GLOBALRAIN

CS 305 Project One
Artemis Financial Vulnerability Assessment Report
Kennedy Uzoho

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	9/27/2022	Kennedy Uzoho	Security assessment

Client

ARTEMIS
FINANCIAL

Developer
Kennedy Uzoho

1. Interpreting Client Needs

- **What is the value of secure communications to the company?**
  As a financial institution, the security of all transactions and communications is important. More importantly, the **Artemis Financial** RESTful web API. The web API needs to be reviewed and analyzed. Then run a full dependency security check to identify all security vulnerabilities that may exist in the company's code base. Existing and future company code releases should be patched securely. All communications, including the software objects and tunnels for example, username, password, and account IDs must be secure since the client participates in individualized financial services.

- **Does the company make any international transactions?**
  Based on the scenario provided, the company that I work for (Global Rain), is a company that specializes in software development and design for clients in several types of organizations around the world. Based on Artemis Financial company's structure, it will be helpful to have a cyber security analyst or DevSecOps engineer who will assist in supporting its online and international transaction tunnels.

- **Are there governmental restrictions on secure communications to consider?**
  Based on the scenario, there are no government restrictions to consider. If there is one, it would be as important to discuss them. We can know this information by going to the CISA portal for verification.

- **What external threats might be present now and in the immediate future?**
  As time progresses there may be a chance for external threats, for example, database errors and process hacking threats. Some of these threats can cause system function failures or data access blockage and these threats can be avoided by hardening the network protection layer and patching codes correctly and as necessary; however, in some cases like spyware or malware, it may be impossible to avoid some of the threats completely.

- **What are the "modernization" requirements that must be considered, such as the role of open-source libraries and evolving web application technologies?**
  Technology evolvement and innovation have grown over time. Today there are so many resources available to utilize when creating applications and other software features, in addition, you can manage the whole SDLC with CI/CD tools available in the market. There are so many open-source libraries, databanks, data centers, different various APIs, and their pipelines for automation protocols.

2. Areas of Security
The areas of security that apply to Artemis's software application include:
Code Quality
Code Errors

APIs, and Input Validation.

Code Quality refers to the designation of a well-simplified and structured code that can be easily monitored to prevent outside modifications/intrusion by unethical hackers. Code Errors should all be addressed within the code base before deploying, for example, defining parameters and conditions outside the code function in the case of an unspecified input. APIs and their communication tunnels are required to be securely implemented into the web applications, there must be a specified tunnel for communication within the software, if not other servers/virtual systems with OS can also come in. Input validation is necessary because REST APIs take user input, and that input value needs to be cleared, validated, and channeled to the appropriate databank securely.

3. Manual Review

After a manual code review, I noticed that the service is not using TLS (SSL) to encrypt HTTP:// to avoid vulnerabilities.

There should be an authentication system in place

All requests should be validated, so the system can be secure from outside intrusion.

I noticed that the business is sent as a request parameter in the CRUDConttoller class, this is not a secure practice, and it provides information to an outside source.

4. Static Testing

A dependency check was run on Artemis Financials' software applications to identify all security vulnerabilities in the code base. Here is a recorded output from the dependency check report. Including the following:

a. The names or vulnerability codes of the known vulnerabilities
b. A brief description and recommended solutions provided by the dependency check report
c. Attribution (if any) that documents how this vulnerability has been identified or documented previously.

Dependencies Checked.

classmate-1.5.1.jar	Library for introspecting types with full generic information including resolving of field and method types Plan. Updating the dependency to the most recent version	**Identifier:** pkg:maven/com.fasterxml/classmate@1.5.1. No codes/IDs found
hibernate-validator-6.0.18.Final.jar	Hibernate's Bean Validation (JSR-380) reference implementation. Plan. Use a fix patch for each of the named items	**CVE-2020-10693**
jackson-core-2.10.2.jar	Core Jackson processing abstractions (aka Streaming API), implementation for JSON	**No codes found:** **Identifier:** pkg:maven/com.fasterxml.jackson.core/j

	Plan. Updating the dependency to the most recent version	ackson-core@2.10.2
Jackson-databind-2.10.2.jar	General data-binding functionality for Jackson: works on core streaming API Plan. Updating the dependency to the most recent version	**CVE-2020-25649** **CVE-2020-36518**
jakarta.annotation-api-1.3.5.jar	Jakarta Annotations API Plan. Updating the dependency to the most recent version	Identifier: pkg:maven/jakarta.annotation/jakarta.annotation-api@1.3.5
jakarta.validation-api-2.0.2.jar	Jakarta Bean Validation API Plan. Updating the dependency to the most recent version	Identifier: pkg:maven/jakarta.validation/jakarta.validation-api@2.0.2
jboss-logging-3.4.1.Final.jar	The JBoss Logging Framework Plan. Updating the dependency to the most recent version	Idenitfier: pkg:maven/org.jboss.logging/jboss-logging@3.4.1.Final
jul-to-slf4j-1.7.30.jar	JUL to SLF4J bridge Plan. Updating the dependency to the most recent version and checking for update on a regular basis can help with these issues.	Identifier: pkg:maven/org.slf4j/jul-to-slf4j@1.7.30 (*Confidence*:High)
log4j-api-2.12.1.jar	The Apache Log4j API	**CVE-2020-9488**
log4j-to-slf4j-2.12.1.jar	The Apache Log4j binding between Log4j 2 API and SLF4J.	Identifier: pkg:maven/org.apac

	Plan. Updating the dependency to the most recent version and checking for update on a regular basis can help with these issues and help to prevent man-in-the-middle type of attacks.	he.logging.log4j/log4j-to-slf4j@2.12.1 (*Confidence*:High)
logback-core-1.2.3.jar	logback-core module Updating the dependency to the most recent version and checking for update on a regular basis can help with these issues.	**CVE-2021-42550**
mongo-java-driver-2.4.jar	Java Driver for MongoDB	**CVE-2021-20328**
slf4j-api-1.7.30.jar	The slf4j API	**Identifier:** pkg:maven/org.slf4j/slf4j-api@1.7.30
snakeyaml-1.25.jar	YAML 1.1 parser and emitter for Java Plan. Some APIs can cause a false positive during dependency vulnerability check. If the source is internal, we can restrict the alternative names for collections.	**CVE-2017-18640** **CVE-2022-25857** **CVE-2022-38749** **CVE-2022-38751** **CVE-2022-38752** **CVE-2022-38750**
spring-boot-2.2.4.RELEASE.jar	Spring Boot Plan. Updating the dependency to the most recent version and checking for update on a regular basis can help with these issues.	**CVE-2022-27772**
spring-core-5.2.3.RELEASE.jar	Spring Core Plan. Update	**CVE-2022-22965** **CVE-2021-22118**

		CVE-2020-5421 CVE-2022-22950 CVE-2022-22971 CVE-2022-22968 CVE-2022-22970 CVE-2021-22060 CVE-2021-22096
spring-web-5.2.3.RELEASE.jar	Spring Web Plan. Update and analyze the API for security testing	CVE-2022-22965 CVE-2016-1000027 CVE-2021-22118 CVE-2020-5421 CVE-2022-22950 CVE-2022-22971 CVE-2022-22968 CVE-2022-22970 CVE-2021-22060 CVE-2021-22096
tomcat-embed-core-9.0.30.jar	Core Tomcat implementation Updating the Restful API can help alleviate some errors and vulnerabilities.	CVE-2020-1938 CVE-2020-11996 CVE-2020-13934 CVE-2020-13935 CVE-2020-17527 CVE-2021-25122 CVE-2021-41079

		CVE-2022-29885 CVE-2020-9484 CVE-2021-25329 CVE-2021-30640 CVE-2022-34305 CVE-2021-24122 CVE-2021-33037 CVE-2019-17569 CVE-2020-1935 CVE-2020-13943
tomcat-embed-el-9.0.30.jar	Core Tomcat implementation	CVE-2020-1938 CVE-2020-8022 CVE-2020-11996 CVE-2020-13934 CVE-2020-13935 CVE-2020-17527 CVE-2021-25122 CVE-2021-41079 CVE-2022-29885 CVE-2020-9484 CVE-2021-25329 CVE-2021-30640 CVE-2022-34305 CVE-2021-24122

		CVE-2021-33037 CVE-2019-17569 CVE-2020-1935 CVE-2020-13943

5. Mitigation Plan

To mitigate the Identified vulnerabilities, it will be necessary to ensure that all the company's client information is secured. The security must be to maintain confidentiality, integrity, and availability of information banks to support the business operation successfully. In IT some security issues can only be maintained, monitored, and tracked, it would be important to secure HTTP to HTTPS in all domains of cyber communications to prevent outside control. The request parameters should be moved to the header. Any client, business, or personal information within a hard-coded database should have a different reference name. Two-factor authentication and general system log-in authentication are recommended. All dependencies should be regularly updated as outdated software malfunctions sometimes.