****

# 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** | **2/26/2025** | **Westley Hunter** |  |

## 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

Westley Hunter

## Algorithm Cipher

When looking into exactly what SHA-256 means, we can find that it is a Secure Hash Algorithm that will create a 256-bit hash. The reason I chose the SHA-256 algorithm is because it was recommended by the FIPS Pub 180-4 created by the National Institute of Standards and Technology. Additionally, the SHA-256 algorithm is secure and it is computationally infeasible to determine the message from the hash or find a collision. The purpose of the hash function is to securely mask the data and ensure it has not been altered or manipulated in any way. This is achieved by comparing received hash with newly generated hash with all the same parameters. If received hash does not match the new hash, then it was manipulated somewhere on its journey. Hash functions and bit level go hand in hand to ensure the strongest level of security. The more bits used when hashing, the more secure the program. Additionally, a greater bit size prevents collisions (where two different messages are hashed to the same value). A critical aspect of cryptography and creating secure encryptions is the use of random numbers. However, while some numbers may appear to be random, they actually are not and generate the same sequence of numbers upon start-up. However, implementing the SecureRandom class provides the application and code with an unpredictable sequence of random numbers. Symmetric cryptography uses the same key to encrypt and decrypt the information. Therefore, the key must remain secret as its leakage would give attackers complete access to the data. Asymmetric cryptography uses two keys: a public and private key. The private key is kept secret and signs the data while the public key is used to validate the signature. The private key is kept secret but the public key can be known by anyone. Encryption can be seen as far back as 60 BC when Roman Leader Julius Caeser used a simple algorithm that shifted letters by a couple places and encoded the message. However, the growth of data needing protection and its’ subsequent amount of attackers, increased knowledge in mathematical fields, and the widespread availability and strength in computing power, encryption algorithms have changed drastically. Modern encryption algorithms like ECC, AES, or RSA securely encrypt data in minimal time. However, the brand new field of quantum computing posses many significant challenges for the field of cryptography. The computing power that quantum computing can produce may result in the RSA 2048 algorithm being broken by 2035.

## Certificate Generation

Insert a screenshot below of the CER file.

A computer screen shot of a black screen

AI-generated content may be incorrect.

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

AI-generated content may be incorrect.

## Secure Communications

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

A screenshot of a computer

AI-generated content may be incorrect.

## Secondary Testing

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

A screenshot of a computer

AI-generated content may be incorrect.

## Functional Testing

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

A screen shot of a computer program

AI-generated content may be incorrect.

## Summary

Through refactoring the code and employing security testing protocols, the code now addresses vulnerabilities in secure API interactions, cryptography, client/server, and code quality. Changing the protocol to HTTPS ensures for secure API interactions and a better client/server interface. Additionally, using the SHA256 for text encryption and within the certificate generation and checksum processes.

## Industry Standard Best Practices

Applying the OWASP dependency checker is a great way to check your code against known vulnerabilities and find industry standard solutions for said vulnerabilities. Additionally, using the HTTPS protocol is a good industry standard to apply as it is significantly more secure than the traditional HTTP protocol and has been tested to ensure its integrity. Using industry standard best practices for coding is a great way to create, development, and maintain solid code that has been tested again and again. Testing, identifying vulnerabilities, and finding and implementing solutions in code takes a significant time investment. The timeline of most projects do not have the time nor money to spare to ensure that the code is time-tested. However, using industry standard practices provides coding patterns and practices that have already been tested and tried by a multitude of other developers.