****

# 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** | **3/17/2023** | **Eric Farkas** | **Initial Comments & Structure Added** |
| **1.1** | **3/18/2023** | **Eric Farkas** | **Section 1 and 2 Details** |
| **1.2** | **3/19/2023** | **Eric Farkas** | **Section 3, 4, 5 Details** |

## 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

Eric Farkas

## Interpreting Client Needs

Artemis Financial provides investment and account management services to their customers. Artemis is in the process of modernizing their client interface. The addition of a web user interface increases the usability and availability of account information. This new interface also introduces risks to security vulnerabilities and data leaks. Artemis Financial has retained Global Rain to perform an assessment and propose a mitigation plan regarding their security vulnerabilities.

This analysis was performed using the Vulnerability Assessment Process Flow Diagram (VAPFD) with a focus on the three pillars of data security as defined by the National Institute of Standards and Technology (NIST) regarding financial institution. These pillars are confidentiality, integrity and availability. This assessment is augmented by guidance provided by the US regulations defined by the Gramm-Leach-Bliley Act and the European regulations defined by the General Data Protection Regulation. Although these regulations are supported by processes independent of this system, support for these processes should be considered.

Implementing a web-based user interface introduces the potential for threats against these pillars. Malware, phishing, trojan-based and ransomware attacks are likely the most relevant external types. Customers may also become susceptible to socially engineered attacks through email, phone and social media inquiries for information. Many of these threats can be addressed within the code, but some require end-user education through system notices, email messaging and reminders.

Several questions were posed by Artemis during requirements gathering. These questions are addressed here:

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

Secure communications are extremely important in this system. Data packets will be transacted here, including credentials, personal information and financial information. All possible strategies to encrypt and secure these data packets should be implemented to keep customer information private.

**Does the company make any international transactions?**

The simple answer is yes. Investment portfolio strategies are typically not limited to specific geographic regions. It is unclear in the requirements whether the customer can perform actual trading or request the trade and Artemis will perform it. It is also possible that an Artemis customer may be traveling, working or living away from their mailing address. Both cases qualify as international transactions by definition.

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

Governments and agencies provide guidelines, restrictions and best practices information regarding secure communications. Adherence to these restrictions is influenced by the geographic location of the company headquarters, the countries or regions the company does business in and the type of business being performed. Artemis would likely be required to comply with Payment Card Industry Data Security Standards (PCI-DSS) and General Data Protection Regulation (GDPR) but may have additional obligations beyond the information presented in the scope of this project’s specifications.

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

There is a large pool of current external threats and the number is expected to grow in the future. Current threats include malware, ransomware, phishing, network intrusion and vulnerabilities in the application or system framework. Social threats are also external and often overlooked. Artemis must remind its customers not to share sensitive information via phone, email and social media.

**What are the modernization requirements that you must consider?**

Web applications are widely developed using frameworks to minimize development time, budget and to increase compatibility. While this is justifiable and considered a best practice, using a framework increases the desirability of hackers to find a vulnerability or an exploit. Open-source libraries have the same benefits regarding development time, budget and large communities of support but similarly, increase the risk of vulnerabilities being discovered.

## Areas of Security

Input validation is extremely important to this system. System access begins with user credentials and continues with access to appropriate resources. Artemis must ensure the identity of the person logging in before sharing sensitive data. Proper input validation will also aid Artemis with compliance, governance and reporting for regulatory audits.

This system was created using a REST API. Implementing a maintenance program or automation is crucial to ensure the latest releases are being used. This will minimize the risk of a discovered vulnerability being used to gain entry or information from the system. It is common for system interactions to be authenticated upon initial connection in this environment. Due to the amount of sensitive data in this application, a method of multiple authentications may be justified. This could be done by establishing a reasonable session limit to decrease user inconvenience.

This system contains a large amount of sensitive data and will require cryptography. This would not only be a best practice but is likely required for regulatory audits. All information transmitted from user login to user logout should be encrypted due to the sensitive nature and regulations. Encrypted sessions will increase user confidence in the Artemis user interface and should prevent hijacking and interception-based attacks.

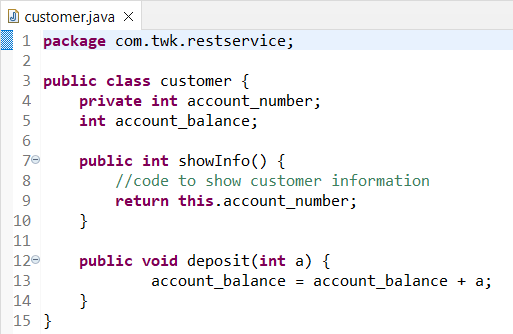
This system leverages a client/server relationship due to its web-based nature. It is reasonably easy to control the security of the server side but more challenging or impossible to control the client side. This fact puts additional focus on the API interactions. Additional diligence must be put on vulnerability and version maintenance to ensure the interactions are not susceptible to cross-scripting, hijacking or injection-based attacks.

These four areas are extremely important to the security of this system. This does not mean that code error, code quality and encapsulation are unimportant. All three should be addressed through the development and testing cycle. They were deprioritized here as they primarily represent internal threats and the scope of this review is external.

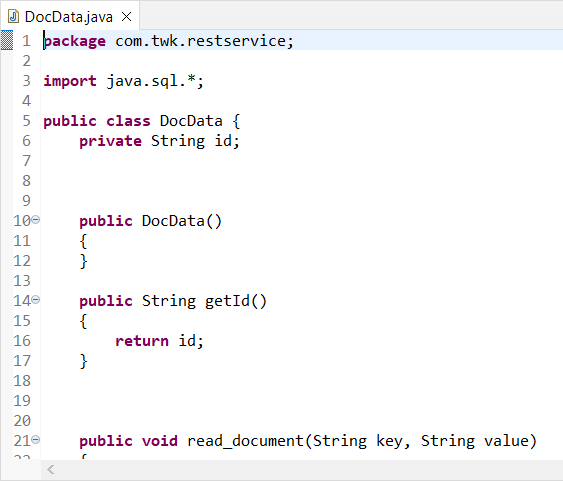
## Manual Review

The manual review of the submitted code revealed many opportunities for security improvement. This review includes code samples and descriptions of the potential vulnerability.

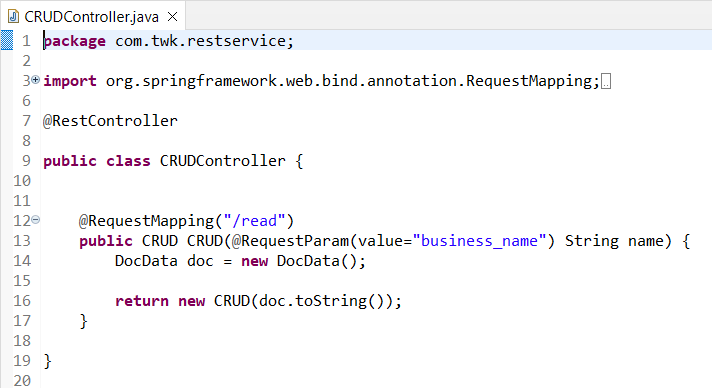
The customer class (customer.java) contains many improper access control definitions that could allow sensitive data to be revealed to an attacker. It would be possible to leverage the show info method to return the account number to other fields in the application. All variables within this class should be set to private to prevent external code from viewing or exploiting them.



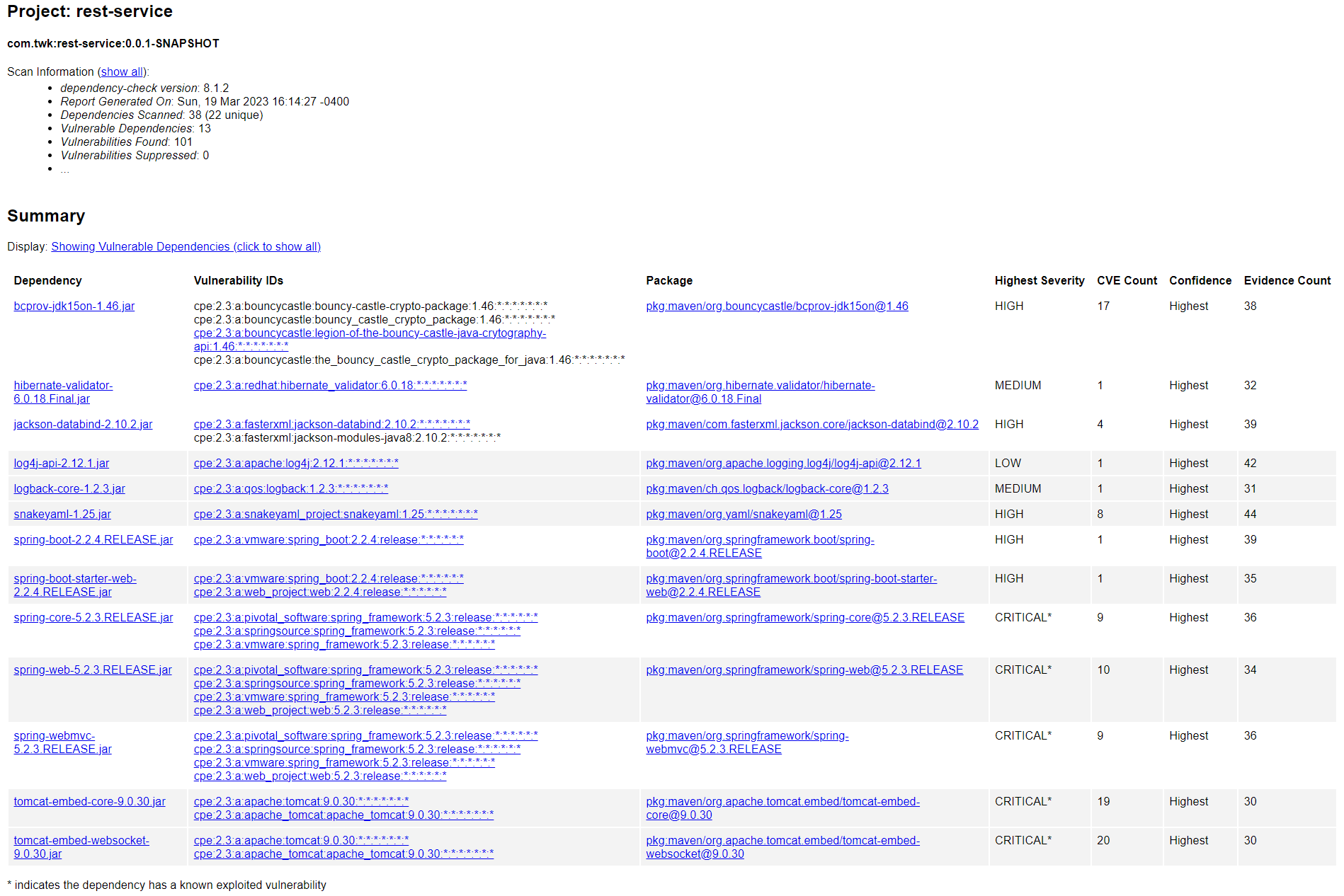
The DocData class (DocData.java) suffers from similar access control definition issues that could allow sensitive data to be revealed to an attacker. It would be possible to leverage the getId function to return the ID elsewhere in the application. All variables within this class should be set to private to prevent external code from viewing or exploiting them.



The CRUDController class (CRUDController.java) contains a method that makes direct reference to DocData. This method could be exploited using code injection to expose data within the DocData object.



## Static Testing



**Vulnerabilities Identified:**

Bouncy Castle

* Dependencies: bcprov-jdk15on-1.46.jar
* CVE Count: 17
* Severity: HIGH
* Description: This package serves to implement cryptographic algorithms into Java which adds to the concern regarding the multiple documented vulnerabilities. Of the published vulnerabilities, CVE-2018-5382 is the most concerning as it applies to all versions prior to 1.47 of the Bouncy Castle API. The default BKS keystore uses an HMAC of only 16bits which can allow attackers to compromise the integrity of the keystore. This will allow attackers to infiltrate the application and gain access to confidential information. As this improper validation of integrity check directly affects the cryptography of the application, Artemis could be in violation of regulatory standards should this vulnerability not be addressed.

Hibernate’s Bean Validation

* Dependencies: hibernate-validator-6.0.18.Final.jar
* CVE Count: 1
* Severity: MEDIUM
* Description: This issue involves the message interpolation processor which allows invalid expressions to return as valid. This can allow attackers to bypass input validation controls. CVE-2020-10693 specifies version 6.1.2 or newer versions should remedy the vulnerability.

Jackson Databind

* Dependencies: jackson-databind-2.10.2.jar
* CVE Count: 4
* Severity: HIGH
* Description: This flaw involves the FasterXML Jackson Databind where entity expansion is not secured properly. This creates a vulnerability to external entity attacks which can affect the data integrity. The severity is high, but the vulnerability has been modified and is undergoing reanalysis according to CVE-2020-25649 on the NIST NVD database.

Apache Log4j

* Dependencies: log4j-api-2.12.1.jar
* CVE Count: 1
* Severity: LOW
* Description: This issue is caused by improper certificate validation with a host mismatch in the SMTP append. Attackers can use this vulnerability to intercept SMTP connections and leak data. This issue has been resolved with the latest version 2.14.1 of the API and is awaiting reanalysis according to CVE-2020-9488.

SnakeYAML Parser

* Dependencies: snakeyaml-1.25.jar
* CVE Count: 8
* Severity: HIGH
* Description: The alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, allowing attackers to cause a denial-of-service attack. CVE-2017-18640 only specifies version 1.18 and we are using 1.25 which may indicate a false positive. Version 1.26 or newer should provide definitive resolution.

Spring Framework

* Dependencies: spring-aop-5.2.3.RELEASE.jar, spring-core-5.2.3.RELEASE.jar, spring-web-5.2.3.RELEASE.jar, spring-webmvc-5.2.3.RELEASE.jar
* CVE Count: 29
* Severity: CRITICAL
* Description: Spring AOP and Spring Core depend on the Spring Framework which is vulnerable to an RFD attack (CVE-2020-5421). Certain browsers can allow attackers to use a jsession path parameter to bypass protections. Webflux applications are vulnerable to privilege escalation. The newest version 5.3.9 corrects this issue.

Core Tomcat Implementation

* Dependencies: tomcat-embed-core-9.0.30.jar, tomcat-embed-websocket-9.0.30.jar
* CVE Count: 29
* Severity: CRITICAL
* Description: Multiple vulnerabilities ranging from medium to critical impact have been identified. The most severe of the issues involves the usage of the AJP where Tomcat gives it a higher trust than an HTTP connection. The AJP connector should be disabled, but the default configuration is enabled (CVE-2020-1938). Attackers can read and write to files and make remote code executions through this vulnerability. Version 9.0.31 has hardened the default configuration. The system version (9.0.30) should be updated to a newer version to overcome these vulnerabilities.

## Mitigation Plan

The Artemis system has many vulnerabilities in its current state. Fixing many of these vulnerabilities is critical to restoring the system to secure and compliant The vulnerabilities allowing attackers to compromise the application and gain access to client information need immediate attention. These vulnerabilities are putting sensitive data at risk. The vulnerabilities regarding data structure practices and outdated APIs are important but carry a lower risk until an attacker realizes the vulnerability exists. No code development or significant integration is required to fix these vulnerabilities, just an update to the affected frameworks. A step-by-step plan has been included below.

**Input Validation Issues:**

Hibernate’s Bean Validation (CVE-2020-10693)

File Path - Maven Dependencies -> hibernate-validator-6.0.18.Final.jar

Action – Upgrade the API version to the latest release

Access Control

File Path - DocData.java: lines 21 - 31

Action - Define wrapper around public native methods

**API Issues:**

Tomcat Core (multiple CVEs)

Spring Framework Core (multiple CVEs)

File Path - Maven Dependencies -> tomcat-embed-core-9.0.30.jar

File Path – Maven Dependencies -> tomcat-embed-websocket-9.0.30.jar

File Path – Maven Dependencies -> spring-aop-5.2.3.RELEASE.jar

File Path – Maven Dependencies -> spring-core-5.2.3.RELEASE.jar

File Path – Maven Dependencies -> log4j-api-2.12.1.jar

File Path – Maven Dependencies -> jackson-databind-2.10.2.jar

Action – Upgrade framework to the latest release and update dependencies

**Cryptography Issues:**

Bouncy Castle Crypto Package (CVE-2018-5382)

File Path - Maven Dependencies -> bcprov-jdk15on-1.46.jar

Action – Upgrade the API version to the latest release

**Client/Server Issues:**

Spring Framework (multiple CVEs)

File Path – Maven Dependencies -> Spring Boot Application v2.2.4.RELEASE.jar

File Path – Maven Dependencies -> RestServiceApplication.java

File Path – Maven Dependencies -> spring-boot-2.2.4.RELEASE.jar

Action – Upgrade the Spring Boot framework to the latest release and update dependencies

**Code Quality Issues:**

Secure Code Implementation

File Path - customer.java: lines 4 - 9

Action - Implement proper access allocation of variables to hide sensitive data from external attack

File Path - DocData.java: lines 5 - 20

Action - Change data structures to use a hidden or guarded constructor