****

# 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** | **9/16/23** | **Amy Houseal** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also choose to include images or supporting materials. If you include them, make certain to insert them in all the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Amy Houseal

## Interpreting Client Needs

With Artemis Financial being a financial consulting company, the value of secure communication should be a top priority. When working with clients, they will have access and most likely a database of client information that includes tax identification, social security numbers, bank account numbers and other sensitive information. When communicating with other applications or digital communication, protecting this information should be prioritized.

Since this company is a financial company, it would be more secure to assume the international transactions take place. There is no documentation that states otherwise.

This company would need to consider the Social Security Number Privacy act, the Payment Card Industry Data Security Standard and to consider the protection of any trading information. With these regulations in mind, the current and future external threats would be to the client’s information. The modernized requirements that would be necessary for Artemis Financial would be to encrypt all client data and maintain any bug fixes.

## Areas of Security

* Input Validation – Input validation is necessary to protect user information.
* APIs – Since we are reviewing security for an API, creating this API is necessary to access the data.
* Cryptography – Cryptography is necessary to ensure that client information is not threatened by other areas of the world since there are international transactions.
* Code Quality – Having quality code will ensure what and how the user is able to access the application. This will keep both the company and its information secure.

## Manual Review

When reviewing the GreetingController.java file against the areas of security I listed, I noticed that there was a lack of input validation. Addressing input validation in this file will be necessary to protect the company and its data from external threats.

Across all files I noticed a lack of comments and header documentation for the program. This will make it difficult to update and maintain any bug fixes throughout the program, in turn this can affect the code quality.

When reviewing the API, it was not written through a POST method which is a risk to user input.

## Static Testing

|  |  |  |
| --- | --- | --- |
| **Dependency** | **Description** | **Solution** |
| bcprov-jdk15on-1.46.jar | The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7. | Updating to latest version |
| hibernate-validator-6.0.18.Final.jar | Hibernate's Bean Validation (JSR-380) reference implementation. | Updating to latest version |
| jackson-databind-2.10.2.jar | General data-binding functionality for Jackson: works on core streaming API | Updating certificate |
| log4j-api-2.12.1.jar | The Apache Log4j API | Updating to latest version |
| logback-core-1.2.3.jar | logback-core module | Updating to latest version |
| snakeyaml-1.25.jar | YAML 1.1 parser and emitter for Java | Use SnakeYaml's SafeConstructor when parsing |
| spring-boot-2.2.4.RELEASE.jar | Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0 | Updating to latest version of Spring Boot |
| spring-boot-starter-web-2.2.4.RELEASE.jar | Starter for building web, including RESTful, applications using Spring | Updating to latest version of Spring Boot |
| spring-core-5.2.3.RELEASE.jar | Spring Core | Run on Tomcat as a WAR deployment |
| spring-web-5.2.3.RELEASE.jar | Spring Web | Run on Tomcat as a WAR deployment |
| spring-webmvc-5.2.3.RELEASE.jar | Spring Web MVC | Run on Tomcat as a WAR deployment |
| tomcat-embed-core-9.0.30.jar | Core Tomcat implementation | Upgrade to latest version and make changes to configurations |
| tomcat-embed-websocket-9.0.30.jar | Core Tomcat implementation | Upgrade to latest version and make changes to configurations |

## Mitigation Plan

From my manual review and the static testing, I would suggest upgrading all necessary packages and frameworks. After that is complete, I would update the API to include input validation and review the code quality. In addition I would address the cryptography of the API.