****

# 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 5](#_Toc32574614)

[5. Mitigation Plan 8](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **11/12/22** | **Jonathan Boeglin** | **Filled out sections** |

## 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

Jonathan Boeglin

## Interpreting Client Needs

The client, Artemis Financial, is a consulting company that develops financial plans for its customers. Their goal is modernization of their practices. They currently have a RESTful API and wish to use the most current and effective software security.

In this sense, secure communication is a must-have for the company, as their customers will be sending and receiving personal financial data. The company may also be making international transactions, depending on customer location. With both things in mind, there are certain governmental restrictions to consider, including financial and personal information transfers between an individual and an organization. Additionally, external threats to this system include viruses, ransomware, or spyware sent from compromised or malicious accounts, interceptions of data being transferred, false claims of identity on the customer and organization’s side, and interruptions of service like DoS attacks.

Modernization requirements to consider include proper previewing and maintenance of any open-source libraries used. These libraries may be uploaded with malicious code included or may become vulnerable to attacks if not updated to the newest version. With use of these libraries, coding and maintenance time for the application can be reduced. Another aspect of modernization is the fact that technology is constantly evolving. For the application to keep up, it needs to be able to adapt to any new devices that can run web applications. Therefore, keeping the security and code streamlined and not overcomplicated will help in updating later.

## Areas of Security

The areas of security that need to be focused on in this web application are Input Validation, APIs, Cryptography, and Code Quality. Input Validation is important to make sure customers are entering valid information into the system for accounts and any financial manipulation. API integration needs to be secured for the same reasons as open-source libraries, and in this case the company is using a RESTful API. Cryptography’s importance lies with the ability to secure financial data being transferred from the client to their customers, as money is involved, and the distances crossed could be international. Finally, code quality needs to be addressed, because improper coding practices in security creation can lead to vulnerabilities in other areas of the system.

## Manual Review

* @RequestParam is used in CRUDController. This is not “injection safe.”
* @RequestParam is also used in GreetingController.
* DocData connection is hard coded.
* pom.xml uses outdated Java 1.8 version.
* pom.xml dependency check needed update.

## Static Testing

Graphical user interface, text, application, email

Description automatically generated

Affected dependencies found:

* bcprov-jdk15on-1.46.jar
  + Vulnerability code(s):
    - 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)
  + Recommendation: Update to version 1.70, currently has no known vulnerabilities
  + Attribution: NIST National Vulnerability Database/MITRE
* hibernate-validator-6.0.18.Final.jar
  + Vulnerability code(s):
    - CVE-2020-10693
  + Recommendation: Update to version 6.2.5, no known vulnerabilities.
  + Attribution: NIST National Vulnerability Database/Red Hat, Inc.
* jackson-databind-2.10.2.jar
  + Vulnerability code(s):
    - CVE-2020-25649
    - CVE-2020-36518
    - CVE-2022-42003
    - CVE-2022-42004
  + Recommendation: Update to version 2.14.0
  + Attribution: NIST National Vulnerability Database/MITRE
* log4j-api-2.12.1.jar
  + Vulnerability code(s):
    - CVE-2020-9488
  + Recommendation: Upgrade to 2.13.2, supports verification
  + Attribution: NIST National Vulnerability Database/Apache Software Foundation
* logback-core-1.2.3.jar
  + Vulnerability code(s):
    - CVE-2021-42550
  + Recommendation: Update to version 1.4.4
  + Attribution: NIST National Vulnerability Database/Switzerland Government Common Vulnerability Program
* snakeyaml-1.25.jar
  + Vulnerability code(s):
    - CVE-2017-18640
    - CVE-2022-25857
    - CVE-2022-38749
    - CVE-2022-38751
    - CVE-2022-38752
    - CVE-2022-38750
  + Recommendation: Update to version 1.33
  + Attribution: NIST National Vulnerability Database/Google, Inc.
* spring-boot-2.2.4.RELEASE.jar
  + Vulnerability code(s):
    - CVE-2022-27772
  + Recommendation: Update to version 2.7.5, if looking for a RELEASE version, upgrade to 2.3.12.RELEASE, neither have vulnerabilities.
  + Attribution: NIST National Vulnerability Database/MITRE
* spring-core-5.2.3.RELEASE.jar
  + Vulnerability code(s):
    - 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
  + Recommendation: Upgrade to either 5.3.23(latest) or 5.2.22.RELEASE(latest RELEASE)
  + Attribution: NIST National Vulnerability Database/VMware
* spring-web-5.2.3.RELEASE.jar
  + Vulnerability code(s):
    - CVE-2016-1000027
    - 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
  + Recommendation: Upgrade to 5.3.23, and do not deserialize untrusted data. Vendor will not fix issue because untrusted data is not considered a use case.
  + Attribution: Sonatype OSS Index/Maven
* tomcat-embed-core-9.0.30.jar
  + Vulnerability code(s):
    - CVE-2020-1938
    - CVE-2020-11996
    - CVE-2020-13934
    - CVE-2020-13935
    - CVE-2020-17527
    - CVE-2021-25122
    - CVE-2021-41079
    - CVE-2022-29885
    - CVE-2022-42252
    - CVE-2020-9484
    - CVE-2021-25329
    - CVE-2021-30640
    - CVE-2022-34305
    - CVE-2021-24122
    - CVE-2021-33037
    - CVE-2019-17569
    - CVE-2020-1935
    - CVE-2020-13943
    - CVE-2021-43980
  + Recommendation: Upgrade to 10.1.1 or later and ensure rejectIllegalHeader is set to true.
  + Attribution: NIST National Vulnerability Database/Apache Software Foundation
* tomcat-embed-websocket-9.0.30.jar
  + Vulnerability code(s):
    - 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-2022-42252
    - CVE-2020-9484
    - CVE-2021-25329
    - CVE-2021-30640
    - CVE-2022-34305
    - CVE-2021-24122
    - CVE-2021-33037
    - CVE-2019-17569
    - CVE-2020-1935
    - CVE-2020-13943
    - CVE-2021-43980
  + Recommendation: Upgrade to 10.1.1 or later.
  + Attribution: Sonatype OSS Index/Maven

## Mitigation Plan

* Update all dependency vulnerabilities to recommended versions
* Ensure rejectIllegalHeader is set to true
* For parameter requests in CRUDController and GreetingController use the PreparedStatement class instead of @RequestParam.
* Pass connection as a variable containing the address to DocData instead of hardcoding it
* Change JRE version to more recent one in pom.xml
* Make sure pom.xml dependency check version stays up to date
* Make sure system is not deserializing untrusted data