****

# Thomas Corrigan

# 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** | **11/21/2021** | **Thomas Corrigan** |  |

## 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

Thomas Corrigan

## 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?

The value of secure communications to Artemis Financial is that if someone finds any vulnerability in one of their applications it could compromise the entire system. Surely, the database at Artemis Financial contains at a minimum, employee and customer private information up to and including social security numbers which could wreak havoc on anyone involved. Even more troubling is the fact that they regularly have international transactions and occasionally a government could have restrictions about secure communications to consider so it is important to stay informed of the current state and local policies. One important process the company will need to have encryption to prevent the spying on the transfer of data. They will need to use secure protocols to accomplish this, and it is important for them to use open source technologies like the maven-dependency-check plugin to find any potential issues to get them fixed before they become serious. Artemis Financial has external threats in the form of vulnerabilities in their software code.

## 2. Areas of Security

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

Almost every area in the Vulnerability Assessment Process Flow Diagram is applicable to Artemis Financial’s software application. Input validation is the first relevant type of vulnerability from the vulnerability assessment process flow diagram because if an attacker can get past input sanitization then they will be able to make the system malfunction. Secure APIs are applicable because it is important to know if any dependencies are vulnerable to a bug which would extend down into their own software. Secure distributed composing is important because if any node in a network is convinced to operate improperly the entire network is compromised. Secure error handling is another part of the vulnerability process flow diagram which is relevant because the application does not have any type of error handling so these can cause unexpected consequences. Secure Coding practices is the next relevant type because the code needs to show consistency. One example of this consistency is that if dependencies are unused then they must be removed from the code. Running the Maven Dependency Plugin reflected four unused dependencies,

## 3. Manual Review

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

Manually reviewing the code of the client’s web application showed several more issues. For example, the unused dependencies that were shown by the unused dependency report are com.jayway.jsonpath:json-path:jar:2.4.0:test, org.bouncycastle:bcprov-jdk15on:jar:1.46:compile, org.springframework.boot:spring-boot-starter-web:jar:2.2.4.RELEASE:compile, org.springframework.boot:spring-boot-starter-test:jar:2.2.4.RELEASE:test. Another flaw that I noticed was that there was no error handling present. There is also no input sanitization when the greeting template is assembled. This could allow an attacker to send a specially crafted input to trick the system into giving them sensitive information.

## 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

The client’s web application has external threats present currently ina range from low to severe with most vulnerabilities high severety. The most critical vulnerability is the

CVE-2020-8022 bug which allows “attackers to escalate from tomcat to root” (National Institute of Standards and Technology, March 17, 2021) so the dependency tomcat-embed-core-9.0.30.jar is most at risk. It needs to be updated to 9.0.31 or later immediately. There are also several high severity vulnerabilities such as in the dependency snakeyaml-1.25.jar there is CVE-2017-18640 which used a memory leak in CompressedChunkReader to gain access to unauthorized resources. (dc…@apache.org, Oct 9, 2020). This was patched in the 1.26 version of snakeyaml. Also in Jackson-databind-2.10.2.jar, CVE-2020-25649, in which a flaw was found in FasterXML Jackson Databind dependencies (NetApp, August 24, 2021). This can be prevented with the updated version Jackson-databind-2.10.5.1. There are also medium level severity vulnerabilities such as CVE-2020-10693 in the dependency hibernate-validator-6.0.18.Final.jar which “allows invalid EL expressions to be evaluated as if they were valid” (National Institute of Standards and Technology, July 14, 2021) allowing for attackers to bypass input sanitization. This was resolved in version 6.0.20.Final (Neil Griffin, July 14, 2021). The dependency spring-jcl-5.2.3.RELEASE.jar has another medium level vulnerability called CVE-2020-5421. This attack allows “disclosure of sensitive information or addition or modification of data” (NetApp, July 13, 2021) so the spring framework must be updated to at least version 5.2.9. Spring-core-5.2.3.RELEASE.jar from the same package has another high severity vulnerability. Finally, a low severity vulnerability in the log4j-api-2.12.1.jar there is vulnerability CVE-2020-9488 where “improper validation of certificate with host mismatch in Apache Log4j SMTP appender allows an SMTPS connection to be intercepted by a man-in-the-middle attack (NetApp, July 13). Apache Log4j must be update to version 2.14.2.

## 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.

**After reviewing the source with static testing and the maven-dependency-check plugin, Several courses of action must be taken immediately.**

* Update TomCat to version 9.0.31
* Update SnakeyAML to version 1.26
* Update FasterXML Jackson Databind to version 2.10.5.1
* Update Hibernate-Validator to version 6.0.20.Final
* Update Spring Framework to version 5.2.9
* Update Apache Log4j to version 2.14.2
* Remove unnecessary dependencies
* Implement input sanitization
* Implement error handling

Sources:

National Institute of Standards and Technology. (March 17, 2021). *CVE-2020-8022 Detail.* National Vulnerablity Database. <https://nvd.nist.gov/vuln/detail/CVE-2020-8022>

dc…@apache.org. (Oct 9, 2020). *[cassandra] branch trunk updated: Upgrade to snakeyaml >= 1.26 version for CVE-2017-18640 fix.* Apache Pony Mail. <https://lists.apache.org/thread/b8bwgp04v43wos4zrvlp00tp1jcljp66>

NetApp. (August 24, 2021). *CVE-2020-25649 FasterXML Jackson Databind Vulnerability in NetApp Products.*

NetApp Product Security. <https://security.netapp.com/advisory/ntap-20210108-0007/>

National Institute of Standards and Technology. (July 14, 2021). *CVE-2020-10693 Detail.* National Vulnerablity Database. <https://nvd.nist.gov/vuln/detail/CVE-2020-10693>

Neil Griffin. (July 14, 2021). *[jira] [Closed] (PLUTO-791) Upgrade to hibernate-validator-6.0.20.Final due to CVE-2020-10693 and CVE-2019-10219.* Apache Pony Mail. <https://lists.apache.org/thread/6xr81qfr9gorl7ghs8dooffz7lnxlc7j>

NetApp. (July 13, 2021). *CVE-2020-5421 Spring Framework Vulnerabliity in NetApp Products.* NetApp Product Security. <https://security.netapp.com/advisory/ntap-20210513-0009/>

NetApp. (July 13, 2021). *CVE-2020-9488 Apache Log4j Vulnerability in NetApp Products.* NetApp Product Security. <https://security.netapp.com/advisory/ntap-20200504-0003/>