GLOBALRAIN

CS 305 Project One
Artemis Financial Vulnerability Assessment Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	11/14/2021	Matt Zindler	Initial Vulnerability Assessment

Client

Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found. Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

Developer
Matt Zindler

1. Interpreting Client Needs
Determine your client's needs and potential threats and attacks associated with their application and software security requirements. Consider the following regarding how companies protect against external threats based on the scenario information:

- What is the value of secure communications to the company?
- Are there any international transactions that the company produces?
- Are there governmental restrictions about secure communications to consider?
- What external threats might be present now and in the immediate future?
- What are the "modernization" requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

Artemis Financial requires security to protect their investors and their assets from malicious attackers. They deal worldwide with financial planning, and so value security for their customers and their investments. Many governments will have requirements and standards on how money will be able to be accessed and traded within their countries. Some examples include taxes and proper documentation. Their regulations will certainly need to be addressed and followed for our product to be used in their countries. As for external threats, there will be many malicious users who wish to access private information for their own gain: they may want company records, access to bank accounts, or to shut down services. We have to ensure proper protection from these attacks by ensuring proper security and software updates. Modernization is important for security as software is ever evolving. Ensuring that we take use of the most up to date security software available. Using older or less tested software leaves exploitable holes in security.

2. Areas of Security
Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial's software application. Justify your reasoning for why each area is relevant to the software application.

The areas that affect the project the most are cryptography, and secure APIs. Cryptography is the most important aspect to get right for this application. Having any data that is being transferred, potentially worldwide, be encrypted is essential. Failing to secure data in this way leaves malicious users open to tracking and reading what data is being transferred. Secure APIs help in a similar way, by securing the interactions done by the program with all associated interfaces. Insufficient API security can lead to data breaches and exposed data after the information has reached different interfaces. Since we will be dealing with sensitive financial information, taking every possible precaution and security level is essential to give our customers the peace of mind and security they need.

3. Manual Review
Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

- In the doc data class, there is an error catch, where the username and password are the same and stored in plaintext (test, test).
- In the customer class, account_balance is not private, which can cause issues with the code (encapsulation).
- The spring framework maven plugin was not the most up to date version.

4. Static Testing

Run a dependency check on Artemis Financial's software application to identify all security vulnerabilities in the code. Record the output from the dependency check report. Include 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

1.cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.46:*:*:*:*:*:*:*
cpe:2.3:a:bouncycastle:bouncy_castle_crypto_package:1.46:*:*:*:*:*:*:*
cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:*:*:*:*:*:*:*
cpe:2.3:a:bouncycastle:the_bouncy_castle_crypto_package_for_java:1.46:*:*:*:*:*:*:*

Bouncy Castle appears to be using an out of date version, and has several flaws with more recent versions of Java and it's libraries. Mainly with timing attacks and some key issues. Recommended to use a more up to date version that fixes these vulnerabilities.

2. cpe:2.3:a:redhat:hibernate_validator:6.0.18:*:*:*:*:*:*:*

message interpolation can validate invalid inputs, bypassing input sanitation.

3.cpe:2.3:a:fasterxml:jackson-databind:2.10.2:*:*:*:*:*:*:*
cpe:2.3:a:fasterxml:jackson-modules-java8:2.10.2:*:*:*:*:*:*:*

Data integrity is at risk from improperly secured entity expansion.

4. cpe:2.3:a:apache:log4j:2.12.1:*:*:*:*:*:*:*\

Improper certification can be compromised by a min-in-the-middle attack, where they can view log messages.

5. cpe:2.3:a:snakeyaml_project:snakeyaml:1.25:*:*:*:*:*:*:*

The Alias feature can be used for entity expansion during a load operation

6. cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*
cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*
cpe:2.3:a:vmware:springsource_spring_framework:5.2.3:release:*:*:*:*:*:*

Older versions of Spring allow RFD attacks to bypass security depending on the browser used..
Recommend to ensure that spring is not running on the older unsupported versions.

7. cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:*
cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*
cpe:2.3:a:vmware:spring_framework:5.2.3:release:*:*:*:*:*:*:*
cpe:2.3:a:vmware:springsource_spring_framework:5.2.3:release:*:*:*:*:*:*:*

Older versions of Spring allow RFD attacks to bypass security depending on the browser used.. Recommend to ensure that spring is not running on the older unsupported versions.

8. cpe:2.3:a:apache:tomcat:9.0.30:*:*:*:*:*:*:*:*
cpe:2.3:a:apache_software_foundation:tomcat:9.0.30:*:*:*:*:*:*:*:*
cpe:2.3:a:apache_tomcat:apache_tomcat:9.0.30:*:*:*:*:*:*:*:*

A regression was possible in certain versions of Apache Tomcat that some headers could be processed resulting in HTTP Request Smuggling. Advised to not use these versions, using a more recent or supported version.

9. cpe:2.3:a:apache:tomcat:9.0.30:*:*:*:*:*:*:*:*
cpe:2.3:a:apache_software_foundation:tomcat:9.0.30:*:*:*:*:*:*:*:*
cpe:2.3:a:apache_tomcat:apache_tomcat:9.0.30:*:*:*:*:*:*:*:*

A regression was possible in certain versions of Apache Tomcat that some headers could be processed resulting in HTTP Request Smuggling. Advised to not use these versions, using a more recent or supported version.

5. Mitigation Plan
After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial's software application.

Many of the flaws are due to varied flawed versions of programming. Some may be fixed with manual overriding, but many may be changed to updated or more stable past versions that don't have those vulnerabilities. Encapsulation is also another important issue, as all critical variables should only be seen and used by it's specific class.

Sources:

What is API security? (2019, January 8). Retrieved November 14, 2021, from https://www.redhat.com/en/topics/security/api-security.