****

# 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** | **03/22/24** | **Brandon Porter** | **Updating** Client NeedsAreas of SecurityManual ReviewStatic TestingMitigation Plan |

## 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

Brandon Porter

## Interpreting Client Needs

Artemis is a growing company that wishes to modernize their operations. They will need to be-to-date on the most current security needs to work with their clients. Artemis is working with some of the most personal information on the market. Dealing with retirement, savings, investments, and insurance needs of their customers makes Artemis a high value target for those seeking to obtain this private information and sell it to the highest bidder. Their needs will be met with the most robust securities and vulnerability testing available. Because of the company’s widespread base with users and customers globally we will need to follow best practice for security coding and launch plans before launching. At this moment Artemis uses REST API, as good as this is as far as how they are starting out, they need some improvements from Global Rain to reenforce any of their issues. We will review and analyze the most in-depth parts of the client vulnerability tests and put best practices for word. We know that Artemis deals with clients of all sorts of needs and locations, this means secure communication is a must and needs to be built into the service.

This aspect becomes particularly crucial given the potential for international transactions involving clients. Variations in security and privacy regulations across different regions necessitate careful consideration. For instance, in Europe, the General Data Protection Regulation (GDPR). Many countries have crucial regulations on security breaches. Consequently, Artemis Financial must either adhere to these regulations or confront potential repercussions stemming from non-compliance with diverse governmental mandates. Additionally, there exists a persistent threat of cyber attacks from state adversaries seeking to conduct espionage, data theft, or sabotage, further emphasizing the importance of robust security measures.

To guarantee the modernization of the client's systems, it's imperative to ensure they utilize the latest web technologies. This not only enhances the functionality and security of their web application but may necessitate periodic system updates to maintain compatibility with the most recent advancements in web technologies.

## Areas of Security

API:

The security aspect of web applications is critical mainly for API’s because they ensure the protection of the relationship and prevent the potential security attacks. APIs are mediators and they communicate between the different applications, facilitating communication. OAuth known as Open Authorization is a protocol within the API and it allows the user to grant access to the web resources without sharing a password. OAuth is used widely to grant access to websites and applications without the need to share a password. Security measures, such as HTTPS and other protocols, are integrated into RESTful APIs to prevent attacks. However, precautions like escaping specific characters in untrusted input are essential to thwart injection attacks, including command injections.

Cryptography:

The data that Artemis will be dealing with is going to be very sensitive given the information that we have and knowing what documentation they are using. This means that CryptomorphyIs going to play a crucial role in securing the information that is going to be transmitted over the internet by encrypting that data. This safeguards its integrity and confidentiality. Encryption is a fundamental process because of the way the data is transformed into a format that prevents unauthorized access. Cryptography ensures that data remains protected even when parties involved lack mutual trust, using techniques like commitment schemes and computational securities to establish trust in the encryption process despite mutual distrust between parties.

Code Error:

Clients/ server communication is a fundamental process in web application. Clients will interact with the backend of the application through the front-end user interface. This requires various programs that are architecturally involved. Security features within client/server systems encompass physical security, safeguarding hardware, networks, and data from physical threats, and change control security, guaranteeing systems can defend against vulnerabilities autonomously. These measures are indispensable for ensuring system security.

Code Quality:

Code quality and best practices are crucial to the security of a program. By following the best practices and having good code quality this puts the program in the right path starting out. When adding and manipulating adjusted coding lines, we want the best quality code that we can get with good commenting and understand as to what each main point of code is doing. OWASP can help with secure coding practices making clients understand the importance of secure coding.

## Manual Review

There are vulnerabilities in the DOCDATA.java file. It's not advisable to use root user credentials for accessing data. Root usernames and passwords are vulnerable, as the root password can be easily guessed. Unauthorized users could gain system access through brute force attacks. Here the data is access method retrieves information about the locations of data that includes usernames and passwords, this poses a massive threat. In the CRUD Controller file there might be exposure for internal objects allowing them to be retrieved and manipulated through code injection. More so “business\_name” values is passed through the CRUD method. This exposes high vulnerability.

## Static Testing

A screenshot of a computer

Description automatically generated

* Spring Boot

A screenshot of a computer error

Description automatically generated

Mitigation: Upgrade to the latest versions

* Bouncy CastleA screenshot of a computer

  Description automatically generatedA screenshot of a computer

  Description automatically generated
* Mitigation: The Bouncy Castle allows for malicious application to access the private data stored on the database. The recommended solution for vulnerability is to frequently update the application among the operating system to prevent a compromise of the security of the application.
* Snakeyaml

A screenshot of a computer error message

Description automatically generated

Mitigation:

* Upgrade to latest versions!
* Utilize SafeConstructor when parsing.
* Address denial of service vulnerability.
* Consider depth limitations.
* Consider future vulnerabilities.
* logback

A screenshot of a computer error

Description automatically generated

Mitigation: Upgrade to latest version and address denial of service vulnerabilities

* Apache Log4j

A screenshot of a computer error

Description automatically generated

Mitigation: The Apache Log4j SMTP appender allows for log messages to be revealed if they are sent through the appended. The recommended solution for vulnerability is to upgrade to the newer version and having a built-in feature for verification.

* Redhat

A screenshot of a computer

Description automatically generated

Mitigation: Upgrade to the latest patch version and implement the input sanitization. Further review the error message handling and monitor for future responces.

## Mitigation Plan

* Create string and combination of alphanumeric characters for the username and for that of the password. This will help improve the risk from head on attacks into the system.
* Review and modify the code., integrating secure coding and best practice would help greatly when addressing the quality of the code and catching errors and handling of them. By doing this we know it will help in the mitigation process of authentication and vulnerabilities to attacks.
* Updating and performing all necessary versions and running through the mitigations listed above for each vulnerability is a must and will prove to solve many of the issues and create secure application.
* These steps should be followed:
  + Upgrading to the patch version and latest upgrades
  + Implementing input sanitization
  + Reviewing error message handling
  + Providing security awareness training
  + Security audits and code reviews
  + Monitoring and incident response
  + Vendor notification
  + Backup and recovery
* For each of these listed approaches the company should have an in-depth team and plan of action to ensure that when performing upgrades the teams are doing it appropriately, auditing teams should exist within the company to ensure accountability to the company and the program and a team of to monitor and respond to the incidents should be in place.