****

# Practices for Secure Software Report

Table of Contents

[Document Revision History 3](#_Toc102040754)

[Client 3](#_Toc102040755)

[Instructions 3](#_Toc102040756)

[Developer 4](#_Toc102040757)

[1. Algorithm Cipher 4](#_Toc102040758)

[2. Certificate Generation 4](#_Toc102040759)

[3. Deploy Cipher 4](#_Toc102040760)

[4. Secure Communications 4](#_Toc102040761)

[5. Secondary Testing 4](#_Toc102040762)

[6. Functional Testing 4](#_Toc102040763)

[7. Summary 4](#_Toc102040764)

[8. Industry Standard Best Practices 4](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **6.18.24** | **Tyler Rape** |  |

## Client



## Instructions

Submit this completed practices for secure software report. Replace the bracketed text with the relevant information. You must document your process for writing secure communications and refactoring code that complies with software security testing protocols.

* Respond to the 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 Two Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Tyler Rape

## Algorithm Cipher

Artemis Financial is seeking to modernize their operations. They wish to add a file verification step to their web application to ensure secure communications. For this project, SHA-256 has been utilized. Touted as being nearly optimal collision resistance, SHA-256 is a cryptographic hash algorithm (Synnada, 2024). Secure Hash Algorithms were first published by the National Security Agency (NSA) in 2001 (The NSA and Bitcoin, 2024). Cryptographic algorithms utilize random numbers in order to help cipher the data they are securing. SHA is considered symmetric as it allows not only encryption but also decryption. Asymmetric decryption relies on public and private keys to cipher and decipher data. Collision avoidance is important because hackers can utilize collisions to steal or misuse data. While there are higher bit levels (384 and 512) SHA-256 offers optimal security while leveraging performance. Higher bits require more resources and can make the program run slower. User experience should always be considered when implementing security. In conclusion, SHA-256 offers an optimal level of security while offering the best experience.

## Certificate Generation

Insert a screenshot below of the CER file.

A computer screen with white text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

## Secure Communications

Insert a screenshot below of the web browser that shows a secure webpage.

A screenshot of a computer

Description automatically generated

## Secondary Testing

Insert screenshots below of the refactored code executed without errors and the dependency-check report.

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

Insert a screenshot below of the refactored code executed without errors.

A screenshot of a computer program

Description automatically generated

Pom file: Updated Spring boot to 3.3.0, Java to 9 (Spring would not work otherwise), and Maven to 9.2.0.

A screenshot of a computer

Description automatically generated

In the application.properties file, I had to update the “????” with the above. I am working on a Mac. I had to move the keystore file to the system resources for it to work.

A screen shot of a computer program

Description automatically generated

In the SslServerApplication file, I had to add the above code.

## Summary

For this project, a full code review has taken place. Specifically, the Vulnerability Assessment Process Flow was utilized and the areas of APIs, Cryptography, Client/Server were addressed. This application relies on RESTful APIs. It was using an outdated form of Spring Boot and Maven. For these, both were updated to the most recent releases. To be able to utilize the most recent version of Spring Boot (3.3.0), we had to update to Java 9. This likely could have used an even more current version of Java but 9 was the minimum requirement. Prior to updating the application, there over 100 vulnerabilities. Just by updating the Spring Boot version, we were able to clear all of these vulnerabilities, as demonstrated by the Dependency Check. A requirement of this project was to incorporate cryptography. To satisfy Artemis Financial’s need, we incorporated a cipher. This was accomplished through SHA-256. This application also relies on a client/server interaction. We created a self-signed certificate authority to secure the connection.

## Industry Standard Best Practices

Security should be considered at every level of a software’s lifecycle. From development to maintenance, it should always be a concern. Above, we spoke about updating to Spring Boot 3.3.0, which eliminated over 100 vulnerabilities. This is why keeping applications updated is one of the best security practices. Known vulnerabilities are quickly negated by updates. Hence why there are so many versions of a given application or product. Staying current keeps an application from being vulnerable to known threats. We also updated Maven to the most recent version. Which continues the previous idea. The value of adding industry best standards to a company’s overall well-being cannot be quantified. To be known as an organization with a data leak could be lethal to that company. To a customer, this loss could come in the form of financial loss, or even worse their information being compromised. Consumer confidence in a company could be the difference between its downfall or its success. When a company has great data security, along with a great user experience, their possibility for growth is limitless.

References:

*Synnada | AI-native Data Infrastructure*. (n.d.). Www.synnada.ai. Retrieved June 16, 2024, from https://www.synnada.ai/glossary/collision-resistance#:~:text=Cryptographic%20hash%20algorithms%20like%20SHA

*The NSA and Bitcoin: Origins of the SHA-256 Hashing Algorithm*. (2024, March 25). Https://Supra.com/. https://supra.com/academy/the-nsa-and-bitcoin-origins-of-the-sha-256-hashing-algorithm/