### Southern New Hampshire University

3-1 Project One Submission

David Vega

### Dr. Vivian Lyon

### CS-305 – Software Security

01/22/2023

****

# Artemis Financial Vulnerability Assessment Report

Table of Contents

[Southern New Hampshire University 1](#_Toc125386718)

[Dr. Vivian Lyon 1](#_Toc125386719)

[CS-305 – Software Security 1](#_Toc125386720)

[Artemis Financial Vulnerability Assessment Report 2](#_Toc125386721)

[Document Revision History 4](#_Toc125386722)

[Client 4](#_Toc125386723)

[Developer 5](#_Toc125386724)

[1. Interpreting Client Needs 5](#_Toc125386725)

[2. Areas of Security 6](#_Toc125386726)

[3. Manual Review 7](#_Toc125386727)

[4. Static Testing 8](#_Toc125386728)

[bcprov-jdk15on-1.46.jar 9](#_Toc125386729)

[hibernate-validator-6.0.18.Final.jar 10](#_Toc125386730)

[jackson-databind-2.10.2.jar 10](#_Toc125386731)

[log4j-api-2.12.1.jar 11](#_Toc125386732)

[logback-core-1.2.3.har 11](#_Toc125386733)

[snakeyaml-1.25.jar 12](#_Toc125386734)

[spring-boot-2.2.4.RELEASE.jar 13](#_Toc125386735)

[spring-boot-starter-web-2.2.4.RELEASE.jar 13](#_Toc125386736)

[spring-core-5.2.3.RELEASE.jar 14](#_Toc125386737)

[spring-web-5.2.3.RELEASE.jar 14](#_Toc125386738)

[spring-webmv-5.2.3.RELEASE.jar 15](#_Toc125386739)

[tomcat-embed-core-9.0.30.jar 15](#_Toc125386740)

[tomcat-embed-websocket-9.0.30.jar 16](#_Toc125386741)

[5. Mitigation Plan 16](#_Toc125386742)

[References 17](#_Toc125386743)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **01/21/2023** | **David Vega** | **Initial Release** |
| **1.1** | **1/23/2023** | **David Vega** | **Resubmission: Addt’l descriptions added for remaining vulnerabilities** |

## Client



## Developer

David Vega

## Interpreting Client Needs

Artemis Financial puts secure communications at the top of their list of priorities. Without the proper security channels in place, there is no business. The company services a wide variety of portfolios for a varying customer base and a security event can severely impact their customers. There is no single product offering that Artemis Financial offers that won’t be impacted by a security event. For many of their customers, this will be a devastating blow to their livelihood as well as future financial security.

Considering that Artemis Financial has contracted with Global Rain, a global leader in software design and development whose customer base expands globally, it stands to reason that Artemis Financial has customers that have international assets. The expectation is to experience growth with the implementation of this new software. Artemis Financials’ portfolio may include a large proportion of US assets but with the implementation of this new software, they are expecting to accommodate the global markets. The security features will allow the company to better position themselves to provide international offerings.

Artemis Financial will need to adhere to governmental restrictions when it comes to secure communications. There are several government agencies that regulate the financial markets and the companies that engage in financial activities. The purpose of these regulatory agencies is to mitigate fraud and provide transparency within the markets and with consumers.

There are multiple external security threats out there, and it seems that the list continues to grow as processors become more advanced and existing encryption methods become less reliable. Some of the common threats are phishing, and ransomware just to name a few. These threats prey on the trust of employees and or consumers to gain access to systems that store personal information. As we enter the age of automation, these cyber threats are expanding into personal hardware devices such as smartphones.

The role of open-source libraries is mainly for development. Open source is meant to be a resource for developers to develop software quickly and efficiently by using pre-defined code snippets or entire code bases in their projects. Open-source code is deployed with open-source licensing in which it gives the developer freedom and flexibility to use the code as they see fit. Some of these licenses contain clauses that require users of the code to release their version back to the open-source community so that it may be used in future adaptations. Not all open-source code has the same licensing agreements. For example, a clause that can be found on an open-source code base called a *“copyleft”* can require the user of the code to release their version as open-source in its entirety which may contradict other open-source licensing agreements of other code bases that you may have implemented in your project potentially rendering your version unusable in proprietary software and less attractive for commercial use.

## Areas of Security

The areas of security that are relevant to the software application for Artemis Financial include:

* **Input Validation:** There are areas of input within the code especially in the controller files.
* **Cryptography –** Depending on the nature of the business transactions, consideration for international transactions should be discussed. If client’s have accounts overseas, then Artemis Financial should be implementing the most current encryption methods.
* **APIs:** The controller files handle the RESTful architecture.
* **Client/Server:** This code is primarily focused on activities associated with server side functionality. There are methods requesting for data from a database
* **Code Quality:** It is possible that implementations of multiple open-source code can vary in syntax as well as functionality. It is important to review these components and maintain a consistent syntactical flow for readability.
* **Code Error:** The DocData.java file includes a call to the database. The connection is established, and a try/catch statement is implemented to throw an exception if the connection is unsuccessful.

## Manual Review

A manual review of the application code revealed some potential vulnerabilities. First, I determined that the version of maven dependency check in the pom.xml file is outdated. I would make sure to review all dependencies and plug-ins to ensure that the most up-to-date releases are used. These updated releases could resolve vulnerabilities identified in previous versions.

I also noticed the code is incomplete. For example, there are no setter methods in the Greeting.java file. The methods to get and return the id and content are defined but nothing sets them to the private variables. I’m also certain that the customer, Greeting, and myDateTime.java files are missing imports.

In addition, there are no validations in place for the inputs. Some examples of validation should be checking for appropriate data types, and field length. The account\_balance in the customer.java file is neither set to private or public. However, I believe the default is set to public. This could cause conflict if access to this variable is permitted outside of its intended scope. I also do not see in the controller files where a route-endpoint is defined. A dependency check is also implemented using a maven plug-in.

## Static Testing

Graphical user interface, text, application, email

Description automatically generated



### bcprov-jdk15on-1.46.jar

**Graphical user interface, text, application, email

Description automatically generated**

This package is a collection of cryptographic APIs to enable cryptography. One of the published vulnerabilities was the allowed use of ECB mode (CV-2016-1000344). This mode is weak because if there is an identical block of text the cipher text will be identical too; allowing an intruder to guess the mapping of a block text to the cipher (Common Vulnerabilities and Exposure "CVE", 2023).

Figure 1: CV-2016-1000344

**Graphical user interface, text, application, Teams

Description automatically generated**

### hibernate-validator-6.0.18.Final.jar

**Graphical user interface, text, application, email

Description automatically generated**

The bug was identified in version 6.1.2.Final of Hibernator Validator. The bug allowed for carefully crafted user input as EL expressions to be passed to a constraint violation message which would then execute the user input before returning the message to the user. This would essentially bypass any input validation. Interpolation for constraint violations would mitigate arbitrary code within a string literal to be executed (Common Vulnerabilities and Exposure "CVE", 2023).

### jackson-databind-2.10.2.jar

Graphical user interface, text, application, email, Teams

Description automatically generated

This vulnerability affects the functionality of the Jackson data processor which is used to parse and return JSON from Java code. An intruder can exploit the vulnerability through the processing of deeply nested objects and array values as was identified by Debian which is a non-profit organization of volunteers who produce an open-source operating system distribution (Common Vulnerabilities and Exposure "CVE", 2023).

### log4j-api-2.12.1.jar

Graphical user interface, text, application, email, Teams

Description automatically generated

This vulnerability would allow an intruder to inject malicious code using a JNDI Lookup pattern to extract data from the log4j config file (Common Vulnerabilities and Exposure "CVE", 2023).

### logback-core-1.2.3.har

Graphical user interface, application, Teams

Description automatically generated

This vulnerability would allow anyone with the appropriate roles to inject malicious code from LDAP servers. LDAP servers are used to store user credentials (Common Vulnerabilities and Exposure "CVE", 2023).

### snakeyaml-1.25.jar

Graphical user interface, text, application, Teams

Description automatically generated

This vulnerability would allow an attacker to provide input that could cause the parser to become overwhelmed and crash resulting in a DoS (“Denial of Service”) attack (Common Vulnerabilities and Exposure "CVE", 2023).

### spring-boot-2.2.4.RELEASE.jar

Graphical user interface, text, application, email

Description automatically generated

### spring-boot-starter-web-2.2.4.RELEASE.jar

Graphical user interface, text, application, email

Description automatically generated

These vulnerabilities impact a method within a version of spring-boot < v2.2.11. The method can create a directory for web servers that would contain config files, class files, etc. Should an attacker be able to write in this directory, they could elevate their privilege and take full control of the application (Common Vulnerabilities and Exposure "CVE", 2023).

### spring-core-5.2.3.RELEASE.jar

Graphical user interface, text, application, email

Description automatically generated

### spring-web-5.2.3.RELEASE.jar

Graphical user interface, text, application, email

Description automatically generated

### spring-webmv-5.2.3.RELEASE.jar

Graphical user interface, text, application, email

Description automatically generated

These vulnerabilities are dependent on a particular situation where the application is run from Tomcat (Java web server) as a WAR deployment. A WAR file is basically a collection of jar files that contain everything that is related to a web application (i.e. – html files, Java classes, etc.). This type of exploit can result in a RCE (Remote Code Execution) (Common Vulnerabilities and Exposure "CVE", 2023).

### tomcat-embed-core-9.0.30.jar

Graphical user interface, text, application, email, Teams

Description automatically generated

This vulnerability could result in a DoS because HTTP/2 requests weren’t being handled by the web server. HTTP/1 was not replaced with HTTP/2 during the upgrade. This exploit could be used to cause unnecessary consumption of resources (Common Vulnerabilities and Exposure "CVE", 2023).

### tomcat-embed-websocket-9.0.30.jar

Graphical user interface, application

Description automatically generated

This exploit can result in a XSS (Cross Site Scripting) vulnerability when user input is not filtered when it should be on a site. Interaction with a vulnerable web application may prompt an attacker to respond with malicious code to the client’s browser (Common Vulnerabilities and Exposure "CVE", 2023).

## Mitigation Plan

Many of these vulnerabilities can be fixed with an update to the latest release. However, when this option is not available, often the vulnerability is left untouched. Depending on the vulnerability, not every one of them can be fixed. This is likely due to budget constraints and the complexity of the application. It would be prudent to run a severity study to determine the impact a successful exploit will have on the company before fixing any vulnerabilities.

# References

(2023, January 15). Retrieved from Common Vulnerabilities and Exposure "CVE": https://cve.mitre.org/cve/search\_cve\_list.html