****

# 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 5](#_Toc102040763)

[7. Summary 6](#_Toc102040764)

[8. Industry Standard Best Practices 6](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **04/17/2023** | **Chayne Pieri** | **Add Algorithm Cipher, Certificate Generation, Deploy Cipher, Secure Communications, Secondary Testing, Functional Testing, Summary, and Industry Standard Best Practices responses.** |

## 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

Chayne Pieri

## Algorithm Cipher

Advanced Encryption Standard, or AES, is a very common encryption algorithm that can be utilized with varying security levels based on your needs. It is recommended to use AES as it provides the developer with strong defenses against data leaks. A collision means two inputs will have the same output when hashed. The more bits there are within an algorithm, the less likely a collision is to happen. Because AES-256 has a 256-bit value, there are 2^256 values that the key can contain, the possibility of a collision is one in 2^256, which is very improbable. As newer algorithms become standard practice, the security of our data becomes stronger; this, however, requires that our processing standards include upgraded hardware to compensate for the more strenuous complexity of the ciphers.

## Certificate Generation

Insert a screenshot below of the CER file.

Text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

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



## Secondary Testing

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

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Functional Testing

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

A picture containing graphical user interface

Description automatically generated

## Summary

To increase application security, I implemented input validation to prevent injection attacks, secure API interactions to prevent Denial-of-Service attacks, and cryptography for end-to-end encryption to ensure data is not compromised. Secure error handling is used to prevent unexpected crashes, and encapsulation is utilized to ensure unintended access to private class fields. Continuous code review was also implemented as vulnerabilities were eliminated to verify of any issues that were created along the way.

## Industry Standard Best Practices

I have refactored the code to improve security flaw mitigation. A self-signed certificate and keystore were added to allow a secured address (https) to be used instead of a standard address (http). Maven was upgraded to version 8.2.1 to include newer known vulnerabilities. Vulnerability suppression was also used to exclude vulnerabilities from dependency checks if they cannot yet be modified. Using these best practices helps prepare your application for known attacks and unintended occurrences. As the life of the application continues, it is always recommended to check your dependencies and code for any new vulnerabilities that have been identified.