

# 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 | 11/12/2023 | Matt Tranchina |  |

## 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:

Matt Tranchina

## Interpreting Client Needs

Artemis Financial is a consulting company that develops individualized financial plans. These plans include savings, retirement, investments, and insurance. Considering that important personal information, such as social security numbers, are needed for such investments, security is of top priority. There is no indication that these transactions between client and server are strictly national, so it is safe to assume that these transactions could be international. On the same note, if international transactions are to occur then governmental restrictions regarding trade information need to be considered. For example, if information were to be intercepted, insider trading could occur. With such sensitive information being transferred, external threats to intercept such information are great. Depending on the client, the threat could rise. One double edge sword that could be used is open-source libraries. This could provide security against malicious behavior, but the downside is that these libraries are available to hackers, and vulnerabilities could be found and exploited. With evolving web application technologies, staying up to date with software algorithms is crucial to security. Having a team member dedicated to ensuring updates and bug fixes should be considered.

## Areas of Security

* Input Validation – In order to access personalized accounts and profiles, input validation is needed. This will allow access to sensitive information, once information provided by the user is submitted correctly. Two step verification or biometric authorization could be used.
* APIs – An API would be needed for the user and the server to communicate. The API is what would transfer the data between the two, thus a secure API is a necessity.
* Cryptography – Another necessity is Cryptography. This would ensure that the information being passed is encrypted and secure. Even if a breach were to happen, the information would be safe. Considering the data in question would be financial information, cryptography is a must.

* Code Error – A detailed list of errors within the API would help developers indicate where vulnerabilities lie, or at the very least bugs that need to be fixed.
* Code Quality – Following proper procedures of good code quality would control the level of information provided to each user. Following a principle of privilege would only allow the user to perform actions allowed by the user, and not an administrator for example.

## Manual Review

After reviewing the code, I have noticed that there is an absence of input validation. This could invite multiple vulnerabilities and should be addressed. There is also no indication of handling errors, which could help identify further issues. After running a dependency check, the API had multiple vulnerabilities, including improper input validation, exposure of sensitive information, HTTP request/response smuggling, and unprotected transport of credentials to name a few.

## Static Testing

The following is a list showing vulnerable dependencies, listing from greatest threat to lowest. This test was to exclude false positives.

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency | Vulnerability IDs | Description | Solution |
| tomcat-embed-websocket-9.0.30.jar  Highest Severity: CRITICAL\*  CVE Count: 25 | cpe:2.3:a:apache:tomcat:9.0.30:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:apache\_tomcat:  apache\_tomcat:9.0.30:  \*:\*:\*:\*:\*:\*:\* | Apache Tomcat treats Apache JServ Protocol (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. | Upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. Several changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. |
| tomcat-embed-core-9.0.30.jar  Highest Severity: CRITICAL\*  CVE Count: 24 | cpe:2.3:a:apache  :tomcat:9.0.30:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:apache  \_tomcat:apache\_tomcat:9.0.30:  \*:\*:\*:\*:\*:\*:\* | 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. | Upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. Several changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. |
| Spring-webmvc-5.2.3.RELEASE.jar  Highest Severity: CRITICAL\*  CVE Count: 11 | cpe:2.3:a:pivotal\_software:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\*  cpe:2.3:a:web\_project:  web:5.2.3:release:  \*:\*:\*:\*:\*:\* | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. A WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data. | Apply updates per vendor instructions. |

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency | Vulnerability IDs | Description | Solution |
| spring-web-5.2.3.RELEASE.jar  Highest Severity: CRITICAL\*  CVE Count: 12 | cpe:2.3:a:pivotal\_software:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\* cpe:2.3:a:springsource:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\* cpe:2.3:a:vmware:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\* cpe:2.3:a:web\_project:  web:5.2.3:release:\*:\*:\*:\*:\*:\* | Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data. Depending on how the library is implemented within a product, this issue may or not occur, and authentication may be required. | Apply updates per vendor instructions. |
| spring-core-5.2.3.RELEASE.jar  Highest Severity: CRITICAL\*  CVE Count: 11 | cpe:2.3:a:pivotal\_software:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:  spring\_framework:5.2.3:release:  \*:\*:\*:\*:\*:\* | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition. | Apply updates per vendor instructions. |
| spring-boot-starter-web-2.2.4.RELEASE.jar  Highest Severity: CRITICAL  CVE Count: 3 | cpe:2.3:a:vmware:  spring\_boot:2.2.4:release:  \*:\*:\*:\*:\*:\* cpe:2.3:a:web\_project:  web:2.2.4:release:  \*:\*:\*:\*:\*:\* | In Spring Boot versions 3.0.0 - 3.0.5, 2.7.0 - 2.7.10, and older unsupported versions, an application that is deployed to Cloud Foundry could be susceptible to a security bypass, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache. | 3.0.x users should upgrade to 3.0.6+.  2.7.x users should upgrade to 2.7.11+.  Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+. |
| spring-boot-2.2.4.RELEASE.jar  Highest Severity: CRITICAL  CVE Count: 3 | cpe:2.3:a:vmware:  spring\_boot:2.2.4:release:  \*:\*:\*:\*:\*:\* | Spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache. | 3.0.x users should upgrade to 3.0.6+.  2.7.x users should upgrade to 2.7.11+.  Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+. |
| Dependency | Vulnerability IDs | Description | Solution |
| snakeyaml-1.25.jar  Highest Severity: CRITICAL  CVE Count: 1 | cpe:2.3:a:snakeyaml\_project:  snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\* | 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, resulting in deserialization of untrusted data. | Use SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization.  Upgrade to version 2.0 and beyond. |
| logback-core-1.2.3.jar  Highest Severity: MEDIUM  CVE Count: 1 | cpe:2.3:a:qos:  logback:1.2.3:\*:\*:\*:\*:\*:\*:\* | 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 server, resulting in deserialization of untrusted data. | Update to current version. |
| log4j-api-2.12.1.jar  Highest Severity: LOW  CVE Count: 1 | cpe:2.3:a:apache:  log4j:2.12.1:\*:\*:\*:\*:\*:\*:\* | 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. | Upgrade to Apache Log4j 2.12.3 or 2.13.1 |
| jackson-databind-2.10.2.jar  Highest Severity: HIGH  CVE Count: 6 | cpe:2.3:a:fasterxml:  jacksondatabind:2.10.2:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:fasterxml:  jackson-modulesjava8:2.10.2:  \*:\*:\*:\*:\*:\*:\* | 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 current version. |

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency | Vulnerability IDs | Description | Solution |
| hibernate-validator-6.0.18.Final.jar  Highest Severity: MEDIUM  CVE Count: 1 | cpe:2.3:a:redhat:  hibernate\_validator:6.0.18:  \*:\*:\*:\*:\*:\*:\* | A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages. | Update to current version. |
| bcprov-jdk15on-1.46.jar  Highest Severity: HIGH  CVE Count: 18 | cpe:2.3:a:bouncycastle:  bouncycastle-cryptopackage:1.46:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:  bouncy\_castle\_crypto\_package:1.46:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:  legion-of-the-bouncy-castle-  java-crytography-api:1.46:  \*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:  the\_bouncy\_castle\_  crypto\_package\_for\_java:1.46:  \*:\*:\*:\*:\*:\*:\* | In Bouncy Castle JCE Provider version 1.55 and earlier the DSA does not fully validate ASN.1 encoding of signature on verification. It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' data into a signed structure. | Update to current version. |

## Mitigation Plan

The first step in the mitigation plan is to update all dependencies to current versions, or versions suggested. The next step would be to implement an input validation method, along with an authorization method. This would help ensure that the information requested is correct and if the information is accessible to the requested user.