****

# 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 5](#_Toc102040759)

[3. Deploy Cipher 6](#_Toc102040760)

[4. Secure Communications 6](#_Toc102040761)

[5. Secondary Testing 7](#_Toc102040762)

[6. Functional Testing 8](#_Toc102040763)

[7. Summary 8](#_Toc102040764)

[8. Industry Standard Best Practices 9](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **2/20/2025** | **Joshua Shoemaker** |  |

## 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

Joshua Shoemaker

## Algorithm Cipher

To ensure secure financial transactions and protect sensitive client data, AES-256 with Galois/Counter Mode (AES-GCM) is the recommended encryption algorithm. AES-256 is a symmetric cipher, meaning it uses the same key for encryption and decryption, making it efficient for large data sets. AES-GCM provides authenticated encryption, ensuring both confidentiality and integrity, making it resistant to tampering and data breaches. However, encryption security depends on proper key management. Keys should be stored in Hardware Security Modules (HSMs) or encrypted vaults to prevent unauthorized access (NIST, 2023). Additionally, strict access controls are required to comply with the Gramm-Leach-Bliley Act (GLBA) and financial security regulations (Lamb & Nguyen, 2025).

To verify data integrity, SHA-256 (Secure Hash Algorithm 256-bit) should be used alongside AES-256. Hash functions convert input data into a fixed-length digest, ensuring that even minor modifications result in a completely different hash (Wagner et al., 2024). These one-way functions are essential for digital signatures and password storage. Additionally, cryptographic random number generators (RNGs) play a vital role in key generation and encryption security (Keyfactor, 2024). Poor randomness can make encrypted data vulnerable to attacks, so AES-256 requires high-entropy random numbers for key generation, and AES-GCM relies on unique initialization vectors (IVs) to prevent replay attacks (NIST, 2023).

AES-256 is a symmetric encryption algorithm, meaning both parties must securely exchange the encryption key. While this makes AES faster and more efficient than asymmetric encryption, it requires a secure key exchange mechanism, such as TLS (Transport Layer Security) (NIST, 2020). In contrast, asymmetric encryption (e.g., RSA 2048-bit) uses a public-private key pair and is commonly used to secure key exchanges before symmetric encryption begins (Keyfactor, 2024). AES-256, when combined with TLS for secure key transmission, provides the best balance of security and performance.

Encryption has evolved from simple ciphers to advanced cryptographic systems. In 1976, the Diffie-Hellman key exchange enabled secure key sharing, leading to the development of RSA encryption a year later. As computing power increased, AES became the standard, replacing weaker encryption algorithms like DES (Schneider, 2024). Today, AES-256 remains unbroken, but researchers are preparing for future threats posed by quantum computing.

To secure Artemis Financial’s systems, AES-256 with AES-GCM is the ideal encryption method, offering strong security, efficiency, and compliance with financial regulations. SHA-256 should be used for data integrity checks, and TLS with RSA should be implemented for secure key exchange. By following best practices in key management, random number generation, and encryption implementation, Artemis Financial can ensure the confidentiality and integrity of client data.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer screen

AI-generated content may be incorrect.

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A close up of a green and white background

AI-generated content may be incorrect.

## Secure Communications

Insert a screenshot below of the web browser that shows a secure webpage.

A screenshot of a computer

AI-generated content may be incorrect.

## Secondary Testing

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

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

## Functional Testing

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

A screenshot of a computer program

AI-generated content may be incorrect.

## Summary

The vulnerability assessment process helped guide the refactoring process by focusing on key areas like API security, encryption, and securing communication between the client and server. By reviewing the application architecture, providing a working example of hashing and a checksum, and making sure encryption is used properly, the updated code helps prevent security issues like unauthorized access or data breaches. The code review process also looked at securing controllers and data access.

To make the software more secure, multiple layers of protection were added. TLS encryption was enforced to make sure communication between clients and servers stays secure. Data security was improved by using AES-256 encryption and better key management practices to protect sensitive information. These changes help ensure the software is more secure and can better protect financial data from cyber threats. The refactored code follows security testing protocols by implementing static and dynamic security testing, ensuring vulnerabilities are identified and mitigated. Automated security scanning was used to check for known weaknesses, and manual code reviews were performed to validate secure coding practices. These practices comply with security standards of the National Institute of Standards and Technology and Internet Engineering Task Force (IETF) RFC 8446.

## Industry Standard Best Practices

To keep Artemis Financial’s software secure, industry best practices were used to reduce security risks and follow important security standards like OWASP Top 10, NIST guidelines, and TLS encryption protocols. Updates to the code included TLS and HTTPS to protect data during transmission, AES-256 encryption to keep stored information safe, and SHA-256 hashing to check that data hasn’t been changed. These improvements help meet the Payment Card Industry Data Security Standard (PCI-DSS) by making sure sensitive financial data is encrypted when sent over the network, stored securely, and protected from tampering. By following these best practices, the software is better protected against cyber threats and helps keep customer financial data safe.

References

Advanced Encryption Standard (AES). (2023). Category: Computer Security, Subcategory: Cryptography. Information Technology Laboratory, National Institute of Standards and Technology. Retrieved from https://doi.org/10.6028/NIST.FIPS.197-upd1

National Institute of Standards and Technology (NIST). (2020, January). Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC (NIST SP 800-38D). Retrieved from https://csrc.nist.gov/publications/detail/sp/800-38d/final

Lamb, B. A. and H., & Nguyen, S. T. (2025, January 15). Gramm-Leach-Bliley Act. Federal Trade Commission. https://www.ftc.gov/business-guidance/privacy-security/gramm-leach-bliley-act

Wagner, D., Weaver, N., Kao, P., Shakir, F., Law, A., & Ngai, N. (2024, August). Cryptographic hashes. Computer Security. https://textbook.cs161.org/crypto/hashes.html

Types of encryption algorithms + pros and cons for each. Keyfactor. (2024, April 24). https://www.keyfactor.com/education-center/types-of-encryption-algorithms/

Schneider, J. (2024, November 25). The history of Cryptography. IBM. https://www.ibm.com/think/topics/cryptography-history