****

# 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** | **04/15/2024** | **Drew Payton** |  |

## 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

Drew Payton

## Algorithm Cipher

The algorithm I chose was SHA-256 hash algorithm, SHA-256 helps makes the SSL/TLS certificates more secure by creating a digital signature that people’s devices can verify. The way it works is by processing the information in 512-bit blocks after padding it to the correct length requirements. The algorithm uses a series of bitwise operations, additions, and rotations in 64 rounds to change the input to 256-bit hash value. It is deterministic though, meaning the same input will always make the same output, and its statistically unreversible or to find the same two inputs that make the same hash, which makes it very secure. All these properties lead to SHA-256 being the most widely used hashing algorithm to date.

## Certificate Generation

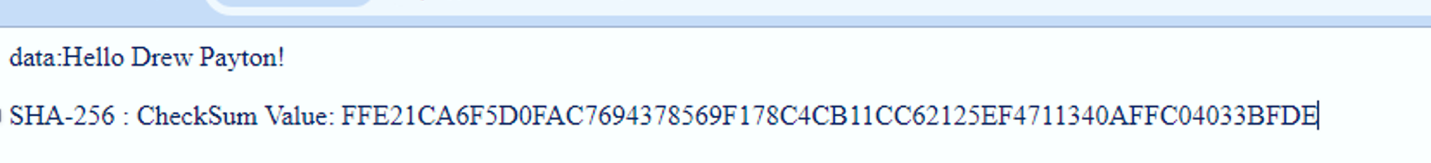
Insert a screenshot below of the CER file.

A computer screen with white text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.



## 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 computer screen shot of a program

Description automatically generated

## Functional Testing

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

[Insert screenshots here.] A screenshot of a computer screen

Description automatically generated

A screenshot of a computer error

Description automatically generated

## Summary

By refactoring code, I updated the package dependencies and edited the applications.properties to pass in the keystore file for a self-signed certificate. Then adding a securityconfig file to override the HTTP to make it HTTPS. Added cryptography from the VAPF chart by adding a SHA-256 Hashing algorithm.

## Industry Standard Best Practices

Implementing industry-standard best practices for secure coding, by using things like OWASP dependency checks, and updating dependencies is crucial for maintaining the security of software applications. These practices will help protect the company's information, including sensitive data, from a means of cyber-attacks. When you build trust through secure measures, you can reduce the of things like security incidents and data breaches, which in turn enhances the business overall. Secure coding practices contribute vastly to the overall wellbeing and success of the company.