****

# 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** | **09/17/2023** | **Armando Gomez** |  |

## 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

Armando Gomez

## Interpreting Client Needs

Artemis Financial is a company that focuses on creating plans that cover savings, retirements, investments, and insurance. They need to ensure communication to protect sensitive client information like social security numbers and tax details. Since the company's transactions can have a global reach, they have to comply with global data protection regulations and take a proactive approach to safeguarding trade secrets. The presence of threats highlights the importance of strong encryption and staying up-to-date with modern technologies for web applications, including incorporating open-source libraries.

## Areas of Security

**Input validation:** Crucial for verifying the ownership of information and protecting users. It is achieved through the implementation of string validations.

**Code Quality**: Vital role in ensuring controlled access to methods, based on user roles. This helps maintain data privacy and enhances system security.

**APIs:** Essential for defining parameters for data access both internally and externally. They serve as a measure against potential data breaches.

**Code Error:** Implementing error handling mechanisms in the API allows us to identify areas that require fixes thereby improving system resilience.

**Cryptography:** Critical in securing user information across geographical regions and currency domains. It acts as a safeguard against breaches.

## Manual Review

After examining the POM.XML and the Greeting Controller, we discovered weaknesses. These include the absence of an Apache Validator and insufficient input validation in the Greeting Controller. The API has some drawbacks as well, such as a potential vulnerability that could expose data because there is no POST method. We also noticed that cryptography is not implemented, which is an area that needs improvement.

## Static Testing

During the testing process, we discovered several vulnerabilities in the dependencies utilized for the project. Here are the specific vulnerabilities that were identified, along with the steps to address and resolve them.

1. **hibernate-validator-6.0.18.Final.jar**

**Issue:** There is a vulnerability in the message interpolation processor that allows for circumvention of input sanitation measures.

**Remediation:**

1. **Jackson-databind-2.10.2.jar**

**Issue:** Lack of security measures can make systems susceptible, to XML external entity (XXE) attacks, which exploit inadequate entity expansion protocols.

**Remediation:**

Upgrade to a version that addresses the XXE vulnerability.

Take steps to implement security controls that can prevent XXE attacks like disabling DTDs and external entities in XML parsers.

It's important to provide developers with training on secure coding practices to avoid XXE vulnerabilities.

Regarding the "log4j api 2.12.1.jar" issue;

One concern is the possibility of man in the middle attacks due to certificate validation in the SMTP appender.

To address this its recommended to enable verification globally for SMTPS connections, which will help mitigate potential man in the middle attacks.

Additionally implementing SSL certificate management practices will ensure secure connections are maintained.

Lastly it's crucial to establish monitoring mechanisms that can detect and alert on any man, in the middle attacks.

1. **log4j-api-2.12.1.jar**

**Issue:** The SMTP appender is vulnerable to man, in the middle attacks because it does not properly validate certificates.

**Remediation:** Make sure to activate verification globally to prevent any possible man in the middle attacks when establishing SMTPS connections.

To ensure connections it is crucial to implement strong SSL certificate management practices.

Set up monitoring mechanisms that can detect and notify any man, in the middle attacks thus ensuring security.

1. **snakeyaml-1.25.jar**

**Issue:** During the process of loading operations, it enables the expansion of entities, which can be seen as a vulnerability, to CVE 2003 1564.

**Remediation:** Please make sure to update to the version, which includes fixes for the vulnerability. It's important to implement input validation to prevent any attempts at expanding entities. Additionally, conducting security audits will help identify and address potential vulnerabilities effectively.

1. **spring-aop-5.2.3.RELEASE.jar and spring-core-5.2.3.RELEASE.jar**

**Issue:** Some browsers may have vulnerabilities that could allow for RFD attacks to bypass protections, in specific conditions.

**Remediation:** Update to the version, which includes fixes, for the RFD vulnerability.

To enhance security and protect against RFD attacks it is advisable to implement security headers.

Mitigate the risk of RFD attacks by establishing browser security policies.

1. **tomcat-embed-core-9.0.30.jar and tomcat-embed-websocket-9.0.30.jar**

**Issue:** Vulnerability to requests, allowing smuggling to occur when using a proxy because the HTTP transfer encoding request header is parsed incorrectly.

**Remediation:** Upgrade to the recent version as it addresses the request smuggling vulnerability. Additionally, it is essential to configure your reverse proxy settings in order to prevent any potential request smuggling. Lastly conducting security testing will help identify and fix any vulnerabilities related to request smuggling.

## Mitigation Plan

To effectively address the vulnerabilities that have been identified, Artemis Financial should promptly update to the latest versions of critical dependencies such as Snakeyaml, Hibernate Validator, and Apache Tomcat. At the time, it was crucial to strengthen the application's security infrastructure by implementing robust input validation mechanisms and enhancing error handling procedures. Incorporating encryption techniques will play a key role in protecting sensitive data during transactions and storage. Additionally, promoting a culture of security awareness through training sessions and adopting secure coding practices can further strengthen the application's defense against potential threats. This proactive approach, focused on improvement, aims to establish a resilient security stance that fosters trust and reliability among our clientele.