****

# 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** | **7/14/2022** | **Nicholas Ciesla** | **Initial** |

## 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

Nicholas Ciesla

## 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 are extremely valuable to any company looking to adapt their operations to include more technology. Secure communications help to mitigate potential data breaches and prevent hackers or bad actors from accessing information that doesn’t belong to them through vulnerabilities in your communications. If a financial company such as Artemis Financial experienced a data breach, it would be buried in litigation for years. Artemis Financial doesn’t want that, and neither do their customers. One could say that Artemis Financial owes this to their customers for trusting them with their finances. Mitigating vulnerabilities and avoiding data breaches ensures that the customers will continue to trust Artemis Financial for years to come.**

* Are there any international transactions that the company produces?

**Although it is not stated specifically in the provided information, I would assume that Artemis Financial has some international clients. Additionally, there will also be potential international investments taking place.**

* Are there governmental restrictions about secure communications to consider?

**In the United States, governmental restrictions tend to deal with customer information and credit card transactions rather than communications in general, and the legislation is quite vague in its wording. However, the EU has much more specific legislation regarding online communication in general. Despite most business likely being US based, it would be a good idea to cover all bases regarding restrictions globally.**

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

**External threats of the highest concern will be related to attempts on the customer’s private information, financial or otherwise. These will generally include attempts to skim data from HTTPS request s, modifying HTTPS requests maliciously, attempts at code injection to return information or escalate permissions, and prodding for errors that cause the site to fail open. The second type of attack would possibly be a DoS or Denial of Service attack with the goal of crashing the site.**

* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

**While the term “open source” may seem to be counter productive to security if the code is open for anyone to observe, it is better for security as the software community at large is free to investigate possible vulnerabilities. However, it is very important to keep the versions of these libraires up to date and to frequently run dependency checks to track vulnerabilities as they are discovered and act accordingly. Modernization is not a one and done operation, it is a daily responsibility.**

## 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.

* Input Validation:
  + **With one of the most common points of attack being abusing input for code injection, all customer input or untrusted data of any kind should be handled like nuclear waste. Especially considering the type of information being handled. Limits need to be put on what the user can input, and it should be checked before it is used anywhere in the application.**
* APIs
  + **Because this project uses several libraries, the version of these libraires and their dependencies must be kept up. To mitigate vulnerabilities and protect customer data, Artemis Financial must routinely run dependency checks to ensure all communications are free from any known issues with the versions of these libraires they are using to keep customer data as safe as possible.**
* Cryptography
  + **All customer information that is sent and stored must be encrypted, not only for it to be safer, but to comply with international and federal regulations regrading personal information in the financial sector.**
* Client/Server
  + **When it comes to client server communications, Artemis Financial must do their due diligence to ensure that anyone who intercepts their HTTP communications cannot fabricate a request or obtain valuable information from the request to guarantee the safety of their customer’s data and the security of their account.**
* Code Error/Code Quality
  + **Because of the sensitive nature of Artemis Financial’s business, it is imperative that the underlying code is free from errors and well written, and that any errors that do occur and handled gracefully. This is primarily to avoid what is known as a “fail open” situation where an error allows a malicious user to access something without proper authentication or access.**

## 3. Manual Review

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

**Initially, by looking at the POM file, I can see that many of the dependent libraries are using out of date versions, these will likely have many known vulnerabilities and will need to be updated to the latest stable versions. CRUD.java, CRUDController.java, Greeting.java, and GreetingController.java all use untrusted data without any input validation which needs to be addressed. Customer.java sends private customer information without any type of encryption; an attacker would easily be able to steal this information. Artemis Financial will need both input validation and encryption to ensure that their customer data is protected adequately. The database can be accessed directly through the URL using a username and password that could easily be cracked considering that they are hardcoded as “root” for each. This information could easily be leaked or found through probing by attackers.**

**It is difficult to comment further on this project as it is very early in development and in a state meant for testing, but there are still plenty of issues to be addressed.**

## 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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dependency** | **Dependency Desc** | **Vulnerability ID** | **Vulnerability Desc** | **Recommended Action** |
| bcprov-jdk15on-1.46.jar | The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7. | CVE-2016-1000338 | In Bouncy Castle JCE Provider version 1.55 and earlier the DSA does not fully validate ASN.1 encoding of signature on verification. It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' data into a signed structure. | UPDATE TO CURRENT VERSION |
| CVE-2016-1000343 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DSA key pair generator generates a weak private key if used with default values. If the JCA key pair generator is not explicitly initialised with DSA parameters, 1.55 and earlier generates a private value assuming a 1024 bit key size. In earlier releases this can be dealt with by explicitly passing parameters to the key pair generator. | UPDATE TO CURRENT VERSION |
| CVE-2016-1000341 | In the Bouncy Castle JCE Provider version 1.55 and earlier DSA signature generation is vulnerable to timing attack. Where timings can be closely observed for the generation of signatures, the lack of blinding in 1.55, or earlier, may allow an attacker to gain information about the signature's k value and ultimately the private value as well. | UPDATE TO CURRENT VERSION |
| CVE-2016-1000345 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES/ECIES CBC mode vulnerable to padding oracle attack. For BC 1.55 and older, in an environment where timings can be easily observed, it is possible with enough observations to identify when the decryption is failing due to padding. | UPDATE TO CURRENT VERSION |
| CVE-2017-13098 | BouncyCastle TLS prior to version 1.0.3, when configured to use the JCE (Java Cryptography Extension) for cryptographic functions, provides a weak Bleichenbacher oracle when any TLS cipher suite using RSA key exchange is negotiated. An attacker can recover the private key from a vulnerable application. This vulnerability is referred to as "ROBOT." | UPDATE TO CURRENT VERSION/SUPPRESS not applicable to used version. |
| CVE-2020-15522 | Bouncy Castle BC Java before 1.66, BC C# .NET before 1.8.7, BC-FJA before 1.0.1.2, 1.0.2.1, and BC-FNA before 1.0.1.1 have a timing issue within the EC math library that can expose information about the private key when an attacker is able to observe timing information for the generation of multiple deterministic ECDSA signatures. | UPDATE TO CURRENT VERSION |
| **CVE-2020-0187** | In engineSetMode of BaseBlockCipher.java, there is a possible incorrect cryptographic algorithm chosen due to an incomplete comparison. AndroidVersions: Android-10Android ID: A-148517383 | UPDATE TO CURRENT VERSION/ SUPPRESS, not using android. |
| CVE-2016-1000339 | In the Bouncy Castle JCE Provider version 1.55 and earlier the primary engine class used for AES was AESFastEngine. Due to the highly table driven approach used in the algorithm it turns out that if the data channel on the CPU can be monitored the lookup table accesses are sufficient to leak information on the AES key being used. | UPDATE TO CURRENT VERSION |
| CVE-2020-26939 | In Legion of the Bouncy Castle BC before 1.61 and BC-FJA before 1.0.1.2, attackers can obtain sensitive information about a private exponent because of Observable Differences in Behavior to Error Inputs. | UPDATE TO CURRENT VERSION |
| CVE-2015-7940 | The Bouncy Castle Java library before 1.51 does not validate a point is withing the elliptic curve, which makes it easier for remote attackers to obtain private keys. "invalid curve attack." | UPDATE TO CURRENT VERSION |
| CVE-2018-5382 | The default BKS keystore use an HMAC that is only 16 bits long, which can allow an attacker to compromise the integrity of a BKS keystore. | UPDATE TO CURRENT VERSION |
| CVE-2013-1624 | The TLS implementation in the Bouncy Castle Java library before 1.48 and C# library before 1.8 does not properly consider timing side-channel attacks on a noncompliant MAC check operation during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets | UPDATE TO CURRENT VERSION |
| CVE-2016-1000346 | In the Bouncy Castle JCE Provider version 1.55 and earlier the other party DH public key is not fully validated. | UPDATE TO CURRENT VERSION |
| CVE-2015-6644 | Bouncy Castle in Android… | UPDATE TO CURRENT VERSION/SUPPRESS, not using android. |
| hibernate-validator-6.0.18.Final.jar | Hibernate's Bean Validation (JSR-380) reference implementation. | CVE-2020-10693 | Bug in message interpolation processor allows invalid EL to be evaluated as valid allowing attackers to bypass input sanitation. | UPGRADE TO CURRENT VERSION or pass user input as an expression variable by unwrapping the context to HibernateConstraint  ValidatorContext |
| jackson-databind-2.10.2.jar | General data-binding functionality for Jackson: works on core streaming API | CVE-2020-25649 | FasterXML Jackson Databind, where it did not have entity expansion secured properly. vulnerability to (XXE) attacks | UPGRADE TO CURRENT VERSION |
| CVE-2020-36518 | allows a Java StackOverflow exception and denial of service via a large depth of nested objects | UPGRADE TO CURRENT VERSION |
| jakarta.annotation-api-1.3.5.jar | Jakarta Annotations API | CVE-2022-31569 | The RipudamanKaushikDal/projects repository through 2022-04-03 on GitHub allows absolute path traversal because the Flask send\_file function is used unsafely. | Do not use code from this project in implementation/edit to use send\_file in a secure way. |
| log4j-api-2.12.1.jar | The Apache Log4j API | 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. | UPGRADE TO CURRENT VERSION |
| logback-core-1.2.3.jar | logback-core module | CVE-2021-42550 | In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers. | UPGRADE TO CURRENT VERSION |
| snakeyaml-1.25.jar | YAML 1.1 parser and emitter for Java | CVE-2017-18640 | The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation | Suppress, using a later version. |
| spring-boot-2.2.4.RELEASE.jar | Spring Boot | CVE-2022-27772 | This vulnerability only affects products and/or versions that are no longer supported by the maintainer. | UPGRADE TO CURRENT VERSION |
| spring-core-5.2.3.RELEASE.jar | Spring Core | CVE-2022-22965 | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | UPGRADE TO CURRENT VERSION |
| CVE-2021-22118 | WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory | UPGRADE TO CURRENT VERSION |
| CVE-2020-5421 | 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. | UPGRADE TO CURRENT VERSION |
| CVE-2022-22950 | older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition. | UPGRADE TO CURRENT VERSION |
| CVE-2022-22971 | old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user. | UPGRADE TO CURRENT VERSION |
| CVE-2022-22968 | older unsupported versions, the patterns for disallowedFields on a DataBinder are case sensitive which means a field is not effectively protected unless it is listed with both upper and lower case for the first character of the field | UPGRADE TO CURRENT VERSION |
| CVE-2022-22970 | old unsupported versions, applications that handle file uploads are vulnerable to DoS attack if they rely on data binding to set a MultipartFile or javax.servlet | UPGRADE TO CURRENT VERSION |
| CVE-2021-22060 | older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. follow-up to CVE-2021-22096 | UPGRADE TO CURRENT VERSION |
| CVE-2021-22096 | older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. | UPGRADE TO CURRENT VERSION |
| spring-web-5.2.3.RELEASE.jar | Spring Web | CVE-2016-1000027 | through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data. Depending on how the library is implemented within a product, this issue may or not occur, and authentication may be required. | UPGRADE TO CURRENT VERSION |
| tomcat-embed-core-9.0.30.jar | Core Tomcat implementation | CVE-2020-1938 | When using the Apache JServ Protocol (AJP)… | SUPPRESS, NOT USING JServ Protocol |
| CVE-2020-11996 | A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat. could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive. | UPGRADE TO CURRENT VERSION |
| CVE-2020-13934 | An h2c direct connection to Apache Tomcat. did not release the HTTP/1.1 processor after the upgrade to HTTP/2. If a sufficient number of such requests were made, an OutOfMemoryException could occur leading to a denial of service. | SUPPRESS, NOT USING h2c direct connection |
| CVE-2020-13935 | The payload length in a WebSocket frame was not correctly validated in Apache Tomcat. Multiple requests with invalid payload lengths could lead to a denial of service. | UPGRADE TO CURRENT VERSION |
| CVE-2020-17527 | While investigating bug 64830 it was discovered that Apache Tomcat could re-use an HTTP request header value from the previous stream received on an HTTP/2 connection for the request associated with the subsequent stream. While this would most likely lead to an error and the closure of the HTTP/2 connection, it is possible that information could leak between requests. | UPGRADE TO CURRENT VERSION |
| CVE-2021-25122 | When responding to new h2c connection requests, Apache Tomcat versions could duplicate request headers and a limited amount of request body from one request to another meaning user A and user B could both see the results of user A's request. | UPGRADE TO CURRENT VERSION |
| CVE-2021-41079 | Apache Tomcat When Tomcat was configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS, a specially crafted packet could be used to trigger an infinite loop resulting in a denial of service. | UPGRADE TO CURRENT VERSION |
| CVE-2022-29885 | The documentation of Apache Tomcat for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. | ADD FURTHER ENCRYPTION TO SENSITIVE COMMUNICATIONS/ UPGRADE TO CURRENT VERSION |
| CVE-2020-9484 | When using Apache Tomcat the attacker knows the relative file path from the storage location used by FileStore to the file the attacker has control over; then, using a specifically crafted request, the attacker will be able to trigger remote code execution via deserialization of the file under their control. | UPGRADE TO CURRENT VERSION |
| CVE-2021-25329 | The fix for CVE-2020-9484 was incomplete. | UPGRADE TO CURRENT VERSION |
| CVE-2021-30640 | A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid user name and/or to bypass some of the protection provided by the LockOut Realm | UPGRADE TO CURRENT VERSION |
| CVE-2022-34305 | In Apache Tomcat the Form authentication example in the examples web application displayed user provided data without filtering, exposing a XSS vulnerability. | UPGRADE TO CURRENT VERSION WHEN RELEASD AND REMOVE ANY EXAMPLE OF FORM AUTHENTICATION AS DEMONSTRATED IN THE APACHE TOMACAT SECURITY EXAMPLES |
| CVE-2021-24122 | Apache Tomcat were susceptible to JSP source code disclosure in some configurations. The root cause was the unexpected behaviour of the JRE API File.getCanonicalPath() which in turn was caused by the inconsistent behaviour of the Windows API (FindFirstFileW) in some circumstances. | UPGRADE TO CURRENT VERSION |
| CVE-2021-33037 | Apache Tomcat did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. Specifically: - Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response | UPGRADE TO CURRENT VERSION |
| CVE-2019-17569 | The refactoring present in Apache Tomcat introduced a regression. The result of the regression was that invalid Transfer-Encoding headers were incorrectly processed leading to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely. | UPGRADE TO CURRENT VERSION |
| CVE-2020-1935 | Apache Tomcat the HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. This led to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. | UPGRADE TO CURRENT VERSION |
| CVE-2020-13943 | If an HTTP/2 client connecting to Apache Tomcat exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. This could lead to users seeing responses for unexpected resources. | UPGRADE TO CURRENT VERSION |
| tomcat-embed-websocket-9.0.30.jar | Core Tomcat implementation | CVE-2020-8022 | A Incorrect Default Permissions vulnerability in the packaging of tomcat on certain Linux/OpenStax versions. | UPGRADE TO CURRENT VERSION |

## 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.

**This Dependency check contained many false positives. While these vulnerabilities are important to remember, it would be helpful to suppress such vulnerabilities with the use of a suppression file to clear up the feed. Artemis Financial would want to suppress codes that they will never encounter based on the nature of their application and implementation of the APIs. This will allow them to focus on the vulnerabilities that are most important to their specific application.**

**The overwhelming majority of relevant vulnerabilities could all be fixed by updating the version of the API being used. It would be beneficial for Artemis Financial , to update all Dependencies to their latest versions and run the dependency check again to be safest. This is true even if the current application does not utilize features of the API that are affected by the vulnerability. It is always better safe than sorry with these types of things. Generally, the longer a software version is out, the more vulnerabilities will be found.**

**Some specific vulnerabilities that can be addressed are CVE-2022-34305 and CVE-2022-31569 that deal with Tomcat and Jakarta respectively. These are both instances of sample projects containing security vulnerabilities. This is why it is so important to run these static tests. If Artemis Financial had used the techniques from these projects, they may have been open to attack.**