****

# 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** | **2/22/2024** | **Veronica Guzman** |  |

## Client



## Developer

Veronica Guzman

## Algorithm Cipher

The Advanced Encryption Standard (AES) is recommended for Artemis Financial, given its status as a symmetric encryption algorithm celebrated for its security, efficiency, and widespread adoption in protecting sensitive information. AES encrypts data in 128-bit blocks, allowing for key sizes of 128, 192, or 256 bits, which provides flexibility in balancing security needs and operational efficiency. It uses symmetric keys, meaning the same key is used for both encryption and decryption, necessitating secure key management practices. The algorithm's reliance on random numbers for key generation and initialization vectors enhances security by adding unpredictability to encrypted data. Developed by Vincent Rijmen and Joan Daemen and selected by the U.S. National Institute of Standards and Technology (NIST) in 2001, AES replaced the older Data Encryption Standard (DES) due to its stronger security features and has remained the gold standard in encryption, unbroken by cryptanalysis to this day. Its adoption ensures Artemis Financial can safeguard client data against modern security vulnerabilities, aligning with Global Rain’s commitment to implementing cutting-edge and effective software security measures.

## Certificate Generation

## Deploy Cipher

## Secure Communications

## Secondary Testing

## Functional Testing

## Summary

In developing and unit testing the SSL server application, I focused on verifying the SHA-256 hashing functionality's security and reliability, aligning closely with software requirements for secure data handling. Through a comprehensive suite of JUnit tests, including checks for consistent hash outputs, handling of varied inputs, and robustness against null or unexpected inputs, I ensured the application's technical soundness and efficiency. This testing strategy, highlighted by specific tests like assertThrows for null inputs and assertDoesNotThrow for large string processing, proved effective in not just achieving high coverage but also in affirming the application's preparedness for real-world scenarios, ultimately reinforcing its capacity to secure sensitive data communications.

## Industry Standard Best Practices

In developing the SSL server application, I applied industry standard best practices for secure coding, such as input validation, using strong cryptographic algorithms like SHA-256, adhering to the principle of least privilege, ensuring regular updates of dependencies, secure configuration through HTTPS, and incorporating code reviews and static analysis. These practices not only mitigated known security vulnerabilities, enhancing the application's security integrity, but also significantly contributed to the company's overall wellbeing by building client trust, mitigating financial and legal risks, improving operational efficiency, and fostering a culture of innovation. By prioritizing security in the software development lifecycle, the company can maintain a strong market reputation as a secure and reliable software provider, supporting its long-term growth and sustainability.