Artemis Financial Vulnerability Assessment Report

Table of Contents

Document Revision History 3

Client 3

Instructions 3

Developer 4

1. Interpreting Client Needs 4

2. Areas of Security 4

3. Manual Review 4

4. Static Testing 4

5. Mitigation Plan 4

Document Revision History

**Version**

**Date**

**Author**

**Comments**

**1.0**

**February 17th 2024**

**Corey Hamilton**

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

Corey Hamilton

• Interpreting Client Needs

​​**Value of Secure Communications​:**

Given Artemis Financials handling of classified client information, including SSNs and tax details, secure communications are absolutely crucial. It serves as the backbone for safeguarding sensitive data and maintaining the trust of clients.

International Transaction**s:**

Since there's no indication that Artemis Financial operates solely within the US, it's reasonable to assume international transactions are part of their operations. This underscores the importance of robust security measures that are adaptable to different regulatory environments.

**Governmental Restrictions:**

A key governmental restriction revolves around protecting trade secrets. Artemis Financial must ensure that secure communications prevent any exposure of classified information, aligning with regulations to safeguard vital financial data.

**External Threats:**

The primary external threat revolves around the targeted compromise of client information. To counter this, a robust encryption strategy is necessary to shield client data from any unauthorized access outside the company.

**Modernization Requirements:**

Keeping up with modern security standards is imperative. ​Regular maintenance​ ​checks, bug fixes, and​ addressing evolving security threats are essential modernization requirements. Artemis Financial should prioritize staying current to fortify its defense against potential vulnerabilities.

• Areas of Security

**Input Validation:** Artemis Financial requires stringent input validation to confirm data ownership. Implementing this validation as strings adds a layer of protection, mitigating unauthorized access and ensuring the integrity of user information.

**Code Quality:**Controlling method access based on user roles is vital for a secure user experience. Code quality ensures that users only access their own information, preventing unauthorized entry to other user data or server resources.

**APIs:**The creation of an API is crucial for both internal and external operations. APIs define acceptable data access, regulating how information is accessed. This measure is essential for controlling data flow and preventing unauthorized access.

**Code Error:**  Implementing error handling is proactive in identifying and rectifying issues within the API. This ensures that Artemis Financial is shielded from potential exposure of user information due to unforeseen errors, contributing to system robustness.

**Cryptography:**   Integrating cryptography is paramount, especially for international transactions involving different currencies. This safeguards user information from potential compromises, providing an additional layer of security in a diverse financial landscape.

• Manual Review

Aspect  | Findings

POM.XML. - Verified the presence of an Apache Validator, ensuring secure handling of dependencies.

Aspect  | Findings

Greeting Controller. - Identified a lack of input validation, highlighting a key aspect for future development. - Acknowledged acceptable code quality but noted a deficiency in error handling, warranting attention. - Observed API vulnerabilities, specifically the absence of a secure POST method, posing a risk to user input exposure.

• Static Testing

• Mitigation Plan

Upgrade Bouncy Castle, Hibernate Validator, SnakeYAML, Spring Boot, and Spring Framework components to the latest versions. Monitor for updates, conduct thorough testing, and maintain a backup.