

# 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** | **May 21, 2023** | **Joshua Wozny** |  |

## Client



## Developer

Joshua Wozny, CS-305, Southern New Hampshire University

## Interpreting Client Needs

“Security is everyone’s responsibility.”

As a company with with locations, clients, and employees across the globe, it is vital that Artemis Financial has secure communications across their organization. As the company grows, international financial transactions will continue to become a larger share of the company’s total transaction. Securing software solutions is a vital step to ensure that Artemis Financial’s sensitive and regulated data is safe from malicious actors.

Having a secure infrastructure is required by government regulation, particularly the General Data Protection Regulation (GDPR) in the EU and California’s Consumer Privacy Act, Consumer Privacy Rights Act, Virginia Consumer Data Protection Act, and Colorado Privacy Act in the US. As a multi-national company, they must comply with all of these. Even in the absence of regulations, secure practices and best of class solutions are important to implement to maintain consumer confidence (CCPA vs GDPR. What, n.d.).

Upgrading to a modern technology stack for application development like Springboot, will aid in the facilitation of securing Artemis Financial’s full-stack applications (The New Stack, n.d.). It is important that you do not rely solely Springboot’s defaults but implement best practices – however, Spring provides a great base to start from.

Utilizing open source libraries in the application can provide easy to implement solutions to business problems. It is critical that documentation be reviewed and known vulnerabilities considered when implementing to ensure that these vulnerabilities are properly mitigated. This is especially true as the web application evolves and additional security vulnerabilities are identified. Keeping abreast of industry-wide security concerns and frequent vulnerability assessments of the application are necessary to the maintenance of the application.

## Areas of Security

As a client-server, with an API centered communication, Artemis Financial’s full-stack application has the following major areas of security concerns:

* Input Validation
* API
* Cryptography
* Client/Server
* Code Quality

## Manual Review

Reviewing the current code base, I have identified the following security concerns.

* Classes are not marked final. All classes should be marked final to prevent malicious subclassing from overriding random methods and adding finalizers.
* Input Validation of string should occur ti limit the length of strings entering methods within the application.
  + Greeting.java: content parameter is not currently validated
  + GreetingController.java: the /greeting API endpoint is not validated
* Code quality and Client/Server security could be better secured. The connection should be closed in the DocData.java class. It is bad practice to leave connections open and can create performance problems as well.

## Static Testing

The following dependencies with known vulnerabilities were identified:

* bcprov-jdk15on-1.46.jar: The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms.
* hibernate-validator-6.0.18.Final.jar: Hibernate's Bean Validation (JSR-380) reference implementation.
* jackson-databind-2.10.2.jar: General data-binding functionality for Jackson: works on core streaming AP
* log4j-api-2.12.1.jar: The Apache Log4j API.
* logback-core-1.2.3.jar: logback-core module
* snakeyaml-1.25.jar: YAML 1.1 parser and emitter for Java
* spring-boot-2.2.4.RELEASE.jar: Spring Boot
* spring-boot-starter-web-2.2.4.RELEASE.jar: Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container
* spring-core-5.2.3.RELEASE.jar: Spring Core
* spring-web-5.2.3.RELEASE.jar: Spring Web
* spring-webmvc-5.2.3.RELEASE.jar: Spring Web MVC
* tomcat-embed-core-9.0.30.jar: Core Tomcat implementation
* tomcat-embed-websocket-9.0.30.jar: Core Tomcat implementation

## Mitigation Plan

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Vulnerability** | | |
| **Dependency** | **Code** | **Description** | **Mitigation** |
| Bcprov-jdk15on-1.46 | 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 1.73 |
|  | CVE-2016-1000344 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider. |  |
|  | CVE-2016-1000341 | In the Bouncy Castle JCE Provider version 1.55 and earlier DSA signature generation is vulnerable to timing attack. Where timings can be closely observed for the generation of signatures, the lack of blinding in 1.55, or earlier, may allow an attacker to gain information about the signature's k value and ultimately the private value as well. |  |
|  | CVE-2020-15522 | Bouncy Castle BC Java before 1.66, BC C# .NET before 1.8.7, BC-FJA before 1.0.1.2, 1.0.2.1, and BC-FNA before 1.0.1.1 have a timing issue within the EC math library that can expose information about the private key when an attacker is able to observe timing information for the generation of multiple deterministic ECDSA signatures. |  |
| Hibernate-validator-6.0.18 | 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. | You can pass user input as an expression variable by unwrapping the context to HibernateConstraintValidatorContext. |
| Jackson-databind-2.10.2 | 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. | Avoid the use of the DOMDeserializer class.  Limit the amount of resources that an unauthorized user can cause to be expended |
|  | CVE-2020-36518 | jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects. |  |
|  | CVE-2021-46877 | (OSSINDEX) The software does not properly restrict the size or amount of resources that are requested or influenced by an actor, which can be used to consume more resources than intended. |  |
| Log4j-api-2.12.1 | CVE-2020-9488 | 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 to version 2.12.3 or 2.13.1 |
| Logback-core-1.2.3 | CVE-2021-42550 | 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. | None Required |
| Snakeyaml-1.25 | 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 to version 2.0 or higher. |
|  | CVE-2017-18640 | The Alias feature in SnakeYAML before 1.26 allows entity expansion during a load operation |  |
|  | CVE-2022-38749 | Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow. |  |
| Spring-boot-2.2.4  Spring-webmvc-5.2.3  Spring-core-5.2.3  Spring-boot-starter-web-2.2.4 | 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. | None Required |
| Spring-core-5.2.3 | CVE-2022-22965 | 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 version to to 5.2.20+ |
|  | CVE-2021-22118 | In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, 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. |  |
| Spring-web-5.2.3 | CVE-2016-1000027 | 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. | Do not expose HTTP Invoker endpoints to untrusted clients |
| Tomcat-embed-core-9.0.30  Tomcat-embed-websocket-9.0.30. | 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 | upgrade to Apache Tomcat 9.0.74 |
|  | CVE-2020-11996 | A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive. |  |
|  | CVE-2020-13935 | The payload length in a WebSocket frame was not correctly validated in Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M1 to 9.0.36, 8.5.0 to 8.5.56 and 7.0.27 to 7.0.104. Invalid payload lengths could trigger an infinite loop. Multiple requests with invalid payload lengths could lead to a denial of service. | upgrade to Apache Tomcat 9.0.74 |

**Citations**

CCPA vs GDPR. What. (n.d.). Cookieyes. <https://cookieyes.com/blog/ccpa-vs-gdpr/>.

The New Stack. (n.d.). Thenewstack. https://thenewstack.io/use-of-spring-shifts-toward-modern-apps-and-architecture//blog/ccpa-vs-gdpr/.