****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

**Victoria Kaloudis**

**Southern New Hampshire University**

**CS-305-T5606**

**Dr. Vivian Lyon**

**May 21, 2022**

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 10](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **21May2022** | **Victoria Kaloudis** | **Initial vulnerability findings** |

## 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

Victoria Kaloudis

## 1. Interpreting Client Needs

Artemis Financial is looking to update its software to the most current and adequate software security. Software security is critical to the client. Artemis Financial is a financial consulting company that works with their customers’ earnings to create financial, retirement and investment plans, and help provide insurance to them. Having secure software is imperative to keeping their customers. As this company is working with various customers, there is the potential for global clientele that would lead to international transactions. As international transactions would need to consider currency exchange rates and import and export laws, there is required importance needed for knowledge of international laws in general. With international laws comes various new threats to security and unknown vulnerabilities. Researching those vulnerabilities is crucial. There is always the possibility of new viruses attacking the hardware to gain access to private information. Adding more levels of security to the software will help with modernizing it. Overprepared is better than underprepared.

## 2. Areas of Security

* Input validation: Input validation will play a significant role in ensuring Artemis Financials remains secure. A couple of vulnerabilities listed come from improper input validation, where unauthorized users can access the software when validating external inputs. Essentially, as input from untrusted sources is validated before use, unauthorized users can craft malicious attacks to cause problems (Oracle, 2022). Ensuring the system validates only trusted input will be essential and could save Artemis Financials a lot of time and money.
* API: Artemis Financials use a RESTful API which will help with providing a level of protection to their client’s data. Ensuring the API is accessing outside the network safely is imperative. With many of the vulnerabilities listed in the dependency check involving APIs, having secure ways to communicate between at least two different applications will ensure Artemis Financials’ clients’ private information is safe.
* Cryptography: A high level of encryption is needed to ensure Artemis Financials’ clients are safe. A few of the vulnerabilities found in this report are due to cryptography API and extension issues. Finding ways to safeguard the information transferred remains encrypted and secure will be necessary for Artemis Financials to stay in business.
* Client/Server: As security is tightened up within Artemis Financials’ software, we move on to client/server vulnerability. The best way to handle this is to ensure all information, communication, transaction, etc., are encrypted. Within the dependency check report, many vulnerabilities could render the server unresponsive. Another thing to consider is a denial of service (DoS) attacks. Unauthorized users could potentially spam the server with overloaded information where the clients cannot access the server and, ultimately, access private information. Guaranteeing encryption here will be best.
* Code Quality: Secure coding is fundamental to maintaining the integrity of Artemis Financials’ software. Many of the vulnerabilities found in the dependency check are caused by code that misuses things or accesses the wrong things. If sanitation, serialization, and deserialization are attacked within the code, it is no longer secure. Writing secure code will be key for Artemis Financials’ customers.

## 3. Manual Review

* In the pom.xml file, line 8, the version of Spring Framework book listed is 2.2.4. The latest version of Spring Boot is 2.6.6. Therefore, an update to the version in the code is needed.
  + Graphical user interface, text, application

    Description automatically generated
* In the pom.xml file, line 30, the version of Bouncy Castle listed is 1.46. The latest version of Bouncy Castle is 1.71. Therefore, an update to the version in the code is needed.
  + Graphical user interface, text, application

    Description automatically generated
* In the CRUD Controller Java file, on lines 13-16, the CRUD class calls the business name as the string. In other java files, it references an account number. I would recommend switching this to an account number to maintain security. This also does not provide any security as far as input validation goes, and it could allow unauthorized users to gain access to the system through code injection.
  + A screenshot of a computer

    Description automatically generated
* In the DocData java file, on lines 25-31, the connection is called, showing the username and password as roots. This is a vulnerability as the default is set to root, where any unauthorized user could gain access to the software to change anything in the code or gain private information. Essentially, the unauthorized user has access to modify any file if they so choose. This issue does not seem to be with Tomcat but with openSUSE.
  + Graphical user interface, text, application

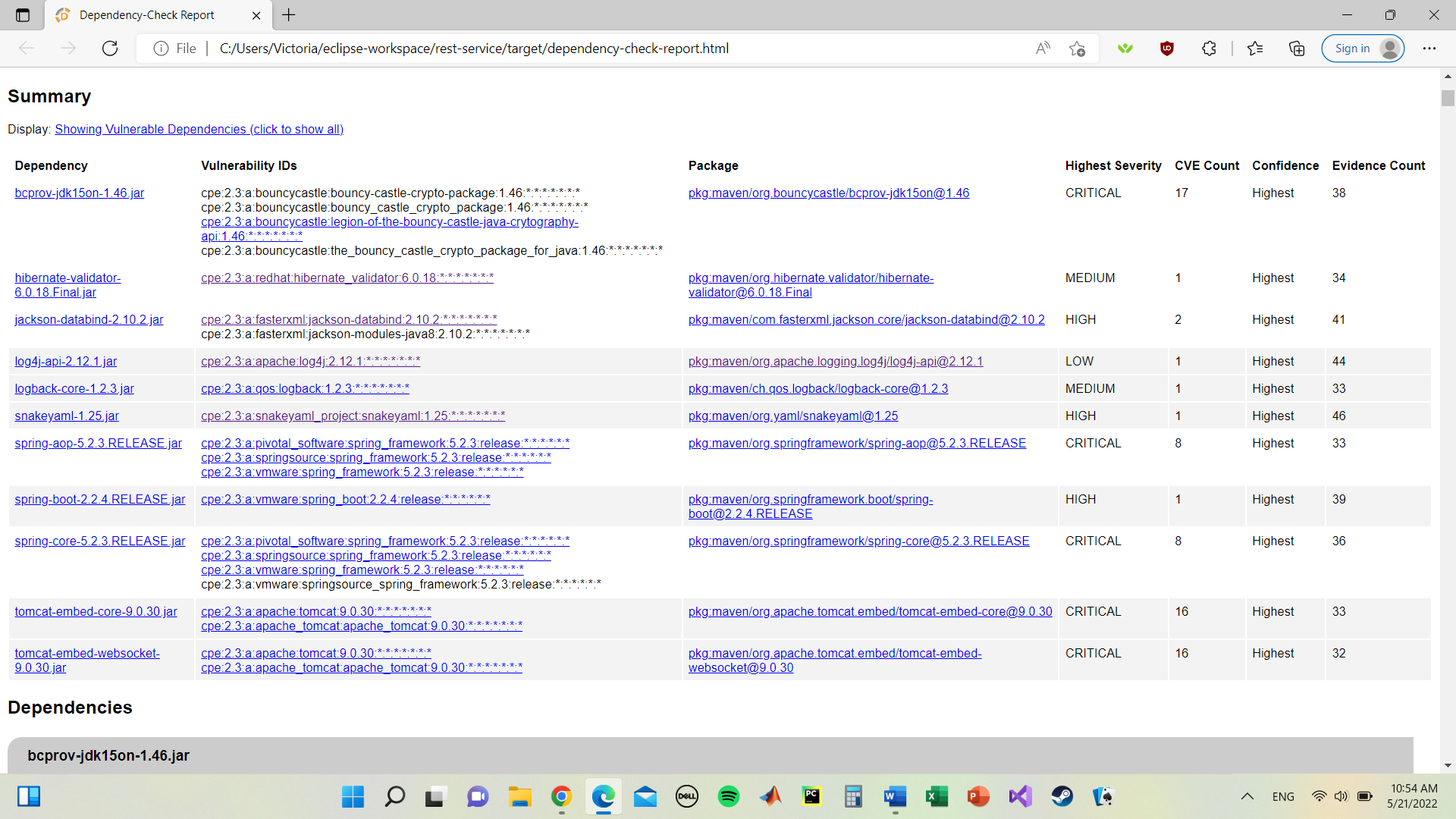
    Description automatically generated
* In the GreetingController Java file, after lines 15-17, no authenticator or parser is used to determine who is accessing the system. A parser could take in the data and create a message from the string. For example, when someone signs in, the system should be able to validate the user. An API here would be valuable to help secure this code. Limiting the size of the string is a good idea as unauthorized users can create a string with malicious code gaining access to the system.
  + A screenshot of a computer

    Description automatically generated

## 4. Static Testing

Graphical user interface, text, application, email

Description automatically generated



Dependencies:

* Bcprov-jdk15on-1.46.jar
  + Description: “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” (National Institute of Standards and Technology [NIST], n.d.).
  + CVE-2018-10000613 – has now been fixed
    - This vulnerability could reconstruct the data structure or objects to execute unexpected code through code injection.
      * Was fixed in version 1.60 and later (NIST, n.d.).
  + CVE-2015-6644 – for android users
    - This was a disclosure vulnerability that could allow application to gain access to android user’s private information (NIST, n.d.).
      * Updating to the latest software version should fix the issue.
  + CVE-2016-1000338 – currently undergoing reanalysis by NVD
    - There could be a fake signature verification where someone could inject code and the system would still validate the signature (NIST, n.d.).
  + CVE-2016-1000342 – currently undergoing reanalysis by NVD
    - There is a possibility to introduce malicious data into a signature verification and still have the system validate it as an accepted signature (NIST, n.d.).
  + CVE-2016-1000343 – currently undergoing reanalysis by NVD
    - The DSA key pair generate assigns a private value assuming a 1024 bit key size (NIST, n.d.).
  + CVE-2016-1000344 – currently undergoing reanalysis by NVD
    - This implements ECB which is regarded as unsafe (NIST, n.d.).
  + CVE-2016-1000352 – currently undergoing reanalysis by NVD
    - This implements ECIES which is regarded as unsafe (NIST, n.d.).
  + CVE-2016-1000341 – currently undergoing reanalysis by NVD
    - Timing attacks can give intruders information on the signature’s k value and private value (NIST, n.d.)
  + CVE-2016-1000345 – currently undergoing reanalysis by NVD
    - DHIES/ECIES CBC modes are vulnerable to padding attacks where an intruder can break the encryption (NIST, n.d.)
  + CVE-2017-13098 – currently undergoing reanalysis by NVD
    - An intruder can gain access to the TLS server with use of an RSA encryption (NIST, n.d.).
  + CVE-2020-15522 – currently undergoing reanalysis by NVD
    - An intruder is able to gain access to the private key through a timing issue, which will then allow the intruder to break the encryption (NIST, n.d.).
  + CVE-2016-1000339 – currently undergoing reanalysis by NVD
    - The AES key can cause leaks of information if the data channel on the CPU can be monitored (NIST, n.d.)
  + CVE-2020-26939 – has now been fixed
    - An unauthorized user can gain access to the private portion of the RSA and give them access to the encrypted data (NIST, n.d.)
  + CVE-2015-7940 – currently undergoing reanalysis by NVD
    - Unauthorized users can easily perform an “invalid curve attack” to gain access to private keys and ultimately gain access to encrypted data (NIST, n.d.).
  + CVE-2018-5382 – has now been fixed
    - The BKS keystore is compromised with the default HMAC length of 16 bits (NIST, n.d.).
  + CVE-2013-1624 – currently undergoing reanalysis by NVD
    - Unauthorized users can perform attacks from analyzing timing data when the TLS is implemented (NIST, n.d.).
  + CVE-2016-1000346 – currently undergoing reanalysis by NVD
    - The DH public key is not fully validated and unauthorized users can gain access to the private key when the user is using DH (NIST, n.d.)
* Hibernate-validator-6.0.18.Final.jar
  + Description: “Hibernate’s Bean Validation (JSR-380) reference implementation” (NIST, n.d.).
  + CVE-2020-10693 – improper input validation
    - Invalid EL expressions are seen as valid which can cause issues with the sanitation controls put in place (NIST, n.d.).
* Jackson-databind-2.10.2.jar
  + Description: “General data-binding functionality for Jackson: works on core streaming API” (NIST, n.d.).
  + CVE-2020-25649 – no longer supported by Java 8
    - The entity expansion is not secured properly which allows external entity attacks that can affect data integrity (NIST, n.d.)
  + CVE-2020-36518 – currently undergoing reanalysis by NVD
    - Previous software version allowed exception through a large amount of nested object (NIST, n.d.)
* Log4j-api-2.12.1.jar
  + Description: “The Apache Log4j API” (NIST, n.d.).
  + CVE-2020-9488 – has now been fixed
    - Leakages could be sent through an appender which mismatched in Apache Log4j SMTP (NIST, n.d.)
* Logback-core-1.2.3.jar
  + Description: “logback-core module” (NIST, n.d.).
  + CVE-2021-42550 – has now been fixed
    - Malicious configurations can cause reconstructing of the object and execute code (NIST, n.d.)
* Snakeyaml-1.25.jar
  + Description: “YAML 1.1 parser and emitter for Java” (NIST, n.d.).
  + CVE-2017-18640 – has now been fixed
    - As a load operation is called, there could be an expansion of data causing many reoccurring entities to be parsed and overload the system (NIST, n.d.)
* Spring-aop-5.2.3.RELEASE.jar
  + Description: “Spring AOP” (NIST, n.d.).
  + CVE-2016-1000027 – no fix for this vulnerability
    - There is a potential RCE issue when the data structure or object is reconstructed with invalid data. This vulnerability may or may not occur and currently there is no fix (NIST, n.d.).
  + CVE-2022-22965 – has now been fixed
    - When Tomcat is run as a WAR deployment, unauthorized users can use data binding to RCE (NIST, n.d.).
  + CVE-2021-22118 – has now been fixed
    - In previous Spring Framework versions, privilege escalation allowed unauthorized users to access data uploaded to WebFlux (NIST, n.d.)
  + CVE-2020-5421 – currently undergoing reanalysis by NVD
    - In previous versions of the software, unauthorized users can perform RFD attacks by bypassing the browser, just depending on the particular browser (NIST, n.d.).
  + CVE-2022-22950 – has now been fixed
    - In previous versions of the software, unauthorized users can cause a denial of service condition through the SpEL expression (NIST, n.d.).
  + CVE-2022-22968 – has now been fixed
    - In previous versions of the Spring Framework software, case sensitivity causes the vulnerability of the nested fields unless it is listed with both capital and lower case characters (NIST, n.d.).
  + CVE-2021-22060 – follow up to CVE-2021-22096 - has now been fixed
    - In previous versions of the Spring Framework software, an unauthorized user can inject malicious code inserting additional log entries (NIST, n.d.).
  + CVE-2021-22096 – has now been fixed
    - In previous versions of the Spring Framework software, an unauthorized user can inject malicious code inserting additional log entries (NIST, n.d.).
* Spring-boot-2.2.4.RELEASE.jar
  + Description: “Spring Boot” (NIST, n.d.).
  + CVE-2022-27772 – has now been fixed
    - Early versions of this software made it easy for unauthorized users to hack into the spring framework boot web server, giving them access to private information (NIST, n.d.)
* Spring-core-5.2.3.RELEASE.jar
  + Description: “Spring Core” (NIST, n.d.) – refer to Spring-aop-5.2.3.RELEASE.jar for CVE information
  + CVE-2016-1000027
  + CVE-2022-22965
  + CVE-2021-22118
  + CVE-2020-5421
  + CVE-2022-22950
  + CVE-2022-22968
  + CVE-2021-22060
  + CVE-2021-22096
* Tomcat-embed-core.9.0.30.jar
  + Description: “Core Tomcat implementation” (NIST, n.d.).
  + CVE-2020-1938 – has now been fixed
    - AJP connections should be disabled if not in use as these connections are not as secure as HTTP connections (NIST, n.d.).
  + CVE-2020-11996 – has now been fixed
    - The server could become unresponsive due to high CPU usage caused by HTTP/2 requests (NIST, n.d.)
  + CVE-2020-13934 – has now been fixed
    - A lack of memory could be caused by HTTP/1.1 processor not releasing after the upgrade to HTTP/2 (NIST, n.d.).
  + CVE-2020-13935 – has now been fixed
    - An infinite loop could be triggered by invalid payload lengths giving unauthorized users “authorized” access to the network (NIST, n.d.)
  + CVE-2020-17527 – has now been fixed
    - The possible re-use of an HTTP request header value could lead to the closure of the HTTP/2 connection or a possible leakage of information (NIST, n.d.).
  + CVE-2021-25122 – has now been fixed
    - Sensitive information could be exposed to an unauthorized user when duplicate request headers are used (NIST, n.d.).
  + CVE-2021-41079 – has now been fixed
    - Previous versions of Apache Tomcat software could be configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS where an unauthorized user could trigger an infinite loop giving them access to the network (NIST, n.d.).
  + CVE-2021-42340 – occurred after a fix for 63362 - has now been fixed
    - Unauthorized users could access the network via a memory leak through the HTTP upgrades connections through WebSocket (NIST, n.d.).
  + CVE-2020-9484 – still under analysis but has a fix listed
    - When four conditions are met an unauthorized user can directly trigger the code to execute by reconstructing the data structure for access (NIST, n.d.).
  + CVE-2021-25329 – relations to CVE-2020-9484 - has now been fixed
    - Meant to support the fix for CVE-2020-9484 however it is still vulnerable to CVE-2020-9494 (NIST, n.d.).
  + CVE-2021-30640 – has now been fixed
    - Previous software versions of Apache Tomcat saw a vulnerability in the JNDI Realm where unauthorized users could bypass LockOut Realm giving them access to the network (NIST, n.d.).
  + CVE-2021-24122 – has now been fixed
    - Unauthorized users could access the data resources and other private information protected in the NTFS file system (NIST, n.d.).
  + CVE-2021-33037 – has now been fixed
    - Unauthorized users could use a reverse proxy to bypass the security response of the HTTP transfer-encoding process (NIST, n.d.).
  + CVE-2019-17569 – has now been fixed
    - Unauthorized users could bypass security responses via a reverse proxy due to a regression (NIST, n.d.).
  + CVE-2020-1935 – has now been fixed
    - The HTTP parser used parsed certain headers as valid when they were actually invalid (NIST, n.d.).
  + CVE-2020-13943 – has now been fixed
    - Users could see responses for unexpected resources due to overloading the maximum number of streams for the connection (NIST, n.d.).
* Tomcat-embed-websocket-9.0.30.jar
  + Description: “Core Tomcat Implementation” (NIST, n.d.) – refer to Tomcat-embed-core.9.0.30.jar for CVE information for those not elaborated upon.
  + CVE-2020-1938
  + CVE-2020-8022– has now been fixed
    - Unauthorized users can modify the root service file if the default is not already set (NIST, n.d.).
  + CVE-2020-11996
  + CVE-2020-13934
  + CVE-2020-13935
  + CVE-2020-17527
  + CVE-2021-25122
  + CVE-2021-41079
  + CVE-2020-9484
  + CVE-2021-25329
  + CVE-2021-30640
  + CVE-2021-24122
  + CVE-2021-33037
  + CVE-2019-17569
  + CVE-2020-1935
  + CVE-2020-13943

## 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 Financials’ software application.

* An update to the latest version of Spring Boot and Bouncy Castle will help mitigate the incorrect version used in the pom.xml file.
* In the CRUD Controller Java file, I would use an account number. An account number is harder to guess compared to a business name. Also, the account number should have numbers and letters to make it more difficult for unauthorized users to guess.
* In the DocData java file, there is a security issue with openSUSE. Updating to the latest software version of openSUSE or SUSE should fix this issue. There is no update needed for Tomcat as this issue was with SUSE (NIST, n.d.).
* In the GreetingController Java file, there is no parser listed in the code to review any string or even an HTTP request. This could allow unauthorized users to access the system through request smuggling. Updating the latest version of Apache Tomcat should help fix the issue with the HTTP request (NIST, n.d.). Adding code to parse a string in this java file would also benefit the overall security.

Works Cited

*GitHub Advisory Database*. (2022). GitHub. Retrieved May 21, 2022, from https://github.com/advisories

National Institute of Standards and Technology (NIST). (n.d.). *NVD - Home*. NIST - NVD. Retrieved May 21, 2022, from https://nvd.nist.gov/

Oftedel, E. (2022). *Retire.js*. Retire.Js. Retrieved May 21, 2022, from <https://retirejs.github.io/retire.js/>

Oracle. (2022, January). *Secure Coding Guidelines for Java SE*. Retrieved May 21, 2022, from https://www.oracle.com/java/technologies/javase/seccodeguide.html

Sonatype. (2018). *Sonatype OSS Index*. Retrieved May 21, 2022, from https://ossindex.sonatype.org/