GLOBALRAIN

Artemis Financial Vulnerability Assessment Report

Marisa Kuyava

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	01.20.23	Marisa Kuyava	Recommendations

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.

1. Interpreting Client Needs

- What is the value of secure communications to the company?

  Secure communications are extremely important for a company that handles sensitive financial information. To ensure data privacy, information security and to protect from external threats, all communications should be secure. The value of secure communication is holding customer's confidence that Artemis Financial will keep their data safe and secure.

- Does the company make any international transactions?

  There is no indication that Artemis Financial makes international transitions in the information provided by the brief, however as they are a web-based company this should be clarified.

- Are there governmental restrictions about secure communications to consider?

  Yes, because Artemis Financial is dealing with sensitive financial information and personal data it is extremely important to ensure government regulations regarding financial communications and transactions are implemented.

- What external threats might be present now and in the immediate future?

  Financial institutions will always face external threats because the data that they utilize would be extremely valuable to a data broker. Financial institutions must ensure there are security measures in place to protect against the many Malware/Ransomware, Phishing, Spoofing, and other types of threats that attackers may use to gain access to customer's data and financial information.

2. Areas of Security

- Input Validation
  - All input should be validated to help prevent injection attacks and to keep the program secure. Because there is a command input function input validation is important.
- APIs
  - RESTful API should be implemented to protect from system attacks.
- Cryptography
  - Proper cryptography should be used on web applications to ensure data is encrypted and protected, especially those that use APIs.
- Code Error
  - To ensure proper security and protect from attacks Secure error handling is important.
- Code Quality
  - Secure coding practices and patters should always be used.

3. Manual Review

Not having input or request parameter validation can lead to vulnerabilities in the system, the string can be manipulated and cause the database to be compromised.

Passing the business name as a request parameter in the CRUDController.java class creates access vulnerability to the DocData doc object.

```
    @RequestMapping("/read")
    public CRUD CRUD(@RequestParam(value="business_name") String name) {
        DocData doc = new DocData();

        return new CRUD(doc.toString());
    }

}
```

Having the connection parameters hard coded into DocData.java creates a vulnerability that could allow unauthorized users to gain access.

```
public void read_document(String key, String value)
{
    /* implement read method */
    //Class.forName("com.mysql.jdbc.Driver");
    try {
        Connection con=DriverManager.getConnection(
        "jdbc:mysql://localhost:3306/test","root","root");
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    //here test is database name, root is username and password
```

4. Static Testing

The dependency check for rest-service returned a total of 13 vulnerable dependencies. After removing duplicate entries, the final number of vulnerable dependencies was 10. These are listed below.

Dependency: bcprov-jdk15on-1.46.jar

- **Code**: CVE-2016-1000338
- **Description**: 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.

Dependency: hibernate-validator-6.0.18.Final.jar

- **Code**: CVE-2020-10693
- **Description**: 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.

Dependency: jackson-databind-2.10.2.jar

- **Code**: CVE-2020-25649
- **Description**: 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.

Dependency: log4j-api-2.12.1.jar

- **Code**: CVE-2020-9488
- **Description**: 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

Dependency: logback-core-1.2.3.jar

- **Code**: CVE-2021-42550
- **Description**: 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.

Dependency: snakeyaml-1.25.jar

- **Code**: CVE-2022-1471
- **Description**: 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.

Dependency: spring-boot-2.2.4.RELEASE.jar

- **Code**: CVE-2022-27772
- **Description**: ** 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.

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

- **Code**: CVE-2022-27772
- **Description**: ** 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.

Dependency: spring-core-5.2.3.RELEASE.jar

- **Code**: CVE-2022-22965
- **Description**: 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.

Dependency: tomcat-embed-core-9.0.30.jar

- **Code**: CVE-2020-1938
- **Description**: 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.

5. Mitigation Plan

Follow mitigation for each of the following dependencies to decrease security risks. Review and update cod following security coding practices to ensure proper error catching and input validation to help mitigate vulnerabilities.

Dependency: bcprov-jdk15on-1.46.jar

- **Code**: CVE-2016-1000338
- **Mitigation**: Update Bouncy Castle JCE Provider version. Issue is only for version 1.55 and earlier.
    - Refer to: https://nvd.nist.gov/vuln/detail/CVE-2016-1000338

Dependency: hibernate-validator-6.0.18.Final.jar

- **Code**: CVE-2020-10693
- **Mitigation**: Pass user input as an expression variable by unwrapping the context to HibernateConstraintValidatorContext
    - Refer to: https://in.relation.to/2020/05/07/hibernate-validator-615-6020-released/
    - Refer to: https://docs.jboss.org/hibernate/stable/validator/reference/en-US/html_single/#_the_code_constraintvalidatorcontext_code.

Dependency: jackson-databind-2.10.2.jar

- **Code**: CVE-2020-25649
- **Mitigation**: There is currently no known mitigation for this flaw
    - Refer to: https://access.redhat.com/security/cve/cve-2020-25649

Dependency: log4j-api-2.12.1.jar

- **Code**: CVE-2020-9488
- **Mitigation**: Update to version 2.13.2 which supports this feature
    - Refer to: https://issues.apache.org/jira/browse/LOG4J2-2819

Dependency: logback-core-1.2.3.jar

- **Code**: CVE-2021-42550
- **Mitigation**: Update to version 1.2.9.
    - Refer to: https://jira.qos.ch/browse/LOGBACK-1591

Dependency: snakeyaml-1.25.jar

- **Code**: CVE-2022-1471
- **Mitigation**: Undergoing reanalysis. Recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization.
    - Refer to: https://nvd.nist.gov/vuln/detail/CVE-2022-1471

Dependency: spring-boot-2.2.4.RELEASE.jar

- **Code**: CVE-2022-27772

- **Mitigation**: Update to version 2.2.11 - NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.
    - Refer to: https://github.com/JLLeitschuh/security-research/security/advisories/GHSA-cm59-pr5q-cw85

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

- **Code**: CVE-2022-27772
- **Mitigation**: Update to version 2.2.11 - NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.
    - Refer to: https://github.com/JLLeitschuh/security-research/security/advisories/GHSA-cm59-pr5q-cw85

Dependency: spring-core-5.2.3.RELEASE.jar

- **Code**: CVE-2022-22965
- **Mitigation**: Users of affected versions should apply the following mitigation: 5.3.x users should upgrade to 5.3.18+, 5.2.x users should upgrade to 5.2.20+. No other steps are necessary.
    - Refer to: https://tanzu.vmware.com/security/cve-2022-22965

Dependency: tomcat-embed-core-9.0.30.jar

- **Code**: CVE-2020-1938
- **Mitigation**: Update Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users.
    - Refer to: https://nvd.nist.gov/vuln/detail/CVE-2020-1938