****

# 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** | **10/15/2023** | **Jarrid Kamphenkel** |  |

## Client



## Developer

Jarrid Kamphenkel

## Algorithm Cipher

After analyzing the security vulnerabilities inherent in Artemis Financial's current system and considering the sensitive nature of financial data, the encryption algorithm I recommend deploying is the Advanced Encryption Standard (AES).

The Advanced Encryption Standard, commonly referred to as AES, stands as a hallmark in the realm of encryption. Introduced as the encryption gold standard by the U.S. government, AES operates as a block cipher, encrypting and decrypting data in 128-bit blocks. The design offers unparalleled security, making it a prime choice for entities like Artemis Financial that handle a vast amount of sensitive data.

One of the standout attributes of AES is its versatility. It supports three distinct key lengths: 128, 192, and 256 bits. While each increment in bit length amplifies the strength of the encryption, it simultaneously demands more computational power. To strike a balance between efficiency and security, deploying AES-256, with its 256-bit key length, seems fitting for a financial powerhouse like Artemis Financial. Furthermore, the data integrity is reinforced when paired with the cryptographic hash algorithm SHA-256, producing a steadfast 256-bit hash representation of input data.

The cryptographic world contains things called "nonces" or numbers utilized singularly. In AES's context, these random numbers are pivotal in formulating an "Initialization Vector" (IV), ensuring that encrypted outputs vary even with identical data inputs. This variability is a silent guardian against potential breaches. Also, deliberating between symmetric and asymmetric keys is a battle between speed and distribution safety. While symmetric encryption, as seen in AES, employs an identical key for both encryption and decryption phases, its asymmetric counterpart utilizes a dual key system: a public key for encryption and a secretive private key for decryption. For an institution like Artemis Financial, where swift data processing is paramount, AES's symmetric encryption is the clear winner in my book, promising speed without compromising security.

Tracing back to its roots, AES has been referred to as the encryption standard by the National Institute of Standards and Technology (NIST) in the early 2000s. Its rise to prominence was no accident; it was chosen to succeed the older, and increasingly vulnerable, Data Encryption Standard (DES). Winning a global competition aimed at identifying a robust successor to DES, AES has, over the years, solidified its reputation, becoming the go-to standard for diverse applications, from safeguarding resting data to encrypting transient online data. Its resilience is evident, remaining undefeated and secure against all practical onslaughts.

## Certificate Generation

A screenshot of a computer

Description automatically generated

## Deploy Cipher & Secure Communications

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.

## Functional Testing

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Summary

In the initial phase of the project, a Spring Boot application was presented with basic functionalities and no defined endpoint to carry out hash computations. The subsequent refactoring led to the development of a /hash endpoint which not only aligns with the project requirements but also guarantees a SHA-256 checksum of a pre-established string.

From a security perspective, the application leverages Spring Boot and runs over HTTPS, ensuring encrypted communications. Nevertheless, for bolstered security:

SSL/TLS configurations should be robust with the implementation of solid ciphers.

The application should be designed such that it avoids exposing sensitive details through error messages or logs.

It's crucial to ensure all application dependencies are contemporary and free from recognized vulnerabilities.

## Industry Standard Best Practices

Staying up to date with and applying industry-standard best practices is vital for any application's security:

Software Updates: Regularly updating all software dependencies ensures the application remains shielded from known vulnerabilities present in outdated libraries.

Secure Communication: The HTTPS configuration is crucial, as it encrypts the data transmitted between the client and server, assuring both confidentiality and integrity.

Data Integrity Assurance: The integration of SHA-256 hashing is pivotal to ensure that the data's integrity remains uncompromised during transmission.

By adopting these industry standards, businesses can secure sensitive data, foster trust among clientele, and align with regulatory requirements. Furthermore, it substantially reduces the risk of security breaches, which could otherwise lead to substantial financial and reputational damages. Adhering to these practices not only ensures technical robustness but also boosts the overall wellbeing of the organization in the digital landscape.