****

# 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** | **06/19/2024** | **Mitchell Fontaine** | **First Release** |

## Client



## Instructions

Submit these completed practices for a 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

Mitchell Fontaine

## Algorithm Cipher

When defending against various types of security attacks there are a few security protection best practices to consider. The first would be to use a secure key management system so we can generate, distribute, and store encryption keys. Artemis will also need to implement a backup that is secure and have a system to recover data in case it is lost. They will also need to update and patch their encryption software and systems regularly. Finally, they will need to ensure that only authorized personnel are given access to encrypted data (Fontaine, 2024).

Since Artemis Financial is in the finance sector and handles large amounts of sensitive PII, I would recommend they use AES, Advanced Encryption Standard, for their encryption needs. An AES algorithm with a 256-bit key size is considered the most secure and efficient algorithm and is widely accepted and trusted by industry professionals. AES is also compliant with NIST standards, FIPS 140-2, and meets the requirements of PCI-DSS and GDPR, which are required for financial data protection. There are a few risks with using AES like side-channel attacks and brute-force attacks, but if Artemis uses 256-bit keys then brute-force attacks become much harder to accomplish (N-Able, 2024). If Artemis doesn’t regularly update its security software/systems or uses weak passwords, then it may be more vulnerable to attacks (Fontaine, 2024).

So why should we use AES? It is widely trusted and accepted. AES uses hash functions that provide data integrity and authenticity by taking data and transforming it into an unreadable cipher. There are various bit levels that can be used based on Artemis’ needs which can be increased up to 256-bit to provide a high level of security. The larger the bit size the harder it is for brute-force attacks to occur (N-Able, 2024). Random numbers are used to create encryption keys to make sure they are unique and less susceptible to guesses (Boyle, 2018). It also uses symmetric keys which help provide better performance and efficiency by using the same key for encrypting and decrypting. Non-symmetric keys use a public-private key for encryption and a private key for decryption which is more secure but less efficient (Cyware, 2020) (Fontaine, 2024).

The history of encryption goes back as far as ancient Mesopotamia and Greece. Spartans would send secret messages during battles through the use of a scytale. Julius Ceasar even created the famous Ceasar Cipher which shifts characters in the alphabet by three places (A Brief History of Encryption (and Cryptography), 2023). The current state of encryption algorithms has drastically improved thanks to the creation of computers. There are two encryption algorithms commonly used today which are Symmetric Key Algorithms and Asymmetric Key Algorithms (non-symmetric) (Cyware, 2020). With quantum computers being developed there has been a growing concern with how to protect against them. NIST announced in 2022 that they created the first four quantum-resistant algorithms which is a huge step in ensuring we remain secure in a quantum world (Boutin, 2022) (Fontaine, 2024).

References

A brief history of encryption (and cryptography). (2023, February 1). Thales Group. <https://www.thalesgroup.com/en/markets/digital-identity-and-security/magazine/brief-history-encryption>

Boutin, C. (2022, July 7). NIST Announces First Four Quantum-Resistant Cryptographic Algorithms. NIST. <https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms>

Boyle, M. S. T. E. B. B. J. M. K. K. a. M. M. L. B. M. (2018, November 10). Recommendation for the entropy sources used for random bit generation. NIST. <https://www.nist.gov/publications/recommendation-entropy-sources-used-random-bit-generation>

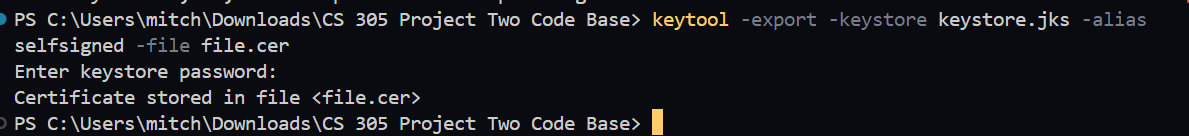
Cyware. (2020, February 22). Exploring the differences between symmetric and asymmetric encryption. *Cyware Labs*. <https://cyware.com/news/exploring-the-differences-between-symmetric-and-asymmetric-encryption-8de86e8a>

Fontaine, M. (2024, May 30). Algorithm Ciphers. SNHU CS-305, 4-2 Written Assignment.

N-Able. (2024, March 4). Understanding AES 256 encryption. N-able. <https://www.n-able.com/blog/aes-256-encryption-algorithm>

## Certificate Generation

Insert a screenshot below of the CER file.



A screenshot of a computer

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer program

Description automatically generated

## 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.

I refactored my code with what I previously used in module 5’s checksum assignment. It looks better to me but contains the same vulnerabilities, with most being fixable by updating dependencies.

A screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

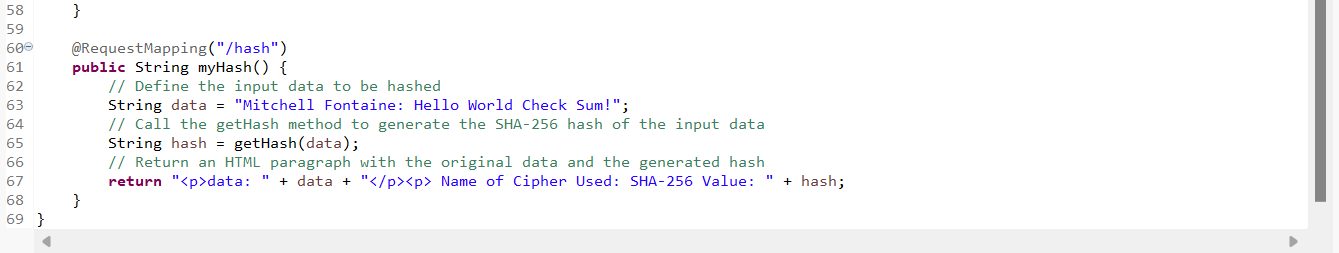
Description automatically generated

## Functional Testing

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

A screenshot of a computer code

Description automatically generated



## Summary

Sha-256 was implemented to generate a checksum for our data. I configured security settings and used the dependency check tool to identify potential vulnerabilities. By refactoring the code to enable SSL/TLS, implementing SHA-256 hashing, configuring the security settings, and conducting static testing, the code complies as best as I can get it with security testing protocols.

## Industry Standard Best Practices

I used HTTPS and SHA-256 hashing, which are industry-standard secure protocols. I used the dependency-check tool to identify and address potential vulnerabilities in dependencies. I followed secure coding practices, such as input validation and error handling. The value of applying industry standard best practices for secure coding to the company’s overall well-being is immeasurable. Secure coding practices protect sensitive data from unauthorized access, ensuring the confidentiality, integrity, and availability of data. It prevents financial losses due to cyber-attacks, data breaches, and other security incidents. It also increases the public's trust in companies using these practices, which makes them more popular.