Southern New Hampshire University

Elizabeth Marticello

CS 305 Module Two Written Assignment

1. Areas of Security

All seven of the listed areas in the Vulnerability Assessment Process Flow Diagram are important to ensure secure code. However, not all these areas are relevant to the current project. The Module Two project doesn't seem to take user input, therefore doesn't have a need for cryptography. Cryptography is used to validate password authenticity, protects users from eavesdropping attacks, and prevents fraud with e-signatures (IBM, n.d). Every other area in the Flow Diagram is relevant to ensure a secure product as explained more below.

2. Areas of Security Justification

It is essential to assess multiple areas in a project to ensure security is sufficient. Assessing:

- Input Validation is important to maintain project security, data integrity, and preserve a good user experience.
- APIs handles data transfers between different systems and must ensure that the data is protected.
- Cryptography, although not used in this project, excels at protecting sensitive information like passwords, web browsing, and electronic signatures (IBM, n.d). It's essential to assess the cryptography because it maintains the security when handling sensitive information.
- Client/Server is important to ensure the reliability of the application using a client/server model. Assessing this portion of the project will ensure that the data exchanged between both client and server is accurate and protected from unauthorized users (Jain, 2025).
- Code Error Handling is important as mentioned in Fail Open section of chapter 3 in Iron-Clad Java. The example in the book mentions that the lack of error handling outside of normal operations could lead to a breach in security where an attacker could gain unintended access. This example included a NullPointerException which is a subclass of RuntimeExceptions, which are not flagged by a complier, so it falls on the developer to ensure error handling is tested both in normal operations and unexpected ways too (Manico & Detlefsen, 2014).
- Code Quality can make or break a project. Having poor code quality could cause the misuse of features and create vulnerabilities to make the code hard to audit or maintain.

- Encapsulation limits access to internal data and helps limit the information available to attack. The Greeting class uses private fields, but the controller has opportunities as mentioned more below.

3. Code Review Summary

- Views. To my knowledge, the only View is the Dependency-Checker. As it's generated by the pom.xml I don't see how it could have any vulnerabilities.
- Models are classes in java that hold data. The Greeting.java does this with its getters and I see no vulnerabilities.
- Controllers. The GreetingController.java is a Controller that has vulnerabilities where it's using unvalidated user input as a Spring expression. As mentioned in Iron-Clad Java, an attacker can pass a malicious entry to the project to gain unintended access.
- Services are not present in this project.
- Plug-ins would include the spring boot maven plugin which seems to be an older version. This has the potential to be a vulnerability.

4. Mitigation Plan

The only vulnerabilities I've found in the Module Two Code Base were in the controller and plug-in. The controller seems to be using an unvalidated user input, like what was used as an example in the Iron-Clad Java text. Some potential mitigation techniques would be to use a centralized enforcement layer to ensure all input is validated in one place. Another would be to implement the ability to change a user's entitlement in real time to allow suspending users with suspicious activity. As for the plug-in. An update to the most recent version would be recommended to ensure that the most up-to-date security is being implemented.

Resources

(n.d.). *What is cryptography?* IBM. https://www.ibm.com/think/topics/cryptography

Jain, R. (2025, June 26). *Client Server Testing | What it is, Advantages & Challenges*. Testsigma. https://testsigma.com/blog/client-server-testing/

Manico, J., & Detlefsen, A. (2014). *Iron-Clad Java: Building Secure Web Applications* (pp. Anti-Pattern6FailOpen). Oracle Press. https://learning.oreilly.com/library/view/iron-clad-java/9780071835886/ch03.html#ch03lev2sec3