****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

[Document Revision History 3](#_Toc32574607)

[Client 3](#_Toc32574608)

[Instructions 3](#_Toc32574609)

[Developer 4](#_Toc32574610)

[1. Interpreting Client Needs 4](#_Toc32574611)

[2. Areas of Security 4](#_Toc32574612)

[3. Manual Review 4](#_Toc32574613)

[4. Static Testing 4](#_Toc32574614)

[5. Mitigation Plan 4](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **3/15/22** | **Christopher Clark** |  |

## Client



## Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found.

Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Christopher Clark

## 1. Interpreting Client Needs

*Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following regarding how companies protect against external threats based on the scenario information:*

* *What is the value of secure communications to the company?*
* *Are there any international transactions that the company produces?*
* *Are there governmental restrictions about secure communications to consider?*
* *What external threats might be present now and in the immediate future?*
* *What are the “modernization” requirements that must be considered, such as the role of open-source libraries and evolving web application technologies?*

This team has been tasked with determining the security requirements for our new client, Artemis Financial. Artemis is modernizing their systems and would like to implement the latest security protocols to protect their web-based software. It is crucial to maintain the highest level of security for this financial institution. Secure communications are vital. This is especially true given the nature of the data that is transferred. Client information needs to be always protected against attacks. Given the global nature of our financial systems, Artemis will require all protocols for secure transaction to follow any government regulations in place on either side of the transaction.

As we have witnessed in recent years, brute force attacks, denial of service attacks, ransomware, and data corruption attacks are on the rise. Security protocols must be in place and enforced to ensure protection of client data and prevent such attacks. This firm will need to provide essential guidance for hardware and software security to the client. Artemis intends to modernize and should take advantage of this time to upgrade aging equipment and utilize the most recent and secure versions of any code or software employed within the system.

## 2. Areas of Security

*Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.*

Understanding that this is a review and modernization of existing software, the following areas of security should be the main focus for our team.

* Input Validation. This is a critical step to ensuring secure transactions within a web-based system. All client-side data entry should be verified and determined accurate and safe using the latest version and most secure coding for packet transfer.
* Cryptography. Data encryption methods should meet the highest standards provided by operational software. All software should be of the latest versions and updated when patches or system upgrades are available.
* Client-Server. Given this is a web-based application, a review of all security protocols in place for client-server communications should be conducted. This goes hand in hand with input validation. HTML code should be in place that prevents data insertion and corruption via packet interception.
* Error Checking. Proper error checking and reporting is needed for secure database and system control. If improper or incorrect data is entered or transferred, the system should have a catch in place for this and be able to disable user account and report to the system administrators.

Other areas of security, i.e., APIs, and code quality would be considered. However, this project already has existing code. A review will be conducted but given the current use of RESTful API’s and existing operational code, the focus will remain on version updates and the issue listed above.

## 3. Manual Review

*Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.*

Upon manual review of the client’s provided code, there are a few areas of concern that should be addressed to improve security within the code. There are software versions indicated within the pom.xml file that should be updated. One example would be the spring framework version 2.2.4-RELEASE. Spring framework versions have reached 5.3.17. There is also liberal use of public access modifiers throughout the code. This should be reviewed, and all data not shared between classes should be set to private. One example is the variable for account balance within the customer.java file. That variable does not look to be shared within other classes and should have a private modifier. Input validation is required for data requests and client input. The DocData.java file has a server connection request with error checking. However, input validation and cryptography should be applied to the connection request. This means, user input for username and password should be encrypted. There should also be more stringent input validation with the use of multi-factor authorization. This also applies to the GreetingController.java file. There are mapping and parameter requests withing the greeting method that require secure API validation. Cryptography should be applied as needed. Some thought should be given to the possibility of porting this application over to the latest version of Java and JDK packages. All dependencies should be checked, and versions updated.

## 4. Static Testing

*Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from dependency check report. Include the following:*

1. *The names or vulnerability codes of the known vulnerabilities*
2. *A brief description and recommended solutions provided by the dependency check report*
3. *Attribution (if any) that documents how this vulnerability has been identified or documented previously*

The dependency report for Artemis Financial’s web-based software is shown in Figure 1.

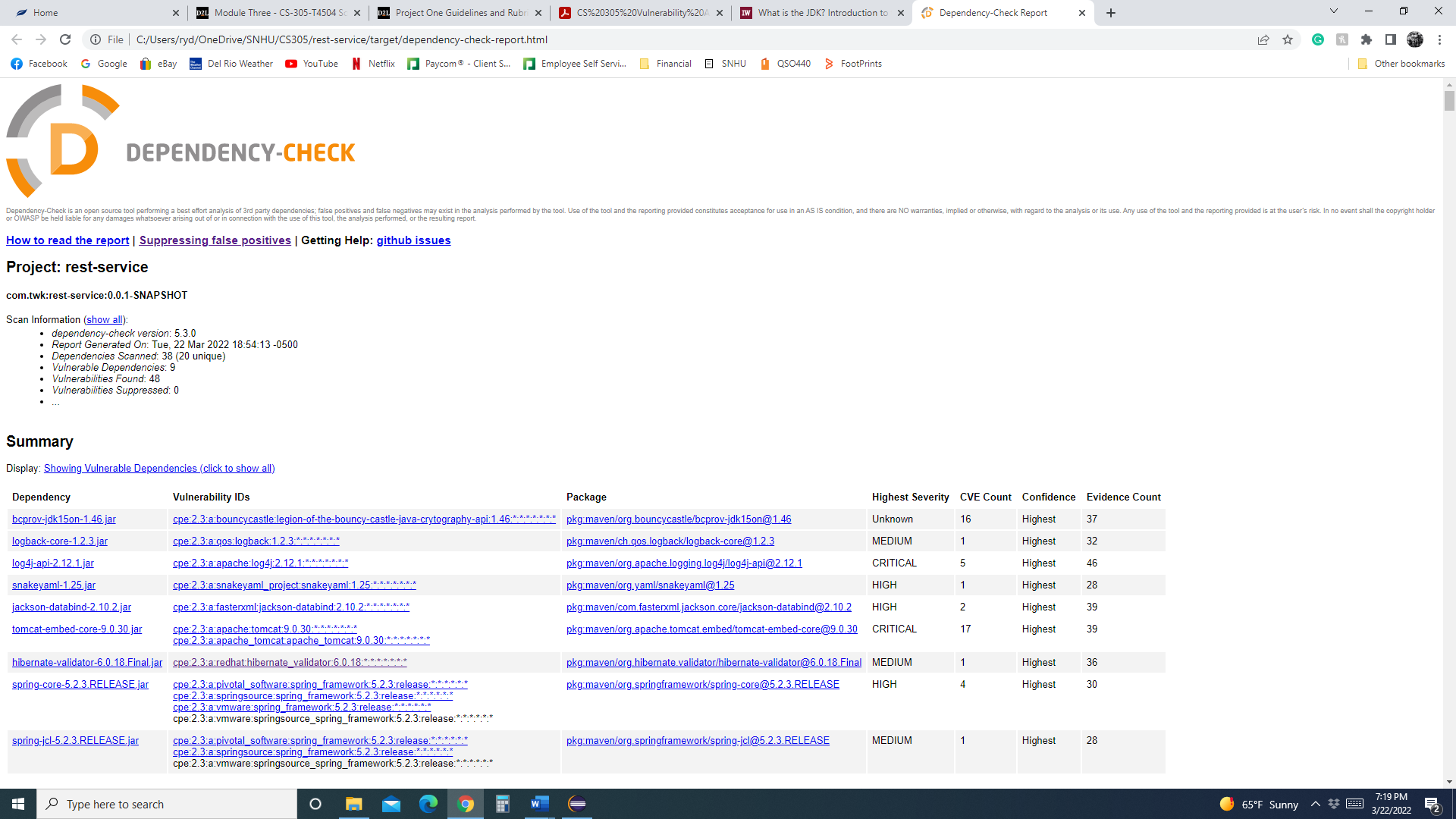


Figure 1

There are two critical and three high severity issues shown. Those include the log4j-api-2.12.1 and tomcat-embed-core-9.0.30 of which both are CRITICAL. The report also indicates that the snakeyaml-1.25, Jackson-datablind-2.10, and the spring-core-5.2.3-RELEASE are of HIGH severity.

**CRITICAL**

The log4j-api-2.12.1 dependency can lead to outside control of log messages or message parameters. This could lead to malicious code insertion. The widely accepted solution is to upgrade to the latest version of 2.17.x. This can eliminate the security issue.

The tomcat-embed-core-9.0 dependency include multiple vulnerabilities such as remote code execution, denial of service, and privilege escalation. As with many other dependencies, using the latest versions will provide the accepted solution to the issue.

**HIGH**

The snake-yaml-1.25 dependency can also lead to a DoS attack. This vulnerability leaves the system open to attack when using certain Alias features. A common solution to this is the upgrade to 1.26. The vulnerability has been removed.

The Jackson-datablind-2.10.2 version is at risk of data breach or denial of service attacks. Some solutions found include the use of the latest versions on the Jackson-datablind and being more specific about the use of polymorphism within system code.

The spring-core-5.2.3 dependency leads to a vulnerability to insertion attacks. An update to the latest version of the Spring Framework will resolve this vulnerability.

## 5. Mitigation Plan

*After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.*

There are several solutions to help mitigate the issues identified and address others yet to be found. One suggestion would be to start with a full port over to Java 17. Even though other versions are more popular at the moment, this is the latest long term support release for Java. Updating all dependencies to the latest versions is also recommended. Ensuring the continued use of REST API’s and secure HTML practices will be beneficial. During the port project, all issues identified with secure coding practices can be addressed. All classes and variables should be revisited, and the correct modifiers will be used.

Input validation methods, i.e., dual factor authorization, and client-side verification will be implemented. Protection from brute force, DoS, and ransomware attacks will be addressed with secure coding, input validation, and admin tools. One example would be account lock out upon login failure. Locking out accounts with multiple login failures, or accounts that are unused is one way to limit access to bad actors.

The client will receive recommendations with regard to equipment upgrades that will enhance security. Utilizing the latest hardware configurations can enhance security options. Database servers using RAID configurations can be used to secure data for moss and corruption. There are many other system upgrades that will allow for the latest encryption to work more efficiently and lower costs in the long run.

Finally, a battery of secure code testing should be accomplished once all changes and upgrades have been made. Ethical hacking methods could be employed to identify any vulnerabilities not yest addresses. This testing should occur incrementally throughout the project. Dependency checks will also help to weed out potential issues. All best secure coding and work practices will be employed to ensure the successful completion of this project. All business requirements will be addressed.