****

# 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** | **[8/25/2024]** | **[Eskinder]** |  |

## 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

[Eskinder Kassahun]

## Algorithm Cipher

In this project, SHA-256 is the algorithm used for hashing to keep data integrity. SHA-256 is the hash function employed here, which given any input provides us a 256-bit (32-byte) unique fixed-length hash. The hash allows to use the data as a sort of fingerprint and check whether it were altered or not. SHA-256 is different than traditional encryption, because it's what known as a one-way function, you can't go backwards and get the original text from the hash. It can verify the authenticity of sensitive information in one shot. SHA-256 is hash function, it does not require encryption keys but can be “salted” to increase the security against brute force attacks by introducing random data ("salts") before hashing. SHA-256, created by the NSA and now widely used for applications like securing passwords with a hash or verifying files though another checksum often accompanied digitally detached (but typically insecure) signature. SHA-256: This project, with its strong security and the trust it enjoys by many people across borders is safe to say for SHA-256 being an ideal option in ensuring integrity of data.

## Certificate Generation

A screenshot of a computer

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

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

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The functional testing phase confirmed that the refactored code executed successfully, with the Spring Boot application starting correctly and the secure communication setup functioning as intended. Accessing the /hash endpoint via HTTPS returned the expected data, validating both the hash function and secure communication. These successful tests, along with the absence of security vulnerabilities during static analysis, demonstrate that the refactored code meets security standards and complies with the required security testing protocols.

## Summary

The code in this project was refactored and enhanced for security improvement, while addressing vulnerabilities found using a vulnerability assessment process. The findings resulted in problems such as encrypted messages, lack of verification on data consistency & correctness, and missing input validation. All evaluated vulnerabilities for potential severity, with unencrypted communications flagged as a high-risk concern. According to the risk assessment, industry-standard security best practices were implemented for addressing these vulnerabilities, secured communications, and encrypted verification systems integrity.

Measures taken for security included ensuring SSL/TLS encryption to secure communications between the client and server (self-signed certificate created using Java Keytool) running on port 8446 over HTTPS. SHA-256 cryptographic hashing was used for data integrity, input validation, and sanitization were strengthened. Dependency-Check, a tool from OWASP, was used to detect vulnerabilities in third-party libraries. These security enhancements were substantiated through functional testing results and static analysis, where the app fulfilled security test cases and addressed risks identified.

## Industry Standard Best Practices

This project was carried out by adhering to secure coding best practices, which are industry standards, in order to address common security vulnerabilities and protect the application. Important best practices were implementing input validation and sanitization to prevent things like SQL injection or cross-site scripting (XSS) attacks. It also featured data transmission using SSL/TLS encryption for securing sensitive information. We also made use of the OWASP Dependency-Check tool to check for vulnerabilities in third-party libraries, via static code analysis, ensuring no external dependencies introduce security risks into the system.

Following these guidelines reduces the chance of a security breach, potentially resulting in data loss, legal penalties, and damage to reputation. Taking a proactive stance on security issues, the company protects itself not only from potential fines for violating industry standards but also retains customer confidence in the brand. These efforts secure the future of our company and its software products, with minimal risk to assets.