Secure Coding in an Introductory-level Android Mobile Application Development Course

This series of secure coding modules for introductory Android mobile application development courses was developed and made available with Intel's generous support.

OVERVIEW OF SECURE CODING MODULES FOR ANDROID

Every computing platform is vulnerable to security threats and attacks making it essential for future software developers to learn and practice secure coding techniques. One effective teaching strategy introduces secure coding in foundational courses and infuses more challenging concepts across the curriculum so that students study security early, often, and with increasing complexity. First semester programming courses are ideal for introducing students to secure coding.

Embracing this strategy, a series of secure coding modules were developed for beginning Android mobile application development courses and designed with the assumption that students will come into the course with little or no knowledge of secure coding. Due to the close relationship between Java and Android, the modules include (1) an overview of common coding vulnerabilities, (2) secure coding techniques for Java, and (3) secure coding techniques for Android. More specifically, the modules include:

1. Common coding vulnerabilities
    - Integer errors
    - Improper input validation
    - Buffer overflow
2. Secure coding techniques for Java
    - Accessibility
    - Extensibility
3. Secure coding techniques for Android
    - Application Sandboxing
    - Data Storage
    - Accessibility (content provider)

These security modules were developed during the fall of 2014 and initially used in Android classes during the spring of 2015. Updates will be incorporated into the modules over time as warranted by software updates or editing requirements. Although the modules were developed for an Android CS1 programming course (all three categories being applicable), the first two categories are also suitable for Java CS1 programming courses.

OVERVIEW OF SECURE CODING LAB 1: COMMON CODING VULNERABILITIES

Security Injections @Towson (http://www.towson.edu/securityinjections) is a National Science Foundation project administered by Blair Taylor and Siddharth Kaza at Towson University. *Security Injections* are security-related modules, free and available to any instructor, to be strategically placed into existing undergraduate classes. The *Security Injections* modules have been used by faculty across the United States for several years. To not reinvent the wheel, **Secure Coding Lab 1** utilizes three of the *Security Injections @Towson* modules as an introduction to secure coding:

1. Integer errors
2. Improper input validation
3. Buffer overflow

There are versions of these labs (called Security Injections 2.0) which are completed online with automatic grading. More details can be found online at the website above.

The compressed file for this lab includes:

- This readme file
- A PDF of a laboratory exercise for students

OVERVIEW OF SECURE CODING LAB 2: ENCAPSULATION AND ACCESSIBILITY

Because Android uses Java classes, it is important for Android students to understand the vulnerabilities caused by improper implementation of encapsulation and misuse of access modifiers. This secure module provides a(n):

1. Review of how to create a new class in Java (for students who are just learning this)
    - Includes 4 videos
2. Overview of the public and private access modifiers
3. Laboratory exercise which highlights security vulnerabilities in classes with improper encapsulation or misuse of the private and public access modifiers

The compressed file for this lab includes:

- This readme file
- A PDF of the laboratory exercise for students
- CardGame.zip which included the files the students need to complete the lab
- EncapsulationExample.zip which includes the files discussed in the videos referenced in the lab

OVERVIEW OF SECURE CODING LAB 3: EXTENSIBILITY THROUGH INHERITANCE IN JAVA

Because Android uses Java classes, it is important for Android students to understand the vulnerabilities caused by improper implementation of extensibility through inheritance, the misuse of access modifiers and the final modifier. This secure module provides a(n):

1. Overview of the public, **protected**, and private access modifiers. Explanation of the **final** modifier.
2. Laboratory exercise which highlights security vulnerabilities in classes with improperly designed inheritance structures.

The compressed file for this lab includes:

- This readme file
- A PDF of the laboratory exercise for students
- BoxExample.zip which included the files the students need to complete the lab example
- InheritanceLab.zip which includes the files needed to complete the lab assignment (homework)