****

# 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** | **07/16/2023** | **Jerrod Salzer** |  |

## 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

**Jerrod Salzer**

## Interpreting Client Needs

Artemis Financial is a consulting company that develops individualized financial plans for their clients. These plans include retirement plans, insurance as well as savings plans. This means they are in possession of many people’s financial information and because of this access to private financial information security must be the company’s highest priority when dealing with customer data. Being a financial institution with access to personal financial information they are regulated by the government on all communications and transactions to ensure the utmost security for the client as well as the company itself seeing as vulnerabilities don’t just affect the client but the company as well, being that they (Artemis Financial) are liable for any data breaches to client information while in possession of financial data. Unauthorized users accessing client information is of concern because of the financial ramifications for both client and company as well as possible phishing attempts to gather client information could be a potential threat. Most if not all vulnerabilities to this company involve unwanted access to user information so a secure API is important to discourage information leaks, as well as communicating via HTTPS when using customer or company data, also a two-factor authentication system could provide the necessary security against false login attempts.

## Areas of Security

API- Let’s think of the API as a meditator as well as a bouncer, we call it a mediator because the API works as a mediator or negotiator between any number of applications. We call it a bouncer because the API is used to authenticate users into a system, any user not on the list isn’t allowed in. The API is responsible for the safety and security of everything in the system from any attacks either internal or external.

Cryptography- Cryptography is used to ensure that any information that moves through a system is secure and can’t be accessed by external attacks or unauthorized users.

Client Server- A temporary storage of a system’s data to be used when information is sent to another server.

Code Error – Correct and precise code are important to all applications since it is the foundation of the application and even the smallest error in code can lead to bugs that could possibly allow for attacks to customer or company data.

## Manual Review

When reviewing the code, we see that request are not always being validated which can cause external attack vulnerabilities, we also can see that some information is being sent to the controller as a request which can cause vulnerabilities to client and company information. HTTPS is not being used and will need to be implemented since we are dealing with very important financial information for both parties.

## Static Testing

bcprov-jdk15on-1.46.jar- The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7

Update to latest version:

CVE-2016-1000338

CVE-2016-1000342

CVE-2016-1000343

CVE-2016-1000344

CVE-2016-1000352

CVE-2017-13098

CVE-2022-15522

CVE-2018-1000613

CVE-2018-5382

Log4j-api-2.12.1.jar - 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. Fixed in Apache Log4j 2.12.3 and 2.13.1

CWE-295 Improper Certificate Validation

CVE-2020-9488

Snakeyalm-1.25.jar - 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.

Update to latest version:

CVE-2022-1471

CVE-2017-18640

CVE-2022-25857

CVE-2022-38749

CVE-2022-38751

CVE-2022-38752

CVE-2022-41854

CVE-2022-39750

Jackson-databind-2.10.2.jar -A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.

Update to latest version:

CVE-2020-25649

CVE-2022-36581

CVE-2021-46877

CVE-2022-42003

CVE-2022-42004

CVE-2023-35116

Tomcat-embed-core-9.0.30.jar -When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.

Update to latest version:

CVE-2019-17569

CVE-2020-11996

CVE-2020-13934

CVE-2020-13935

CVE-2020-13943

CVE-2020-17527

CVE-2020-1935

CVE-2020-1938

CVE-2020-8022

CVE-2020-9484

CVE-2021-25122

CVE-2022-41079

CVE-2022-29885

CVE-2022-42252

CVE-2021-25329

CVE-2021-30640

CVE-2022-34305

CVE-2021-24122

CVE-2021-33037

CVE-2023-28708

CVE-2021-43980

## Mitigation Plan

To ensure both client and company security and safety we must implement the following:

Move to HTTPS for all future communications to discourage any external threats.

Implement a two-factor authentication system to ensure security of client/company data and fight against false login attempts.

Move request to headers.

Update all dependencies from our dependency check report.