**Secure Programming**

© Elephant Scale

November 28, 2018

Today, hackers have achieved tremendous successes while the software developers are just beginning to learn about secure coding. This course is for developers who want to defend against hackers.

The course is intended for software architects and engineers. It gives them a practical level of experience, achieved through a combination of about 50% lecture, 50% demo work with student’s participation.

**Course objectives**

* Understand what security is
* Add basic, programmatic security to your code
* Understand how to defend against an attack
* Understand how to secure resources=

**Duration**: 2 days

**Audience**: Software Architects, Developers

**Prerequisites**:

* Familiarity with any programming language
  + However, Java is preferred, since many labs are in Java
* Be able to navigate Linux command line
* Basic knowledge of command line Linux editors (VI / nano)

**Lab environment:**

Working environment will be provided for students. Students would only need an SSH client and a browse.

Zero Install: There is no need to install software on students' machines.

**Course Outline:**

**Fundamentals**

* Why and what?
* Characteristics of application security
* Basic security mechanisms

**Basic Security within Java SE 8**

* Mutability of objects (done)
* Variable, method, class, and package scopes (done)
* Thread safety (done)
* Exception handling (done)
* Input validation (done)

**Basic Security at the EE level**

* Role-based authentication (done)
* Specifying Security Constraints
* Programmatic security
* Declarative security

**Programming Against an Attack**

* Denial of Service (DoS)
* SQL Injection
* Large files
* XML and HTML issues
* LDAP injection
* XPath injection
* Password storage

**Defensive Programming**

* Error handling in the Java EE space
* Type annotation syntax and the Checker Framework
* Application-layer security
* Transport-layer security
* Message-layer security
* Secure connection using SSL

**Java Security**

* REST endpoint security (done)
  + oauth2 token based authentication and authorization
  + oauth2 server setup
  + embedded token server, token lifecycle and management
* Developing securing java applications
  + Safe coding and design patterns
  + how to find vulnerable code while doing a code review
  + most common pitfalls
* JavaScript security
  + how to manage secure Personally Identifiable Information (PII) data on the end client apps (mobile app, browser, tablet)
  + Versions: OAuth2, Angular4

**Securing Resources**

* Authentication mechanisms
* Using Form-Based Login
* Digital certificates
* Using JDBC realm
* Securing HTTP resources
* Securing Application clients