****

# 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** | **9/15/2023** | **Robert Cook** |  |

## Client



## Developer

Robert Cook

## Interpreting Client Needs

Secure communications would be a high priority for Artemis Financial since they deal with financial information and transactions ranging from entrepreneurs to government agencies domestically and internationally. Since they are making international transactions and dealing with government agencies, they will have to consider the governmental restrictions about communications and its security. Complying with the regulations imposed by the government is important to avoid any legal fees/penalties and to keep the trust with customers. External threats are almost guaranteed to happen now and in the future due to the sensitive nature of the information involved. When modernizing the clients’ operations, we need to implement the most current and effective software security to help prevent external threats in the future.

## Areas of Security

Based on the client’s needs and the code given these are the areas of security I recommend:

* APIs

Need to validate the user to make sure that they are accessing relevant information they need and nothing else. If there is no validation, then anyone can access the sensitive client files.

* Input Validation

Using APIs for this situation is imperative since the data being transmitted is of a sensitive nature. APIs provide authorization and authentication to keep the information they are sending safe and private.

* Cryptography

Encryption keeps bad actors from getting access to the information by intercepting API requests. In situations like this if encryption is used correctly data is only accessible by the recipient and sender.

* Code error

Errors can lead to corruption of data, privacy leaks, unauthorized access of data, leakage of data, etc. This is why making sure there are no errors in the code is so important.

* Code quality

Having well written and structured code is important because it creates less vulnerabilities such as SQL injections. It also makes sure that input validation is appropriate thus making the application more resilient to attacks.

* Encapsulation

This keeps sensitive information restricted to those with authorization which reduces the chances of successful attacks on it.

## Manual Review

* CrudController.java- Input should be validated before accessing DocData.
* DocData.java- the code attempts to access the database using a hard-coded username and password.
* GrettingController.java- no input validation is being used and the user should be validated properly to prevent security breaches.
* myDateTime.java- classes need to be encapsulated.

## Static Testing

|  |  |
| --- | --- |
| **Vulnerability ID** | **Description** |
| CVE-2016-1000352 | In the Bouncy Castle JCE Provider version 1.55 and earlier the ECIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider. |
| CVE-2023-20883 | In Spring Boot versions 3.0.0 - 3.0.6, 2.7.0 - 2.7.11, 2.6.0 - 2.6.14, 2.5.0 - 2.5.14 and older unsupported versions, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache. |
| CVE-2021-42550 | In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers. |
| CVE-2021-44832 | Apache Log4j2 versions 2.0-beta7 through 2.17.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable to a remote code execution (RCE) attack when a configuration uses a JDBC Appender with a JNDI LDAP data source URI when an attacker has control of the target LDAP server. This issue is fixed by limiting JNDI data source names to the java protocol in Log4j2 versions 2.17.1, 2.12.4, and 2.3.2. |
| CVE-2022-1471 | SnakeYaml's Constructor() class does not restrict types which can be instantiated during deserialization. Deserializing yaml content provided by an attacker can lead to remote code execution. We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. We recommend upgrading to version 2.0 and beyond. |
| CVE-2022-3064 | Parsing malicious or large YAML documents can consume excessive amounts of CPU or memory. |
| CVE-2023-35116 | \*\* DISPUTED \*\* jackson-databind through 2.15.2 allows attackers to cause a denial of service or other unspecified impact via a crafted object that uses cyclic dependencies. NOTE: the vendor's perspective is that this is not a valid vulnerability report, because the steps of constructing a cyclic data structure and trying to serialize it cannot be achieved by an external attacker. |
| CVE-2023-41080 | URL Redirection to Untrusted Site ('Open Redirect') vulnerability in FORM authentication feature Apache Tomcat.This issue affects Apache Tomcat: from 11.0.0-M1 through 11.0.0-M10, from 10.1.0-M1 through 10.0.12, from 9.0.0-M1 through 9.0.79 and from 8.5.0 through 8.5.92. The vulnerability is limited to the ROOT (default) web application. |

## Mitigation Plan

I would recommend the following course of action to correct the vulnerabilities that were identified:

* Use the latest Bouncy Castle JCE version.
* Update Spring Boot to the most recent version.
* Implement a user validation function.
* Ensure high quality code is being used.
* Do not hard code sensitive info on clients.
* Use encapsulation correctly in the code.
* Don’t parse large YAML documents.