****

# 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** | **[3/19/23]** | **[Ashley De Venuto]** |  |

## 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

Ashley De Venuto

## Interpreting Client Needs

The company that this software will help is Artemis Financial. Artemis Financial helps their customers by creating unique financial plans for them. These plans include savings, retirement, investments, and insurance. It is important to have communication with their clients. Since the company is handling a lot of person data on their clients so they want to make sure that software security is at its best. There is a big chance that this company would need to deal with international transaction. The company would have to be careful not to leak confidential information during secure communications. One of the main threats would be the leak of personal client information. This information would need to be encrypted to make it harder for people outside the company to be able to steal it. Some modernization requirements that should be considered by Artemis Financial would be up-to-date maintenance checks for bug fixes and weakened security threads.

## Areas of Security

Here are the areas of security that should be considered for the software security:

Input Validation – This is to make sure that the only person to access information is the owner of that information. This will protect the users and their information.

Code Quality – This will ensure what information is available for the users to see.

APIs – APIs help run the server. It is running internally and externally. It also determines what data is accessible.

Code Error – Checking for errors is needed to make sure that area is fixed. This will help protect client/user information.

Cryptography – This will ensure that private information would be protected from outside entities.

## Manual Review

After looking through the code I noticed that the pom.xml file does not have a Apache Validator. This was also notice in the dependency check because that was one of the vulnerabilities. There was no input validation that I saw. This was also reflected in the dependency check. There was no error check in the application within the RestServiceApplicationTests.java. I was unable to find anything verifying cryptography. I could not find anything about user input within the API. There is code in the application customer.java where you would be able to see the information but nowhere to input user information.

## Static Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency | Vulnerability | Description | Solution |
| Bcprov-jdk15on-1.46.jar | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | | [CVE-2016-1000338](https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-1000338) | |  |  |  |  |  | |  |  |  |  |  |  | | 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. | Upgrade to version 1.61 or higher |
| Hibernate-validator-6.0.18.Final.jar | CVE-2020-10693 | A flaw was found in Hibernate Validator version 6.1.2.Final. 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. | Upgrade to the newest version 8.0.0. |
| Jackson-databind-2.10.2.jar | CVE-2020-25649 | 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. | Upgrade to version 2.6.7.4, 2.9.10.7, 2.10.5.1 or higher. |
| Log4j-api-2.12.1.jar | **CVE-2020-9488** | 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 | Upgrade to version 2.13.2 or higher. |
| Logback-core-1.2.3.jar | 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. | Upgrade version |
| Snakeyaml-1.25..jar | 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. | Upgrade version to 2.0 or higher |
| Spring-boot-2.2.4.RELEASE.jar | CVE-2022-27772 | \*\* UNSUPPORTED WHEN ASSIGNED \*\* spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer. | Upgrade to the latest version 3.0.3 |
| Spring-boot-starter-web-2.2.4.RELEASE.jar | CVE-2022-27772 | \*\* UNSUPPORTED WHEN ASSIGNED \*\* spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer. | Upgrade to the latest version 3.0.3 |
| Spring-core-5.2.3.RELEASE.jar | CVE-2022-22965 | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it. | Upgrade to latest version 6.0.6 |
| Spring-web-5.2.3.RELEASE.jar | CVE-2016-100027 | 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. NOTE: the vendor's position is that untrusted data is not an intended use case. The product's behavior will not be changed because some users rely on deserialization of trusted data. | Upgrade to latest version 6.0.6 |
| Spring-webmvc-5.2.3.RELEASE.jar | CVE-2022-22965 | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it. | Upgrade to latest version 6.0.6 |
| Tomcat-embed-core-9.0.30.jar | CVE-2020-1938 | 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. | 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. |
| Tomcat-embed-websocket-9.0.30.jar | CVE-2020-1938 | 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. | 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. |

## Mitigation Plan

The best way to remedy the vulnerabilities that were found during the dependency check is to run the current versions of those vulnerabilities.