****

# 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** | **7/8/21** | **Francis Naughton** |  |

## 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

Francis Naughton

## 1. Interpreting Client Needs

Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?
* Are there any international transactions that the company produces?
* Are there governmental restrictions about secure communications to consider?
* What external threats might be present now and in the immediate future?
* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

Since Artemis Financial uses sensitive client data to form finance plans for their clients, a price cannot be put on secure communications between the clients and company. Artemis financial does preform financial transactions. One breach of security could mean the financial end for a company. Artemis Financial does also conduct international transactions. Clients from around the world will use the application to access their accounts and preform transactions with other clients. Strict policies will be needed to protect client from scammers.

Due to Artemis being a finance company, it is crucial that it follows all government rules and restrictions. There are many restrictions that the application must adhere to. To fall in line with the General Data Protection Regulation, we must ensure that the application protects all data being sent and received from clients. If the company were to get breached, depending on the circumstances the company could face company fines.

Outside attacks will continue to be an ongoing battle when it comes to software development. Whether after the client’s or the company’s personal data, an outside attacker can cause major issues for a company that needs to build trust with their clients. The software can be modernized and improved upon a couple different ways. Implementing a two-factor authentication will help fight against log in breaches. Restricting all communications to secure HTTPS requests is a smart way to transmit sensitive data.

## 2. Areas of Security

Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial software application. Justify your reasoning for why each area is relevant to the software application.

Input validation is relevant because user input should be restricted to only allow what is necessary to get through. Since this company uses RESTful API, API’s are very relevant to this application. Cryptography will be very important and essential to encrypt sensitive company and client data. Secure error handling is also very important in prevent some exploits like input validation. The code quality is very important throughout the entirety of the applications development. Using secure coding practices from the beginning is a great way to start preventing possible vulnerabilities early on. Lastly, encapsulation is so important in keeping the sensitive data protected and held together.

## 3. Manual Review

Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

Reviewing the code revealed a possible vulnerability in the CRUDController.java file. This file could easily expose inner components of the application if not addressed. In lines 13-15, if the value “business\_name” was passed through the method, the DocData database could become vulnerable. Another possible vulnerability lies withing the DocData.java file. The username and password are both root which makes it easy to carry out a brute force attack. This could have drastic consequences if an untrusted user were to gain access to the system.

## 4. Static Testing

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from dependency check report. Include the following:

1. The names or vulnerability codes of the known vulnerabilities
2. A brief description and recommended solutions provided by the dependency check report
3. Attribution (if any) that documents how this vulnerability has been identified or documented previously

* bcprov-jdk15on-1.46.jar -- several vulnerabilities, prevention by frequent updates
  + CVE-2013-1624
  + CVE-2015-6644 --can result in disclosure of information & allow malicious apps

to access private data in the database(Android, 2016)

* + CVE-2015-7940
  + 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
* Log4j-api-2.12.1.jar -- one vulnerability, prevented by frequent updates
  + CVE-2020-9488 --due to the improper validation of certificates, SMTPS

connections can be intercepted which can cause log messages to be leaked(dependency-check report)

* Snakeyaml-1.25.jar -- one vulnerability, needs an update
  + CVE-2017-18640 --allows entity expansion during load

operation(dependency-check report)

* Jackson-databind-2.10.2.jar -- one vulnerability, needs an update
  + CVE-2020-25649 --entity expansion is not secured properly, is vulnerable

to XML external entity attacks(dependency-check report)

* Tomcat-embed-core-9.0.30.jar -- several vulnerabilities, prevented by updating
  + CVE-2019-17569
  + CVE-2020-11996
  + CVE-2020-13934
  + CVE-2020-13935
  + CVE-2020-13943
  + CVE-2020-17527 --certain versions of Apache Tomcat can re-use an HTTP

request header value from the previous stream,

this can potentially leak information between requests

(dependency-check report)

* + CVE-2020-1935
  + CVE-2020-1938 --must be cautious when trusting incoming connections

to Apache Tomcat, the connection could be

exploited in many ways(dependency-check report)

* + CVE-2020-8022
  + CVE-2020-9484
  + CVE-2021-24122
  + CVE-2021-25122
  + CVE-2021-25329
  + CVE-2021-33037
* Hibernate-validator-6.0.18.Final.jar -- one vulnerability, prevent by updating
  + CVE-2020-10693 -- a bug that enables invalid EL expressions as if they

were valid, hackers could bypass the input validation

* spring-core-5.2.3.RELEASE.jar -- two vulnerabilities, prevent by updating
  + CVE-2020-5421
  + CVE-2021-22118
* Spring-jcl-5.2.3.RELEASE.jar -- one vulnerability, prevent by updating
  + CVE-2020-5421

## 5. Mitigation Plan

After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.

First off, the issue regarding username and password needs to be fixed. The username and password should be set securely with a combination of upper and lowercase letters as well as some numbers. This will protect against potential brute force attacks to get into the system. Next, I would fix the certificate validation issue by updating the Log4jAPI. To prevent XML entity expansion attacks YAML needs to be updated. Updating Jackson will also remove the one known vulnerability in the dependency-check. I would then update Apache Tomcat. This will clear the known vulnerabilities in the dependency. To improve upon the input validation, Hibernate’s Bean Validation will need to also be updated. The underlying operating systems should also be updated frequently. Keeping the dependencies and system up-to-date will mitigate many vulnerabilities that could normally put the company and its clients at risk.

**Sources:**

<https://source.android.com/security/bulletin/2016-01> 01#information\_disclosure\_vulnerability\_in\_bouncy\_castle