****

# 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 | **1/22 /2023** | **James Fowles** |  |

## Client



## Developer

James Fowles

## Interpreting Client Needs

All clients need a comprehensive security plan, and some organizations are even legally bound to protect client data. There are many reasons companies value security, the first being companies compete and the possibility of intellectual property being leaked could have damaging effects on business.

The second is cost, it is very expensive to recover from a cyber-attack, not only is there fixing and securing the breach there is the PR campaign to do damage control, but finally companies also have reputations to maintain and having a bad track record when it comes to people’s money is especially bad for a company whose primary product is managing and investing other people’s money.

It is hard to tell exactly what type of transactions this company makes but one can safely assume it includes international transactions as the modus operandi for investment companies is to reduce risk by diversifying investments that often includes investing in international bonds and stocks and as such security considerations should be considered to whatever API or connections the organization has to any international entity.

Not only is it important to be a secure company it is also legally required. For example, the US Securities and Exchange Commission regulation S-P requires investment companies to adopt written polices and procedures that address technical safeguards for the protection of customer records and information along which is covered under the 34-42974 SEC release. Additionally in 1999 the Gramm-Leach-Bliley act (GLBA) was enacted which included additional legal requirements for financial companies to protect and safeguard data at the threat of legal repercussions and fines.

Financial organizations are always going to be a target for attacks because exploiters know what type of information they are bound to store. It is safe to assume these companies will store users SSN, dates of birth, address’s and of course their assets such as bank accounts, investments, 401k’s. One consistent and current threat is social engineering which is the act of tricking a legitimate employee to give you privileged access. Steps can be taken via software and training to help avoid these type of attacks, in the future Artemis needs to keep up with changing technologies and exploits as more sophisticated attacks are being designed every year such as advanced API exploits, SQL injections, and Zero day hacks based on companies not following a consistent update schedule.

Modernization efforts should be focused on checking to make sure dependencies are up to date, while reducing the amount of API’s and dependencies that are needed. Making sure two factor authorization is used where possible while verifying end points in external libraries are secure and valid by verifying, they are still well maintained and researching any exploits that are found. Finally keeping up with modern web technology when appropriate due to the cost of maintaining out of date software.

## Areas of Security

**Input Validation**

Input validation should be reviewed to avoid against malicious inputs such as injection type attacks and denial of service (DoS) exploits. This can be done by limiting string size and putting in validation code to make sure whatever is passed to the application meets certain criteria.

**API**

Organizations such as Artemis use RESTful API’s to request and send information over HTTP protocol. Correct authorization tokens should be used and updates to this API should be maintained.

**Cryptography**

Following and maintaining encryption standards on all areas where sensitive information is being moved around. This includes adding encryption around input and API transactions.

**Client/Server**

Since this product is using RESTful API’s. Artemis will need to make efforts to secure web tokens and additional considerations will need to be taken for stopping denial of service and limiting resource usage will need to be addressed.

**Code Error**

Making sure that all errors and cached data are handled appropriately will be important as well as ensuring that only trusted commands and transactions can be executed. Additionally, making sure that any logs/outputs do not include any sensitive data such as passwords in the clear.

**Code Quality**

Code quality is important to ensure that functions and methods are efficient and using the appropriate amount of memory when running and not causing stack overflows. Checking loops to make sure they don’t become infinite are all important steps in quality code.

## Manual Review

Manual review of code showed a few areas of concern. The general concern is there is still no validation around input data in any of the classes this could be resolved with proper getters and setters on fields. There are two main areas of concern the first being in DocData.java. The major issue is the default password is being used and the server is being accessed as the root user. Security articles have stated that default passwords “are easy to find on forums and other easy-to-find places on the internet.” Default Settings, and Why the Initial Configuration is not the Most Secure. (2017, March 8).

Case in point, “one POS manufacturer used the same password for 25 years” Default Settings, and Why the Initial Configuration is not the Most Secure. (2017, March 8). It is good practice to create a user which has only the minimal access necessary and use that for connections to cut down on possible issues if the password is compromised. Additionally, there is an unused variable “con” while not a direct threat at present it could become an issue as the application gets bigger.

In the Customer.java the field variables need to be private so they can not be accessed outside of the class. In the Greetings.java we need to have some validation around what can be entered as it takes a long int and string and as such could be targeted by an inject style attack.

In the CRUDController.java since this instantiates the DocData class and takes in a unvalidated parameter with all public fields there is the possibility it could lead to objects being exposed to attacks while also sending the business name along as the request param.

Finally, I do not see where the application is using the more secure HTTPS considering the type of information it is dealing with for this API.

## Static Testing

Graphical user interface, text

Description automatically generated

|  |  |
| --- | --- |
| [bcprov-jdk15on-1.46.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l1_991c96a4e31e6c19e2b9136c8955bd423f2dc4c7) | This is a crypto package for cryptographic algorithms. Issue: Provider version 1.55 and prior the DSA does not fully validate ASN  Possible Solution: update to version above 1.55 |
| [hibernate-validator-6.0.18.Final.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l3_7fd00bcd87e14b6ba66279282ef15efa30dd2492) | Hibernates beans validation. Issue: includes a bug in the message interpolation process that enables invalid EL expressions to be evaluated as if they were valid.  Possible solution: update to newer version |
| [jackson-databind-2.10.2.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l5_0528de95f198afafbcfb0c09d2e43b6e0ea663ec) | General data-binding functionality for Jackson  Issue: Flaw found in FasterXML Jackson data bind where it did not have entity expansion secured.  Possible solution: avoid use or verify issue before using |
| [log4j-api-2.12.1.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l10_a55e6d987f50a515c9260b0451b4fa217dc539cb) | Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. Could allow a man in the middle attack.  Possible solution: upgrade version |
| [logback-core-1.2.3.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l12_864344400c3d4d92dfeb0a305dc87d953677c03c) | In log back 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.  Possible solution: update past 1.2.7 |
| [snakeyaml-1.25.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l14_8b6e01ef661d8378ae6dd7b511a7f2a33fae1421) | Deserializing yaml content provided by an attacker can lead to remote code execution  Possible solution Using SnakeYmals safe constructor when parsing untrusted content |
| [spring-boot-2.2.4.RELEASE.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l15_225a4fd31156c254e3bb92adb42ee8c6de812714) | Unsupported when assigned version. Opens application to directory hijacking.  Possible solution: update to latest version |
| [spring-boot-starter-web-2.2.4.RELEASE.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l16_ec75d01d212b5229c16d872fb127744c0ed46ed8) | Unsupported when assigned version. Opens application to directory hijacking.  Possible solution: update to latest version |
| [spring-core-5.2.3.RELEASE.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l17_3734223040040e8c3fecd5faa3ae8a1ed6da146b) | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.  Possible solution: make sure framework is being used properly on JDK+9 |
| [spring-web-5.2.3.RELEASE.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l18_dd386a02e40b915ab400a3bf9f586d2dc4c0852c) | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding  Possible solution: make sure framework is being used properly on JDK+9 |
| [tomcat-embed-core-9.0.30.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l20_ad32909314fe2ba02cec036434c0addd19bcc580) | Apache Tomcat treats Apache JServ Protocol (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.  Possible solution: Verify configuration and access levels |
| [tomcat-embed-websocket-9.0.30.jar](file:///C:\Users\jfowl\eclipse-workspace\rest-service\target\dependency-check-report.html#l22_33157f6bc5bfd03380ebb5ac476db0600a04168d) | Apache Tomcat treats Apache JServ Protocol (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.  Possible solution: Verify configuration and access levels |

## Mitigation Plan

There are a couple of steps needed to mitigate the issues found in this software. Most of the dependencies can be resolved by updating to the latest version or a version that is higher than the affected one. The tomcat-embed-websocket can be resolved by verifying configuration of the server or updating the version.

Additionally, the application should be using HTTPS when connecting to RESTful API’s to ensure encryption. When creating classes developers should use as many private fields as possible, so that objects created cannot be accessed by unauthorized code and input validation should be added when taking data directly from the customer. This can be as simple as limiting the number of characters a field can take or checking the input against certain logic to make sure it’s a bonafide entry and not an injection attack, adding limitations to data input also cuts down on DoS attacks.

The application should not be exposing business names in its parameters or having any passwords in the clear. Finally, the database connection should not be the root user and it should not be using the default username and password as these are readily available online for bad actors to try.

References

Federal Trade Commission. (n.d.). *Gramm-Leach-Bliley Act*. Federal Trade Commission. https://www.ftc.gov/business-guidance/privacy-security/gramm-leach-bliley-act

‌

*Default Settings, and Why the Initial Configuration is not the Most Secure*. (2017, March 8). Panda Security Mediacenter. Click here to enter text.<https://www.pandasecurity.com/en/mediacenter/security/default-settings-initial-configuration-not-secure/>

NIST. (2019). *NVD - Vulnerabilities*. Nist.gov. https://nvd.nist.gov/vuln

‌

‌