****

# 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** | **March 26, 2023** | **Michelle Ou** |  |

## 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

Michelle Ou

## Interpreting Client Needs

We must emphasize the fact that our client, Artemis Financial, is a consulting company that devises “individualized” financial plans for their customers. What does that mean? It means that our client might and most likely holds even more customers’ personal information than financial/bank companies, because only do they know more about the customer can they make a better customized plan for the customer. It is because they know how much information they hold and how important the information are that they come to use majorly for security. And so we know that they greatly value secure communications. They want to ensure that the information their customers provided to them and the information they gave to their customers remain confidential. Although there are not any information specifying that Artemis Financial works internationally, but it wouldn’t be a surprise if they did or are planning to do so. Afterall, we are a global company, and it could have been one of the reasons they chose us.

There are some information that each individual have and must think through before they provide it to any company, organization, or business. The government have these information, so they know the possible consequences of these information getting into the hands of others. That is why they have restrictions on the use of these information and secure communication to ensure that the company/organization/business that have these information can keep them safe to the best of their ability. Knowing what type of company our client is, we know that there are no doubt governmental restrictions that need to be considered, such as encrypting information to disable others from reading it. When it comes to information on cloud/web there are a lot of possible external threats. It doesn’t even have to be a company, or business, or another country; even an individual attacker can use the information to do many things. In order for users to access their information, there is most likely a sign-in username/password, and this can become a way for the attacker to access the information if we are not careful. Therefore, what we can consider having are different identity verification processes and regular system check for vulnerabilities.

## Areas of Security

The area of security that need to be assessed are:

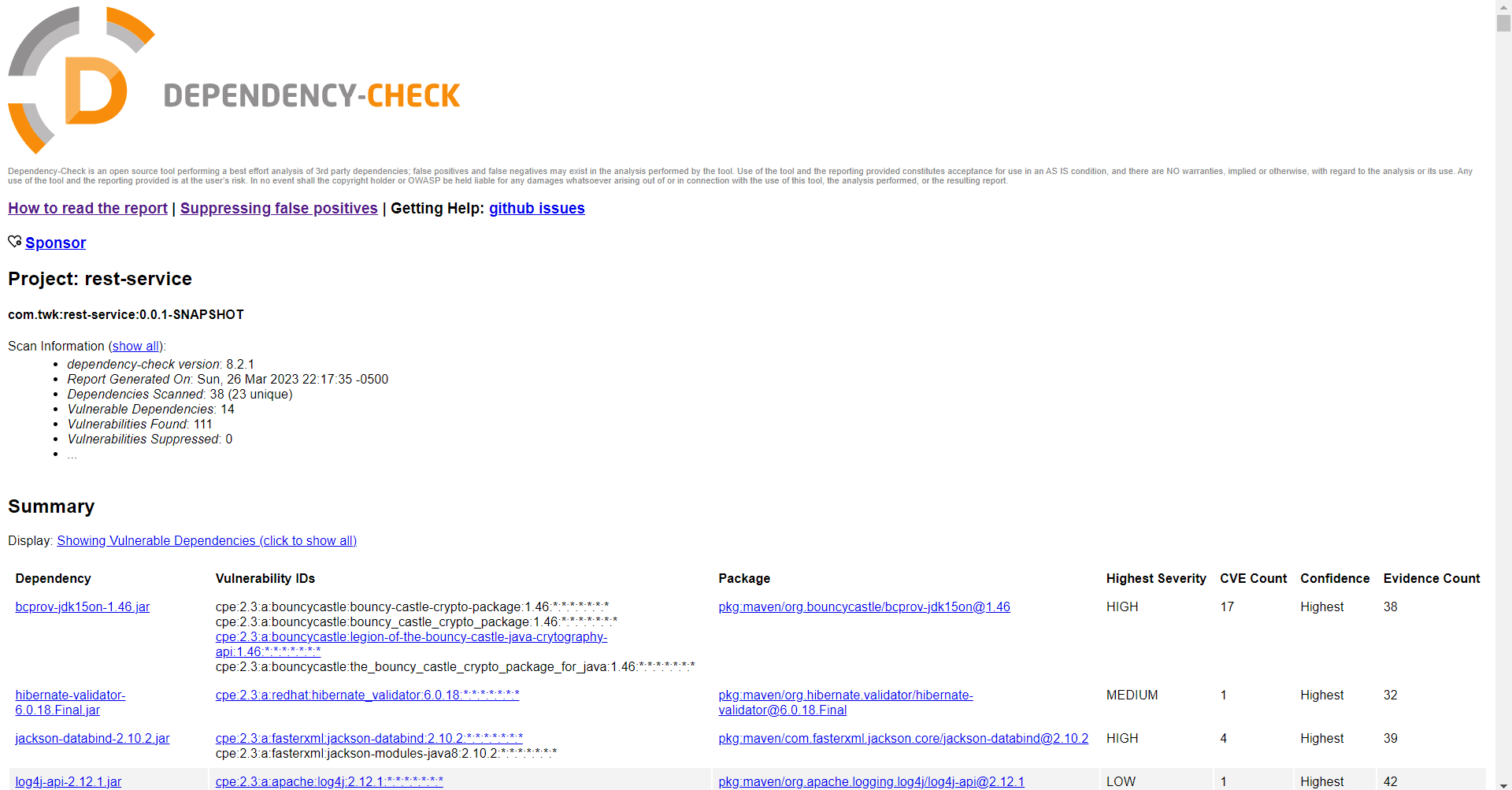
* **Input Validation-Secure input and representations:** There has to be a way for users to access their information, so inputs are inevitable. And it being a way to access all of the personal information makes it a major area of security we need to assess.
* **APIs-Secure API interactions:** When our client, Artemis Financial, was introduced, the fact that they have RESTful web application programming interface, which is a API. Therefore, this is an area of security that need to be assessed.
* **Cryptography-Encryption use and vulnerabilities:** In order to avoid the information being used by others, it is important to encrypt the information. This is both another security for if the information gets out of hand, and one way to work with governmental restrictions. Hence, ensuring that there are not flaws in cryptography is important.
* **Code Error-Secure error handling:** It is alright to have errors, but it is bad if both us and the user are not aware of it. Being able to handle the error would allow us to lessen the possibility of attacker using it to their advantage, especially during information input and transactions.
* **Code Quality-Secure coding practices and patterns:** When other area of security such as Input Validation, APIs, and Code error are mentioned, we know code quality will eventually be mentioned as well. If there are flaws in the code quality, it might not only affect other areas of security, but also how the system functions. Especially since our client touches both personal information and financial information, which requires a lot of data and layer of security, we have to carefully assess the code quality.

## Manual Review

The first thing I noticed was the issue of the versions not being up to date. If we place our cursor on the lines with @ and the class, or method, or tool name, it shows more information about it and the version. Seeing the fairly small digits, I am pretty sure they are not the recent versions. Although we cannot say that there will be no flaws in current versions, but older versions tend to have more flaws that could be avoided if we switch to a more current one. This was also the case when I was looking through pom.xml file, which shows the versions for many things such as the model version, java version, and dependency check version (Which is also not up to date, but I will change it in the static testing). Another thing I noticed was that the codes are pretty basic. In other words, they lack a lot of security. For instance, even with the importance of cryptography, I do not see it anywhere in the codes. Also, in DocData.java, I see a bit of mentioning the username/password (Although, the class is mostly used for reading file), and do not see it anywhere else. For something as important as username/password which can access the personal information, I find it lacking in input validation and verification.

## Static Testing

Although this project already has a dependency check in place, I still followed the tutorial and change the version to current, because I was not sure if an non up to date version would affect anything.



|  |  |  |  |
| --- | --- | --- | --- |
| **Dependency** | **Description** | **Vulnerability** | **Solution** |
| bcprov-jdk15on-1.46.jar | The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7. | CVE-2016-1000338  CVE-2016-1000342  CVE-2016-1000343  CVE-2016-1000344  CVE-2016-1000352  CVE-2016-1000341  CVE-2016-1000345  CVE-2017-13098  CVE-2020-15522  CVE-2020-0187 (OSSINDEX)  CVE-2016-1000339  CVE-2020-26939 (OSSINDEX)  CVE-2015-7940  CVE-2018-5382  CVE-2013-1624  CVE-2016-1000346  CVE-2015-6644 (OSSINDEX) | Use latest version |
| hibernate-validator-6.0.18.Final.jar | Hibernate's Bean Validation (JSR-380) reference implementation. | CVE-2020-10693 | Use latest version |
| jackson-databind-2.10.2.jar | General data-binding functionality for Jackson: works on core streaming API | CVE-2020-25649  CVE-2020-36518  CVE-2022-42003  CVE-2022-42004 | Use latest version |
| log4j-api-2.12.1.jar | The Apache Log4j API | CVE-2020-9488 | Use latest version |
| logback-core-1.2.3.jar | logback-core module | CVE-2021-42550 | Use latest version |
| snakeyaml-1.25.jar | YAML 1.1 parser and emitter for Java | CVE-2022-1471  CVE-2017-18640  CVE-2022-25857  CVE-2022-38749  CVE-2022-38751  CVE-2022-38752  CVE-2022-41854  CVE-2022-38750 | use SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization and use latest version |
| spring-boot-2.2.4.RELEASE.jar | Spring Boot | CVE-2022-27772 | Use latest version |
| spring-boot-starter-web-2.2.4.RELEASE.jar | Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container | CVE-2022-27772 | Use latest version |
| spring-core-5.2.3.RELEASE.jar | Spring Core | 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 | Use latest version |
| spring-expression-5.2.3.RELEASE.jar | Spring Expression Language (SpEL) | CVE-2022-22965  CVE-2021-22118  CVE-2020-5421  CVE-2022-22950  CVE-2022-22971  CVE-2022-22968  CVE-2022-22970  CVE-2023-20861 (OSSINDEX)  CVE-2021-22060  CVE-2021-22096 | Use latest version |
| tomcat-embed-websocket-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-2021-43980 | Use latest version |

## Mitigation Plan

Through the dependency check we can see to importance of versions. The best solution for most of the vulnerabilities is to keep the versions of the tools as current as possible. Therefore the steps we can take is to first use SnakeYaml's SafeConsturctor instead of snake yaml, then update all the versions to latest. Then, we can work on the input validation and verification, strengthening API security, and cryptography.