****

# 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** | **05/17/2023** | **Afahri Kerr** | **Initial Evaluation** |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also choose to include images or supporting materials. If you include them, make certain to insert them in all the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Afahri Kerr

## Interpreting Client Needs

Artemis Financial wants to modernize their operations. As a crucial part of the success of their custom software, they also want to use the most current and effective software security. Artemis Financial has a RESTful web application programming interface (API).

1. What is the value of secure communications to the company?

Since this company deals with sensitive personal financial information, secure communications will be very important.

1. Does the company make any international transactions?

It has not been specified if this is the case or not but if it is then we must comply with the foreign country’s laws as well as ours.

1. Are there governmental restrictions about secure communications to consider?

I have not located any government regulations on secure communications at this time but that does not mean that it will not come up in the near future.

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

Since the company deals with sensitive financial and personal information there is a good chance that it will be attacked.

1. What are the modernization requirements that you must consider? For example:
   1. The role of open-source libraries

Open-source libraries are a security risk in themselves because they allow everyone to see the underlying code and manipulate it. This alone presents many threats.

* 1. Evolving web application technologies

Technology is advancing at quick pace so we should try to keep up with current practices but not at the expense of security. These new technologies often have faults when they are newly released that can be exposed by hackers.

## Areas of Security

The areas of security to address are Input Validation, API’s, Cryptography, Code Error, and Code Quality

We need input validation since we are asking user to input information such as in CRUDController.java

They plan to use REST API so we need to make sure that there are no major security flaws when we begin to build the program.

Cryptography to encrypt the sensitive financial data and information that is being delivered to the clients.

Code Error handling to make sure that incorrect data or attempts to access the software are reported and logged for evaluation.

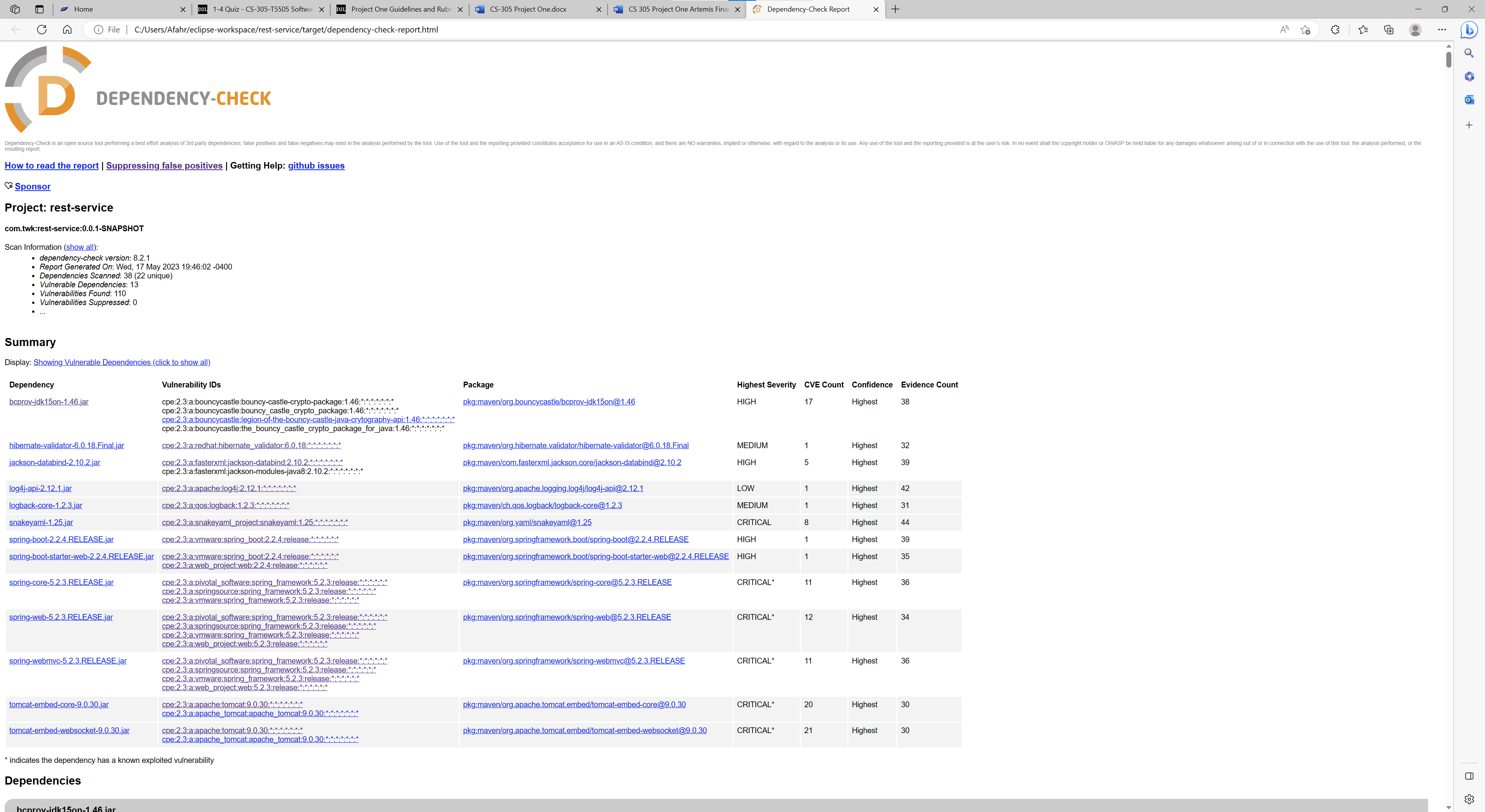
Code Quality to make sure that best practices are used to avoid common threats such as SQL injection or DDoS attacks.

## Manual Review

In the program is currently incomplete but it looks like it will allow the user in input data into the CRUDController and GreetingController files. These should be validated (input validation) and be parameterized (code quality) to ensure the system is not under attack. It should also return an error (error handling) if bad data is entered. The only place that currently has any error handling is in the

DocData file and it needs to be updated. I have not found any encryption code (Cryptography) in the program yet so this will also need to be added to ensure secure communications. The API currently does not have any way to interact with it.

## Static Testing



## Mitigation Plan

1. bcprov-jdk15on-1.46.jar

In the Bouncy Castle JCE Provider version 1.55 and earlier the ECIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider.

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.

FIX: Update to later version than 1.56 to allow validation

1. hibernate-validator-6.0.18.Final.jar

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.

Fix:

1. jackson-databind-2.10.2.jar

In FasterXML jackson-databind before 2.13.4, resource exhaustion can occur because of a lack of a check in BeanDeserializer.\_deserializeFromArray to prevent use of deeply nested arrays. An application is vulnerable only with certain customized choices for deserialization.

Fix: Update to later version than 2.13.4 to add check

1. log4j-api-2.12.1.jar

Apache Log4j2 versions 2.0-beta7 through 2.17.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable to a remote code execution (RCE) attack when a configuration uses a JDBC Appender with a JNDI LDAP data source URI when an attacker has control of the target LDAP server. This issue is fixed by limiting JNDI data source names to the java protocol in Log4j2 versions 2.17.1, 2.12.4, and 2.3.2.

Fix: Update to newer versions and implement parameterized input and validation

1. logback-core-1.2.3.jar

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.

Fix: update to newer version and use best coding practices to make sure attacker doesn’t acquire priveleges

1. snakeyaml-1.25.jar

SnakeYaml's Constructor() class does not restrict types which can be instantiated during deserialization. Deserializing yaml content provided by an attacker can lead to remote code execution. We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. We recommend upgrading to version 2.0 and beyond.

Fix: Upgrade to newer version and use input validation to ensure input isn’t malicious.

1. spring-boot-2.2.4.RELEASE.jar

spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.

Fix: Upgrade to newer version

1. spring-boot-starter-web-2.2.4.RELEASE.jar

spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer

Fix: Upgrade to newer version

1. spring-core-5.2.3.RELEASE.jar

In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.

Fix: upgrade to newer version and provide input validation to protect against DoS

1. spring-web-5.2.3.RELEASE.jar

In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.

Fix: upgrade to newer version and provide input validation to protect against DoS

1. spring-webmvc-5.2.3.RELEASE.jar

In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.

Fix: upgrade to newer version and provide input validation to protect against DoS

1. tomcat-embed-core-9.0.30.jar

When using the RemoteIpFilter with requests received from a reverse proxy via HTTP that include the X-Forwarded-Proto header set to https, session cookies created by Apache Tomcat 11.0.0-M1 to 11.0.0.-M2, 10.1.0-M1 to 10.1.5, 9.0.0-M1 to 9.0.71 and 8.5.0 to 8.5.85 did not include the secure attribute. This could result in the user agent transmitting the session cookie over an insecure channel.

If Apache Tomcat 8.5.0 to 8.5.82, 9.0.0-M1 to 9.0.67, 10.0.0-M1 to 10.0.26 or 10.1.0-M1 to 10.1.0 was configured to ignore invalid HTTP headers via setting rejectIllegalHeader to false (the default for 8.5.x only), Tomcat did not reject a request containing an invalid Content-Length header making a request smuggling attack possible if Tomcat was located behind a reverse proxy that also failed to reject the request with the invalid header.

If Apache Tomcat 8.5.0 to 8.5.82, 9.0.0-M1 to 9.0.67, 10.0.0-M1 to 10.0.26 or 10.1.0-M1 to 10.1.0 was configured to ignore invalid HTTP headers via setting rejectIllegalHeader to false (the default for 8.5.x only), Tomcat did not reject a request containing an invalid Content-Length header making a request smuggling attack possible if Tomcat was located behind a reverse proxy that also failed to reject the request with the invalid header.

Fix: upgrade to newer version and use best coding practices and input validation to protect against attacks.