****

# 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** | **[Date]** | **Cameron Ziefel** |  |

## 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

Cameron Ziefel

## Interpreting Client Needs

1. **What is the value of secure communications to the company?**

The value of secure communications at Artemis Financial is of upmost importance.

Considering Artemis Financial handles sensitive data like customer social security numbers, names, birthday, and account numbers, it is extremely important for Artemis Financial to have the most secure communications possible.

1. **Does the company make any international transactions?**

The scenario does not say if Artemis Financial makes international transactions. It would be safer to assume that Artemis Financial DOES make international transactions. This would provide extra security to the system whether they make international transactions or not.

1. **Are there governmental restrictions about secure communications to consider?**

I am not aware of any restrictions at this time

1. **What external threats might be present now and in the immediate future?**

The main threat with the type of company that Artemis Financial is, is the threat of theft of the customer’s sensitive data. Social Security Numbers, account numbers, names, and other information regarding the customer will always be the first target.

1. **What are the modernization requirements that you must consider?**

When considering the modernization of technology, everyday it becomes more advanced and new ways to hack and deceive are created. Regular maintenance and attention to the security of the system will be key to keeping up with the new techniques to break security.

## Areas of Security

* Input Validation: Artemis Financial will require strong input validation so the system can verify that authorized users are the ones accessing their account. Unique usernames for each customer will be required, along with passwords that require specific criteria such as one uppercase letter, one number, one special character, and the password would be required to meet a minimum length.
* API’s: Artemis Financial uses a RESTful API. Proper security of the API will be reviewed and improved if necessary.
* Cryptography: Cryptography will be needed within this system to provide more security for sensitive data that resides in the databases of the system.

## Manual Review

After reviewing the code, the first noticeable problem was the lack of input validation within the GreetingController class. Input validation will need to be added to provide security from unauthorized access. Cryptography was also missing, this will need to be added to also provide additional security. The API would benefit from adding a POST method. The POST method is a secure way of transferring sensitive data.

## Static Testing

After conducting a dependency check, I received the results of dependencies that have security vulnerabilities

1. **Bcprov-jdk15on-1.46.jar**

**Description:** The TLS implementation in the Bouncy Castle Java library before 1.48 and C# library before 1.8 does not properly consider timing side-channel attacks on a noncompliant MAC check operation during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets, a related issue to CVE-2013-0169.

**Solution:** Update The Bouncy Castle Crypto Package to the latest version.

1. **Spring-boot-2.2.4.RELEASE.jar**

**Description:** spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking.

**Solution:** Update spring-boot to the latest version.

1. **Logback-core-1.2.3.jar**

**Description:** In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuring allowing to execute arbitrary code loaded from LDAP servers.

**Solution:** Update the logback-core module to the latest version.

1. **Log4j-api-2.12.1.jar**

**Description:** Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender.

**Solution:** This vulnerability was fixed in Apache Log4j 2.12.3 and 2.13.1. Updating to the latest version of Apache Log4j API will fix this vulnerability.

1. **Snakeyami-1.25.jar**

**Description:** The Alias feature in SnakeYAML before 1.26 allows entity expanstion during a load operation.

**Solution:** Update SnakeYAML to the latest version.

1. **Jackson-Databhind-2.10.2.jar**

**Description:** A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly.

**Solution:** Update the Jackson-Databind to the latest version.

1. **Tomcat-embed-core-9.0.30.jar**

**Description:** The refactoring present in Apache Tomcat 9.0.28 to 9.0.30, 8.5.48 to 8.5.50 and 7.0.98 to 7.0.99 introduced a regression

**Solution:** Update Apache Tomcat to the latest version.

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

**Description:** A flaw was found in Hibernate Validator version 6.12.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid.

**Solution:** Vulnerable versions of this software include 5.0.0 up to 6.0.20 (excluding 6.0.20), The fix would be to update Hibernate’s Bean Validation to version 6.0.20

## Mitigation Plan

To resolve the issues found within the code and the dependency check. Proper input validation and cryptography are required to be added to meet an acceptable level of security. Also, updating all the affected dependencies will correct their vulnerabilities.