****

# 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** | **11/14/21** | **Ashli Campbell** |  |

## 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

Ashli Campbell

## 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?
  + Secure communications is a top priority for Global Rain because of the clients they serve and the sensitive data they handle. It is imperative for the company to establish trust with their customers and prevents data leaks and attacks that could ruin their reputation and business.
* Are there any international transactions that the company produces?
  + Global Rain deals with clients all over the world, as well as with government agencies.
* Are there governmental restrictions about secure communications to consider?
  + Governmental restrictions exist to protect the vast amount of sensitive data held by government agencies. There are data security laws in the US that vary by state, so the company should be aware of the states where they are doing business and the laws that govern data communication.
* What external threats might be present now and in the immediate future?
  + An attacker will be interested in hacking the assets that the company holds, such as credit card information, passwords, personally identifiable information, and more. There are constantly-evolving methods of attack that may be used, including privilege escalation, horizontal privilege escalation, abuse of business logic or workflow, and more.
* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?
  + Open source libraries are a good tool to maintain security because they allow the developer to manipulate the code. You can inspect the code and write tests against it, or you can modify it to fix bugs. Cloud computing is on the rise with an increased focus on cloud security, making them a safe alternative to traditional systems.

## 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.

* APIs: Secure API interactions
  + Secure Application Programming Interface, or API, prevents unexpected access to a system. The API acts as an intermediary between applications, never allowing a user’s data to be fully exposed to the server, and vice versa.
* Code Quality: Secure coding practice/patterns
  + Secure coding practices and patterns keep security vulnerabilities to a minimum and protect against common attacks. There are coding guidelines for different languages that should be adhered to no matter what a developer is working on.
* Input Validation: Secure input and representations
  + The best practice in accepting user input is never to trust it. Input validation is a fundamental defense that helps to prevent vulnerabilities and protect the system from things like injection attacks. For every piece of input, the system should allow only the proper data to be entered; for instance, telephone number fields should only accept numbers, name fields should only accept letters, etc.

## 3. Manual Review

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

* The code is missing input validation, leaving it exposed to attacks. Input validation can include a white list of allowed characters.
* HTTPS should be utilized when sharing sensitive data over the web.
* There isn’t an authentication system in place, which is used to verify users. This is especially relevant where the customer’s account number is being shown.

## 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
   1. Dependency: [bcprov-jdk15on-1.46.jar](#l2_991c96a4e31e6c19e2b9136c8955bd423f2d)
      1. The highest severity for **bcprov-jdk15on-1.46.jar** (Bouncy Castle Crypto package) is unknown, but the dependencyhas 16 published vulnerabilities: **CVE-2013-1624, CVE-2015-6644, 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**, and **CVE-2020-26939.** For all of these vulnerabilities, an update of Bouncy Castle to version 1.69 will resolve them.
   2. Dependency: [log4j-api-2.12.1.jar](#l5_a55e6d987f50a515c9260b0451b4fa217dc5)
      1. **log4j-api-2.12.1.jar**, had the published vulnerability **CVE-2020-9488**: “Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender.” The National Vulnerability Database listed a link to an [issue page on apache](https://issues.apache.org/jira/browse/LOG4J2-2819), which recommended the following mitigation: “Upgrade to 2.13.2 which supports this feature.” The severity of this CVE was low.
   3. Dependency: [snakeyaml-1.25.jar](#l8_8b6e01ef661d8378ae6dd7b511a7f2a33fae)
      1. Next, the dependency **snakeyaml-1.25.jar**: “The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564.” The published vulnerability was **CVE-2017-18640**, which is high severity, and the [mitigation technique I found](https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/PTVJC54XGX26UJVVYCXZ7D25X3R5T2G6/) through the CVE List is to update to a new version (version 1.26) to resolve the CVE.
   4. Dependency: [jackson-databind-2.10.2.jar](#l9_0528de95f198afafbcfb0c09d2e43b6e0ea6)
      1. **jackson-databind-2.10.2.jar**:“A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.” The high-severity CVE associated with this dependency is **CVE-2020-25649**, [which was fixed in versions 2.6.7.4, 2.9.10.7, 2.10.5.1, 2.11.0 and later.](https://github.com/FasterXML/jackson-databind/issues/2589)
   5. Dependency: [tomcat-embed-core-9.0.30.jar](#l13_ad32909314fe2ba02cec036434c0addd19b)
      1. **tomcat-embed-core-9.0.30.jar** and is associated with 17 different CVE’s, bringing the severity count to critical. The CVE’s are: **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**, and **CVE-2021-42340**. I found that one update addressed three of the associated vulnerabilities: [version 9.0.31 resolved](https://lists.opensuse.org/archives/list/security-announce@lists.opensuse.org/message/F3FOVKJAK2YR7UVBYBATR7JKLD5IA6WI/) **CVE-2019-17569, CVE-2020-1935**, and **CVE-2020-1938.** [Version 9.0.35 resolved](https://lists.opensuse.org/archives/list/security-announce@lists.opensuse.org/message/KEGFHXFDNNML32KIYPBCJAKPCF3Q6VJ6/) **CVE-2020-9484;** [version 9.0.36 resolved](https://lists.opensuse.org/archives/list/security-announce@lists.opensuse.org/message/3V7X3IWA53FRK7AHHX6TLLX6ZDE5CAEM/) **CVE-2020-11996.** Upgrading to version 10.0.0 resolves **CVE-2021-24122** and **CVE-2021-25329**. Because this list is extensive, I looked up the latest version of tomcat, which is version 10.0.11, released in September of 2021. I would likely just upgrade to this one and run the check again to see if there are any vulnerabilities. If there were, I would go through the resources on the CVE List and in the NVD to try to find patches until the next version with a fix for the vulnerabilities is released.
   6. Dependency: [hibernate-validator-6.0.18.Final.jar](#l16_7fd00bcd87e14b6ba66279282ef15efa30d)
      1. The medium-severity CVE associated with **hibernate-validator-6.0.18.Final.jar** is described here: “A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages.” **CVE-2020-10693** [is fixed in versions hibernate-validator 7.0.0.Alpha2, hibernate-validator 6.1.5.Final, hibernate-validator 6.0.20.Final](https://bugzilla.redhat.com/show_bug.cgi?id=CVE-2020-10693).
   7. Dependency: [spring-core-5.2.3.RELEASE.jar](#l19_3734223040040e8c3fecd5faa3ae8a1ed6d)
      1. Vulnerabilities for **spring-core-5.2.3.RELEASE.jar** are considered high-severity and include **CVE-2020-5421** and **CVE-2021-22118**. **CVE-2020-5421** is described here: “In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.” The recommendation is to upgrade the spring framework dependency; [releases that have fixed this issue include](https://tanzu.vmware.com/security/cve-2020-5421) 5.2.9, 5.1.18, 5.0.19 and 4.3.29. The description for **CVE-2021-22118** is: “In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data.” I found [the following recommendation](https://tanzu.vmware.com/security/cve-2021-22118) through the NVD: “5.3.x users should upgrade to 5.3.7. 5.2.x users should upgrade to 5.2.15. No other steps are necessary.”
   8. Dependency: [spring-jcl-5.2.3.RELEASE.jar](#l20_fcba3ae00724ee18f37aa49ac887a5fd4ad)
      1. The medium-severity published vulnerability for **spring-jcl-5.2.3.RELEASE.jar** is **CVE-2020-5421**, which is described in the dependency above.

## 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.

There were many security practices missing from the code. The company should implement the use of HTTPS, two-factor authentication, and input validation against a white list. All dependencies should be updated to the most current versions and additional dependency checks should be run to ensure the vulnerabilities are eliminated.