****

# 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** | **06/16/2023** | **Carmen Kingery** | **Updated security** |

## 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

Carmen Kingery

## Algorithm Cipher

I would recommend using the Advanced Encryption Standard (AES) as the encryption algorithm cipher. AES is a symmetric encryption algorithm widely adopted for its security, efficiency, and flexibility.

*Overview of AES:*

AES is a symmetric block cipher that operates on fixed-size blocks of data. It uses a series of transformations, including substitution, permutation, and mixing operations, to encrypt and decrypt data. AES has three key sizes: AES-128, AES-192, and AES-256, indicating the number of bits in the encryption key.

*Hash functions and bit levels:*

AES does not directly involve hash functions. Instead, it relies on a combination of substitution, permutation, and mixing operations to provide encryption. The bit levels of AES depend on the key size used: AES-128 operates on 128-bit blocks, AES-192 on 192-bit blocks, and AES-256 on 256-bit blocks.

*Use of random numbers and symmetric vs. non-symmetric keys:*

AES primarily uses symmetric key encryption, which means the same key is used for both encryption and decryption. Random numbers are used to generate the encryption key. In AES, the key is shared between the sender and the receiver, ensuring that both parties can encrypt and decrypt the data securely. Non-symmetric key algorithms, such as RSA, involve a pair of keys: a public key for encryption and a private key for decryption. However, for most practical use cases in data encryption, symmetric key algorithms like AES are more efficient and commonly employed.

*History and current state of encryption algorithms:*

Historically, encryption algorithms have evolved to meet the increasing need for secure communication and data protection. Some notable encryption algorithms include Data Encryption Standard (DES), Triple DES (3DES), and RSA. DES, once widely used, is now considered relatively weak due to its short key length. AES was established as the new standard in 2001, replacing DES. It was selected through a competition process and is widely accepted as a secure encryption algorithm. AES has been extensively studied, tested, and analyzed by experts worldwide, making it a trusted choice for encryption in various industries. In recent years, there has been ongoing research and development of new encryption algorithms to address emerging threats and advances in computing power. However, AES remains the most widely deployed and recommended encryption algorithm for most applications due to its security, efficiency, and established track record.

## Certificate Generation

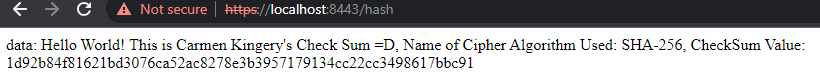
Insert a screenshot below of the CER file.

A screenshot of a certificate

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 with low confidence

## Secondary Testing

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

A picture containing text, screenshot, font

Description automatically generatedRefactored Code

A picture containing screenshot, text, line, font

Description automatically generatedSuccessful Build

Dependency Check Before Refactoring

A screenshot of a computer

Description automatically generated with medium confidence

Dependency Check After Refactoring

A screenshot of a computer

Description automatically generated with low confidence

## Functional Testing

A screen shot of a computer program

Description automatically generated with low confidenceInsert a screenshot below of the refactored code executed without errors.

A picture containing screenshot, text, line, font

Description automatically generated

## Summary

The refactored code incorporates the use of the SHA-256 algorithm for hashing the data. SHA-256 is a widely used cryptographic hash function, offering a high level of collision resistance. The top priority of Artemis Financial was to add a data verification step to their web application. By adding this cryptographic hash function, we can now check and verify that the sensitive data being transferred over the internet is correct and has not been tampered with.

The main areas of security we focused on in the refactoring of this application include Cryptography, Code Errors, and Code Quality. We included a new encryption method to their application to enable the verification of transferred data, and we checked to make sure this new function did not introduce any new dependency vulnerabilities. We also made sure our code handled any and all exceptions thrown in a safe and productive manner, and we produced quality code by following secure coding practices and patterns.

## Industry Standard Best Practices

We ensured industry standard best practices were followed by making sure we did not introduce any new vulnerabilities while implementing the cryptographic functions. We followed secure coding practices and patterns, and on top of that we ran Maven dependency checks before and after refactoring the code to make sure we did not add dependencies with known vulnerabilities.

Applying industry-standard best practices for secure coding provides a range of benefits that contribute to a company's overall wellbeing. It helps protect against data breaches, ensures regulatory compliance, fosters customer trust, minimizes financial losses, reduces development costs, prevents service disruptions, and safeguards a company's reputation in the market. By prioritizing security from the outset, companies can build a robust and resilient foundation for their software applications, enhancing their overall security posture and mitigating potential risks.