****

# 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** | **01/11/2023** | **David J Allen** | **Initial vulnerability report** |

## 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

David J Allen

## Interpreting Client Needs

Working as a developer for the software engineering company, Global Rain, the custom design and development is for entrepreneurs, businesses, and government agencies around the world. The client, Artemis Financial, is a consulting company that develops individualized financial plans for their customers including savings, retirement, investments, and insurance. Artemis Financial works with both internal and external clients and their information. Being a financial firm there is bound to be transactions both domestic and foreign. For regulation purposes Artemis Financial will need to ensure their communications external and internal are secure to prevent leaking information. For the financial transactions, data encryption is highly necessary for storing and transmitting. For modernization purposes, Artemis Financial will need to stay up to date on their code library and necessary updates to patch vulnerabilities.

## Areas of Security

For Artemis Financial areas of security, the following are areas of security:

* APIs:
  + APIs allow transmission of data and functionality to third-party developers and business partners. This allows applications and systems to communicate with one another and utilize one another's data and functionality.
* Code Error:
  + Error handling is important to prevent unauthorized access and validate inputs.
* Code Quality:
  + Quality is necessary with liabilities involved in the data and its encryption of clients proprietary personal information.
* Cryptography
  + Cryptography should be used because of the transfer of customer information both domestically and internationally. The data needs to be secure to comply with all regulations.
* Input validation:
  + All the information obtained is client driven. Input validation prevents malicious or poorly qualified data from entering an information system.

## Manual Review

Manual code review involves the reading of source code line by line to look out for security vulnerabilities. While reviewing the code it did appear that the dependency check was outdated, so that correction was necessary to ensure every vulnerability was accounted for. The code base did not appear to use any input validation. Further digging in for other areas of security, no working API was revealed within the coding. The data was also lacking in encryption methods which are mandatory for an application accessing so much proprietary information. The data is accessed via URL but utilizing POST method would be more secure to prevent possible exploitation. While the data did possess high quality it was riddled with a lack of pertinent features in the area of security to make it a full faceted secure banking application for Artemis Financial.

## Static Testing

bcprov-jdk15on-1.46.jar

Description: 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. 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.

License: Bouncy Castle Licence: http://www.bouncycastle.org/licence.html

hibernate-validator-6.0.18.Final.jar

Description: Hibernate's Bean Validation (JSR-380) reference implementation. 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.

License: http://www.apache.org/licenses/LICENSE-2.0.txt

jackson-databind-2.10.2.jar

Description: General data-binding functionality for Jackson: works on core streaming API. 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.

License: http://www.apache.org/licenses/LICENSE-2.0.txt

log4j-api-2.12.1.jar

Description: The Apache Log4j API. 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.

License: https://www.apache.org/licenses/LICENSE-2.0.txt

logback-core-1.2.3.jar

Description: logback-core module. 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.

License: http://www.eclipse.org/legal/epl-v10.html, http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html

snakeyaml-1.25.jar

Description: YAML 1.1 parser and emitter for Java. 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.

License: Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

spring-boot-2.2.4.RELEASE.jar

Description: Spring Boot. \*\* 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.

License: Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

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

Description: Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container. \*\* 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.

License: Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

spring-core-5.2.3.RELEASE.jar

Description: Spring Core. 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.

License: Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

spring-web-5.2.3.RELEASE.jar

Description: Spring Web. 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.

License: Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

spring-webmvc-5.2.3.RELEASE.jar

Description: Spring Web MVC. 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.

License: Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

tomcat-embed-core-9.0.30.jar

Description: Core Tomcat implementation. 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.

License: Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

tomcat-embed-websocket-9.0.30.jar

Description: Core Tomcat implementation. 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.

License: Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

## Mitigation Plan

Digging through the security exploits and filtering out the false positives presents a list of vulnerabilities that can be remedied by updating the software as many patches have been released since the included frameworks. To mitigate the risks, it would be necessary to update all the software aspects of the code base and verify success through a manual code review which would take a significant amount of attention to detail.

1. **References**

*CVE - Search CVE List*. (n.d.). https://cve.mitre.org/cve/search\_cve\_list.html

Manico, J. (n.d.). *Iron-Clad Java*. O’Reilly Online Learning. https://www.oreilly.com/library/view/iron-clad-java/9780071835886/?sso\_link=yes

*NVD - Search and Statistics*. (n.d.). https://nvd.nist.gov/vuln/search

*Vulnerability Assessment Process Flow*. (2012). SNHU. Retrieved January 11, 2023, from https://learn.snhu.edu/content/enforced/1237126-CS-305-J7341-OL-TRAD-UG.23EW3/course\_documents/CS%20305%20Vulnerability%20Assessment%20Process%20Flow%20Diagram.pdf?\_&d2lSessionVal=m74rHk38W8rbtHNWLqrf8f9Qu&ou=1237126