia. We refer to this book as *GT*. The book acknowledges the SEED project;

Types	Labs	L	MB1	MB2	PP	KPS	GT	DG
Vulnerability and Attack Labs	Buffer-Overflow Lab	U	20, 26	23, 29	3	-	3	10
	Return-to-libc Lab	U	20, 26	23, 29	3	-	3	10
	Race-Condition Lab	U	20, 26	23, 29	3	-	3	10
	Format-String Lab	U	20, 26	23, 29	3	-	3	10
	Chroot Sandbox Lab	U	20, 26	23, 29	3	-	3	-
	TCP/IP Attack Lab	U	20, 23, 26	23, 26, 29	3	-	5	17
	DNS Pharming Attack Lab	U	20, 23, 26	23, 26, 29	3	-	6	17
	Cross-Site Scripting Attack Lab	U	20, 23, 26	23, 26, 29	3	25	7	18
	Cross-Site Request Forgery Attack Lab	U	20, 23, 26	23, 26, 29	3	25	7	18
	ClickJacking Attack Lab	U	20, 23, 26	23, 26, 29	3	25	7	18
	SQL Injection Attack Lab	U	20, 23, 26	23, 26, 29	3, 6	-	7	10, 18
	`Set-UID` Program Vulnerability Lab	U	14	15	4	-	3	7
Exploration Labs	Pluggable Authentication Modules Lab	U	11	12	4	9, 10	3	4, 7
	`Linux` Capability Exploration Lab	U	12, 14, 17	13, 15, 19	4	-	3	5
	Secret-key Encryption Lab	U	8-10	9, 10, 11	2, 12	2-6	8	14
	One-Way Hash Function Lab	U	8-10	9, 10, 11	2, 12	2-6	8	14
	Public-key Cryptography Lab	U	8-10	9, 10, 11	2,12	2-6	8	15
	SYN-Cookie Lab	U	23	26	2, 7	5	5	17
	Packet Sniffing and Spoofing Lab	U	23	26	2, 7	5	5	17
	Web Access Control Lab	U	4, 14	4, 15	4, 7	25	7	18
Design and Implementation Labs	Set-RandomUID Sandbox Lab	G	19	22	-	-	3	-
	`Minix` Capability Lab	G	12, 14, 17	13, 15, 19	4	-	3	5
	`Minix` Role-Based Access Control Lab	G	12, 14, 17	13, 15, 19	4	-	9	5
	Encrypted File System Lab	G	8-10, 17	9-11, 13, 19	2, 4	2-5	9	14
	Address-space Layout Randomization Lab	G	22, 24, 26	25, 27, 29	4, 5	-	-	-
	IP Sec Lab	G	8-10, 17, 23	9-11, 19, 26	2, 7	2-5, 17	6	16
	VPN Lab	G	8-10, 17, 23	9-11, 19, 26	2, 7	2-5, 17	6	16
	`Linux` Firewall Lab	G	17, 23	19, 26	7.4	23	6	17
	`Minix` Firewall Lab	G	17, 23	19, 26	7.4	23	6	17

Table 1: The SEED Labs and Their Mappings to Textbooks (The numbers in the table are chapter numbers. In the column titled L, 'U' means the lab is appropriate for both undergraduate and graduate students, and 'G' means the lab is intended for graduate students)

its web site (`http://www.securitybook.net/`) provides direct links to thirteen of the SEED labs.

- *Computer Security (2nd edition, 2011)*, by Dieter Gollmann. We refer to this book as *DG*.

Instructor Manuals. To create instructor manuals for each lab, we picked the best lab reports from students for each lab and bound them together. Access to the manuals is restricted to instructors only, who must send us specific requests.

Copyright. The SEED labs are licensed under the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; and permission is granted to copy, distribute and/or modify SEED documents under the terms of the license.

5. SUMMARY

The SEED labs have matured after nine years of field testing and improvement; and they have been used by many universities worldwide. They are free for educational use. We are committed to helping other instructors incorporate the SEED labs in their courses. In our future work, we plan to expand our SEED labs to cover additional topics, especially those related to emerging technologies and threats.

6. REFERENCES

[1] M. Bishop. Computer security in introductory programming classes. In *Workshop on Education in Computer Security*, pages 1–2, Monterey, CA, USA, January 1997.

[2] J. M. D. Hill, C. A. Carver, Jr., J. W. Humphries, and U. W. Pooch. Using an isolated network laboratory to teach advanced networks and security. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*, pages 36–40, Charlotte, NC, USA, February 2001.

[3] J. Mayo and P. Kearns. A secure unrestricted advanced systems laboratory. In *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education*, pages 165–169, New Orleans, USA, March 24-28 1999.

[4] N. Memon. CS392/681: Computer Security. http://isis.poly.edu/courses/cs392/.

[5] W. G. Mitchener and A. Vahdat. A chat room assignment for teaching network security. In *Proceedings of the 32nd SIGCSE technical symposium on Computer Science Education*, pages 31–35, Charlotte, North Carolina, United States, 2001. ACM Press.