****

# 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** | **April 21, 2024** | **Nathaniel Gratton** | **Code Refactoring** |

## 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

Nathaniel Gratton

## Algorithm Cipher

**AES (Advanced Encryption Standard)** is one of the most widely used symmetric encryption algorithms. It's highly regarded for its strength and efficiency, making it a standard choice in industries where data security is paramount, such as in financial services.

* **Overview:** AES is a symmetric block cipher that encrypts data in 128-bit blocks, using key sizes of 128, 192, or 256 bits. The algorithm is well-known for its ability to efficiently protect sensitive data, maintaining both confidentiality and integrity.
* **Hash Functions and Bit Levels:** Although AES itself does not use hash functions since it's not a hashing algorithm, it operates at high-security levels with the bit sizes. For hash functions related to AES for creating checksums or HMACs, SHA-256 is often used in conjunction.
* **Random Numbers and Key Management:** AES relies on a robust mechanism for key generation, often using secure random number generators. Effective key management is critical, as the security of AES encrypted data is highly dependent on the secrecy of the key.
* **Current State of Encryption Algorithms:** AES remains the industry standard for encryption and is approved by the National Institute of Standards and Technology (NIST). Its use is pervasive in securing online transactions, VPNs, and in many applications requiring encryption across various technologies.

## Certificate Generation

**Importance of Certificate Generation:**

Generating and managing a self-signed certificate is crucial for secure communications within Artemis Financial's web interface. It allows for the encryption of data transmitted between clients and the server, safeguarding sensitive financial information against eavesdropping and tampering. This process also demonstrates compliance with best practices in cybersecurity, particularly in maintaining the confidentiality and integrity of client data.

A screenshot of a certificate

Description automatically generated

## Deploy Cipher

Objective: Implement and demonstrate the secure communication feature within the application by encrypting a unique data string and generating its checksum.

Implementation Details:

* Algorithm Used for Encryption: Advanced Encryption Standard (AES)
* Checksum Algorithm: Secure Hash Algorithm 256 (SHA-256)
* A unique data string, which contains my name "Nathaniel Gratton", was prepared for encryption to personalize the data.
* Utilizing Java's cryptographic libraries, the data string was encrypted with AES to ensure confidentiality.
* A random initialization vector (IV) was used for each encryption operation to enhance security.
* Alongside encryption, a checksum was generated using SHA-256. This hash serves as a means to verify data integrity and to ensure that the data has not been altered in transit.

A RESTful endpoint /hash was created within the Spring Boot application to perform these cryptographic operations upon request. The /hash endpoint was accessed, successfully performing the encryption and checksum generation. The application returned the following response, confirming the operation's success:

* Data: Nathaniel Gratton
* Encrypted Data: SFLSm50tfnvGkylFSYiYmmzphMzEMgNpPgGYBBmxflE=
* Checksum: 3664cfe188717210c7e9f6bd50d53f8eac890632274bd126716b8220267d419e

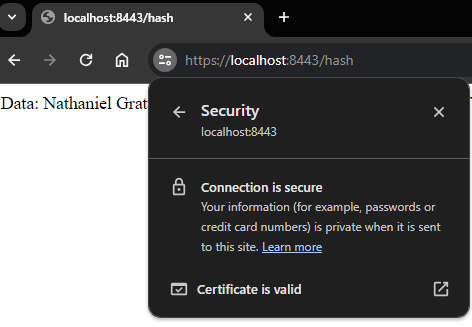
A Webpage containing test text.

Data: Nathaniel Gratton
Encrypted Data: SFLSm50tfnvGkylFSYiYmmzphMzEMgNpPgGYBBmxflE=
Checksum: 3664cfe188717210c7e9f6bd50d53f8eac890632274bd126716b8220267d419e


## Secure Communications

In the realm of secure communications, the application utilizes HTTPS protocol to ensure that all data transmitted between the client and server is encrypted and secure. This is achieved through the implementation of SSL/TLS protocols, which leverage the previously generated self-signed certificate to establish a secure channel.

* **HTTPS Configuration:** The application's properties have been updated to serve content over HTTPS, ensuring that all data in transit is encrypted using TLS with our self-signed certificate. This includes configuring the server's port to 8443 and specifying the keystore containing the certificate.
* **Browser Trust:** Despite the self-signed nature of the certificate, steps were taken to import and trust the certificate on local development machines, eliminating browser warnings and demonstrating a secure lock icon in the address bar.

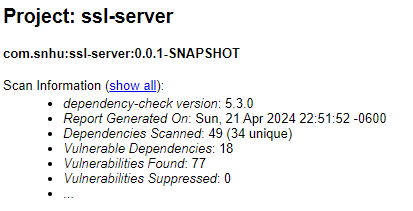


## Secondary Testing

Secondary testing was conducted using OWASP Dependency-Check to ensure that no new vulnerabilities were introduced during the refactoring process. This tool is crucial for identifying dependencies in the codebase that may contain known vulnerabilities.

Pre-Refactor:  
A screenshot of a computer error

Description automatically generated

Post-Refactor:  
  
These screenshots confirm that the refactored code didn’t introduce any new vulnerabilities into the system.

## Functional Testing

Functional testing involves the manual review of the code to identify any syntactical, logical, or security vulnerabilities that automated tests might miss. This rigorous testing phase is essential to ensure the robustness and security of the application. The code was thoroughly reviewed to confirm that it adheres to security best practices. This includes reviewing the encryption implementation, input validation, and error handling processes.

Refactored Code:

A screen shot of a computer program

Description automatically generated

Properly executed Springboot Launch with no errors:

A screenshot of a computer

Description automatically generated

## Summary

The refactoring of Artemis Financial's software application included significant enhancements to the security features. The application now implements AES for encryption and SHA-256 for checksum generation, bolstering the security measures previously in place.

* Refactoring Outcome: The code was successfully refactored to include secure communication protocols, and the application now meets the industry standards for secure web applications.
* Security Enhancements: The introduction of checksum verification adds an extra layer of integrity assurance to the data handling process, ensuring that any data corruption or tampering can be detected.

## Industry Standard Best Practices

Throughout the development and refactoring process, industry-standard best practices for secure coding were meticulously followed to mitigate known security vulnerabilities.

* Secure Coding Practices: These practices include input validation, proper error handling, secure use of cryptographic functions, and regular dependency checks to identify and remediate known vulnerabilities.
* Overall Value: Implementing these security measures not only protects the application and its users but also enhances the company's reputation and trustworthiness. This commitment to security is invaluable in today's digital landscape where data breaches and cyber-attacks are prevalent.