Summary of the SEED Labs – For Authors and Publishers

Wenliang Du, Syracuse University

To help authors reference our SEED labs in their textbooks, we have created this document, which provides a brief summary of each of the SEED labs. Permission is granted to authors and publishers to include this entire document (or part of it) in their books or online resources, as long as the following statement is included:

> The summary of the labs are provided by Wenliang Du (Syracuse University), who maintains the copyrights of these labs. Full details of the labs can be downloaded from the following web site: `http://www.cis.syr.edu/∼wedu/seed/`

1 Vulnerability and Attack Labs

The objective of the vulnerability and attack labs is to provide students with a first-hand experience on various vulnerabilities and attacks. Students discover why some design or implementation errors become vulnerabilities, how vulnerabilities cause security breaches, how they are exploited, and how to apply the security testing principles to detect vulnerabilities. More importantly, students will learn from other people's mistakes.

Buffer Overflow Vulnerability Lab. Students are given a program with the buffer-overflow vulnerability. Their task is to develop a scheme to exploit the vulnerability and finally to gain the root privilege. In addition to the attacks, students will be guided to walk through several protection schemes that have been implemented by the system to counter against the buffer-overflow attacks, including non-executable stack, address space randomization, and StackGuard. Students need to develop methods to defeat some of the countermeasures.

Return-to-libc Attack Lab. One of the countermeasures against buffer-overflow attacks is to make stacks non-executable. This countermeasure can be defeated using the `return-to-libc` attack, which does not need an executable stack. In this lab, students are given a program with the buffer-overflow vulnerability. Their task is to develop a `return-to-libc` attack to exploit the vulnerability and finally to gain the root privilege.

Format String Vulnerability Lab. Students are given a program with the format-string vulnerability, and their task is to develop a scheme to exploit the vulnerability. To goal of the attack is (1) to crash the program, (2) to print out a secret value hidden in the program, and eventually (3) to modify the secret value. In addition to the attacks, students will be guided to walk through a protection scheme that can be used to defeat this type of attacks. Students need to evaluate whether the scheme works or not and explain why.

Race Condition Vulnerability Lab. Students are given a program with the race-condition vulnerability, and their task is to develop a scheme to exploit the vulnerability and gain the root privilege. In addition to the attacks, students will be guided to walk through several protection schemes that can be used to counter the race-condition attacks. Students need to evaluate whether the schemes work or not and explain why.

Set-UID Program Vulnerability Lab. Set-UID is an important security mechanism in Unix operating systems. When a Set-UID program is executed, it assumes the owner's privileges. Set-UID allows us to do many interesting things, but unfortunately, it is also the culprit of many bad things. Therefore, the objective of this lab is two-fold: (1) Appreciate its good side: understand why Set-UID is needed and how it is implemented. (2) Be aware of its bad side: understand its potential security problems. Students will be asked to conduct attacks on several Set-UID programs via environment variables and bad inputs.

Chroot Sandbox Vulnerability Lab. Almost all the Unix systems have a simple built-in sandbox mechanism, called chroot. In this lab, students need to figure out how chroot works, why it works, and why it should only be used by root. Moreover, students will see the vulnerabilities and limitations of this type of sandbox.

Cross-Site Request Forgery Attack (CSRF) Lab. Students are given a version of phpBB, which is an open-source web-based message board; they need to launch Cross-Site Request Forgery attacks on this web application. Using the attacks, attacker can post an arbitrary message on behalf of the victim, as well as modifying the victim's profile.

Cross-Site Scripting Attack (XSS) Lab. Students are given a version of phpBB, which has the XSS vulnerability. Students need to exploit this vulnerability by posting a malicious message to the message board. Users who view this malicious message will become victims, and the attackers can then post forged messages for the victims. Essentially, students need to create XSS worms in this lab. To make the lab more challenging, students need to make this worm self-propagating, so infected victims can further spread the XSS worms for the attackers.

SQL Injection Attack Lab. Students are given a version of phpBB, which has the SQL injection vulnerability. Students need to exploit this vulnerability to achieve the followings: (1) log into another user's account without typing the password, and (2) change other users' profiles without logging into their accounts. After the attacks, students need to experiment with several countermeasures against this type of attacks.

ClickJacking Attack Lab. In the Clickjacking attack, the attacker constructs a malicious web page and misleads the victim into clicking on certain (visible) links/buttons, whereas in reality, the victim is actually clicking on other links/buttons made invisible by the attacker. In this lab, students need to construct such a malicious web page to launch the ClickJacking attack on phpBB. Students will also experiment with the countermeasures against the ClickJacking attack.

TCP/IP Attack Lab. Using tools such as Netwox and Wireshark, students need to conduct a variety of attacks on the TCP/IP protocols, including ARP cache poisoning attack, ICMP Redirect attack, SYN Flooding attack, TCP Reset attack, TCP Session Hijacking, and various ICMP-based attacks.

DNS Pharming Attack Lab. Students need to first set up and configure a DNS server (BIND9), and then they will try various DNS Pharming attacks, including the DNS cache poisoning attack. In particular, students will conduct the famous DNS attack that was developed by Dan Kaminsky in 2008.

2 Exploration Labs

The objective of the exploration labs is 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.

Packet Sniffing & Spoofing Lab. The objective of this lab is for students to master the technologies underlying most of the sniffing and spoofing tools. Students will play with some simple sniffer and spoofing programs, read their source code, modify them, and eventually gain an in-depth understanding of the technical aspects of these programs. At the end of this lab, students should be able to write their own sniffing and spoofing programs.

Pluggable Authentication Module Lab. Students need to explore the Pluggable Authentication Module (PAM) mechanism in `Linux`. In particular, students will configure PAM and write a program that uses PAM for its authentication.

Web Access Control Lab. The objective of this labs is to help students gain a good understanding of the same-origin policy. The understanding will be a precursor for other web-related labs such as cross-site scripting and cross-site request forgery. In this lab, students will deal with JavaScript programs, DOM objects, cookies, Ajax, and iframes. They need to understand the access control policies when dealing with these entities.

SYN Cookie Lab. The learning objective of this lab is for students to explore the mechanism of SYN cookies in `Linux`, and understand how this mechanism helps defend against the SYN flooding attack.

Linux Capability Lab. The learning objective of this lab is for students to gain a first-hand experience on capability, to appreciate the advantage of capabilities in access control, to master how to use capability to achieve the principle of least privileges, and to analyze the design of the capability-based access control in `Linux`. This lab is based on POSIX 1.e capability, which is implemented in recent versions of `Linux` kernel.

Secret-Key Encryption Lab. The learning objective of this lab is for students to get familiar with the concepts in the secret-key encryption. Using the `openssl` tools, students experiment with encryption algorithms, encryption modes, paddings, and initial vector (IV). Moreover, students will use tools and write programs to encrypt/decrypt messages.

One-Way Hash Function Lab. The learning objective of this lab is for students to get familiar with one-way hash functions and Message Authentication Code (MAC). Using the `openssl` tools, students experiment with message digest, message authentication code, keyed hash, and both one-way property and collision-free property of one-way hash functions.

Public-Key Cryptography and PKI Lab. The learning objective of this lab is for students to get familiar with the concepts in the Public-Key encryption and Public-Key Infrastructure (PKI). In this lab, students gain a first-hand experience on public-key encryption, digital signature, public-key certificate, certificate authority, and authentication based on PKI. Moreover, students will be able to use tools and write programs to create secure communication channels using PKI.

3 Design and Implementation Labs

The objective of the design/implementation labs is to provide students with opportunities to apply security principles in *designing and implementing* systems. Students should have a reasonable background in operating systems, because kernel programming and debugging are required for most of the labs in this category.

Virtual Private Network (VPN) Lab. The learning objective of this lab is for students to master the network and security technologies underlying VPNs. The design and implementation of VPNs exemplify a number of security principles and technologies, including crypto, integrity, authentication, key management, key exchange, and Public-Key Infrastructure (PKI). To achieve this goal, students will implement a simple TLS/SSL VPN for `Ubuntu`, called `miniVPN`, which is simplified from the open-source project `openVPN`. Students will use the TUN/TAP technology to build VPN tunnels, and use `openssl` and PKI to implement the authentication and tunnel encryption.

IPSec Lab. The learning objective of this lab is for students to integrate a number of essential security principles in the implementation of IPSec. The design and implementation of IPSec exemplify a number of security principles, including encryption, one-way hashing, integrity, authentication, key management, and key exchange. Furthermore, IPSec demonstrates how cryptography algorithms are integrated into the TCP/IP protocols in a transparent way, such that the existing programs and systems do not need to be aware of the addition of IPSec. In this lab, students will implement a simplified version of IPSec for `Minix`.

Linux Firewall Lab. The learning objective of this lab is for students to gain the insights in how firewalls work by designing and implementing a simple personal firewall for `Linux`. Students need to use Loadable Kernel Module (LKM) and `Netfilter` to implement such a firewall.

Minix Firewall Lab. The learning objective of this lab is for students to learn how firewall works by implementing a simple personal firewall for `Minix`. Students need to add the packet filtering functionality to `Minix`'s kernel.

Role-Based Access Control (RBAC) Lab. Students need to implement a simplified capability-based RBAC system for `Minix`. The simplification on RBAC is based on the RBAC standard proposed by NIST, while the simplification on capabilities is based on the POSIX capabilities. This lab is quite comprehensive, covering a variety of security concepts and principles.

Minix Capability Lab. Students need to implement a simplified capability-based access control system for `Minix`. To make this lab accomplishable within a short period of time, only 5 capabilities need to be implemented.

Encrypted File System Lab. Encrypted File System (EFS) has been widely implemented in a number of operating systems, such as `Solaris`, `Windows NT`, and `Linux`. In this lab, students need to implement a simplified EFS for `Minix`. Students need to deal with various issues related to EFS, such as transparency, key management, encryption algorithms, padding, authentication, etc.

Address Space Randomization Lab. Address space layout randomization (ASLR) is a computer security technique, which involves randomly arranging the positions of key data areas in a process's address space. These key data areas usually include the base of the executable and position of libraries, heap, and stack, etc. Although ASLR does not eliminate vulnerabilities, it can make the exploit of some vulnerabilities much harder. In this lab, students need to implement ASLR in `Minix`.

Set-RandomUID Sandbox Lab. Students need to design and implement a simple sandboxing mechanism called `Set-RandomUID`. When a `Set-RandomUID` program is executed, the operating system randomly generates a non-existing user id, and runs the program with this new user id as the effective user id. This way, the damage of this program is limited even if it is malicious.