****

# 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** | **3/22/2024** | **Victoria Keyser** | **Completed all sections** |

## 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

Victoria Keyser

## Interpreting Client Needs

Artemis Financial recognizes the value of secure communications in safeguarding sensitive financial information and upholding clients' trust. While specifics regarding international transactions remain undisclosed, the company prioritizes adherence to regulatory compliance to protect financial integrity, including considering potential governmental restrictions, executing access control features, maintaining audit trails, and employing encryption measures. Additionally, Artemis Financial should remain vigilant against external threats, such as malicious activities that steal financial data. Global Rain should evaluate the security implications of utilizing open-source libraries to support and maintain cyber defense mechanisms, thereby addressing modernization requirements. Furthermore, Global Rain and Artemis Financial should be mindful of evolving web application technologies as part of their ongoing efforts to enhance security measures.

## Areas of Security

Implementing input validation in a financial environment helps prevent unauthorized access and manipulation of data. By validating user inputs, Artemis Financial can reduce the risk of security breaches and ensure the integrity of its systems. Additionally, employing cryptography adds an extra layer of security by protecting against unauthorized access and data breaches, enhancing defense for the company's customers. Furthermore, ensuring secure interactions with APIs assures that only authorized users and systems can access sensitive information and functionalities, thus preventing unauthorized access to bank information and maintaining the confidentiality of client data.

## Manual Review

pom.xml file: <dependency> <groupId> org.bouncycastle</groupId> (line 27-31) - Bouncy Castle Cryptography could lead to security issues that deal with encryption and security (as seen in the dependency check)

pom.xml file: <version> 9.0.10 </version> (line 57-68) - OWASP Dependency Check Plugin could improperly execute if not updated with newer versions, updated this from 5.3.0 version to 9.0.10 version to run the dependency check.

application.properties: file.upload-dir=./uploads (line 11-12) - storing uploaded files in a directory accessible via URL and could possibly be access by unauthorized users.

DocData.java: ("jdbc:mysql://localhost:3306/test", "root", "root"); (line 25-28)

Username and password are hard coded within the source code and could lead to source code being accessed by unauthorized parties.

Connection is also not closed after use which can result in resource leakage.

customer.java: public int showInfo () (line 7-10) - lacks encapsulation and lead to manipulation of account information (account\_number and account\_balance)

lacks data validation to ensure that the deposit amount is valid (non-negative)

## Static Testing

A screenshot of a computer

Description automatically generated

bcprov-jdk15on-1.46.jar (High severity)

**Description**: 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

**Recommendation**: Update to a version that doesn't contain the reported vulnerabilities

hibernate-validator-6.0.18.Final.jar (Medium severity)

**Description**: Hibernate's Bean Validation (JSR-380) reference implementation

**Recommendation**: upgrade to the latest version to address the identified vulnerabilities

Jackson-databind-2.10.2.jar (High severity)

**Description**: General data-binding functionality for Jackson: works on core streaming API

**Recommendation**: upgrade to the latest version to address the identified vulnerabilities

log4j-api-2.12.1.jar (Low severity)

**Description**: Apache Log4j API

**Recommendation**: Upgrade to the latest version to mitigate potential security risks

logback-core-1.2.3.jar (High severity)

**Description**: logback-core module

**Recommendation**: upgrade to the latest version to address the identified vulnerabilities

snakeyaml-1.25.jar (Critical severity)

**Description**: YAML 1.1 parser and emitter for Java

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

spring-boot-2.2.4.RELEASE.jar (Critical severity)

**Description**: Spring Boot

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

spring-boot-starter-web-2.2.4.RELEASE.jar (Critical severity)

**Description**: Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container.

**Recommendation**: update to a newer version to address the critical vulnerabilities

spring-core-5.2.3.RELEASE.jar (Critical severity)

*\* Dependency has a known exploited vulnerability*

**Description**: Spring Core

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

**Vulnerability code**: CVE-2022-22965

**Description**: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding, allowing attacker to execute arbitrary ode

**Recommendation**: apply updates per vendor instructions

**Attribution**: CWE-94 improper control of generation of Code ('Code injection')

spring-web-5.2.3.RELEASE.jar (Critical severity)

*\*Dependency has a known exploited vulnerability*

**Description**: Spring Web

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

**Vulnerability code**: CVE-2022-22965

**Description**: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.

**Recommendation**: apply updates per vendor instructions

**Attribution**: CWE-94 improper control of generation of code ('Code Injection')

spring-webmvc-5.2.3.RELEASE.jar (Critical severity)

*\* Dependency has a known exploited vulnerability*

**Description**: Spring Web MVC

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

**Vulnerability code**: CVE-2022-2965

**Description**: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via ta binding. This specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.

**Recommendation**: apply updates per vendor instructions

**Attribution**: CWE-94 improper control of generation of code ('Code Injection')

tomcat-embed-core-9.0.30.jar (Critical severity)

*\* Dependency has a known exploited vulnerability*

**Description**: Core Tomcat implementation

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

**Vulnerability code:** CVE-2020-1938

**Description**: Apache Tomcat treats Apache JServ Protocol (AJP) connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited.

**Recommendation**: apply updates per vendor instructions

**Attribution**: NVD-CWE-Other

**Vulnerability code:** CVE-2023-44487

**Description**: HTTP/2 protocol allows a DoS (server resource consumption) because request cancellation can reset many streams quickly, as exploited in the wild in August through October 2023

**Recommendation**: apply mitigation per vendor instructions or discontinue use of the product if mitigations are unavailable

**Attribution**: CWE-400 uncontrolled resource consumption

tomcat-embed-websocket-9.0.30.jar (Critical severity)

*\* Dependency has a known exploited vulnerability*

**Description**: Core Tomcat implementation

**Recommendation**: upgrade to the latest version to address the critical vulnerabilities

**Vulnerability code:** CVE-2020-1938

**Description**: HTTP/2 protocol allows a DoS (server resource consumption) because request cancellation can reset many streams quickly, as exploited in the wild in August through October 2023

**Recommendation**: apply mitigation per vendor instructions or discontinue use of the product if mitigations are unavailable

**Attribution**: CWE-400 uncontrolled resource consumption

**Vulnerability code**: CVE-2023-44487

**Description**: HTTP/2 contains a rapid reset vulnerability that allows for a distributed DoS

**Recommendation**: apply mitigation per vendor instructions or discontinue use of the product if mitigations are unavailable

**Attribution**: CWE-400 uncontrolled resource consumption

## Mitigation Plan

* Implement secure coding practices to prevent common vulnerabilities.
* Implement secure methods for storing and retrieving credentials.
* Close the database connections after use.
* Implement data validation methods.
* Upgrade libraries to address all identified vulnerabilities and mitigate potential security risks.
* Ensure that the OWASP Dependency Check Plugin is updated to the latest version to execute properly.
* Regular monitoring for new security advisories and updates related to dependencies and libraries used in the application.