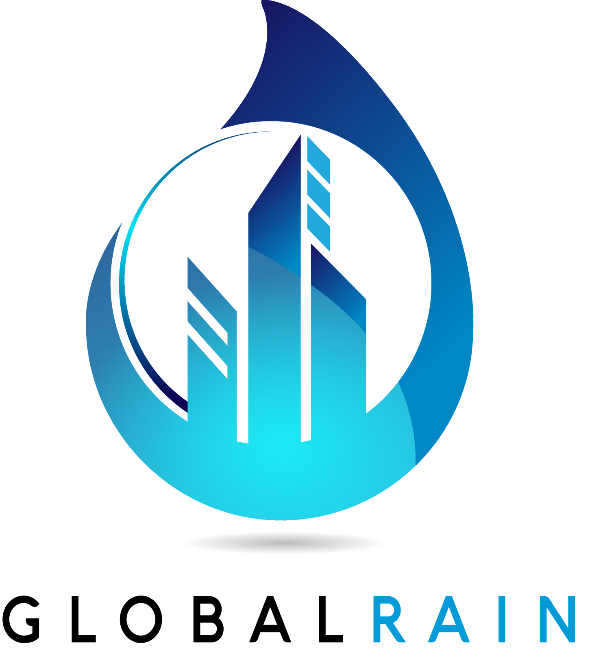
****

# CS 305 Project One

**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** | **11/08/2021** | **Justin Haby** |  |

## 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

Justin Haby

## 1. Interpreting Client Needs

* **What is the value of secure communications to the company?**

Secure communication is a primary priority for every institution, and every organization must ensure this, but it becomes much more critical when an organization handles sensitive data like information that Artemis Financial has. The value of secure communications to Artemis Financial is important because the company holds critical data of its customers and therefore cannot allow a breach to happen, which may lead to reputational damage to the extent of facing legal suits by its customers.

* **Are there any international transactions that the company produces?**

There are a number of international transaction that Artemis Financial conducts. These international transactions include transactions done by its clients outside the company's country. However, despite how risky international transactions are, the organization has placed programs to ensure safe transactions.

* **Are there governmental restrictions about secure communications to consider?**

There are a number of restrictions that the federal governments across various countries have in place to ensure secure communications. For instance, in European Union countries, there is a regulation named "General Data Protection" that safeguards communication and sets measures in which data handled by companies are protected and secured against unauthorized access. As a result, of going against this regulation, companies may be fined and punished when their customers' sensitive data is breached.

* **What external threats might be present now and in the immediate future?**

External threats currently present and may still be a threat in the intermediate future are authentication threats which are the most common since the company grants access to unauthorized parties who may damage the reputation of the company and destroy the company’s ICT infrastructure.

* **What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?**

One of the modernization requirements is that it can make significant changes to existing systems, thus increasing the chance of the system malfunctioning and producing problems for the firm and its customers. This problem may be fixed by regularly updating the servers to figure out which of the updates 'broke' the circuit. Unauthorized access to web application or servers, in general, become possible as a result of modernization demands. This is attributed to an SMTPS connection, which allows unauthorized access to critical information.

## 2. Areas of Security

The study will be focused on only the below areas, but it's crucial to remember that all components of the vulnerability assessment process apply in the program. Encapsulation, code quality, and Code error are the three items that have been left off. All of the items are relevant to this application, though to a lower degree than the other four. To avoid a severe system failure in the event of uncertain data, inaccuracies in the login process or inside account management operations will necessitate some type of error handling. Encapsulation and code quality will apply to the data structures as well as the application code, but faults with either of these should be avoided by adopting best practices. All areas of coding must be considered while securing a system, although depending on the application's design, some elements may play a larger role than others.

**Input Validation:**

Users of this program will have access to secure data that must be kept private and confidential. As a result, we need to verify that anyone accessing this data is permitted. Before authorizing access to the system, the program must appropriately validate the user's credentials. Compliance in this section will aid in the prevention of data breaches and a variety of phishing attacks, as well as retain clients' confidence in the company. Artemis can assure that it is adhering to all regulations put forth by governments worldwide by authenticating each user.

**APIs:**

Since this application was built with the REST API, its proper integration is critical to its success. Importantly, using the most recent API-released versions will provide the strongest defense against known vulnerabilities and prevent attackers from exposing earlier API versions.

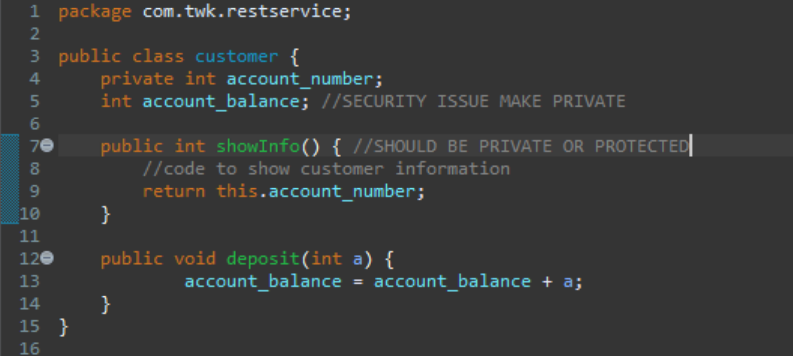
**Cryptography:**

Because most of the data sent and received through the application is critical data protected by several federal rules, it's critical to follow best practices when transmitting and receiving information. Personal client information, in particular, should be packed and encrypted before being transmitted or delivered. Additionally, when used correctly, cryptography can help avoid hijacking and interception-based attacks while also maintaining the secrecy of users' data.

## 3. Manual Review

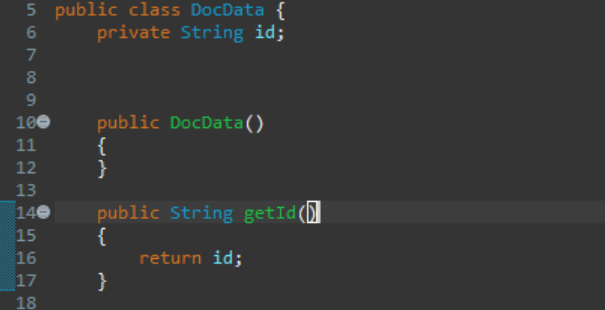
The findings of our manual inspection of the code will be presented in this section, along with any areas of concern. Additionally, there will be visual documentation as well as a technical explanation for each problem.

**Exhibit 1: *customer.java***



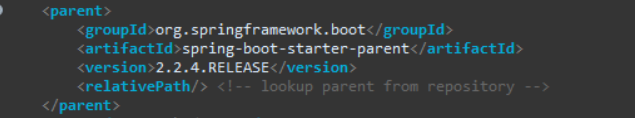
Inspecting the code, multiple cases of incorrect access control definitions within the "customer class” might allow an attacker to access confidential information. For instance, one of the most blatant examples in the code was a variable that holds the value for the customer's account balance and is now inheriting the class's public access level. To protect external programs from reading personal data, it should be saved in private variables utilizing obtaining and setting data structures. The method retrieves the client's account number is now set to be a public function, making it accessible from anywhere in the application. Despite the fact that the account number variable is only available to this class, attackers could still penetrate the program and use the "showInfo" method to get the account number's secret value. This puts the client's data's confidentiality in jeopardy, and corrective action should be undertaken.

**Exhibit 2: *DocData.java***



Like the "DocData" class, the "customer" class has various vulnerabilities that could put confidential data at risk. The class is now public, making it available throughout the code, but the method that returns the private value for "id" is the more worrying aspect. If an attacker accesses the "getId" function, the hidden nature of the variable "id" can be bypassed, allowing access to many more sensitive data. To preserve the integrity and confidentiality of the client's data, corrective actions must be conducted once more.

**Exhibit 3: *“pom.xml”***



The version of the Spring Boot API is the source of the final vulnerability uncovered during our manual review. This dependency is required to build a framework for the Artemis application to function, and the version included with the program is 2.2.4. RELEASE. This version is outdated and is susceptible to various types of attacks that are not present in the latest stable release version of 2.5.4.

## 4. Static Testing

**Dependencies:**

**bcprov-jdk15on-1.46.jar:** “Legion of the Bouncy Castle Legion of the Bouncy Castle Java Cryptography APIs 1.58 up to but not including 1.60 contains a CWE-470”. Earlier versions of Bouncy Castle have a flaw in low-level interface. Updating to versions 1.60 and later should fix vulnerabilities listed below, as this current version is 1.46.

* CVE-2013-1624
* CVE-2016-1000338
* CVE-2016-1000339
* CVE-2016-1000341
* CVE-2016-1000342
* CVE-2016-1000343
* CVE-2016-1000344
* CVE-2016-1000345
* CVE-2016-1000346
* CVE-2016-1000352
* CVE-2017-13098
* CVE-2018-1000613
* CVE-2018-5382
* CVE-2020-26939 (OSSINDEX)

**log4j-api-2.12.1.jar:**

* CVE-2020-9488

**snakeyaml-1.25.jar:** (Status High) - “Allows context-dependent attackers to cause a denial of service (memory and CPU consumption) via a crafted XML document containing a large number of nested entity references, aka the "billion laughs attack." Updating to versions 1.60 and later should fix vulnerabilities listed below, as this current version is 1.46.

* CVE-2017-18640

**jackson-databind-2.10.2.jar:**

* CVE-2020-25649

**tomcat-embed-core-9.0.30.jar:** (Status Critical) - Spring boot tomcat dependency, Update version to most current.

* CVE-2019-17569
* CVE-2020-11996
* CVE-2020-13934
* CVE-2020-13935
* CVE-2020-13943
* CVE-2020-17527
* CVE-2020-1935
* CVE-2020-1938
* CVE-2020-8022
* CVE-2020-9484
* CVE-2021-24122
* CVE-2021-25122
* CVE-2021-25329
* CVE-2021-30640
* CVE-2021-33037
* CVE-2021-41079
* CVE-2021-42340

**hibernate-validator-6.0.18.Final.jar:** (Status Medium) - “Flaw found in Hibernate version 6.1.2. Final. Enables invalid EL expressions to be evaluated as they were valid. Allows attackers to bypass input sanitization due to developer error". Update to the Most current version to eliminate both risk and vulnerability

* CVE-2020-10693

**spring-core-5.2.3.RELEASE.jar:** (Status High) - Three listed vulnerabilities, update to latest version.

* CVE-2020-5421
* CVE-2021-22096
* CVE-2021-22118

**spring-jcl-5.2.3.RELEASE.jar:** (Status Medium) - "Older unsupported versions of Spring Framework, protection against RFD attacks remain vulnerable to being bypassed. Update to the latest version in your Spring framework.

* CVE-2020-5421

## 5. Mitigation Plan

According to the report, the current condition of the Artemis software reveals various weaknesses that might place the company in direct conflict with its obligations to both governmental agencies and their client base. The majority of these flaws can allow attackers to breach the program and obtain access to client data such as account numbers, balances, and personal information. The issues uncovered range from a lack of adherence to proper data structuring principles to old APIs with known flaws. All vulnerabilities will have their own solution; however, most dependencies can be resolved by upgrading to newer APIs. The errors detected during the manual inspection should be easily fixed by reworking the access levels of the java classes within the application. The recommended remedies for the detected vulnerabilities are presented below, along with their associated security areas.

***References:***

Gehem, M., Usanov, A., Frinking, E., & Rademaker, M. (2015). *Assessing Cyber Security: A meta analysis of threats, trends, and responses to cyber attacks*. The Hague Centre for Strategic Studies.

National Institute of Standards and Technology. (n.d.). National Vulnerability Database. NVD. Retrieved September 19, 2021, from <https://nvd.nist.gov/>.

National Vulnerability Database. (n.d.). *NVD - Results*. NIST. Retrieved November 19, 2021, from https://nvd.nist.gov/vuln/search/results

Tikkinen-Piri, C., Rohunen, A., & Markkula, J. (2018). EU General Data Protection Regulation: Changes and implications for personal data collecting companies. *Computer Law & Security Review*, *34*(1), 134-153.