Southern New Hampshire University

CS 305 Module Two

Ronny Z. Valtonen

Version 1.4

1. Areas of Security

Since we are being tasked to implement an expressive command input function for the web application, we need to ensure we have proper security. Upon review of the vulnerability assessment process flow diagram, there are a few areas of security in the first layer that are relevant to assess in the software application. To begin, input validation is important as this is one of the most common areas of attack. API validation falls in-line with input validation, therefore while ensuring you have proper input validation, you should have proper API interactions. Thirdly, having safe error handling is integral to preventing malicious behavior.

2. Areas of Security Justification

To further explain why these areas of security are important, it's needed to justify each area. The main risk factor without having input validation would be SQL injection. SQL injection occurs when a user adds untrusted data to database queries to produce inconsistent behavior in the application.[1] Ensuring proper and secure query parameterization, because when you use it, you will never have an SQL injection attack. Furthermore, proper API interactions will prevent unauthorized access and API abuse. Finally, it is critical to have proper error logging as this will assist security forensics understand what occurred, such as a failed login attempt if an attack succeeds. We must ensure that warnings, errors and so on have time stamps, IP addresses that tie into the log entries to make it easy to track.[2] Ensuring we follow proper secure coding techniques in these particular areas should provide us with a secure command input function for the web application.

3. Code Review Summary

After reviewing the code base provided in Module 2.1 there are some potential vulnerabilities at play. The big, checking over the pom.xml, the explicit version for Spring Boot is outdated, using the latest versions unless absolutely necessary is typically best practise. To further discuss old versioning, the Java version 8 is also not supported, meaning security updates are also not being patched. There is also no Spring security dependency here which would force authentication and authorization. It is also recommended to use this annotation instead of SpringBootApplication for .java files.

As mentioned in Iron-Clad Java, it is important to ensure any output for errors is secure. The Greetings.java does not validate any user-input. The GreetingsController.java has unhandled exceptions, such as IndexOutOfBounds.

---

[1] Manico and Detlefsen, *Iron-Clad Java: Building Secure Web Applications (Oracle Press)*, chap. 7.
[2] Manico and Detlefsen, *Iron-Clad Java: Building Secure Web Applications (Oracle Press)*, chap. 9.

4. Mitigation Plan

While a simple print for ‘Hello World’ is not a security vulnerability, you want to ensure that error handling prints do not provide enough information to allow attackers to build upon strategies. Old versions of plugins may have security vulnerabilities that you are now giving to your code base. The Spring Security dependency enables authentication and authorization of applications.

Input validation is an important step for any program, web-based or not. Not validating user input allows for SQL injection.[3] Similarly to validating input, you must also validate error exceptions which may reveal security vulnerabilities to an attacker. Instead of checking if the user input is OutOfBounds after accessing the array, check if the ID is already beyond the bounds prior to array access.

[3] "Web Applications Basics." *Iron-Clad Java*, O'Reilly Media, n.d., https://learning.oreilly.com/library/view/iron-clad-java/9780071835886/ch01.html#ch01lev1sec12. Accessed 14 Mar. 2025.