****

# 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** | **7/16/2023** | **Jenna Case** | **First version of the document** |

## 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

Jenna Case

## Interpreting Client Needs

Secure communication allows the company to protect personal information when receiving or sending data between the client and the server. Using secure communication methods ensures that there is not a breach of data. Artemis Financial deals with confidential customer information so they need to be utilizing secure communications. There currently does not seem to be any international transactions that the company produces, but that could always be a possibility in the future. There is an executive order on protecting sensitive data for Americans so secure communications are vital for Artemis Financial. External threats will always be present now and in the immediate future. These threats come from hackers and anyone who would want to steal personal information. In the future, Artemis Financial may want to integrate other kinds of software into their system. This means that they need to actively make sure that their current software is secure so that there are not any problems when integrating.

## Areas of Security

* Input Validation – The program allows for input in the form of a string. This input needs to be valid to avoid any failures and to make the program secure.
* APIs – The application is designed to run in a web browser, so a good API is essential.
* Cryptography – Since there is a lot of secure information that is being used, cryptography could be useful to keep the information private.
* Code Error – Secure error handling goes along with the input validation.
* Code Quality – Secure coding practices will help keep the application secure from the start.

## Manual Review

* It does not appear that there is an authentication system in place to verify users. This is the first way to protect against attacks, so it is crucial to include authentication systems.
* There is no input validation. There is the ability to enter information, but no way for it to check and validate.
* There is no encryption present.

## Static Testing

* bcprov-jdk15on-1.46.jar – The version does not fully validate encoding of signature on verification.
* hibernate-validator-6.0.18.Final.jar – A bug in the message interpolation processor enables invalid expressions to be evaluated as if they were valid.
* jackson-databind-2.10.2.jar – There is an issue where the entity expansion was not secured properly.
* log4j-api-2.12.1.jar – There is improper validation of certificate with host mismatch in the appender. This could cause the connection to be intercepted.
* logback-core-1.2.3.jar – An attacker with the required privileges to edit configurations files could make a malicious configuration to execute arbitrary code.
* snakeyaml-1.25.jar – The Constructor() class does not restrict types which can be instantiated during deserialization.
* spring-boot-2.2.4.RELEASE.jar – Old version, unsupported.
* spring-boot-starter-web-2.2.4.RELEASE.jar – Old version, unsupported.
* spring-core-5.2.3.RELEASE.jar – Vulnerable to remote code execution via data binding.
* spring-web-5.2.3.RELEASE.jar – Potential remote code execution issue.
* spring-webmvc-5.2.3.RELEASE.jar – Vulnerable to remote code execution via data binding.
* tomcat-embed-core-9.0.30.jar – Must be more careful with incoming connections. The AJP connections are treated with higher trust and can be exploited in ways that may be surprising.
* tomcat-embed-websocket-9.0.30.jar - Must be more careful with incoming connections. The AJP connections are treated with higher trust and can be exploited in ways that may be surprising.

## Mitigation Plan

I would start with updating the Spring Framework and eliminating a lot of vulnerabilities with the outdated software. I would mitigate the Tomcat threats to keep sensitive information in the safe zone. With this I would also want to add some authentication systems to prevent data leaking. I would then move on to mitigating the rest of the dependencies highlighted in the dependency check to keep data secure and safe. I would also add in input validation measures so that attacks to private data can be reduced.